Universidad de Valladolid

Escuela de Doctorado Universidad de Valladolid

Programa de doctorado en Informática

TESIS DOCTORAL:

Robust and affordable localization and mapping for 3D reconstruction. Application to architecture and construction

Presentada por **Francisco Javier Delgado del Hoyo**

para optar al grado de

Doctor por la Universidad de Valladolid

Dirigida por:

Dra. Belén Palop del Río

For the things we have to learn before we can do them, we learn by doing them.

— Aristotle, The Nicomachean Ethics

ABSTRACT

Three-dimensional reconstruction is the process of estimating the unknown depth of the points in a scene. There are several approaches to solve this problem. For example, depth can be retrieved using active sensing (e. g., laser scans) or passive sensing (e. g., digital images). The former provides more expensive, accurate and clean measurements, whereas the latter provides multiple affordable noisy measurements. Both of them can capture the color of the point according to the radiance perceived by the sensor. Hence, the fundamental differences between active and passive sensing concern the acquisition cost of the device and the accuracy of the delivered measurements.

Recent advances in passive sensing technologies have presented them as a compelling alternative to active sensing. Digital cameras are ubiquitous nowadays thanks to the popularization of smartphones. However, conventional 3D reconstruction is still not feasible on such devices due to the high computational demand of resources. The scientific community has stepped into this possibility with outstanding contributions in the last five years. Our research aims at joining to this effort, advancing in robust solutions that operate in most devices and scenes. At the same time, a good accuracy is advisable, especially in sequences recorded without challenging conditions.

Affordable and ubiquitous technologies for 3D reconstruction pave the way for wider adoption of 3D digitalization. Three-dimensional reconstruction allows users to digitize buildings and other environments that serve as a source of documentation in more advanced applications. This sort of documentation is highly demanded by the Architecture Engineering and Construction (AEC) industry to keep track of the construction progress or to save the state of the building before a refurbishment. In addition, information from traditional 2D images can be augmented with the associated 3D model.

Passive devices capture information about their environment and represent it with images. Image-based 3D reconstruction estimates the depth of any point based on its projection on multiple images. When this estimation is repeated for a large number of corresponding points in images, a point cloud of the scene is generated. In these devices, algorithms must deal with the uncertainty introduced by a large number of noisy measurements. Fortunately, such problems have been investigated by the robotics community in Simultaneous Localization and Mapping (SLAM), which comprises the algorithms for tracking the camera pose at the same time that the map of the scene is being reconstructed. In this sense, SLAM can be perceived as a natural extension of 3D reconstruction when applied to robot navigation.

Our aim is to provide an accurate 3D reconstruction with color point clouds of the scene from an off-the-shelf single-lens camera moving around the scene. In order to achieve this goal, we have developed KN-SLAM, a SLAM system that takes advantage of the contributions introduced by Oriented FAST and Rotated BRIEF (ORB)-SLAM [88]. The system follows a sparse indirect approach, guided by the ORB features detected on each frame. This is the most suitable approach for commodity cameras since their rolling shutter intro-

duces photometric artifacts that difficult the tracking of the pixels. The tracking, mapping, loop detection and relocalization stages of the SLAM pipeline rely on the quality of such features. More concretely, our contributions are the following:

- an adaptive bootstrapping procedure that takes into account the number of failures;
- an exhaustive, non-greedy, 2D-3D guided matching algorithm that ensures correspondences with a minimum distance between descriptors;
- a constrained connectivity graph where each keyframe is linked to its K-best keyframes to keep the minimum number of edges but preserve the accuracy; and
- a loop detection procedure based on smart thresholds selected according to the results achieved in previous optimizations.

We have conducted an exhaustive evaluation of KN-SLAM in 47 sequences belonging to four heterogeneous datasets. The file formats and ground truths provided by these sequences have been normalized so that they can be compared following the same methodology. Several metrics have been considered to assess the performance, accuracy, and robustness of the trajectories estimated by KN-SLAM and ORB-SLAM. Our experimental results determine that, despite the high performance achieved by ORB-SLAM, KN-SLAM is able to improve the accuracy and robustness in more than half of the sequences, although it depends on the challenges introduced by such sequences. To gain more insight, we have carried out a further analysis of the results taking into account such challenges. We have assessed nine challenging characteristics for each sequence with a value on a five-level scale. This characterization has been combined with a class label for training a SVM classifier. This label corresponds to the system that achieves a lower Absolute Trajectory Error (ATE) (KN-SLAM or ORB-SLAM). The coefficients of the trained classifier let us analyze what characteristics of the scene make KN-SLAM more suitable than ORB-SLAM. From this analysis, we have determined that KN-SLAM reduces the ATE in sequences with a good balance between rotations and translations, visual loops and poor illumination conditions. We conclude that the restricted connectivity of the graph harms the accuracy of the tracking when the camera is performing violent movements like pure rotations, but it helps to reach lower ATE when loops are closed and the drift is still tolerable.

The output of a SLAM system is usually discrete (i. e., it is a point cloud instead of a triangulated 3D mesh). The mesh can be achieved by an additional 3D reconstruction stage, which is based on the volumetric fusion of multiple partial meshes. Different systems in different application domains can include the information provided by the reconstructed mesh in certain functionalities. One of the sectors more interested in 3D reconstruction and digitalization is the AEC industry. For example, fast and cheap 3D models can be applied to track the state of a construction w. r. t. the prior design. In addition, the sector is slowly transitioning from a CAD-based methodology to the Building Information Modelling (BIM) methodology, in which stakeholders collaborate

around a shared model of the building. This model includes both the geometry and the semantics of the structural parts and the constructive processes of the building.

Despite the benefits of BIM for the AEC industry, its adoption is being quite lazy. In most cases, it is imposed by the customer of the project. Given our deep knowledge of the sector and the suggestions received from several BIM stakeholders, we aim at fostering the adoption of the BIM methodology with an easy-to-use cross-platform system called 3D-SIMOS. More specifically, our goal is to reduce the gap and pitfalls found at moving from CAD to BIM in the design phase. We address the challenge by providing AEC stakeholders with a solution to manage constructive processes over the structure of the building represented by a 3D mesh. Moreover, 3D-SIMOS does not only allow stakeholders to create, simulate, track and monitor the advances of the construction but also to visualize information about the building (e. g., to showcase the building to an interested customer).

Advanced Visualization techniques have been applied to design smart interfaces over a cross-platform implementation that exploits the WebGL API of modern web browsers. Any project in 3D-SIMOS is generated from standard CAD 3D model and planning where the tasks and resources are arranged. Information from both schemas is integrated into the framework proposed by the Industry Foundation Classes (IFC) using a customized alignment between IFC, MPP and COLLADA.

Furthermore, we have proposed an explicit symbolic representation to include dynamics of constructive processed over static representations of the building like IFC. This representation is based on an expandable dual graph that encodes the interconnected spaces of the building. In this graph, we have defined a computational framework for the functionals that evaluates the dynamic attributes of constructive processes. The evolution of such attributes in time describes a flow, which can be visualized with different types of fields.

This system has been evaluated by combining qualitative and quantitative aspects extracted from interviews with experts of the AEC industry and our experimental results. We have concluded that 3D-SIMOS can promote the adoption of the BIM methodology by stakeholders that are still not accustomed to applying it in their current workflow. In addition, the system operates with good performance on middle-end smartphones and desktops, and it is also easy-to-use. However, some of the stakeholders have noted that the rendering quality of the visualization may not be compelling for showcasing the building to possible customers since computer-generated images are frequently accepted for this task. Besides, the oversimplification of the available functionalities makes the system naive from the perspective of a professional user that would require to integrate more powerful tools with 3D-SIMOS. Fortunately, the system was designed with extensibility in mind so the required effort to aggregate new functionalities would be minimal thanks to the underlying BIM model.

RESUMEN

La reconstrucción tridimensional consiste en ser capaz de estimar la profundidad o tercera dimensión de un punto de la escena. Este problema puede ser abordado con diferentes dispositivos y algoritmos. Por ejemplo, los dispositivos activos (e. g., escáner láser) se caracterizan por emitir una señal (normalmente un haz lumínico) que rebota en el objeto y retorna al dispositivo. El tiempo de retorno es proporcional a la profundidad del punto. Por otro lado los dispositivos pasivos (e. g., una cámara digital) simplemente capturan la información del entorno. Los primeros se caracterizan por ser más caros, pero proporcionan información precisa y densa. Los segundos son mucho más asequibles y proporcionan mucha más información, pero con mucho más ruido e incertidumbre. Obviamente los algoritmos que se utilizan en cada caso son radicalmente distintos.

Las cámaras digitales están integradas en multitud de dispositivos en la actualidad, lo que las convierte en candidatas a multitud de aplicaciones. Una de ellas es la reconstrucción 3D. Sin embargo los algoritmos que se vienen aplicando los últimos diez años no pueden implantarse sin más sobre dispositivos con tan escasos recursos. Desde la aparición de KinectFusion, muchas han sido las aportaciones para intentar conseguir resultados similares a un escáner pero con dispositivos mucho más asequibles. Sin embargo los requisitos hardware siguen siendo muy altos y los resultados obtenidos sin cámaras RGB-D no tienen la misma calidad visual. Existen multitud de aproximaciones distintas en la literatura en función del hardware, del tipo de cámara y de los resultados esperados. Nuestro objetivo es profundizar en este sentido para el caso de una cámara convencional con una sola lente, tratando de obtener una solución precisa y robusta al mismo tiempo en diferentes tipos de escenas.

La simplificación y abaratamiento de las tecnologías de reconstrucción 3D hace más accesibles sus resultados en diferentes aplicaciones. La digitalización de edificios y entornos 3D es de suma utilidad para generar documentación 3D con múltiples fines en diferentes industrias. Por ejemplo, en la industria AEC se utiliza para seguir la evolución de la construcción de un edificio o para capturar el estado actual antes de una rehabilitación. Además un modelo 3D proporciona información suplementaria que se puede superponer sobre la imagen 2D convencional.

La reconstrucción 3D en imágenes está basada en las proyecciones de un punto en el espacio sobre múltiples imágenes. Dicho procedimiento puede repetirse en múltiples puntos correspondientes de imágenes relacionadas para obtener una nubes de puntos. El principal problema radica en manejar la incertidumbre y el ruido cuando se utilizan dispositivos de baja calidad. Afortunadamente estos problemas han sido estudiados por la comunidad en Robótica en la línea de investigación conocida como Simultaneous Localization and Mapping (SLAM). Estos algoritmos permiten seguir la posición de la cámara al mismo tiempo que se reconstruye el mapa de la escena, lo cual encaja perfectamente con nuestro objetivo principal: obtener una reconstruc-

ción 3D precisa en color de una escena arbitraria y la trayectoria de la cámara que la recorre usando una cámara simple, de bajo coste y con una sola lente.

Para conseguir nuestros objetivos hemos desarrollado KN-SLAM, un sistema SLAM basado en las contribuciones desarrolladas por ORB-SLAM. La estrategia seguida es indirecta y dispersa, guiada por las características ORB detectadas en cada nuevo frame. Esta estrategia es la que mejor se adapta a cámaras de baja calidad debido a los artefactos creados por el rolling shutter. Las principales etapas de la tubería de procesamiento dependen de la calidad de las características detectadas: seguimiento, mapeado, relocalización y detección de lazos. Más concretamente, nuestras contribuciones podrían resumirse en:

- un procedimiento automático de arranque que se adapta al número de fallos en los primeros intentos;
- una búsqueda exhaustiva y no avara de correspondencias entre puntos 3D y 2D que garantice que los pares resultantes son realmente los más cercanos;
- un grafo de conectividad entre keyframes en el que el número de vecinos depende del número de observaciones representadas y está limitado a las K mejores;
- un procedimiento de detección de lazos que aplique umbrales variables en los procedimientos de filtrado de candidatos con RANSAC y las optimizaciones.

Para evaluar los resultados hemos utilizado un total de 47 secuencias de 4 conjuntos muy diferentes. Estas secuencias han sido normalizadas con el mismo formato y convenciones para representar el fondo de verdad. Esto permite evaluar todas las secuencias utilizando la misma metodología y agilizar la evaluación de nuevas secuencias. Aunque hemos hemos evaluado diferentes aspectos, nos hemos centrado en la precisión y la robustez en la trayectoria de la cámara. Nuestros experimentos han demostrado que KN-SLAM mejora los resultados de ORB-SLAM en más de la mitad de las secuencias.

Sin embargo hemos detectado que los resultados varían ligeramente dependiendo de los retos que presenta cada una de las secuencias. Hemos profundizado en este análisis a través de una caracterización de las secuencias en términos de sus retos característicos. Para ello hemos evaluado nueve características en una escala Likert de 5 niveles. Esta caracterización ha sido combinada con una categorización binaria de las secuencias según KN-SLAM haya obtenido una trayectoria más precisa o no. Estos datos son utilizados para entrenar un clasificador SVM con un kernel lineal que simplifica su interpretación geométrica. Los valores absolutos de los coeficientes del clasificador son proporcionales a la importancia que cada característica tiene a la hora de obtener mejores resultados y el signo indica si favorece a ORB-SLAM o KN-SLAM.

De nuestro análisis se desprende que KN-SLAM obtiene mejores resultados en secuencias donde la cámara realiza movimientos translacionales y rotacionales a partes iguales, tiene algún lazo visual y presenta algunos tramos de baja iluminación. Podemos concluir que limitar la conectividad del grafo daña la precisión de la trayectoria cuando la cámara realiza movimientos bruscos, tiene relocalizaciones o rotaciones puras. Por otro lado, ayuda a obtener mejores resultados cuando en la optimización de los lazos siempre que la deriva acumulada no sea excesivamente alta.

La salida proporcionada por SLAM es normalmente discreta, i. e., es una nube de puntos y no una malla con textura. Para conseguir esta malla es necesario incluir en la tubería una etapa adicional de reconstrucción 3D que se basa normalmente en integrar todas las mallas parcialmente reconstruidas en una parrilla volumétrica. La información 3D es aplicable en muchos dominios para sus tareas convencionales. Este es el caso de la arquitectura, ingeniería y construcción (AEC), que demandan digitalización 3D para la mayoría de sus tareas cotidianas. Por ejemplo, para seguir los avances de una construcción podrían reconstruir el resultado una vez obtenido y compararlo con el diseño.

Por otro lado, la industria AEC está evolucionando progresivamente desde el tradicional diseño asistido por ordenador (CAD) hacia una nueva metodología basada en el modelado de información del edificio (BIM). En esta nueva metodología las partes involucradas en el diseño, construcción y funcionamiento del edificio colaborar alrededor de un modelo compartido. Este modelo incluye tanto información semántica como geométrica del modelo.

A pesar de las grandes ventajas que propone esta metodología, su penetración está siendo más lenta de lo previsto debido principalmente a la complejidad del nuevo modelo. En la mayoría de casos es un requisito impuesto por el cliente del proyecto. Gracias a nuestras buenas relaciones con los agentes en la industria AEC hemos podido recoger sus sugerencias con el objetivo de desarrollar una aplicación (3D-SIMOS) que ilustre las ventajas del uso de BIM. Esta aplicación es sencilla de usar y entender, y es multi-dispositivo. Con ella pretendemos reducir la brecha y los obstáculos iniciales que encuentran muchos agentes cuando empiezan a utilizar la metodología BIM. Más concretamente la aplicación permite visualización avanzada de proyectos a partir de una planificación de obra y un modelo CAD 3D. Cualquier agente puede crear, simular, seguir y monitorizar los avances del proyecto, además de inspeccionar el modelo 3D y las tareas planificadas. Por ejemplo, el mismo proyecto puede ser usado para determinar el estado actual de la construcción y para mostrar el edificio diseñado a posibles clientes.

El soporte multi-plataforma está basado en WebGL, un API que proporcionan los navegadores web para obtener acceso a la aceleración por hardware de la GPU. Los proyectos se crean reutilizando artefactos generados en la metodología CAD como una planificación en MPP y un modelo 3D en COLLADA. Esta información es integrada en el estándar IFC mediante un alineamiento manual de conceptos que se muestra como una contribución adicional. Además, hemos propuesto un nuevo marco de trabajo para BIM dinámico basado en una representación simbólica en forma de un grafo dual expansible que recoge tanto aspectos estructurales del edificio como de los procesos que se desarrollan en sus espacios. Estos procesos se representan mediante funcionales que evalúan atributos representables mediante diferentes tipos de campos.

Esta segunda parte de nuestra investigación ha sido evaluada combinando aspectos cualitativos (e. g., facilidad de uso) y cuantitativos (e. g., rendimiento)

extraídos de entrevistas con expertos y de nuestros resultados experimentales. Hemos concluido que 3D-SIMOS ayuda a entender mejor las ventajas de la metodología BIM, especialmente para aquellos agentes que no la conocen debido a su facilidad de uso y sencillez práctica. La aplicación funciona correctamente en dispositivos móviles de gama media y equipos de escritorio. Sin embargo algunos agentes han indicado en su evaluación que el modelo 3D visualizado carece de la suficiente calidad para mostrar el edificio a posibles clientes. Esta diferencia se aprecia debido a que se comparan los resultados con las imágenes generadas por computador que se utilizan habitualmente en las promociones. También han hecho notar que las simplificación de las funcionalidades en 3D-SIMOS podría resultar excesiva desde el punto de vista de un profesional. En estos casos el sistema está preparado para extender sus funcionalidades bajo el paraguas del modelo BIM subyacente.

PUBLICATIONS

The results obtained from our research are disseminated across different publications. This page provides a complete list of the papers and scientific publications produced by the author of this thesis that are directly related to the contents of the document. In some cases, they are in preparation (i. e., they have not been published yet) because they contain the latest results of our research. Most ideas and figures are included in the following publications:

[1] Francisco Delgado and Javier Finat. "Complete endomorphisms in Computer Vision." In: *Submitted to the International Journal of Mathematical Imaging and Vision (Under revision)* (2018).

[2] Francisco Delgado and Javier Finat. "Dynamic Building Information Modelling with digital cameras." In: *Proceedings of the 1st Ibero-American Congress of Smart Cities (ICSC-CITIES 2018)*. Springer, 2018. URL: http://www.icsc-cities2018.com/EN_index.html.

[3] Francisco Delgado, Javier Finat, and Belén Palop. "Accurate and robust monocular SLAM in natural sequences." In: *In preparation* (2018).

[4] Francisco Delgado, Javier Finat, and Belén Palop. "Robust mathematical modelling for motion from video sequences with a hand-held camera." In: *Proceedings of the 7th Iberoamerican Congress on Geometry, January 22nd-26th, Valladolid, Spain*. University of Valladolid, 2018. URL: http://iberoamericangeometry2018.uva.es.

[5] Francisco Delgado, M. Mercedes Martinez-Gonzalez, and Javier Finat. "An evaluation of ontology matching techniques on geospatial ontologies." In: *International Journal of Geographical Information Science* 27.12 (2013). (**JCR, Q1**), pp. 2279–2301. DOI: 10.1080/13658816.2013.812215.

[6] Francisco Delgado et al. "Towards a client-oriented integration of construction processes and building GIS systems." In: *Computers in Industry* 73 (2015). (**JCR, Q2**), pp. 51–68.

[7] Jose Luis Martínez et al. "Augmented reality to preserve hidden vestiges in historical cities. A case study." In: *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 40.5 (2015), p. 61.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth[69]

ACKNOWLEDGMENTS

This work has been funded by a scholarship granted by University of Valladolid and Santander Bank.

I would like to express my special appreciation and thanks to my advisor Professor Dra. Belén Palop del Río, you have been tremendous mentors to me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Special thanks also to Dr. Javier Finat Codes for providing me the mathematical background required to understand the basics behind Simultaneous Localization and Mapping and to support me daily at work. Your advice and support have been priceless. You have provided me with brilliant ideas, comments, and suggestions. And thanks also to my fellow Ph. D. students at the University of Valladolid, and especially to Beatriz, who has encouraged me in the most difficult moments.

Thank you very much to every single person that made me feel at home during my research stay at Polytechnic University of Catalonia, especially to Dr. Pere Brunet.

Last, but not least, special thanks to my family. Words cannot express how grateful I am to my mother and sister for all of the sacrifices that they have made on my behalf over this period. I would also like to thank my friends for their advice and revisions in the writing process, which prompted me to strive towards my goal. Finally, I would like to express appreciation to my dearest friend Elena who always supports me in the moments when there was no one to answer my queries.

My special gratitude to anyone who has contributed to make this thesis a reality. You are so many that there is not enough space in this document to name you all.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ADE	Application Domain Extension (p. 52)
AEC	Architecture Engineering and Construction (pp. iv–vii, ix, 2, 3, 5–9, 48–50, 52, 54, 56, 118–121, 129, 131, 137, 139, 141, 143, 145, 148–150, 152–155)
API	Application Programming Interface (pp. vi, ix, 53, 137, 146, 151)
AR	Augmented Reality (pp. 5, 46, 49, 118, 154)
ARM	Advanced RISC Machine (p. 5)
ASL	Autonomous Systems Lab (p. 87)
ATE	Absolute Trajectory Error (pp. v, xiii–xv, 31, 88, 89, 91, 92, 95, 97, 99–101, 103, 104, 108, 110, 114, 115, 118, 161, 167, 168, 167, 169, 167)
ATRE	Absolute Trajectory Rotational Error (pp. xv, 89, 103)
BA	Bundle Adjustment (pp. 16, 17, 20, 21, 23, 32, 33, 36, 38–40, 42, 43, 61, 82, 110, 111, 118)
BIM	Building Information Modelling (pp. v, vi, ix, xiii, 2, 5–10, 49, 48–50, 49–51, 50, 52–59, 119–121, 128, 129, 131, 132, 148–150, 152–156)
CAD	Computed-Aided Design (pp. v, vi, ix, 8, 50, 52, 119, 120, 128, 129, 131–133, 138, 139, 141, 148–150, 153, 155)
CAGR	Compound Annual Growth Rate (p. 3)
CAM	Computer-Aided Manufacturing (p. 49)
CDF	Cumulative Distribution Function (p. 67)
CityGML	City Geography Markup Language (pp. 51, 52, 58, 121, 138)
CMOS	Complementary Metal–Oxide–Semiconductor (pp. 5, 21)
CNN	Convolutional Neural Network (pp. 29–31)
COLLADA	COLLAborative Design Activity (pp. vi, ix, xiv, 53, 128–130, 132, 133, 136, 137, 149, 150, 153)
CPU	Central Processing Unit (pp. 4, 5, 19, 25, 28, 46–48, 97, 117, 141)
CRF	Camera Response Function (pp. 30, 79)
CRS	Coordinate Reference System (p. 51)
CRUD	Create, Read, Update and Delete (pp. 136, 137)

CSS	Cascading Style Sheets (p. 137)
DLT	Direct Linear Transform (p. 34)
DoF	Degrees of Freedom (pp. 18, 25, 36, 41, 43, 77, 89)
DOM	Document Object Model (p. 137)
EKF	Extended Kalman Filter (p. 28)
EuRoC	European Robotics Challenges (pp. xiv, xv, 74, 77, 78, 81, 84, 85, 87, 91, 93, 95, 100, 102, 101–103, 112, 164, 167)
FAST	Features from Accelerated Segment Test (pp. 28, 36)
FOV	Field of View (pp. 5, 24, 78, 82)
FPS	Frames Per Second (pp. 6, 25, 28, 30, 48, 74, 76, 78, 141, 142, 146, 149, 151)
GCC	GNU Compiler Collection (p. 93)
GIS	Geographic Information System (pp. 51, 52, 58, 121)
GLSL	OpenGL Shading Language (p. 151)
glTF	GL Transmission Format (p. 53)
GPS	Global Positioning System (pp. 18, 19, 75)
GPU	Graphics Processing Unit (pp. ix, 24, 25, 29, 30, 46, 47, 53, 146, 151)
GUI	Graphical User Interface (pp. xv, 33, 61, 116, 131, 135–137, 140, 142, 150, 151)
HTML	Hypertext Markup Language (pp. 137, 146)
ICL-NUIM	Imperial College London and National University of Ireland Maynooth (pp. xiv, xv, 74, 78, 79, 81, 84, 87, 92, 93, 95, 97, 100–103, 114, 164, 167)
IDF	Inverse Document Frequency (p. 62)
IFC	Industry Foundation Classes (pp. vi, ix, xiv, 49, 52, 120, 121, 128–130, 138, 139, 145, 148–150, 152–154)
IMU	Inertial Measurement Unit (pp. 19, 20, 77, 78, 81, 95)
iSAM	Incremental Smoothing and Mapping (p. 25)
JSON	JavaScript Object Notation (pp. 136, 137)
JVM	Java Virtual Machine (p. 137)
KITTI	KITTI Vision Benchmark Suite (pp. xiv, xv, 76, 86, 97, 99, 101–103, 164, 167)
KMS	Knowledge Management Systems (pp. 5, 154)
KNN	K-Nearest Neighbors (pp. 44, 45, 60)
LASSO	Least Absolute Shrinkage and Selection Operator (p. 107)
LIDAR	Light Detection and Ranging (p. 75)
LM	Levenberg-Marquardt (pp. 13, 16, 42, 73)
LoD	Level of Detail (pp. 122, 157)
LSD	Large Scale Detection (pp. 25, 28, 29)

MAP	Maximum A Posteriori probability (p. 30)
MAV	Micro Aerial Vehicle (pp. 28, 74, 77, 78, 81, 85, 95)
MLE	Maximum Likelihood Estimation (p. 11)
MLP	Multilayer Perceptron (p. 109)
MPP	Microsoft Project Plan (pp. vi, ix, xiv, 128–130, 132, 133, 136, 149, 150, 153)
MST	Minimum Spanning Tree (p. 122)
MVC	Model-View-Controller (p. 135)
MVD	Maximum Vertex Degree (pp. 67, 68)
MVS	Multiple View Stereo (pp. 2, 6, 19)
NID	Normalized Information Distance (p. 25)
NLLS	Non Linear Least Squares (pp. 16, 102)
ODIN	Indegree Number (p. 45)
OpenGL	Open Graphics Library (pp. 25, 116)
ORB	Oriented FAST and Rotated BRIEF (pp. iv, v, viii, xiv, 8, 17, 25, 27, 28, 31–37, 42, 43, 60–66, 68, 71, 73, 79, 80, 84, 85, 88, 92, 94, 95, 97, 99–104, 108, 110–117, 153, 154, 159, 161, 164, 167)
OS	Operating System (p. 93)
PBR	Physically Based Rendering (pp. 53, 145, 152)
PL	Piecewise Linear (pp. 46, 55, 58, 69–71, 121, 123, 126, 127, 152, 160)
PNG	Portable Network Graphics (pp. 75, 86)
PnP	Perspective-n-Point (pp. 43, 83, 95, 100, 112)
PS	Piecewise Smooth (pp. 121, 126)
PTAM	Parallel Tracking and Mapping (pp. 27, 36)
PTT	Percentage of Tracked Trajectory (pp. xiii, xv, 88, 92, 94, 95, 101, 114, 115, 164, 165, 164, 166, 164)
RAM	Random Access Memory (pp. 93, 141)
RANSAC	Random Sample Consensus (pp. viii, 34, 41, 43, 48, 61, 72, 92, 112)
REST	REpresentational State Transfer (pp. 135, 136)
RFE	Recursive Feature Elimination (p. 107)
RGB-D	RGB and Depth (pp. vii, 1, 6, 25, 26, 29, 46, 47, 60, 78, 86)
RMSE	Root Mean Squared Error (pp. 88, 89, 91, 167)
ROS	Robot Operating System (p. 93)
SCAD	Smoothly Clipped Absolute Deviation (p. 107)
SfM	Structure from Motion (pp. 2, 6, 16, 19, 31)
SIFT	Scale-Invariant Feature Transform (pp. 27, 29)
SIMD	Simple Instruction Multiple Data (pp. xviii, 5, 93)

SLAM	Simultaneous Localization and Mapping (pp. iv, v, vii–ix, xi, xii, xiv, xv, 2–9, 16–21, 23–32, 34–37, 42, 43, 45, 46, 59–66, 68, 71, 73–89, 91–95, 97, 99–102, 101–103, 102–104, 108, 110–118, 152–154, 156, 159, 161, 164, 167)
SLERP	Spherical Linear intERPolation (pp. 54, 55, 138, 145)
SNR	Signal-to-Noise Ratio (pp. 21, 24)
SSE	Streaming Simple Instruction Multiple Data (SIMD) Extensions (p. 93)
SVD	Singular Value Decomposition (p. 36)
SVM	Support Vector Machine (pp. v, viii, xiv, xv, 85, 105–109, 113, 114, 161)
SVO	Semi-direct Visual Odometry (p. 28)
TF	Term Frequency (p. 62)
TFIDF	Term Frequency - Inverse Document Frequency (p. 62)
ToF	Time of Flight (pp. 1, 30, 86)
TUM RGB-D	Technical University of Munich RGB-D (pp. xiv, xv, 30, 34, 37, 39, 44, 74, 76–78, 81, 86, 87, 92–94, 97, 99, 101–103, 110, 113, 164, 167)
TV	Total Variation (pp. 30, 47)
UAV	Unmanned Aerial Vehicle (p. 18)
UML	Unified Modeling Language (pp. 52, 134)
VO	Visual Odometry (pp. 2, 19, 20, 27, 28, 31, 36, 47, 74–77, 79, 81, 82, 154)
VR	Virtual Reality (pp. 46, 49, 118, 154)
VRML	Virtual Reality Modeling Language (p. 53)

1 INTRODUCTION

Information Systems have greatly evolved since the beginning of Computation. Computer Science is relatively young when it is compared to other disciplines, such as Mathematics and Physics. Nevertheless, nowadays most of the business models of innovative companies are powered by some kind of underlying technology and computation model. Moreover, the biggest companies in the world are technology-related (e. g., Apple, Google, Amazon or Microsoft). These companies also invest a huge amount of resources in research and development activities that contribute to the improvement of the computation systems they use in their business.

The advances introduced in computing systems have motivated the development of new applications that were unimaginable before. The more computing power, the higher the application requirements to understand, model and solve complex problems. This power improvement allows us to tackle new problems. Most of them have been introduced by the broad variety of devices able to perceive or "sense" the environment, commonly known as sensors.

The information captured by these sensors is powering a lot of new applications to multimedia and digitalization. Modeling, reality understanding and interaction are involved in both areas. These applications are of interest for diverse sectors, such as entertainment, architecture, civil engineering, medicine or tourism, e.g.. The huge amount of information provided by sensors has led to the emerging of new disciplines like big data or deep learning.

Digital cameras are among these sensors since they capture the light emitted or reflected by the objects in the scene. A matrix of pixels represents this information with functions defined on pixels whose number depends on the resolution of the camera. Since a camera only receives information about the environment without emitting any signal, they are considered passive sensors. Laser scanners and Time of Flight (ToF) cameras are examples of active sensors that transmit and receive a light signal to measure distances according to the time the signal takes to return to the device.

In this work, we have focused on digital cameras understood from a Computer Vision approach. We have dived into the process of capturing and modeling the environment using fundamental concepts of photogrammetry. This technique estimates the depth of a 3D point based on its corresponding projections on multiple images. The distance between two projections is called parallax. Images can be captured by multiple cameras or by placing a single camera at different locations at different times. In this way, it is plausible to estimate the depth of one point in a scene from a single camera if this camera travels around the scene. The recovering of the third dimension of this point is the key for 3D reconstruction. Furthermore, each camera can contain one or more lenses (e. g., binocular cameras or RGB-D cameras). Hence, the selected approach to tackle the problem must always consider the type of camera and the sensors it includes.

The 3D reconstruction problem has been tackled from different approaches. They have evolved along the last 30 years from the very basic perspective 2.5D models to the real-time dense 3D reconstructions that we can achieve nowadays. However, most of them are based on specific sensors, which are usually expensive or only affordable for custom-tailored applications. In other cases, the algorithms require a powerful parallel computer architecture to run properly in real-time. Obviously, these requirements depend on the desired degree of accuracy and completeness for the reconstruction, or on that the desired artifact is a pure cloud or a triangulated mesh.

While the state of the art has advanced in high-performance high-quality reconstruction (e. g., Multiple View Stereo (MVS) and Structure from Motion (SfM) in COLMAP [112]), new low-cost hardware setups have appeared in the recent years associated with mobility and the Internet of Things. These new setups do not provide the same computing power as their traditional alternatives but they are still able to reconstruct scenes with enough level of detail to serve for several different purposes and applications. Algorithms and systems have evolved to support these new computing architectures.

The Robotics community is accustomed to operating in hardware environments with limited resources. There are two approaches coming from this community that can be applied to a low-cost reconstruction system. One of them is Visual Odometry (VO), which is the process of determining the position and orientation of a robot in a local environment by analyzing the images captured by the camera. The other concept is Simultaneous Localization and Mapping (SLAM). SLAM consists in the concurrent construction of a model of the scene (the map) in a global environment at the same time that the camera pose is estimated. Both concepts are related since SLAM can be understood as an improvement over VO or VO can be understood as part of SLAM (more details in Section 2.3.1). The Robotics and Computer Vision communities have made astonishing progress over the last 30 years in both areas, enabling large-scale real-world applications and witnessing a steady transition of this technology to industry.

Our aim is to contribute to the widespread adoption of 3D reconstruction, i. e., to put 3D digitalization within reach of any user with low-cost commodity cameras such as webcams or smartphones. To achieve this goal, our research is focused on the investigation of affordable SLAM algorithms to generate 3D information in devices with limited resources. This research has been addressed without a specific application in mind in Chapter 3. Later, 3D models can be incorporated in applications particularly tailored for a particular purpose, e. g., the building industry or, more generally, the AEC industry.

The building industry is progressively transitioning from the traditional manufacturing processes to more industrialized processes aided by technology. An important piece of this technology is the Building Information Modelling (BIM). Actually, BIM proposes a completely different methodology to integrate all the information about the building in a single framework. Every agent linked to the building in one of its stages across its lifespan produces and consumes this information.

Despite the power of the BIM methodology, there is still a relatively low adoption by the AEC industry. One of the main barriers to extended adoption is the complexity of the model. The other problem concerns the cumbersome task of annotating geometry with semantics like the construction phases of a building. The second part of our research is committed to addressing both issues. Once the model is properly tagged, enriched applications can use the information to provide valuable insights. One of these insights is the monitoring of the construction progress that will let the manager detect offsets in time and space. The model also enables a simulation of the evolution in construction processes. The elements of the building affected by the construction activities can be inspected individually by the user. This application of 3D reconstruction to these tasks will be studied in Chapter 4.

The research in these fields is motivated by the growing interest in the worldwide industry for this kind of technologies. There are two powerful applications to hold this demand steady in the long-term: the necessity of intelligent computing systems able to understand what they are doing, and the digitalization of the environment. They have motivated an equally important growth in the research of new algorithms and technologies for 3D digitalization, autonomous navigation, and augmented/virtual reality. In fact, the development of three-dimensional applications for the industry has grown exponentially in the last decade and this pace is expected to continue. The average Compound Annual Growth Rate (CAGR) estimated for the next five years is currently around 20 %.

The remainder of this chapter is devoted to framing our work in a specific niche inside a wider field of research. It is structured as follows: Section 1.1 proposes the general framework to place the reader in the context of this research and highlights our contributions; Section 1.2 includes the main goals pursued with our contributions.

1.1 THE RESEARCH FRAMEWORK

This section is intended to supply the reader a broader view of the framework in which our research can be described. The conceptual pipeline of this framework is depicted in Figure 1.1. In this pipeline we have distinguished three main components, with our contributions focused on the components one and three. On the one hand, the components in the upper half of Figure 1.1 are application-agnostic, i. e., they can be applied to different purposes in multiple contexts. On the other hand, the component in the lower half is entirely devoted to solving problems for the AEC industry.

The pipeline begins with a sequence of pictures captured by a simple, low-cost camera. These pictures can be analyzed and processed to provide the location of the observer and a map of the scene. This problem is tackled by a general-purpose SLAM system that we call KN-SLAM. The resulting sparse map can be converted into a textured 3D mesh by a 3D reconstruction algorithm, represented as the second component of the pipeline. The textured 3D mesh can be of use in several settings and applications. In particular, we aim at a solution for construction managers in the AEC industry that we call 3D-SIMOS. This solution is represented by the third component of the pipeline.

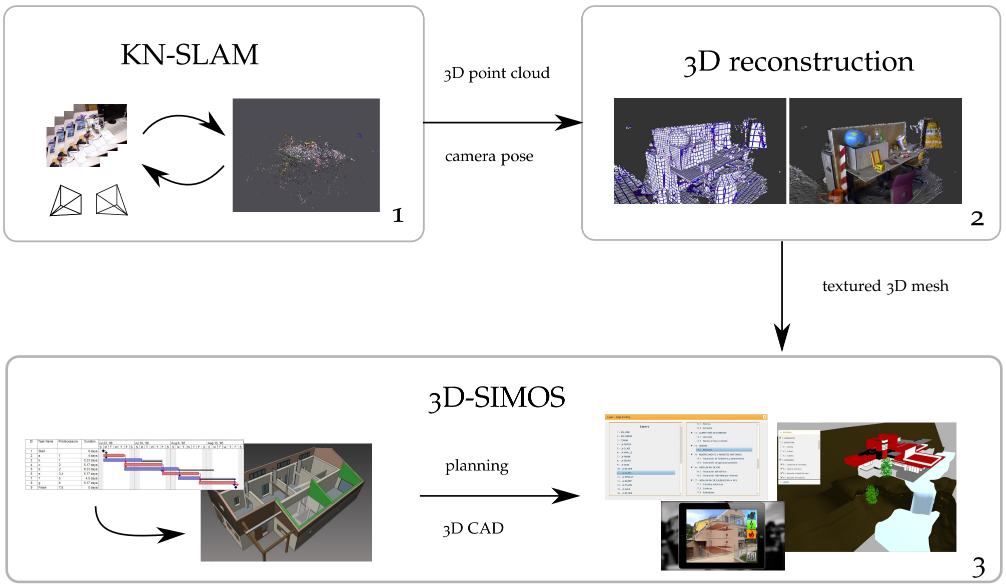


FIGURE 1.1: The three components of the framework of our research. Our contributions are focused on the component one (KN-SLAM) and three (3D-SIMOS).

In the first component, we aim at advancing in the state of the art of low-cost, accurate SLAM. Three-dimensional reconstruction and digitalization is a fast growing industry with many applications. However, there are significant barriers that limit a wide adoption. One of the biggest challenges is to reduce these barriers, allowing SLAM to run to its best in devices with a limited amount of resources. This research has been addressed from a basic point of view in Chapter 3, i. e., without considering the particular interest of any industry.

It is also important to note that besides the ground truth of the sequences, our research must also be compared with the up-to-date state of the art. Thus, we shall start by defining the operating conditions of our system in a quantitative and closed way that allows us to choose the best approaches to compare KN-SLAM.

Despite most SLAM approaches share the same theoretical background, they exhibit subtle differences depending on the operating conditions. These conditions include the number of cameras, the type of lens, the shutter of the camera, the Central Processing Unit (CPU) architecture or even the compiler. Recording conditions of the sequence are also considered since they affect the accuracy and performance of SLAM. For instance, short sequences with strong camera rotations have different requirements than long sequences where the camera describes a straight trajectory. The kind and the number of lenses are the most significant conditions to pick a specific SLAM strategy. The presence of additional sensors would also require modifications in the algorithms. By taking this into account, we shall constrain ourselves to the following operating conditions:

- The acquisition of the images is performed using a low-cost passive device with a single lens, i. e., a commodity monocular digital camera. As a reference, the estimated cost of the camera should be below 50 €.

- Color information provided by the sensor should be included in the points of the map.
- The Field of View (FOV) of the lens is between 60°-70°, i. e., we do not consider fish-eye lenses.
- The sensor area is below $5\,\mathrm{mm}^2$.
- The sensor uses Complementary Metal–Oxide–Semiconductor (CMOS) technology.
- The camera may be equipped with a rolling shutter device, although global shutter is also supported.
- The system exploits parallelism up to CPU level, taking advantage of state-of-the-art CPU architectures with multiple core and SIMD instructions available both in Advanced RISC Machine (ARM) and x86 architectures.
- The sequences are sufficiently large to have a significant impact on the drift along the trajectory.

The goal of KN-SLAM is to accurately track the camera pose while a sparse 3D map is being reconstructed in the background. The 3D map is represented by a sparse point cloud, normally not suitable for applications that require textured 3D meshes. However, these meshes can be generated by the 3D reconstruction component from the output of KN-SLAM, which comprised the poses of the keyframes and a sparse point cloud. The task is usually executed off-line to take advantage of the optimization procedures in the estimated trajectory. In this way, the accuracy of the camera poses and the map points is maximized, as in e. g., [89]. Resulting models are composed of a mesh and a texture, which feed 3D-SIMOS. State-of-the-art approaches for implementing this component are reviewed in Section 2.6.

Once the 3D information has been properly captured and modeled, it can be used in very different contexts. The third component, called 3D-SIMOS, has been designed with this goal in mind. Instead of dealing with a broad range of situations, it exploits our expert knowledge in a particular field where our research group has made numerous contributions over the last years: AEC. We have provided constructors with solutions based on a mix of Computer Vision, Computer Graphics and Knowledge Management Systems (KMS) inspired by the BIM methodology. Our contributions regarding this BIM-based tool are introduced in Chapter 4.

In this context, 3D-SIMOS links 3D models to the planning of the construction so it can be interpreted as a 4D model. Three-dimensional building models are organized in several layers, with multiple meshes per layer. These meshes are linked to stages of the planning, following a BIM-like methodology. This association allows the user to trace the advances in the planning of the building project along its construction. Furthermore, the meshes of the model can be compared to the meshes provided by the second component. This comparison between the 3D capture and the expected 4D mesh can be visualized in an Augmented Reality (AR) environment. The observer is located and the mesh is reconstructed using KN-SLAM, to let 3D-SIMOS integrate this information with the scheduled state of the construction.

The visualization can be implemented in an application that allows the user to inspect the building interactively. Moreover, the associations between meshes and stages can be exploited for simulating the evolution of the construction process. The designed and captured models are useful to estimate deviations in the execution of the project. What is more, this information is available for accurate reporting about the performance of the construction process after it finishes. This also enables the monitoring of the construction processes.

Stakeholders involved in the AEC industry have outlined a collection of demands in the BIM methodology. We have verified such demands in our interviews with involved enterprises that are interested in 3D-SIMOS. The following requirements have been addressed by 3D-SIMOS:

- The geometric information of the building is associated with semantic information concerning the planning.
- The information must be modeled following a BIM methodology.
- The planning is provided by a `.mlp` file. A specific thesaurus limits the different stages and activities available for the planning. The models are imported using the COLLADA format where the system recognizes layers, geometric groups, and objects.
- Associations between geometries and stages of the planning are performed interactively.
- The building can be visualized and inspected in a three-dimensional viewport.
- The stages of the construction can be selected and enabled. Changes in their state are synchronized bidirectionally with the geometry.
- The navigation controls allow the user to translate, rotate and scale the model.
- The evolution of the construction process can be simulated with a progressive display of the geometries associated with the construction stages.
- The tool must be accessible as a web application from any device, anytime, any place.

Despite the relation established between SLAM and BIM in our framework, we shall clarify that the requirements of KN-SLAM have not been posed taking in mind a specific application or operating environment. Actually, they have been stated with the main goal of researching new techniques for affordable real-time solutions to the 3D reconstruction problem. We are aware that there exist better alternatives depending on the requirements of the application domain. However, we do not aim to compete just in quality and accuracy with the state of the art. Other strategies can also run with monocular cameras (e. g., a combination of MVS and SfM) or even in real-time (e. g., a RGB-D camera provides high-resolution real-time depth images at 30 FPS) but they do not operate in real-time. Although some approaches can even work with different types of lenses in the camera (monocular, stereo and RGB-D), we have decided to stay focused only on low-cost alternatives using only monocular cameras with poor lens specifications. It has been the specific combination of requirements what has led to the choice of monocular visual SLAM.

There are more suitable approaches if we focus only on achieving a high-quality reconstruction especially-suited for BIM. However, we decide to put our efforts on a small niche within the huge field of research on three-dimensional reconstruction. This field is still growing fast due to the increasing industrial demand of these solutions, and nowadays many enterprises offer a wide range of devices and methods to generate accurate and low-cost 3D models. Then, we think that our results can be of use in different application domains like BIM, where spatial information captured with other strategies can be also integrated to enhance the resulting system.

1.2 OBJECTIVES

With respect to the reconstruction of the scene, we develop in this work a monocular SLAM implementation that supports the requirements stated above. In particular, we aim at reaching good levels of accuracy and robustness that can improve today's systems. Our closest neighbor concerning sparse indirect SLAM approaches, as we will see in the literature review in Section 2.3, is the one designed by Mur-Artal, Montiel, and Tardos [88]. Therefore, we will establish our goals by taking this system as a reference and describe our challenges with respect to its performance. Our goals are multiple, ranging from the bootstrapping of the system to the map reconstruction. More concretely, the following goals are pursued:

- Reduce the booting times of the system.
- Increase the number of tracked frames of the trajectory.
- Bootstrap the system as soon as possible.
- Develop a smarter adaptive strategy for visual loop detection in the set of candidate keyframes.
- Reduce the trajectory error in sequences recorded in natural conditions, i. e., those sequences recorded naturally by a hand-held camera.
- Deal with more diversity of scenes including, but not only, structured scenes where the material of the object uses a high-frequency texture.
- Achieve a colored sparse three-dimensional map when possible, i. e., when the input images are not represented in grey scale.
- Lower the computing requirements and the memory footprint of the system.

In 3D-SIMOS, we aim at implementing these requirements with the maximum simplicity and good performance. Current alternatives are integrated into non-affordable systems that are usually difficult to setup and manage.

From our collaboration with the stakeholders in the AEC industry, they have recognized an excessive level of sophistication in BIM to accomplish even the most simple tasks. This is one of the main barriers for the adoption of BIM. From now on, our main goal is to provide a simple solution that demonstrates the benefits of BIM. More specifically, the solution should focus on planning and monitoring the advances in the construction of the building. In addition, 3D-SIMOS must be able to:

- Enrich a CAD model with semantic information.
- Simplify the creation of the BIM model from conventional information sources, i. e., from file formats used by the AEC industry in the last decade.
- Develop a light-weight application, able to operate on different web browsers and platforms (both mobile and desktop computation environment).
- The simulation process must run smoothly in desktop platforms and run at least in mobile environments.
- Render appealing three-dimensional models, visually enhanced by textures linked to the materials of the meshes.
- Support different types of building models, such as heritage, residential, offices, and industrial facilities.

1.3 OUTLINE

This document has been organized into five chapters. Current chapter has been devoted to introducing the motivation and scope of our research. We have described the overall framework in which our contributions can be arranged. It involves the three principal concepts to understand the rest of the document: Simultaneous Localization and Mapping, 3D reconstruction and Building Information Modelling.

We have included the background and state of the art in Chapter 2. This chapter describes the mathematical concepts and models behind the areas introduced in Chapter 1. There exist several approaches to SLAM and 3D reconstruction; they are reviewed in this chapter by including specific alternatives in the literature too. A special section is devoted to explaining the SLAM pipeline implemented by ORB-SLAM, our inspiration and reference implementation for this work. The last section describes advanced visualization and BIM in the context of the AEC industry.

Next chapter is Chapter 3, which contains our research concerning sparse monocular SLAM. The main contributions inserted in the SLAM pipeline are described in Section 3.1. Next, Section 3.2 explains our methodology to evaluate the estimated trajectory mainly in terms of accuracy and robustness. It also explains the characterization that we have explicitly developed for this evaluated sequences from four different datasets. Our experimental results are analyzed in Section 3.3 taking into account the challenging characteristics of each sequence. The conclusions are summarized in the Section 3.4, followed by a review of the compliance with our initial goals.

In Chapter 4, we detail the contributions proposed to foster the adoption of the BIM methodology in the AEC industry, including a thorough description of 3D-SIMOS in Section 4.1. Our proposal for a common framework toward dynamic BIM is included here too. The experimental methodology is based on the experimental evaluation of several aspects by means of interviews to experts and direct observation. They are defined in Section 4.2. Results achieved for the evaluated aspects are analyzed in Section 3.3, and finally, conclusions are highlighted in Section 4.4.

The last chapter is intended to summarize our main achievements and discuss the lessons learned. At the end of the chapter, we sketch the guidelines for future research concerning SLAM and BIM.

2 BACKGROUND

Our research has been developed in a field that combines concepts from Robotics, Computer Vision, and Computer Graphics. Most of the mathematical background originates from Algebraic Geometry, as a natural extension of the Projective Geometry, and the Lie Group theory, which provides the structural link between static aspects (relative to the scene) and kinematic aspects (relative to the motion). The bible to understand the geometrical principles is Multiple View Geometry [53]. The other important reference to understand the second part of our research in regard to applications to structures urban environments is [32]. This book introduces a practical approach to BIM from the perspective of different stakeholders.

Most of the concepts are included in this chapter to allow the reader to understand the rest of the contents of this thesis. However, we make reference to the original publication due to space limitations. We will refer to the concepts and notation introduced in this section when appropriate.

2.1 NOTATION

This section defines the notation that will be used throughout the rest of the document. Variables highlighted in bold represent multidimensional arrays whereas non-bold variables are scalars in the field of the real numbers $\mathbb{R}$. Figure 2.1 illustrates the concepts represented by each variable over an example trajectory of just three keyframes.

The image of the frame received at time k is denoted with $I_k : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{R}$, where Ω is the image domain and $I(\boldsymbol{u})$ is the light intensity perceived by the camera sensor at pixel coordinates $\boldsymbol{u}_k = (u, v)^\top \in \Omega$. Each 3D point $\boldsymbol{p}_k = [x, y, z]^\top \in \mathcal{S}$ of the visible scene surface $\mathcal{S} \subset \mathbb{R}^3$ maps to the image coordinates $\boldsymbol{u}_k$ through the camera projection model (e. g., the pinhole model) $\pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$:

$$\boldsymbol{u}_k := \pi(\boldsymbol{p}_k) \; , \tag{2.1}$$

where k denotes the frame in which the coordinates of $\boldsymbol{p}$ are represented. The projection function π is determined by the intrinsic camera parameters obtained from calibration. In the common pinhole camera projection model these parameters are the focal length $\boldsymbol{f} = [f_u, f_v]^\top$ and the principal point $\boldsymbol{c} = [c_u, c_v]^\top$, so that the function π can be expressed as a linear system

$$\begin{bmatrix} u \\ v \end{bmatrix} := \begin{bmatrix} f_x x_k / z_k + c_u \\ f_y y_k / z_k + c_v \end{bmatrix} \tag{2.2}$$

The 3D point $\boldsymbol{p} \in \mathbb{R}^3$ corresponding to an image coordinate $\boldsymbol{u}$ can be recovered, given the inverse projection function $\pi^{-1} : \mathbb{R}^2 \mapsto \mathbb{R}^3$ and the depth $d_u \in \mathbb{R}$:

$$\boldsymbol{p} = \pi^{-1}(\boldsymbol{u}, d_u) \,, \tag{2.3}$$

where $\mathcal{R} \subset \Omega$ is the domain for which the depth is known.

The camera pose (position and orientation) at the frame k is represented by the rigid-body transformation $\boldsymbol{T}_{k,w} \in SE(3)$, where $SE(3)$ is the Special Euclidean Lie group. This group is the semidirect product $SO(3) \ltimes \mathbb{R}^3$ of the Special Orthogonal group $SO(3)$ and Translations group $\mathbb{R}^3$. It maps a 3D point $\boldsymbol{p}_w$ in world coordinates into the coordinates of frame k:

$$\boldsymbol{p}_k = \boldsymbol{T}_{k,w} \cdot \boldsymbol{p}_w \,, \tag{2.4}$$

where the application of $\boldsymbol{T}_{k,w}$ can be viewed as a composition of a rotation and a translation as

$$\boldsymbol{p}_k = \boldsymbol{R}_{k,w}\boldsymbol{p}_w + \boldsymbol{t}_{k,w} \,. \tag{2.5}$$

The relative transformation between two consecutive frames at times $k-1$ and k is computed as

$$\boldsymbol{T}_{k,k-1} = \boldsymbol{T}_{k,w} \cdot \boldsymbol{T}^{-1}_{k-1,w} \tag{2.6}$$

where $\boldsymbol{T}^{-1}_{k-1,w} = \boldsymbol{T}_{w,k-1}$. A minimal parametrization for $\boldsymbol{T}$ consist of a vector $\boldsymbol{\xi} = (\boldsymbol{\omega}, \boldsymbol{\upsilon})^T \in \mathbb{R}^6, \quad \boldsymbol{\omega}, \boldsymbol{\upsilon} \in \mathbb{R}^3$, which is isomorphic with the Lie algebra $\mathfrak{se}(3)$ representing the tangent space to $SE(3)$ at the identity. Each element of the algebra $\boldsymbol{\xi}$ can be mapped to the corresponding element of the group $SE(3)$ by the exponential map [83]:

$$\boldsymbol{T} := \exp(\boldsymbol{\xi}) \tag{2.7}$$

In a similar way, the inverse operation is locally represented by $\boldsymbol{\xi} = \log(\boldsymbol{T})$, which allows to move from the group to the algebra.

Each map point $\boldsymbol{p} \in \mathcal{S}$ is observed from a subset of keyframes $\mathcal{K}_i \in \mathcal{K}$ where π_k , $k \in \mathcal{K}_i$ projects $\boldsymbol{p}$ inside the image boundaries. For the sake of simplicity we will denote the transformation matrix $\boldsymbol{T}_{k,w}$ as $\boldsymbol{T}_k$ and the map points $\boldsymbol{p}_{i,w}$ as $\boldsymbol{p}_i$, which are always represented in world coordinates w. r. t. the first keyframe of the scene.

2.2 OPTIMIZATION FRAMEWORK

The core of the optimization problem is to compute an estimator $\boldsymbol{X}^*$ based on a probabilistic model. This estimator is computed from a plethora of noisy

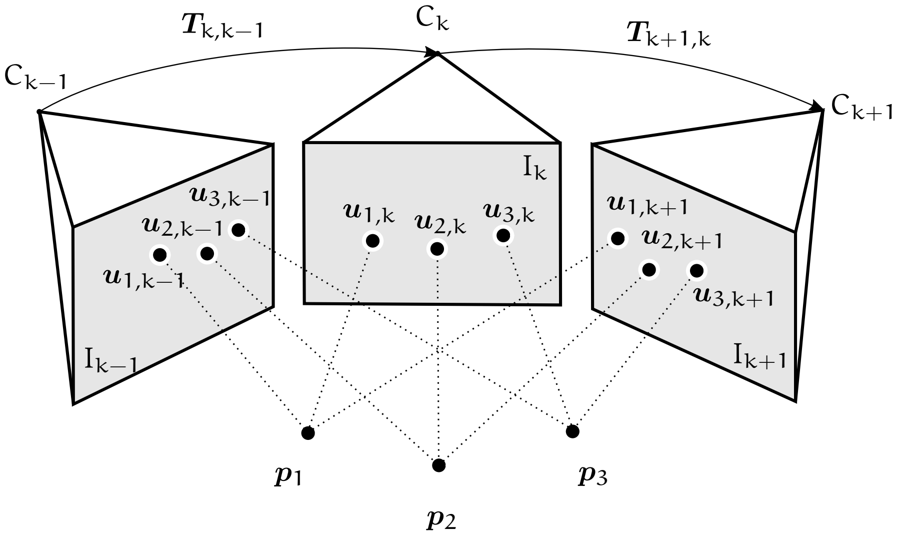


FIGURE 2.1: Notation represented over a classical keyframe-based trajectory. For clarity the trajectory only include three keyframes and three map points but it is easy to generalize the example for N keyframes and M map points.

measurements $\boldsymbol{Y}$ to unveil the hidden model parameters $\boldsymbol{X}$ (i. e., the map points and the camera poses). From an abstract point of view, the Maximum Likelihood Estimation (MLE) maximizes the probability of obtaining the actual measurements $\boldsymbol{Y}$, i. e.,

$$\boldsymbol{X}^* = \arg\max_{X} P(\boldsymbol{Y}|\boldsymbol{X}) .$$

The optimization procedure to compute the estimator depends on the probability distribution of the measurements. This section aims at providing the fundamentals of algorithms for optimizing jointly the points and the poses of the system. A non-linear optimization framework is used over the elements of the Lie algebra of the group of rigid body transformations ($\mathrm{SE}(3)$). This framework can also be extended to other groups like $\mathrm{Sim}(3)$ to optimize the pose graph. Next, the procedure for estimating the Jacobian of the camera motion using the classic interaction matrix in Robotics is explained.

2.2.1 *Least squares and Gauss-Newton optimization*

In estimation problems the goal is to gauge a vector of parameters given a vector of measurements given a likelihood probability distribution f. The most probable solution is the one that maximizes f, which is equivalent to minimize the negative log-likelihood $-\log f$ (commonly known as energy functional). Assuming that f is jointly Gaussian, the measurements and parameters are independent, and the uncertainty in each parameter of every measurement has the same value, the problem can be viewed as a simple sum of squares minimized through a non-linear least squares optimization.

The Gauss-Newton method, an approximation of the Newton method, is commonly used for non-linear least squares minimization. It iteratively updates an initial estimate of the parameter vector $\boldsymbol{p}$ by the rule

$$\boldsymbol{p}^{(i+1)} = \boldsymbol{p}^{(i)} + \boldsymbol{\delta}\,. \tag{2.8}$$

The update vector $\boldsymbol{\delta}$ is solved at each step by the normal equation

$$(\boldsymbol{J}_{\boldsymbol{p}}^{\mathsf{T}} \boldsymbol{\Lambda}_{\boldsymbol{f}} \boldsymbol{J}_{\boldsymbol{p}})\boldsymbol{\delta} = -\boldsymbol{J}_{\boldsymbol{p}}^{\mathsf{T}} \boldsymbol{\Lambda}_{\boldsymbol{f}} \boldsymbol{r} \tag{2.9}$$

where $\boldsymbol{J}_{\boldsymbol{p}} = \frac{\partial \boldsymbol{r}}{\boldsymbol{p}}$ is the Jacobian of the residual error $\boldsymbol{r} = \boldsymbol{f} - \hat{\boldsymbol{f}}(\boldsymbol{p})$ of the estimated parameters $\boldsymbol{p}$ of the model, and $\boldsymbol{\Lambda}_{\boldsymbol{f}} = \sum_{\boldsymbol{f}}^{-1}$ is the *information matrix* or the inverse of the *measurement covariance matrix* $\sum_{\boldsymbol{f}}$ for the likelihood distribution.

Gauss-Newton optimization approaches to the minimum at a quadratic rate of convergence, but gradient descent behaves much better far from the minimum with a higher convergence speed. This behavior is taken into account by the Levenberg-Marquardt (LM) algorithm, in which the normal equation is altered as

$$(\boldsymbol{J}_{\boldsymbol{p}}^{\mathsf{T}} \boldsymbol{\Lambda}_{\boldsymbol{f}} \boldsymbol{J}_{\boldsymbol{p}} + \lambda \boldsymbol{I})\boldsymbol{\delta} = -\boldsymbol{J}_{\boldsymbol{p}}^{\mathsf{T}} \boldsymbol{\Lambda}_{\boldsymbol{f}} \boldsymbol{r} \tag{2.10}$$

where λ is a non-negative damping parameter updated at each iteration i that rotates the update vector $\boldsymbol{\Lambda}$ towards the direction of the steepest descent. Hence, LM behaves as Gauss-Newton if $\lambda \to 0$, whereas it behaves as a gradient descent when $\lambda \to$ inf. LM only updates $\boldsymbol{p}^{(i+1)}$ if the residual error is significantly reduced. In such case λ is decreased as the estimation $\boldsymbol{p}$ is close to the solution; otherwise λ is increased to perform a gradient descent step since the estimation is still far from the minimum.

In Gaussian optimization frameworks, it is convenient to model feature coordinates with the inverse depth $(u, v, \rho)^{\mathsf{T}}$. This feature maps to the Euclidean point $\boldsymbol{x} = \frac{1}{\rho}(u, v, 1)^{\mathsf{T}}$. Inverse depth allows the system to deal with uncertainty over a broad range of depths even for far-away points with little parallax during camera motion. It also enables efficient and accurate representation of depth during bootstrapping, where the equations have a high degree of linearity (see [18]).

2.2.2 *Optimizing Rigid Body Transformations*

Many problems in Robotics and Computer Vision involve manipulation and estimation of the 3D geometry of objects and scenes. Without a coherent and robust mathematical framework for representing and working with 3D transformations, these tasks are onerous and treacherous. Rigid body transformations must be composed, inverted, differentiated and interpolated. Lie groups and their associated machinery address all of these operations, and do so in a principled way, so that once intuition is developed, it can be followed with confidence. In addition, it is feasible to swap between the representation of an element in the algebra and the group. We choose the most convenient representation according to the requirements of the problem we are addressing.

Lie Representation

The optimization of transformation matrices can be simplified if they are parametrized such that singularities are avoided. The group of rotations itself poses issues in angle interpolation since it is only locally Euclidean. However, transformations modelled using a Lie group/algebra avoid this problem and preserve the quadratic convergence rate of the Newton optimization. A transformation matrix $\boldsymbol{T} \in \mathbb{R}^n$ is expressed as an $(n+1) \times (n+1)$ matrix:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix}, \ \boldsymbol{R} \in \mathrm{SO}(n), \ \boldsymbol{t} \in \mathbb{R}^n, \tag{2.11}$$

being $\mathrm{SO}(n)$ the Special Orthogonal Lie group, which is interpreted as the group of rotation matrices if $n = 2$ or $n = 3$. Rigid body transformations given by "rotations" and translations form a smooth manifold too, and therefore a Lie group — the Special Euclidean group $\mathrm{SE}(n)$. Its corresponding Lie algebra $\mathfrak{se}(n)$ - the tangent space of $\mathrm{SE}(n)$ at the identity - provides a minimal representation of a rigid body transformation.

The algebra elements in $\mathbb{R}^3$ are $(\boldsymbol{\omega}, \boldsymbol{v}) \in \mathbb{R}^6$, where $\boldsymbol{\omega} \in \mathbb{R}^3$ is the axis-angle representation of rotations, and $\boldsymbol{v} \in \mathbb{R}^3$ is the rotated translation vector $\boldsymbol{t}$. The elements of the algebra $\mathfrak{se}(3)$ are converted into elements of the group $\mathrm{SE}(3)$ using the application of the exponential map

$$\exp_{\mathrm{SE}(3)}(\boldsymbol{\omega}, \boldsymbol{v}) = \begin{bmatrix} \exp_{\mathrm{SO}(3)}(\boldsymbol{\omega}) & \boldsymbol{V}_v \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.12}$$

where the $\exp_{\mathrm{SO}(3)}$ can be computed using the Rodrigues formula [20] as

$$\exp_{\mathrm{SO}(3)}(\boldsymbol{\omega}) = \boldsymbol{I} + \frac{\sin(\theta)}{\theta}(\boldsymbol{\omega})_\times + \frac{1-\cos(\theta)}{\theta^2}(\boldsymbol{\omega})_\times^2 \tag{2.13}$$

and the translation vector $\boldsymbol{V}_v$ as

$$\boldsymbol{V}_v = \boldsymbol{I} + \frac{1-\cos(\theta)}{\theta^2}(\boldsymbol{\omega})_\times + \frac{\theta - \sin(\theta)}{\theta^3}(\boldsymbol{\omega})_\times^2 \tag{2.14}$$

being $\theta = ||\boldsymbol{\omega}||_2$ and $(\boldsymbol{v})_\times$ the application that maps a vector to its skew-symmetric matrix, which is defined as

$$(\boldsymbol{v})_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \tag{2.15}$$

Finally, the optimization will be performed in the vector space isomorphic to the tangent space $\mathfrak{g} = T_e G$, where the incremental update ξ lives. Then, it is converted back into a manifolds of $SE(3)$ following Equation 2.8.

The Jacobian of a rigid transformation can be expressed using a row decomposition of the rotation matrix $\boldsymbol{R} := [\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3]^T$

$$\frac{\partial \boldsymbol{T}}{\partial \xi} = \left.\frac{\exp_{SE(3)}(\xi)\boldsymbol{T}}{\partial \xi}\right|_{\xi=0} = \begin{bmatrix} -(\boldsymbol{r}_1)_\times & \mathbf{0}_{3\times3} \\ -(\boldsymbol{r}_2)_\times & \mathbf{0}_{3\times3} \\ -(\boldsymbol{r}_3)_\times & \mathbf{0}_{3\times3} \\ -(\boldsymbol{t})_\times & \boldsymbol{I}_{3\times3} \end{bmatrix} \tag{2.16}$$

Jacobian Estimation

The kinematic differential relation between the motion of a point in the scene $\boldsymbol{p} = [x, y, z]^T$ and its projection $\boldsymbol{u} = [u, v]^T$ on the image plane, where $\boldsymbol{u} := \pi(\boldsymbol{p})$ according to the definition of π provided in Section 2.1, can be modelled by the Jacobian of the perspective projection matrix as

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{f_u}{z} & 0 & -\frac{u}{z} \\ 0 & \frac{f_v}{z} & -\frac{v}{z} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \boldsymbol{J}_1(u, v) \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{2.17}$$

Indeed, the velocity of the fixed scene point $\boldsymbol{p}$ is a composition of the rotational and translational movements of the camera, which are defined by its angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ and its linear velocity $\boldsymbol{v} \in \mathbb{R}^3$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = -\boldsymbol{v} - \boldsymbol{\omega} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.18}$$

which can be compactly expressed in matrix form using the skew-symmetric form of the $[x, y, z]^T$ vector as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & -z & y & -1 & 0 & 0 \\ z & 0 & -x & 0 & -1 & 0 \\ -y & x & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix} = \boldsymbol{J}_2(x, y, z) \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix} \tag{2.19}$$

Both matrices can be combined in a single compact Jacobian matrix

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \boldsymbol{J}_1 \boldsymbol{J}_2 \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix} = \boldsymbol{J}_p(u, v, z) \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix} \tag{2.20}$$

They can be finally decomposed into the analytical expression for each component of the Jacobian matrix

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{uv}{f_v} & -(f_u + \frac{u^2}{f_u}) & v\frac{f_u}{f_v} & -\frac{f_u}{z} & 0 & \frac{u}{z} \\ (f_v + \frac{v^2}{f_v}) & -\frac{uv}{f_u} & -u\frac{f_v}{f_u} & 0 & -\frac{f_v}{z} & \frac{v}{z} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix} \tag{2.21}$$

where $\boldsymbol{J}_p(u, v, z)$ is the interaction matrix in robotics (see [35]). It is a good estimate of the Jacobian that only depends on the point image coordinates u, v and its depth z. Some authors tend to simplify the expression by considering both focal lengths equal $\lambda = f_u = f_v$. However, this assumption rarely performs well since well-calibrated lenses have a slightly different focal length in horizontal direction w. r. t. the vertical direction. This split is crucial in order to achieve accurate results.

2.2.3 *Bundle Adjustment*

The accuracy of sparse SLAM relies on the quality of the feature detection and the outcome of the Bundle Adjustment (BA) algorithm that jointly refines the point cloud and the camera poses. Optimizations performed by BA come from SfM where the total error is measured as the sum of the distances between map point observations (keypoints) and their corresponding projections over the keyframes of the trajectory. Three elements intervene in this optimization: the camera poses $\boldsymbol{T_k}$, the map points $\boldsymbol{p}_i$ and the observations $\boldsymbol{u}_{k,i}$. There exist alternatives to the classic BA that consist in fixing the position of some of these objects (i. e., they are optimized in the successive iterations). For example, motion-only BA only optimizes the camera pose since the positions of map points are considered fixed.

Moreover, BA is an example of Non Linear Least Squares (NLLS) optimization that can be solved by a Gauss-Newton algorithm. Due to the strong performance requirements, it is convenient to use a variant formulation of Gauss-Newton such as LM (see Section 2.2.1). This speeds up the convergence to the minimum by interpolating between the Gauss-Newton and the gradient descent algorithms.

Let us define the reprojection error of the observation of a map point $\boldsymbol{p}_i$ in the keyframe k expressed in world coordinates w as

$$\boldsymbol{e}_{i,k} = \boldsymbol{u}_{i,k} - \pi(\boldsymbol{T}_{k,w}\boldsymbol{p}_{i,w}) \ , \tag{2.22}$$

where π is the projection function as described in Section 2.1.

Finally, in the optimization problem the cost function to minimize is formulated as

$$C = \sum_{i,k} \rho_h(\boldsymbol{e}_{i,k}^T \boldsymbol{\Omega}_{i,k}^{-1} \boldsymbol{e}_{i,k}) \tag{2.23}$$

where $\boldsymbol{\Omega}_{i,j}$ is the covariance matrix related to the scale of the level of the pyramid in which the keypoint was previously detected

$$\boldsymbol{\Omega}_{i,k} = \sigma_{i,k}^2 \boldsymbol{I}_{2x2} \tag{2.24}$$

and ρ_h is a robust kernel to reduce the impact of possible outliers. In our case it is defined as the piecewise function corresponding to the Huber norm

$$\rho_h(x) = \begin{cases} \frac{1}{2}x^2, & \text{for } |x| \leqslant h, \\ h(|x| - \frac{1}{2}h), & \text{otherwise.} \end{cases}$$

2.2.4 *Probabilistic modelling of depth hypothesis*

The map of the scene is generated through triangulation of the position of pixels in several keyframes. However, several approaches have been proposed to address the problem of merging several triangulations arising from the different pairs of keyframes. For example, ORB-SLAM [88] optimizes the position of the map point in the multiple BA optimizations performed by the system. However, this approach is not feasible in dense or semi-dense SLAM since the complexity of the BA optimization increases exponentially. Instead, a probabilistic framework is being used for the last five years, based on the original work developed by Vogiatzis and Hernández [129].

The idea is to represent the depth of a pixel with a probabilistic distribution that explicitly models the depth uncertainty. Every subsequent observation of the same map point is used to update the distribution in a Bayesian framework. This framework allows the system to merge several redundant observations of the same 3D point to get a single accurate representation when the variance of the distribution becomes small enough.

A *depth-filter* is a Bayesian distribution that models the depth of a point $\boldsymbol{p}$ projected in the frame I_i through the camera pose $\boldsymbol{C}_i$. Initially, high uncertainty is assigned, with the mean as the average scene depth of the reference keyframe. For every new frame I_j, the system searches for the patch with the highest correlation with the patch in I_i along the epipolar line. This line is computed with the estimated relative transformation $\boldsymbol{T}_{i,j}$ between $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ (see [53]). Then, a new depth hypothesis $d_{i,j}$ can be obtained by conventional triangulation.

The Bayesian distribution for modeling the depth hypothesis can change according to the requirements of the procedure. It is usually a Gaussian distribution or a mixture of a Gaussian and a Uniform distribution (see e. g., [34, 39, 89]). This distribution takes into account the image noise, the parallax and the ambiguity in the matching. A valid measurement is normally distributed around the true depth d_i, while the variance represents the outliers. Usually, depth is represented by the inverse depth $1/\rho$ to handle large scenes. This probabilistic framework is illustrated in Figure 2.2, where the inverse depth $1/\rho$ of the map point $\boldsymbol{p}$ is modeled as a Gaussian distribution.

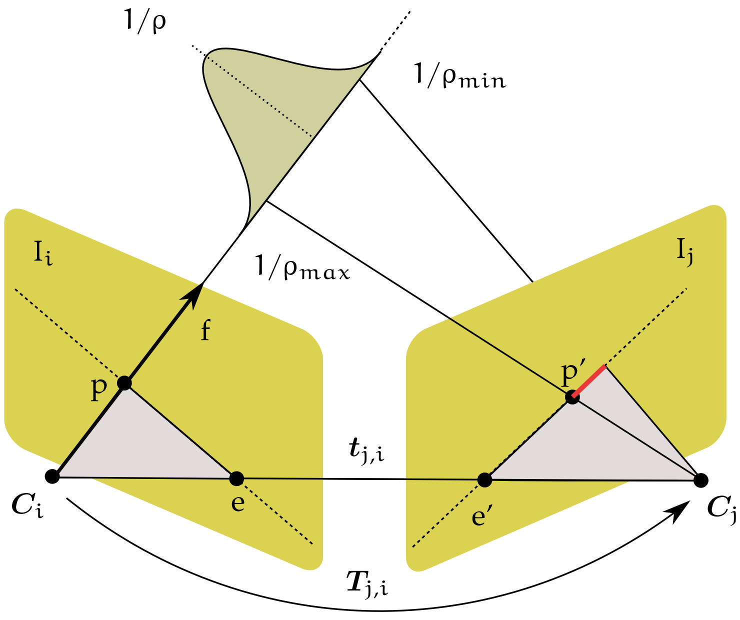


FIGURE 2.2: A probabilistic framework based on Gaussian distributions to integrate and fuse several noisy measurements into a single depth ρ. Red segment represents the limited range of search for epipolar matches of point $\boldsymbol{p}$ through the camera $\boldsymbol{C}_j$. The mean of the Gaussian distribution provides an estimator for the inverse depth $1/\rho$ of $\boldsymbol{p}$ within a range $(1/\rho_{max}, 1/\rho_{min})$.

A set of N hypothesis are fused to yield the inverse depth distribution $N(\rho_p, \sigma^2_{\rho_p})$ for the pixel p with

$$\sigma^2_{\rho_p} = \left(\sum_N \frac{1}{\sigma^2_{\rho_j}}\right)^{-1}, \qquad \rho_p = \sigma^2_{\rho_p} \sum_N \frac{1}{\sigma^2_{\rho_j}} \rho_j \,, \tag{2.25}$$

which corresponds to the merging step for a Gaussian distribution.

When the baseline is large, outlier measurements, generated by similar pixels or occlusions, must be removed before fusing their corresponding hypothesis. In [89], the authors compare the modulo and orientation of the image gradient, besides the usual intensity of the pixel, in the epipolar search. In addition, a hypothesis must pass a compatibility test with the previous hypotheses before fusing them. Besides, at least N are required to consider the estimated depth as reliable.

This method has been shown to be very efficient when the search in the epipolar line is limited to a small range around the current depth estimate (around twice the standard deviation of the depth estimate, see the red segment of Figure 2.2 and [39]). The method is fast since the uncertainty of the distribution is reduced with little camera displacements. In addition, the number of outliers is reduced w. r. t. a conventional triangulation from two views since every filter models multiple erroneous measurements. This also allows the depth to converge even in environments with low contrast or high-frequency textures.

2.3 SIMULTANEOUS LOCALIZATION AND MAPPING

Simultaneous Localization and Mapping (SLAM) makes reference to the set of techniques, methods and algorithms needed to recover a 6 Degrees of Freedom (DoF) camera pose at the same time that the map of the scene is being reconstructed. Nowadays, there exists important emerging technologies and industries demanding robust SLAM algorithms, including autonomous driving, Unmanned Aerial Vehicle (UAV) guidance in Global Positioning System (GPS)-denied environments, augmented reality applications, among others. Some of these products cannot afford expensive camera devices especially manufactured or they are already equipped with a commodity camera that can not be replaced.

One of the main singularities of SLAM is that algorithms must work in real-time. This is the main difference with respect to similar concept such as SfM or MVS that use almost the same mathematical background without performance constraints. Uncertainty is another big difference with those tasks since images have bigger resolutions, less noise and they can be discarded or repeated during 3D reconstruction. Therefore, algorithms in SLAM must be able to deal with lots of redundant and noisy data from images captured in real-time in order to extract valuable information to track the camera position in every possible frame.

2.3.1 *Visual Odometry*

Visual Odometry (VO) is the process of estimating the pose of an agent using only the passive input information captured by a single or multiple cameras. The estimation of the pose is commonly known as *egomotion*. The term VO was coined by Nistér, Naroditsky, and Bergen [93] based on the similarities with wheel odometry. Nowadays it is possible to track the camera pose in real-time with a modern multi-core CPU. In the usual approach, the system incrementally updates the pose through the observation of the changes that motion induces on the images. Furthermore, consecutive frames with sufficient overlap must be considered.

Monocular VO refers to a particular variant of the problem using a camera with a single lens. One of the key concerns of this variant that limits its broad appliance is the inherent scale drift. The Euclidean coordinates of the camera and the map points are always scaled by an unknown scale factor. The distance between the first two camera poses is usually set to one and used as a reference for the rest of the system operation. The relative scale and pose of a new frame w. r. t. the first two frames are determined with the knowledge of the 3D map and the relation with a previously selected frame called keyframe.

The current estimated pose accumulates the errors introduced in previous frames. This produces a drift in the estimated trajectory, which should be kept as small as possible. This goal can be accomplished through local optimization over the last camera poses (sliding window bundle adjustment or windowed bundle adjustment). Obviously, the drift can also be reduced when VO is combined with other sensors, such as GPS, laser or Inertial Measurement Unit (IMU).

For more information, please refer to the tutorial created by Scaramuzza and Fraundorfer [111] and the references therein.

Differences with Simultaneous Localization and Mapping

Both SLAM and VO share a common theoretical background since they estimate the current camera pose taking into account the motion observed in pixels of consecutive frames. Although they are frequently exchanged in the literature, there exist some differences between them that are ignored [111]. Next paragraphs highlight these differences, which we stick to for the rest of the document.

On one hand, VO refers to the problem of estimating the pose of a camera using only information from the image. It could range from one to multiple cameras, or even fuse inertial information from an IMU [77]. It is only concerned with the local consistency of the trajectory, without keeping track of all the previous history of the camera poses.

On the other hand, SLAM extends VO functions by taking into account information about the trajectory of the camera. Its goal is to obtain a global, consistent estimate of the trajectory. This implies the detection of loop closures, the optimization of the positions of the map points and the camera poses or the relocalization of the camera when the tracking is lost. These other tasks allow the system to reduce the drift in the map and the trajectory. Therefore, the map build by SLAM is more accurate and realistic than the local map maintained in VO. This makes SLAM a better choice when robust long-life operation is required.

The sliding window optimization in VO can be considered equivalent to the local BA in SLAM. However, they pursue different goals. The local map is another tool to maintain the global consistency, whereas VO only uses it to obtain a more accurate estimate of the local trajectory. In the same way, VO can be viewed as a starting building block of SLAM. The addition of loop detection and global optimization procedures is what characterized a system as SLAM.

In summary, SLAM extends the scope of VO by including not only the tracking between consecutive frames in a local environment but also the management of the relations between the set of keyframes. They are related to the map points by a graph-like structure called the connectivity graph (see Section 2.3.3). This graph connects keyframes (nodes) sharing a portion of the scene with an edge. Thanks to this data structure the poses and the map points can be jointly optimized when a loop closure is detected, something that VO does not allow.

A visual SLAM system is potentially more accurate since it adds more constraints on the path. However, the system is also more complex, computationally demanding, and it could be even less robust since outliers in the loop can affect to the map's consistency during the optimization. The final choice between VO and SLAM is taken based on the required tradeoff between performance, simplicity, accuracy, and applications. The global consistency of the trajectory is always desirable but VO trades off this consistency for real-time performance.

2.3.2 *Rolling-shutter cameras*

The proliferation of electronic devices equipped with high-quality affordable cameras (e. g., smartphones or webcams) has increased the number of potential SLAM applications. The vast majority of these cameras use CMOS sensors, which are more affordable and easier to manufacture. However, they also install an electronic rolling shutter that, in contrast to global shutter, generates geometric and photometric distortions on images. Most of the common photometric distortions include automatic exposure changes, non-linear response functions (gamma correction and white-balancing), lens attenuation (vignetting) or de-bayering artifacts. Such effects are more remarkable in the presence of fast motion or strong rotation of the camera.

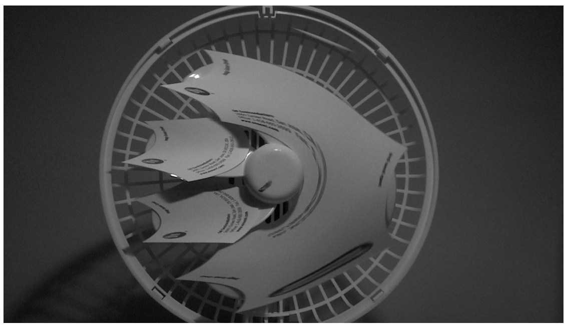
Furthermore, rolling shutter cameras manifest a higher Signal-to-Noise Ratio (SNR) in pixel's intensities. This impedes direct matching between consecutive frames, even though the baseline is short. In these cameras, pixels are written sequentially in different scanlines, according to the readout time (see Figure 2.3). This creates a slight intra-camera movement that should be modeled, besides the conventional inter-camera displacement.

Although continuous-time trajectory models have been proposed to quantify and compensate geometric distortions [97], they are not suitable for real-time operation. These models have been recently extended to the epipolar geometry framework to estimate the relative pose between two frames [24]. In the uniform rolling shutter camera model (i. e., uniform rotation and translation with constant velocity), the classical 8-point linear algorithm is augmented to a 44-point algorithm. Other SLAM algorithms must be adapted to deal with the distortions introduced by rolling shutter. This involves modeling each row of the frame as a different camera pose, which makes Bundle Adjustment (BA) intractable. In [56], the authors propose an alternative BA to optimize only a small subset of the poses and recover the remaining ones using interpolation.

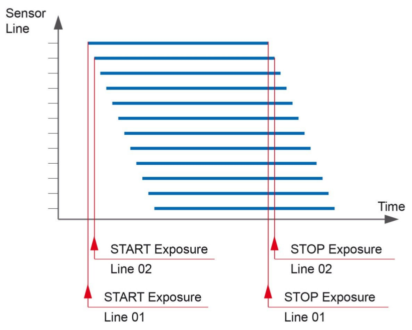
Overall, rolling shutter is not worse than global shutter, but it poses different problems to track the camera using just pixel information. As always it depends on the requirements of the application. Rolling shutter is recommended when low cost, high resolution, and high frame rate are required, whereas global shutter is more appropriate for capturing fast moving objects and Computer Vision applications.

2.3.3 *The connectivity graph*

In order to achieve large-scale and long-life operation, SLAM systems relate their keyframes in a graph, usually known as the *covisibility graph*. This graph is an abstract representation of the relations between the keyframes of the trajectory. Each keyframe is linked to other keyframe according to the portion of the scene shared by both keyframes, i. e., the portion of the map visible



(A) Rolling shutter effect as spatial distortion of fan blades



(B) Rolling shutter readout times influence sensor scanlines

(C) Creative WebCam HD, a low-cost camera with an electronic rolling shutter

(D) MT9V034 camera for Arduino equipped with global shutter

FIGURE 2.3: Rolling shutter introduces artifacts in images that should be taken into account to achieve a robust and coherent SLAM implementation. Images have been extracted from https://www.embedded-vision.com/platinum-members/embedded-vision-alliance/embedded-vision-training/documents/pages/vr.

from both keyframes. This portion can be determined by projecting the map points onto the frustum of each keyframe.

One of the main benefits of this representation is that it allows to manage spatial relations between keyframes used in other algorithms. For example, tracking and mapping tasks are performed in a constrained local environment around the current keyframe, which is independent of the total map size. A mechanism of local sliding window is implemented, clustering the keyframes not only by time but also by space. In this way only a few bundle adjustment iterations are required to accurately estimate the map point position.

Connected keyframes in the connectivity graph share a significant amount of map points. The central question is how many links are required for each keyframe to achieve a simple graph without losing spatial relations. This connectivity is employed by several tasks, such as relocalization, local Bundle Adjustment, loop detection, and keyframe culling. Hence, the topology of the graph influences the accuracy of the trajectory, the robustness of the tracking and the performance of the system. It is our goal to determine a satisfactory connection policy that simplifies the graph while the accuracy of the trajectory is still preserved.

The optimization of a pose graph is a classical non-convex optimization problem, which can be solved with the algorithms described in Section 2.2. If the orientation of cameras were known, the optimization would be a linear least-squares problem, whose solution can be computed efficiently and reliably. Thus, rotations are the main cause of the complexity of the optimization. Techniques for 3D rotation estimation were surveyed in [16]. In this paper, the authors review the contributions to optimization algorithms from three different scientific communities: robotics, computer vision, and control theory. Their evaluation shows that the use of Chordal Relaxation [85] or Riemannian Gradient Descent [125] to bootstrap iterative pose graph solvers entails significant boost in convergence speed and robustness. The procedure works as follows: firstly, the rotations are estimated using one of the reviewed techniques, and then this estimate is used as initial guess for a conventional Gauss-Newton method. Results show that the initial estimations achieved by Chordal Relaxation reduce the computational cost of convergence by half.

Loop detection

Loop detection and closing are still two unsolved problems of the SLAM pipeline. The most common approach consists in: 1) accumulate keyframes; 2) detect loop closures; 3) optimize the pose of the keyframes; and 4) reconstruct the 3D mesh of the scene off-line in a post-processing step (see e. g., [89]). An online 3D reconstruction framework was proposed recently in [65]. In this publication the authors propose to split the scene into several submaps. The poses in each submap are updated only when required in a local framework. Relative pose constraints are progressively accumulated between submaps. Finally, the global optimization is managed in a lightweight and scalable way.

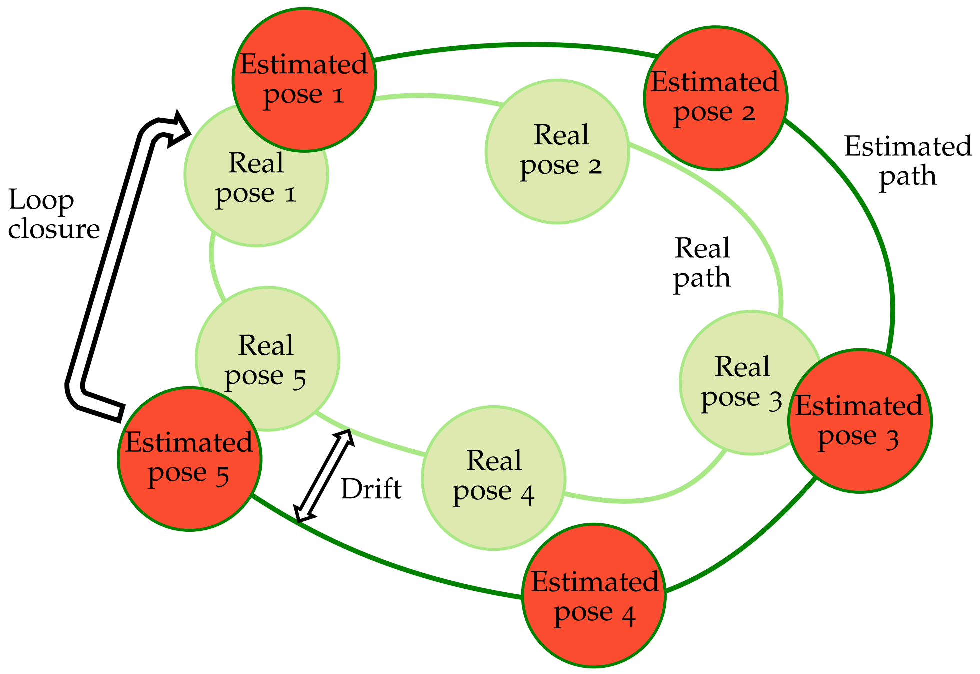


FIGURE 2.4: The connectivity graph links spatially related keyframes. It is mainly employed to optimize camera poses after detecting a loop closure.

2.3.4 *Types of SLAM*

Since the beginning of this century, SLAM methods can be classified into two different categories: sparse / feature-based methods or dense / direct methods. The former ones (also known as indirect methods) abstract the image intensities into a set of sparse geometric features; the latter ones operate directly over the intensity of image pixels. Recently, Engel, Koltun, and Cremers [33] introduced a slightly different classification, distinguishing between direct/indirect and sparse/dense methods. However, in practice, most approaches can be labelled as a combination of indirect and sparse, or direct and dense. Dense indirect approaches are less common since, by definition, features only abstract a subset of pixels of the image. Next sections discuss the most outstanding approaches of each type of SLAM.

Dense SLAM

Dense methods, also known as featureless or direct SLAM, still require additional computing power (i. e., a GPU device) to perform camera tracking by directly aligning two images. Dense SLAM minimizes the photometric error using a Gauss-Newton algorithm in which the Jacobian of the residuals must be computed at each iteration. The photometric error is the distance between two pixels measured in their intensity space, which is affected by the way the sensor perceives the real color of the objects. Actually, this is the main bottleneck of this approach.

The main advantage of this approach is that the system tracks both image regions with a high-intensity gradient (e. g., with edges and corners) and smooth intensity variations (e. g., white walls). Additionally, a dense or semi-dense point cloud is generated by the system without a posterior reconstruction step in the pipeline. However, these methods assume a surface reflectance function of the camera that is usually violated by commodity cameras, generating some artifacts in real-world sequences (see the datasets of [33]). They are extremely sensitive to rolling-shutter, auto-gain, and auto-exposure, which are present nowadays in most of the commodity digital cameras. In addition, points are inserted into the map without correcting the drift. In the best case, a map optimization is performed only in the pose graph, discarding all sensor measurements and map points. All these issues constraint the algorithms to especially suited cameras with global shutter, a wide FOV and a high SNR per pixel (more than 30 dB).

Normalized Information Distance (NID)-SLAM [100] is a direct monocular SLAM system based on the NID metric. This metric replaces the conventional photometric error used for direct pixel comparisons during image alignment. The system is robust to appearance variation caused by lighting, weather and structural changes in the scene. The authors evaluates the system in a synthetic indoor scene generated from real-world data collected from a vehicle-mounted camera. The system runs on a GPU using Open Graphics Library (OpenGL). The accuracy depends on the illuminating conditions of the sequence and it is comparable to Large Scale Detection (LSD)-SLAM and ORB-SLAM. However, the robustness to appearance changes is increased. Robustness is measured as the percentage of frames of the sequence where the camera pose is estimated.

Dense SLAM methods do not pay much attention to the pose graph as a tool for keyframe management; it is only used to detect loop closures. For instance, Whelan et al. [133] perform pose optimization in a limited local region using a combination of dense frame-to-model camera tracking and windowed surfel-based fusion. In addition, frequent loop closure optimizations through non-rigid surface deformations maintain the pose close to the mode of the map distribution. Finally, a global loop closure distributes accumulated drift along the trajectory. In this case RGB-D camera enables accurate camera tracking in consecutive frames taking advantage of the dense nature of the scene map. However, without considering the whole structure of the map in the loop optimizations, the accumulated drift along the trajectory is most times unacceptable (see the results in [133]).

Similar CPU-based alternatives have been developed by imposing some constraint to the environment. For instance, the authors of [60] have developed a KDP-SLAM, a system that reconstructs large indoor environments in real-time using a hand-held RGB-D sensor (see Figure 2.5). They use a fast dense method to estimate odometry. Their method also extracts planes from fused depth maps created from small baseline images. Finally they optimize the poses and planes in a global factor graph using Incremental Smoothing and Mapping (iSAM). They are able to estimate the six DoF pose even without sufficient planes to fully constrain the transformation. Drift is compensated by explicitly modeling plane landmarks in the fully probabilistic global optimization framework. Their system runs at 30 FPS in the CPU but only works with structured scenes.

Direct methods heavily rely on the good lighting conditions of the image since they determine camera poses by means of direct image alignment. Most

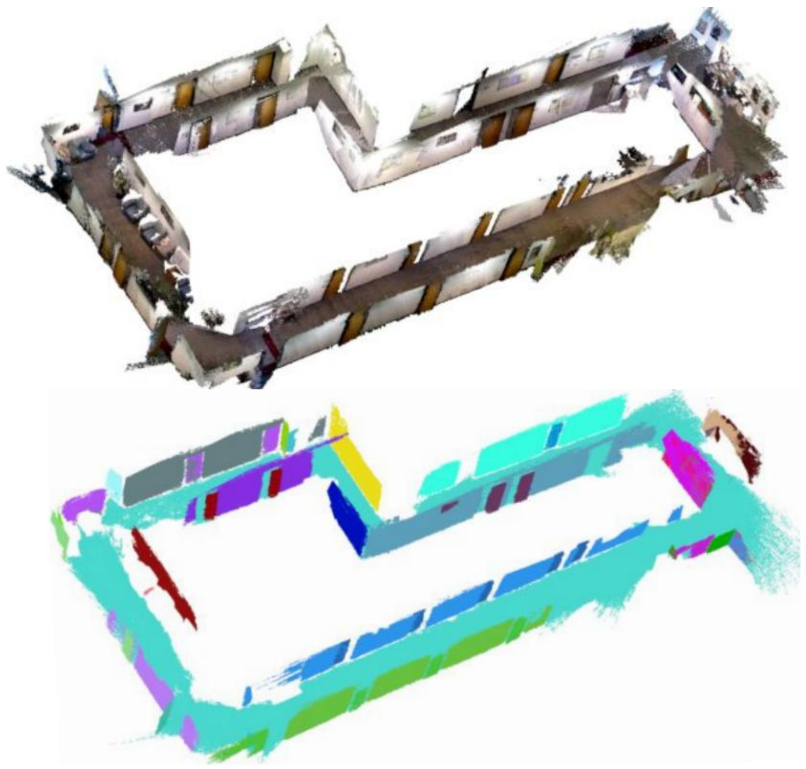


FIGURE 2.5: Example of dense planar reconstruction in large indoor environments with loops (extracted from [60]).

systems incorporate the brightness constancy assumption in the formulation of their alignment cost function [34]. Therefore, they cannot cope with significant illumination changes that are likely to occur for loop closures. In [99], the authors have explored a real-time alternative to this formulation. After a thorough evaluation of the accuracy and robustness in odometry and loop closure, the authors conclude that, for real-world images, a Census-based method [134] is the best. This method is used for depth estimation in stereo images due to its invariance against all intensity transformations that preserve the intensity ordering. However, it may provide less precise information than other metrics about the alignment between two images.

A common methodological framework to evaluate different SLAM systems has been introduced in [91]. This publication presents SLAMBench, a software framework to enable a simplified, quantitative, comparable and reproducible experimental research in SLAM. They illustrate the utility of their framework to investigate the trade-offs in performance, accuracy and energy consumption of a dense RGB-D SLAM system. The constituent algorithmic elements of the SLAM pipeline can be assessed individually in a wide range of computing architectures thanks to the proposed kernels.

Sparse SLAM

Sparse (or feature-based) SLAM references the set of approaches that do not consider direct pixel intensities in their operation. Instead, they preprocess these pixels to generate an indirect representation that abstracts the whole image. This step involves some class of feature detection and description. A feature is a geometric primitive such as points, edges or pieces of curves. A second step involves the establishment of correspondences between the feature set of each frame. Sparse methods do not recover a dense point cloud but a sparse set of 3D map points linked to the 2D keypoints in which they project. Thus, homogeneous textured regions are not tractable (e. g., on-board cameras on a highway) with this approach. On the other hand, they are robust to images with noise, blur, and low-contrast.

There are also other sparse SLAM approaches that do not abstract the image (see [33]). Instead, they work directly over the pixel's intensities. These methods are labeled as direct sparse SLAM, in contrast to the conventional indirect sparse approach of [88]. The latter is the most common approach so we will refer to them when we use the term sparse. We have included these direct sparse approaches in the semi-dense section.

The first real-time VO was introduced in [87], followed by the seminal work of Parallel Tracking and Mapping (PTAM) developed by Klein and Murray [68]. In this work, the authors introduce the first implementation of a real-time monocular SLAM system capable of running on a smartphone. The system employs Scale-Invariant Feature Transform (SIFT) features to detect and describe the keypoints of the frames. Optimization is performed through a parallel windowed bundle adjustment as the pose graph expands. Although the approach is limited to small-scale operations, it includes some of the key elements involved in current SLAM approaches: keyframe selection, feature matching, point triangulation, camera pose prediction with a constant-velocity model, and relocalization after tracking failure. However, the approach misses important features for using the system in real-world scenarios: loop closures, handling of occlusions, and automatic bootstrapping (human intervention was required).

A few years later, Mur-Artal, Montiel, and Tardos [88] introduce the nowadays reference implementation of sparse SLAM called ORB-SLAM. The pipeline of the system is entirely driven by ORB features [108], which are inexpensive to compute and match. Some other remarkable features include frame relocalization, loop detection, and closure, and pose graph optimization in real-time with multiple threads (more details about the complete pipeline have been presented in Section 2.4). Due to its outstanding contributions and open-sourced license, we have selected ORB-SLAM as the reference for developing our contributions.

A homography-based strategy has recently been introduced in [80]. In this work, the authors address the especially challenging situation of fast camera motions with a strong rotation. They introduce RKSLAM, a monocular indirect sparse SLAM system that proposes a multi-homography schema for feature tracking between consecutive frames. This schema allows the system to support different search ranges to correct the perspective distortion of the frame. The system estimates a global homography of the whole image and several 3D plane homographies in the map. If the tracking quality is poor, the system computes multiple local homographies of small image regions around the tracked features. Furthermore, observed 3D points are triangulated immediately. Motion prior constraints between consecutive frames are imposed in camera pose optimization using simulated data. Despite the increased robustness of this approach, the system only works in man-made structured environments that can be described by homographies.

Semi-dense SLAM

Some of the limitations of direct methods against feature-based methods were pointed out by Torr and Zisserman [124]. The authors highlight the superior accuracy of feature-based methods, but they also admit their inability to achieve a dense map of the scene [62]. In addition, dense methods require a huge amount of computational resources, which makes its implementation in a CPU unfeasible.

In order to overcome such limitations more recent efforts are focused on hybrid approaches that combine the best of both worlds in different stages of the pipeline. In the last five years, semi-dense (or semi-direct) approaches have gained a lot of attention [33, 34, 39, 89]. They tackled the problem in real-time using just a representative subset of the pixels of the image. The most common choice is the space of high-intensity gradient pixels, as in LSD-SLAM [34]. This system was one of the first semi-dense proposals that optimizes the photometric error and a geometric prior on pixels with a high-gradient value. In [33], the geometric error has been replaced by a photometric model of the camera. In addition, the system samples pixels evenly throughout the images to reduce the number of constraints and speed up the Gauss-Newton optimizations.

Other interesting semi-dense implementation is Semi-direct Visual Odometry (SVO) [40]. Their authors propose a faster monocular VO algorithm that eliminates the feature extraction and matching steps of the pipeline. The algorithm extends the Bayesian mapping method proposed by Vogiatzis and Hernández [129] to explicitly model map outliers in triangulations. In this method, the depth estimate of a feature is modeled with a Gaussian distribution in a Bayesian framework. A small variance of the distribution indicates that the depth estimate can be transformed to a map point. A map with very few outliers and more reliable points can be achieved in just a few iterations with this model. This framework is further explained in Section 2.2.4.

In SVO, the system detects a collection of Features from Accelerated Segment Test (FAST) features in keyframes, and then it runs a completely direct approach. The algorithm is able to deal with scenes of little, repetitive, and high-frequency texture, thanks to the combination of a high framerate motion estimation and the probabilistic mapping. Due to its simplicity it can run at 300 FPS on a laptop or at 55 FPS in an embedded platform. The speedup is due to the nonexistence of feature extraction and matching steps, which are the most expensive steps of any indirect sparse SLAM. The system has been evaluated with sequences captured by a Micro Aerial Vehicle (MAV). However, results downgrade when the camera is not pointing downwards.

The authors of ORB-SLAM extended the mapping capabilities of their approach in [89] by including a semi-dense mapping stage that operates directly over the set of keyframes. The bundle adjustment optimization of ORB-SLAM allows the system to obtain accurate triangulations even in the presence of wide baselines and noisy outliers. The result is a clean map with very few outliers based on the accurate trajectory of ORB-SLAM.

Other approaches combine a classical Extended Kalman Filter (EKF) with a semi-direct approach, i. e., they work with a subset of pixels. A monocular visual-inertial odometry algorithm of such kind was proposed by Tanskanen et al. [122]. Their method operates only with sparse, small patches within the input images. The key of this approach is the embedding of the optimization algorithm that minimizes the photometric error directly into the EKF algorithm. Thus, inertial data and visual data are considered at the same time for computing the camera pose estimation. Their implementation also reduces the impact of a large number of measurements when minimizing the photometric residual error. The algorithm is able to deal with several pixel patches on the same line. The system bootstraps without manual intervention from a good starting pair of keyframes. This system runs in real-time on the CPU. The results are more stable than LSD-SLAM and SVO, with more robustness to fast camera rotations.

Semi-dense methods have been also applied to RGB-D cameras. For example, BundleFusion [23] combines SIFT features with dense geometric and photometric information to find correspondences. A coarse global alignment is achieved by features; such alignment is refined by including dense photometric and geometric consistency. Both constraints are imposed in an energy minimization framework to obtain the pose of the camera, which is parallelized in the GPU. In addition, a two-level hierarchy is created to estimate the global camera pose in real-time. In the lowest level, consecutive frames are grouped into a local chunk where the camera pose is estimated. Then, chunks are correlated and globally optimized. In this way, the system can consider the complete history of frames and scale to large scenes. Furthermore, the system supports recovery from tracking failures and real-time updates of the 3D model through dynamic integration of frames in a volumetric representation of the scene. The reconstruction quality of the results is comparable to previous offline approaches.

Publications always compare the results of the proposed algorithm with those obtained by other dense and semi-dense approaches [14]. However, there is not a holistic and complete evaluation methodology to make these results reproducible. This problem has been tackled in [137], where the authors notice the nonexistence of individual kernels and parameter spaces for testing individual parts of a SLAM pipeline. In order to solve it, they extend the existing SLAMBench framework to include metrics of accuracy, energy consumption, and processing framerate. The new benchmark is evaluated by comparing the pipelines of KinectFusion [64] and LSD-SLAM [34]. Their methodology is interesting from the viewpoint of system-level design and integration.

Deep Learning and SLAM

Deep Learning methods in general, and more specifically Convolutional Neural Network (CNN) are currently considered the swiss knife to solve most computer-vision related problems. This popularity surged in 2012 when these methods were applied to address the ImageNet classification challenge. The work of Krizhevsky, Sutskever, and Hinton [73] was the pioneer in training a deep network (60 million parameters and 500.000 neurons in five convolutional layers) in a GPU to solve a complex classification problem with an outstanding accuracy. Inspired by this work, a plethora of neural network architectures have arisen in the last three years (see e. g., [45, 55, 104, 105]). On outstanding contribution was recently presented in [54]. This paper presents residual networks by reformulating the layers as learning residual functions, instead of unreferenced functions. These networks are easier to optimize and more accurate, due to the increment in depth.

CNN are part of a bigger research line named *deep learning*, which also comprises other types of networks such as *deep belief networks* and *recurrent neural networks*. These networks have been applied to multiple domains such as speech recognition, natural language processing, automatic machine translation, bioinformatics and recommendation systems. As expected, CNNs have also been recently applied to SLAM. In fact, the presence of big datasets that annotate the pixels of RGB images with their corresponding depth (e. g., the TUM RGB-D dataset) leverage the power of a CNN to predict the depth of an unknown image similar to those used for training.

The main advantage of these methods is that they can estimate the depth per every pixel, i. e., they retrieve a depth image of the current frame like a ToF camera does. However, the main inconvenience is the required computational power to perform the inference step when new frames arrives. Similarly as happens with dense SLAM, they require a powerful GPU, which is not affordable or technically feasible in more conventional devices.

There exist a previous attempt to recover the depth from a single image using a Total Variation (TV) formulation [102]. In this paper, the authors propose a natural extension of the variational approach to estimate the depth map in larger indoor environments with extensive texture-less surfaces. They show that previous approaches degrade in the presence of large affine texture-less areas like walls, floors, ceilings, etc. These regions produce noise and erroneous initial seeds for the optimization procedure. Their proposal introduces a new non-local higher-order regularization term to enforce piecewise affine constraints between far pixels. They assume that the depth at the edges of regions is often well estimated. However, the inner pixels of these regions are deeply problematic. Unfortunately, their system requires 500 ms per frame in a GPU architecture to obtain a median depth error below 2 cm.

The first attempt to apply CNN for depth inference from a monocular image was introduced in [79]. This work does not rely on any cue, e. g., stereo correspondences, motion, etc. Depth estimations are reformulated as a continuous Camera Response Function (CRF) learning problem due to the continuity of the depth values. More concretely, the authors propose to learn the unary and pairwise potentials of continuous CRF within a deep learning framework. The method works without geometric priors nor any extra information. Depth predictions are performed efficiently by solving the Maximum A Posteriori probability (MAP) inference since a closed-form solutions exists. Results show lower errors with comparable or even better performance

Sometimes the depth images used for training purposes are replaced by stereo pairs from which the depth can be inferred. For example, in [46] the authors train the network using binocular stereo data from the KITTI benchmark (in the track of the year 2015). This data is supposed to be easier to achieve for a wide range of environments than the ground truth depth. The authors exploit epipolar geometry to generate disparity maps. Their loss function enforces consistency between the disparities produced relative to both the left and right. The resulting network can estimate the depth of a single frame at 28 FPS in a Titan X GPU.

There are other approaches also based on the monocular content of the KITTI dataset. In [136] the authors propose a completely unsupervised method that requires only monocular video sequences for training. Their loss function is based on warping nearby views to the target using the computed depth and pose. They use two networks (single-view depth network and multi-view pose network) coupled with the loss function during training. However, they can be applied independently at inference time. The training data consists of unlabeled image sequences capturing scene appearance from different viewpoints, where the poses of the images are not provided. However, the achieved ATE greater than ORB-SLAM.

Other authors have tried to reformulated the SfM problem as a learning problem. For example, the authors of DeMoN [127] propose a SLAM system that uses a CNN to compute both the depth and the camera pose from successive, unconstrained image pairs. The system is composed of multiple networks. The core part is an iterative network able to improve its own predictions. Besides depth and motion, the network also infers surface normals, optical flow and confidence of the matching. The loss function is scale invariant and it penalizes relative depth errors between neighboring pixels based on spatial relative differences. The main benefit of this network is that it utilizes the motion parallax, which generalizes better to non-training scenes and enhanced the accuracy of the estimated poses. However, the CNN does not adapt well to other non-training cameras with different intrinsic parameters. Furthermore, it does not work with more than two images.

The previously mentioned approaches are focused on VO, leaving aside the 3D reconstruction of the scene (i. e., the generation of a 3D textured mesh). The reconstruction step has recently been considered in [123], e. g.,. Their approach combines the multiple depth maps predicted by a CNN to produce an accurate and dense monocular reconstruction. The authors propose to refine the predicted depth maps with small-baseline stereo matching obtained from direct monocular SLAM. They privilege depth prediction in regions where monocular SLAM tends to fail, e. g., along low-textured regions. In addition, they use the predicted depth for estimating the absolute scale of the scene, which is one of their main contributions. Finally, they also propose a framework to fuse pixel-wise semantic labels with dense SLAM to yield semantically coherent scene reconstruction from a single view. The resulting system deals with challenging situations such as textureless regions and pure rotational motions while still retaining the robustness and accuracy of direct SLAM. Despite these benefits, the system do not close the loops in the trajectory, which causes problems in lifelong operations in large environments. Other limitation of this approach is that depth estimations are not geometrically refined.

2.4 ORB-SLAM

As it has been mentioned in Figure 2.3.4, there are various indirect sparse approaches to monocular SLAM (see [25, 68, 80, 88]). The I3A Robotics, Perception and Real Time Group of the University of Zaragoza has published several contributions since 2010. Recently, they have implemented ORB-SLAM [88], a monocular SLAM system that is among the most widely cited works in the literature. In addition, the release of the source code of ORB-SLAM under an Open Source license guarantees a quick and widespread adoption by the community. To the best of our knowledge, this is the best reference to consider for any monocular SLAM system. We have used this implementation as the starting point for our contributions in the SLAM component of the framework described in Figure 1.1 at Section 1.1. Despite its sparse nature, many relevant publications (see [33, 40, 123]) have compared its results with ORB-SLAM, especially in terms of accuracy.

The accuracy obtained by ORB-SLAM relies on the accuracy of Bundle Adjustment (BA) as the underlying optimization method (see Section 2.2.3). Although the goodness of this method was previously well-known, this approach was not considered suitable for real-time applications. However, nowadays it is known [88] that accurate results can be achieved at non prohibitive cost if the following conditions are met:

- A map point is represented by a set of corresponding observations of scene features in a set of keyframes.
- The redundancy of keyframes are maintained low with a tradeoff with the length of the sequence.
- Keyframes should introduce significant parallax between them to create a well-spread set of points with a wide range of depths.
- The keyframes should be arranged in a network where the number of loop closures is maximized. These loops are closed in real-time with a fast global optimization procedure.
- The system should provide a good guess estimate of the initial keyframe pose to the nonlinear optimization procedure.
- The optimization should be performed with the local map of the set of keyframes around the current camera pose to make the process scalable in long sequences.

Based on a set of previous SLAM-related works in which the same authors were involved, ORB-SLAM proposes a set of significant contributions:

- The entire pipeline is modeled around the same type of feature (ORB) to simplify the system.
- The bootstrapping procedure handles both planar and non-planar scenes based on a model selection from a homography or a fundamental matrix.

- The smart use of the information collected in the covisibility graph allows ORB-SLAM to constrain the optimizations independently of the global map size. Thus, the system can still operate in real-time in large environments.

- Loops are closed in real-time thanks to a particular pose graph called the *Essential Graph*, which is built from the covisibility graph with a spanning tree, loop closure links, and strong edges (i. e., edges between keyframes sharing more than 100 observations).

- The camera can be relocalized in real-time to recover from tracking failures.

- The keyframes and map points spawn generously to improve tracking robustness under hazard conditions (e. g., rotations, fast movements, etc.). Later, they are culled restrictively to ensure lifelong operation.

The justification of choice of the ORB features is multiple: 1) all the steps of the pipeline use them; 2) they are extracted and matched in real-time, i. e., less than 33 ms per image is required; 3) they are rotational and viewpoint invariant, and 4) they support wide baseline tracking. The ORB descriptor has a size of 32 B or 256 bit.

The system includes three threads running in parallel in the background: tracking, local mapping and loop closing. The main thread is in charge of loading the frames of the sequences. There is an additional optional thread for the GUI visualization. The main data structures managed by the system are the map points and the keyframes. Each map point includes

- the 3D position in the world coordinate system,

- the viewing direction (i. e., the mean unit vector of all its viewing directions),

- the ORB descriptor (i. e., the median of the descriptors of its observations),

- the maximum and minimum distance to the camera at which the point can be observed.

A keyframe is just a special type of frame that stores

- the camera pose represented in the world coordinate system,

- the keypoints and their descriptors extracted from the selected frame,

- the map points associated to the keypoints,

- the covisible keyframes and the number of shared observations.

One of the keys to run a real-time BA is to constrain the number of points and keyframes involved in the optimization. These subsets are created using covibility information between keyframes. This information is useful for other tasks, such as relocalization, loop detection or map spawning. Covisibility is represented by the *covisibility graph* [120], an undirected weighted graph where the keyframes act as nodes. Two keyframes are connected b an edge, which is weighted by the number of shared map points. Only keyframes with a sufficient number of observations (at least 15) are connected.

The rest of this section is intended to describe the fundamentals behind the main stages of the pipeline of ORB-SLAM. We will put the accent in the most significant aspects for our research.

2.4.1 *Bootstrapping*

The bootstrapping procedure is one of the main contributions of ORB-SLAM. The goal of this procedure is to select a good pair of frames from which we can estimate the relative pose and extract the initial map. A good pair of frames must have significant parallax. Ideally, the algorithm must support both planar and general scenes and must be entirely automatic, i. e., it should not require human intervention. This pair was manually selected in previous approaches like [68]. This is a good strategy when the goal is to test the next stages in the pipeline. However, this pair cannot be determined manually if we want to achieve real-time operation. On other side, a poor choice of the pair can lead the system to build a corrupted map. Consequently, we shall look for a good tradeoff between a quick bootstrap and a good pair of keyframes.

The well-known disjunctive between homographies and fundamental matrices requires to evaluate the suitability of both models for the first pair of keyframes. Both models are evaluated in parallel by ORB-SLAM and, then, a heuristic is used to choose the best model. The relative pose is estimated with different methods according to the selected model. In this way, low-parallax pairs and the well-known twofold planar ambiguity [82] are avoided by the system.

The complete procedure includes the following steps:

1. The ORB features $\boldsymbol{x}_c$ are extracted from the current frame F_x. If there is no reference frame F_r, the current frame is labeled as such.

2. The correspondences between the current frame and the reference frame are matched to get $\boldsymbol{x}_c \leftrightarrow \boldsymbol{x}_r$. If there are not enough matches, reset the reference frame. The procedure will start again at the Step 1 when the next frame arrives. Figure 2.6 illustrates an attempt of detecting these matches between a pair of frames.

3. An homography $\boldsymbol{H}_{cr}$ and a fundamental matrix $\boldsymbol{F}_{cr}$ are computed in parallel as

$$\begin{aligned} \boldsymbol{x}_c &= \boldsymbol{H}_{cr}\boldsymbol{x}_r \\ \boldsymbol{x}_c^\top \boldsymbol{F}_{cr}\boldsymbol{x}_r &= 0 \end{aligned} \tag{2.26}$$

   using the normalized Direct Linear Transform (DLT) and the eight-point algorithm in a RANSAC scheme (see [53] and the references therein). The system uses the same number of iterations for both models. Furthermore, the homography is estimated with four points per iteration while

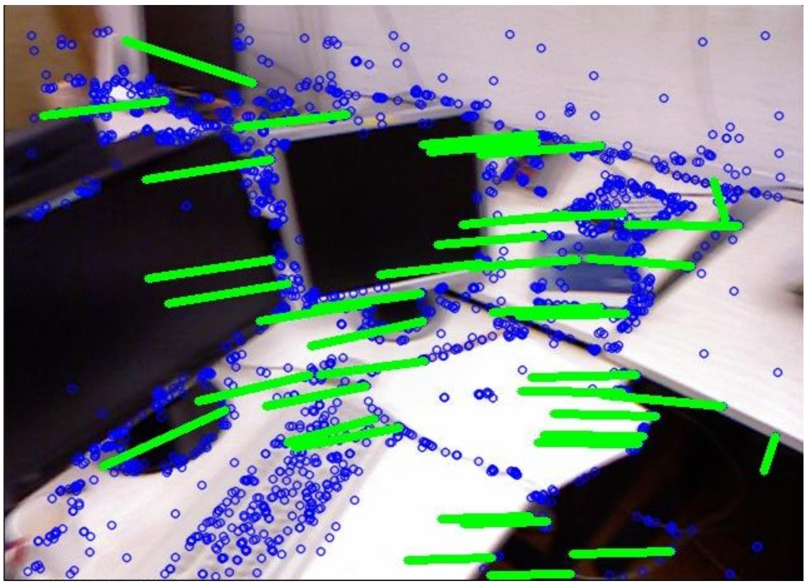


FIGURE 2.6: Example of the automatic bootstrapping of a sequence in the TUM RGB-D dataset. The system tries to detect a pair of keyframes with enough correspondences and good baseline. Green lines represent matched features between both keyframes. Each line starts at the pixel of a feature detected in a previous frame (not represented here) and ends at the pixel of the same feature in the current frame.

linear algorithms for the fundamental matrix requires eight points. A score S_M is computed for each model M (H for the homography and F for the fundamental matrix) taking into account the matches $\boldsymbol{x}_c^i \leftrightarrow \boldsymbol{x}_r^i$ and the symmetric transfer errors [53] d_{cr}^2 and d_{rc}^2.

$$S_M = \sum_i \left(\rho_M(d_{cr}^2(\boldsymbol{x}_c^i, \boldsymbol{x}_r^i, M)) + \rho_M(d_{rc}^2(\boldsymbol{x}_c^i, \boldsymbol{x}_r^i, M))\right)$$

$$\rho_M(d^2) = \begin{cases} T_H - d^2 & d^2 < T_M \\ 0 & d^2 \geqslant T_M \end{cases} \tag{2.27}$$

where T_M is the outlier rejection threshold based on the χ^2 test at 95 %. The system keeps the respective models with the highest scores. If no model is found, the procedure restarts from Step 1.

4. In case of planar or low-parallax scenes the homography should be selected, even though a fundamental matrix can also be found. However, the problem in such cases is not well constrained [53], yielding incorrect results. On the contrary, a fundamental matrix is a model that explains nonplanar scenes with enough parallax. However, it is possible to get a homography explaining a subset of the matches that lie on a plane or that are far away. The authors of ORB-SLAM found that the ratio

$$R_H = \frac{S_H}{S_H + S_F} \tag{2.28}$$

can be used to select the homography when $R_H > 0.45$ or the fundamental matrix otherwise.

5. The relative pose is extracted from the estimated model. Eight motion hypothesis are retrieved for the homography by the method of Faugeras and Lustman [38]. Usual chirality tests fail to detect low-parallax cases since the points easily move to front or back of the cameras. Instead, the points are triangulated for each hypothesis. The best model must have most points in front of cameras, with a low reprojection error and with enough parallax. In addition, the authors of [88] have imposed the following thresholds to accept the model or the procedure will restart from Step 1:
   - The best model must have 33 % more inliers (i. e., map points) than the next model.
   - The parallax of the best model must be higher than 1 degree. The parallax of a model is defined by the parallax of its inliers sorted by the cosine of the angle. The system selects the highest parallax of inliers or the parallax of the 50-th inlier if there are more than 50 inliers.
   - The number of inliers must be greater than $\min(0.9N, 50)$, where N is the original number of inliers.
6. Four motion hypothesis are computed from the essential matrix $\boldsymbol{E}$ with the Singular Value Decomposition (SVD) method described in [53]. In turn, the essential matrix is extracted from the fundamental matrix $\boldsymbol{F}$ using the calibration matrix $\boldsymbol{K}$ as $\boldsymbol{E}_{rc} = \boldsymbol{K}^\top \boldsymbol{F}_{rc} \boldsymbol{K}$. The best model is selected as for the homography.
7. A full BA refines both the positions of the map points and the poses of the first pair of keyframes.

2.4.2 *Tracking*

Tracking is the main stage of every pipeline in SLAM and VO. The goal of this thread is to estimate the rigid body transformation of the current frame and decide whether a new keyframe should be created. The algorithms used by ORB-SLAM are extracted mostly from the work of PTAM. All the optimizations performed to improve the camera pose are motion-only BA, i. e., the map points are fixed while the 6 DoF transformation is iteratively refined.

When a new frame arrives, the system extracts the FAST features at eight-scale levels (scale factor of 1.2). One feature can be extracted in real-time per each 250 - 350 pixels. Usually, most features lie in well-textured regions of the image. Each level is divided in a grid to guarantee a homogeneous distribution of the features. The system tries to detect at least 5 features per cell. The detector threshold is lowered if not enough features are found. Similarly, the number of features per cell changes if some cells are empty (high-frequency textures or low-contrast images). Finally, the ORB descriptors are computed.

Most poses can be estimated based on a simple motion model that assumes constant linear and angular velocity. This approach allows ORB-SLAM to figure the next camera pose for almost 90 % of the frames where the previous tracking was successful. However, there are particular sequences where this hypothesis is invalid such as in the presence of fast movements or strong rotations. In such cases, ORB-SLAM falls back to the reference keyframe to calculate the camera pose. If this second approach also fails the tracking is considered lost.

The initial rough estimation of the pose enables to deliver the power of guided matching between keypoints and map points (or more briefly named as 2D-3D matching). The system aims at finding correspondences between the keypoints of the features and the projection of the map points with the estimated pose. The size of the search window is adjusted according to the number of inliers achieved in previous searches. For instance, if the first search returns only 20 inliers of less, the size is increased from 15 to 30. If the second search also returns less than 20 inliers, the search is performed in the reference keyframe. These associations are used to refine the prior estimation with a $\mathrm{Sim}(3)$-guided optimization.

The number of map points involved in the 2D-3D matching is constrained in large maps. This local map only contains the map points viewed from the last frame augmented with the points observed from some neighbors. The initial set of keyframes includes the keyframes where the map points in the last frame were previously observed. Then, the 10-best covisible keyframes, the children and the parent of each included keyframe expand the original set. In guided matching, each map point in the local map is searched in the current frame by performing the following steps:

1. The map point is projected with the estimated pose.
2. The map point is discarded if
   - it lays out of the camera bounds;
   - the angle between the ray from the point to the camera center and the mean viewing direction of the point is lower than 60°;
   - the distance d between the map point to the camera center is out of the scale invariance region $[d_{min}, d_{max}]$
3. The radius of the search window is proportional to the scale factor of the level of the pyramid in which the feature was detected.
4. The descriptor of the map point is compared against the descriptors of the unmatched features within the search window. Descriptors are compared at the same level of the pyramid. The map point is linked to the closest keypoint.

The result of these steps is a set of 2D-3D correspondences, as shown in Figure 2.7. These matches are the key for the rest of procedures in ORB-SLAM since they allow the system to extend the set of points and to optimize again. This strategy is repeated multiple times in some steps like in relocalization.

The tracking thread is also responsible for deciding whether the current frame should be selected as keyframe or not. However, keyframe culling is

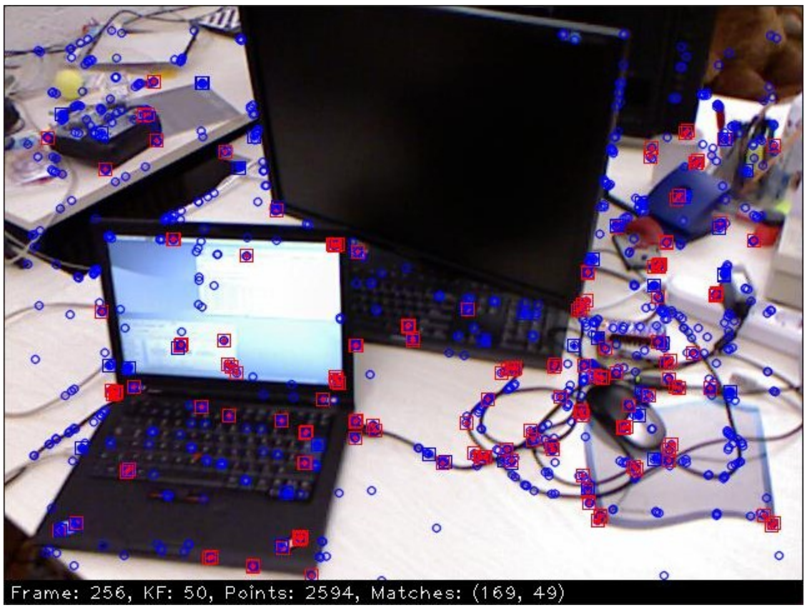


FIGURE 2.7: An example of the resulting matches between map points and ORB features for tracking the current frame. These matches are generated by the guided search in a sequence of the TUM RGB-D dataset. Blue circles represents detected ORB features. Squares represent 2D-3D matches. Red squares correspond to new map points (i. e., with fewer than three observations) whereas blue squares correspond to well-established map points (i. e., with more than two observations). Note that red squares always match detected features as result of the guided search.

performed by the mapping thread. More keyframes make the tracking robust to challenging movements, so the system tries to insert as many as possible. Posterior keyframe culling will remove redundant keyframes. A new keyframe is inserted only if the following conditions are met:

1. The last relocalization was completed at least one second ago.
2. The last keyframe was inserted at least one second ago, or the local mapping is idle.
3. The number of map points in the current frame is between 50 and 90 % of the points tracked by the reference keyframe.

Condition 1 ensures a good relocalization whereas Condition 2 warrants a brisk pace in keyframe insertion. If the first part of Condition 2 is fulfilled, then BA will stop. Condition 3 imposes a minimum visual change and a good tracking between consecutive keyframes.

2.4.3 *Mapping*

In this thread, the system performs some tasks required when a new keyframe arrives. These tasks are executed in a separate thread. Even though new keyframes are created in the tracking thread, the insertion in the covisibility graph and the map are handled by this thread. One of the most important responsibilities is the culling of redundant keyframes when the camera revisits a part of the scene. Moreover, this thread generates new map points from the epipolar correspondences found between the new keyframe and their covisible keyframes.

The thread is waiting for new keyframes in a queue. When a new keyframe is detected, a new node is appended to the covisibility graph. The connectivity of such node is computed based on the observations shared with other keyframes. If two keyframes share at least 15 observations, an edge is created between them. The keyframe with most points in common becomes the parent of the new keyframe. Furthermore, the bag of words representation of the keyframe is computed to add it to the database.

The minimum number of observations discards distant keyframes that share only few observations while maintains a dense graph to manage spatial relations between keyframes. However, the value should depend on the number of points of the map and the aspect ratio of the frames since different sequences have special requirements.

Once a new keyframe is selected, map points created in the latest three keyframes are checked to maintain a low number of outliers. Each point must fulfill the following two conditions:

1. An observation of the point must be found in more than 25 % of the frames in which it can be projected.

2. The point must be observed in more than two keyframes if two or more keyframes have passed since it was created.

The last condition is also checked when a keyframe is culled from the graph or a local BA is executed. The point is removed from the map if it does not pass these tests.

Once the connectivity graph is updated, the creation of new map points is performed by triangulating unmatched keypoints in connected keyframes. The matching is guided by the bag of words representation of each keyframe filtered by the epipolar constraint. This increases the robustness of the tracking since more points allow the system to find more matches. Furthermore, previous outliers are replaced with more refined triangulations (see e. g., Figure 2.8 to see how the map density is increased). A new map point is only accepted if

- The ratio between baseline and the median depth of the keyframe is higher than 0.01.

- The parallax between the rays connecting the point and the camera centers is between 65° and 90°.

- The point is in front of both cameras.

- The reprojection error passes the χ^2 test at 95 %.

- The distance to camera centers is not zero and the ratio between distances is inside an interval of 1.5 times the ratio between the scale factors of each feature.

New points can be projected and matched in the covisible keyframes if there are not new keyframes waiting in the queue. Duplicated matchings are

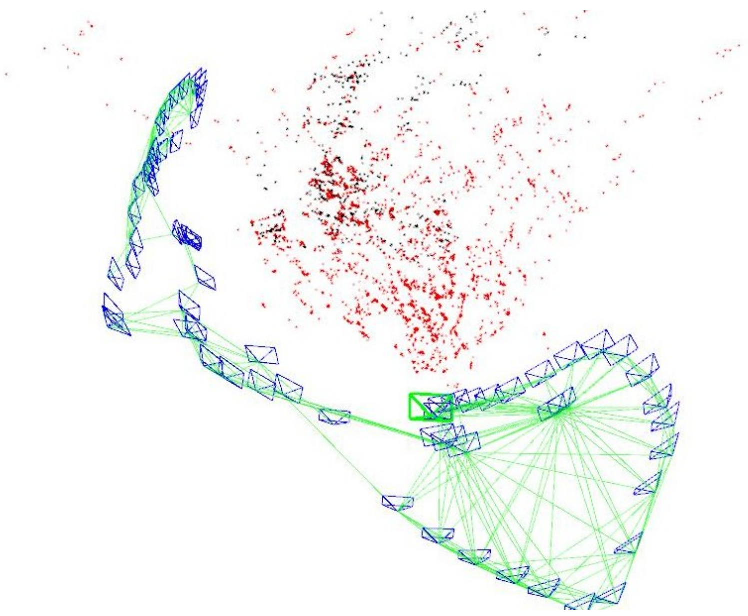


Figure 2.8: Map points for the scene of a sequence in the TUM RGB-D dataset. New points created in the last keyframe are drawn in read. Black points represent those points that were previously created and have been excluded from the last optimization.

conveniently detected and fused. The descriptor, mean viewing direction and d_{min}, d_{max} of each map point are recomputed at the end. Finally, the connections of the current keyframe in the connectivity graph are updated again taking into account the augmented set of map points.

If the mapping thread were also idle (i. e., no stop has been requested), a local BA would be performed to optimize the new keyframe, all its covisible keyframes and their map points. Other keyframes sharing some map point are included in the optimization as a fixed pose. As a consequence, some observations marked as outliers in the process will be discarded. It can be seen in Figure 2.8 that most of the points are included in the optimization in convergent scenes.

The culling of the local redundant keyframes is the last step. It is a mandatory step to guarantee lifelong operation and maintain a compact reconstruction. The number of keyframes only grows as the visual contents of the scene increase. Furthermore, it bounds the complexity of the BA, which grows with the number of keyframes. The policy is to discard the keyframes where 90 % of the points are seen at least by other three keyframes. Points must be viewed in the same or finer scale to preserve those keyframes where points are accurately measured.

2.4.4 *Loop closure*

A key component of the loop closing thread is a place recognition module based on the bag of words metaphor. A visual word is a discretization of a feature descriptor subspace. The complete collection of words constitutes the vocabulary. This vocabulary is created off-line from a large set of general images so it can be reused among different environments.

This place recognition module summarizes the contents of a new keyframe with its most representative words. Therefore, previously seen keyframes can be quickly matched to the current keyframe. Moreover, an inverted index is maintained to speed up the search of the most similar keyframes. For each word, the index stores the keyframe in which it appears. Besides the loop detection, this module also helps to relocalize the camera when the tracking is lost. The module also intervened to speed up the match between two sets of features. The traditional brute force search can be constrained to the features in the same node at a level of the vocabulary tree (e. g., the second out of six).

When a new keyframe is detected, the system looks up the most similar keyframes in the database. Firstly, the system computes the lowest similarity d_{min} score among the current keyframe and its neighbors in the covisibility graph. Then, the keyframes in the database with a score higher than s_{min} are selected.

This set of keyframes is composed by the covisible keyframes of the closest keyframe (i. e., the keyframe with the highest similarity score in the database) and the keyframes with a score higher than the 75 % of the best score. Directly connected keyframes are removed from the resulting set. A loop is recognized as a candidate only if the system detects three consecutive loop candidates whose covisibility groups are consistent. We say that a group is consistent with a previous group if they share at least a keyframe. There may be several loop candidates if several places have a similar visual appearance.

Next step validates the loop candidate by matching its features with the current keyframe. In this step, the features detected by the place recognition module are used to constrain the brute force matching, as stated above. The result is the set of points corresponding to the matching features. If fewer than 20 matches are found, the candidate is discarded.

A loop closure requires to compute the $\mathrm{Sim}(3)$ transformation between the current keyframe and the loop candidate. This transformation represents the accumulated error in the loop, which will be distributed along the trajectory. A solver is created to calculate the $\mathrm{Sim}(3)$ transformation of each candidate using the method of Horn [58]. Then, the system alternatively performs 5 RANSAC iterations for each candidate. If the solver provides a transformation with enough inliers, the matches are increased by a $\mathrm{Sim}(3)$-guided search. An additional optimization is performed with 7 DoF and, if enough inliers are found, the transformation is finally accepted. However, one more test is executed before accepting the loop. A new guided search is performed using the $\mathrm{Sim}(3)$ projection of the map points augmented with the points of the neighbors of the loop candidate. The loop is accepted if at least 40 correspondences are found by the new search.

Once the loop has been detected, the system proceeds to correct the loop. Firstly, the connections of the current keyframe are updated by fusing duplicated map points. Then, the current pose is corrected with the $\mathrm{Sim}(3)$ transformation. The same correction is propagated to the covisible keyframes and the map points they observe so that they align with the other side of the loop. The map points seen by the loop keyframe are projected into the current keyframe and its neighbors using the corrected poses. Matches are searched in a narrow window around the projection following the algorithm explained

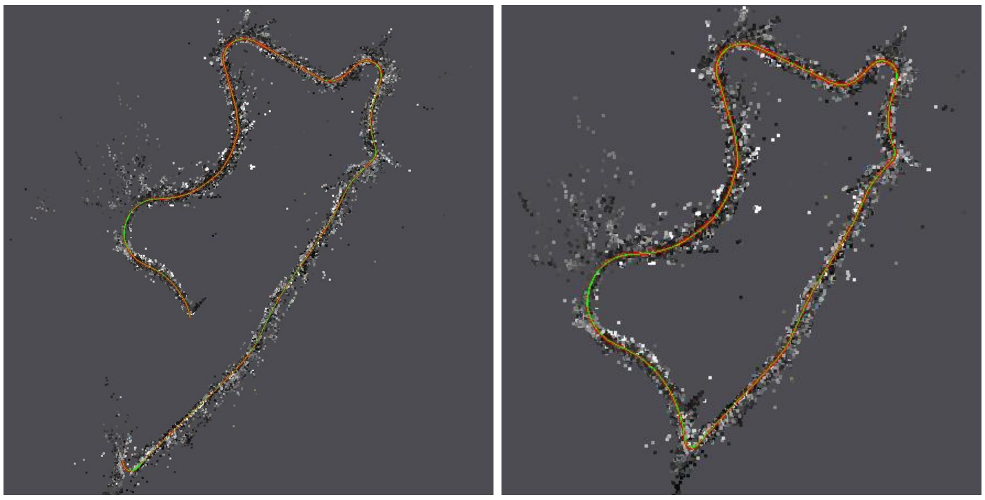


Figure 2.9: A comparison between the trajectory of the same sequence before and after performing the loop closure with KN-SLAM. Note that the scale drift accumulated along the optical axis is corrected, which reduces the error in one order of magnitude.

in Section 2.4.2. As it happens with the points to estimate the transformation, the existing 2D-3D matches are fused, i. e., they are replaced with the new associations. The connections of all keyframes involved in the fusion are updated. Finally, the system inserts new edges to attach both sides of the loop and starts the optimization of the Essential Graph.

Loop closures are an important tool to reduce the accumulated error of the trajectory. They allow the system to distribute the error along the poses of the graph, as it is shown by Figure 2.9. However, the use of the connectivity graph is discouraged to perform the optimization in real-time. Thus, the *Essential Graph* is a subgraph with the same nodes but fewer edges. The goal is that the graph still preserves a strong network of poses and points to obtain accurate results while it still reduces the complexity of the optimizations. Actually, in our results, we have confirmed that fewer connections can increase the accuracy in some sequences.

In a first step, the thread builds a spanning tree starting from the first keyframe. This spanning tree provides the connected subgraph with the minimal number of edges. The keyframe with the highest number of shared observations is the parent of a new keyframe. The Essential Graph is built from the spanning tree, to whom the system adds the edges with high covibility (at least $\theta_{min} = 100$) and the loop closure edges (i. e., edges connecting the start and the end of a loop). The authors of ORB-SLAM show that the choice of θ_{min} has no significant effect on the accuracy of the final trajectory. However, it directly impacts the time required to optimize the graph since it is proportional to the number of edges. For instance, the same error is achieved with $\theta_{min} = 15$ than with $\theta_{min} = 200$, whereas the number of edges for the latter is the 15 % of the former.

The optimization of the Essential Graph is performed over the space of $Sim(3)$ transformations to fix the scale drift [119]. The system approximates a full BA with the classical LM scheme applied over all the points and keyframes, with the loop keyframe marked as fixed. The edges of the graph are used to constrain the optimization using the following error function for each edge

$$e_{ij} = \log_{\mathrm{Sim}(3)}(\boldsymbol{S}_{i,j}\boldsymbol{S}_{j,w}\boldsymbol{S}_{i,w}^{-1}), \tag{2.29}$$

where $\boldsymbol{S}_{ij}$ is the relative $\mathrm{Sim}(3)$ transformation between the i-th and j-th keyframes.

The transformation is estimated from the $\mathrm{SE}(3)$ original poses before the loop closure with a scale of 1. The method of [58] is instead used for loop closing edges. The $\log_{\mathrm{Sim}(3)}$ [118] is the inverse of the exponential map in the Lie Algebra $\mathfrak{sim}(3)$. It transforms the matrix into an element of the algebra in the tangent space. This space has 7 DoF so each element in the tangent space is represented by $\mathbb{R}^7$, where the Jacobian of the cost function can be estimated at each iteration. The optimization of the poses is finally performed through minimizing the following cost function

$$C = \sum_{i,j} \boldsymbol{e}_{ij}^{T} \boldsymbol{\Lambda}_{ij} \boldsymbol{e}_{ij} \tag{2.30}$$

where $\boldsymbol{\Lambda}_{ij}$ is the information matrix of the edge (see Section 2.2.1), which is set to the identity. Once the minimization is completed, all $\mathrm{SE}(3)$ poses are updated from the $\mathrm{Sim}(3)$ transformation as

$$\mathrm{Sim}(3) = \begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} \Rightarrow \mathrm{SE}(3) = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t}/s \\ 0 & 1 \end{bmatrix} \tag{2.31}$$

Each map point is adjusted with the same correction applied to one of its keyframes. Firstly, the point is transformed into the coordinate system of the keyframe with the pose previous to the optimization. Then, the point is transformed back to the world coordinate system with the optimized pose. In the end, the entire set of map points is fixed.

2.4.5 *Relocalization*

In case that the tracking is lost, the only way to estimate a valid camera pose is to use a relocalization procedure. In ORB-SLAM, the authors reuse their own previous work based on the bag of words metaphor described in Section 2.4.4. In this procedure, the current frame is converted into a bag of words and searched in the keyframe database. The features of each candidate keyframe are compared with those of the current frame. The search is guided by the nodes of the vocabulary tree as explained in Section 2.4.4. If the candidate has enough matches, a $\mathrm{Sim}(3)$ transformation between both frames is estimated. The estimation procedure alternatively performs 5 RANSAC iterations for each candidate until a valid solution is found (i. e., at least 10 inliers are found by the Perspective-n-Point (PnP) algorithm [76]). Each pose is then optimized using a motion-only BA on $\mathrm{SE}(3)$. Again, if more than 10 inliers but fewer than 50 are found, the system searches more matches. In this case, it executes

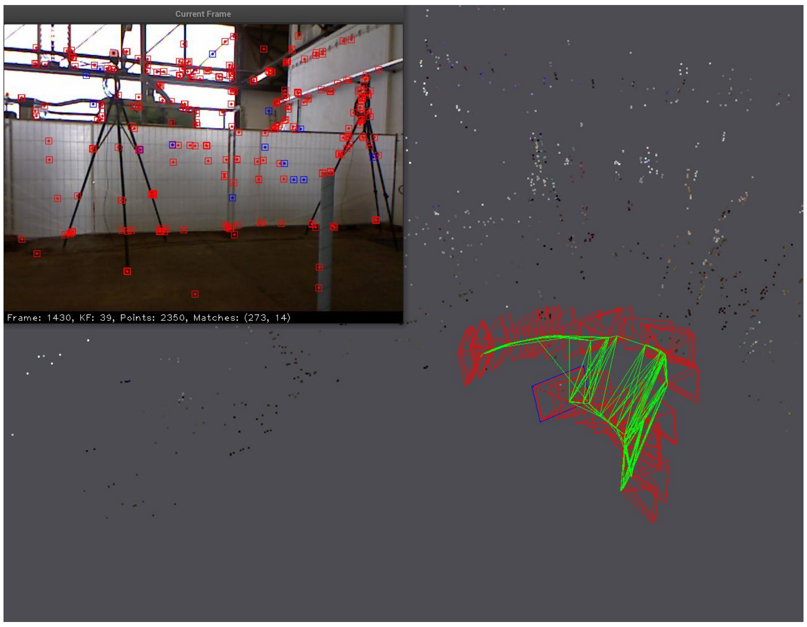


FIGURE 2.10: An example of a forced relocalization in a sequence of the TUM RGB-D dataset. The camera is occluded in the middle of the sequence and the tracking of the camera is lost. The current frame is compared with the previous keyframes until the camera revisits the beginning of the sequence and a robust match is found. Then, the frame is aligned with the first keyframe and the tracking continues. The connectivity graph is shown in the lower right side of the figure. Note that the trajectory is composed of two branches with a slight displacement and the same rotation. The second branch is generated after the relocalization.

a guided search in a wider window including the map points viewed by the candidate keyframe. Finally, the camera pose is optimized again with the same cost function and, if it has at least 50 inliers, the tracking procedure continues.

If the number of inliers were high enough (30) but still below the minimum threshold (50), the system would perform a new guided search in a narrower window with the new points. If there were more than 50 inliers, a final optimization is executed to minimize the error. This workaround avoids situations where the first optimization of the camera pose could include many outliers. This multi-step optimization scheme helps to achieve more accurate results.

An example of execution of the relocalization procedure is shown in Figure 2.10. Here the tracking is lost in the middle of the sequence and the system do not relocalize the camera until it comes back to the beginning of the sequence. Then, the tracking is recovered when the current frame matches the first keyframe.

2.5 K-NEAREST NEIGHBORS

The K-Nearest Neighbors (KNN) algorithm is applied to find the k closest instances given a reference within a feature space [72]. It is among the simplest methods in machine learning, where it represents a type of instance-based learning methods. It infers the features of the new instances based on previously well-known cases, so no training procedures are required. In this case, instances define an underlying model which can be queried at inference time. Usually, graph-like data structures or indexes are maintained to speed up queries. This is also called lazy learning.

The algorithm always relies on a metric of distance between the instances in the feature space. For example, in the three-dimensional Euclidean space where hand-held cameras live, the Euclidean distance is enough to determine the similarity between the points that represent the closest neighbors. In addition, a weigh can be linked to each neighbor, so that the closer neighbors contribute more to the result of the inference (see [30]).

Another important aspect of this algorithm is the value of k that determines the number of instances linked to a new one. High values imply that the inferred solution will be more general, inheriting more traits of the entire dataset, whereas low values of k give a solution focused on the characteristics of specific instances. Therefore, the final choice depends on the level of generalization required in the results and may change with the problem.

In our research, we are especially interested in the three-dimensional case. Paredes and Chávez [98] propose a KNN graph for retrieving the objects closer to a query in a database, a problem commonly known as proximity searching. Their solution is based on creating the database index using the KNN graph, which is a directed graph connecting each element to its k closest neighbors. They introduce two versions of the search algorithm based on both distance and topology to solve nearest neighbor queries. In their results, these algorithms perform 30 % more distance evaluations in the document metric space than the reference implementation, using only a 0.25 % of the memory. This idea can be applied directly on the connectivity graph concept from SLAM, where this graph is used to find the closest neighbors when the camera revisits a space in the scene. We aim at reducing the number of connections to simplify the graph and improve the results in optimization procedures (see Section 3.1).

In the field of graph-based searches, Ismo et al. [63] introduce a method for detecting outliers using the Indegree Number (ODIN) algorithm based also on a KNN graph. They propose two new algorithms to improve existing distance-based methods: MeanDIST or KDIST. Their evaluations on both real and synthetic datasets show that better results are achieved on synthetic datasets, whereas they are worse on real datasets due to the lower number of observations. They concluded that in such case, density-based methods cannot obtain reliable estimates. The concept of ODIN can be also applied to measure distances in the connectivity graph of SLAM.

The KNN algorithm has also been applied to perform queries in spatial networks (e. g., road networks). In this applications, distances between objects depend on the topology of the network and they are expensive to compute. In [70], the authors propose to use a first order Voronoi diagram to evaluate KNN queries in spatial databases. They split large networks into small Voronoi regions with clusters. Then, intra- and inter-distances are precomputed to save both storage and computation time. In addition, computations are only performed on the border of the neighboring regions. They show that this solution outperforms the performance of traditional approaches by up to

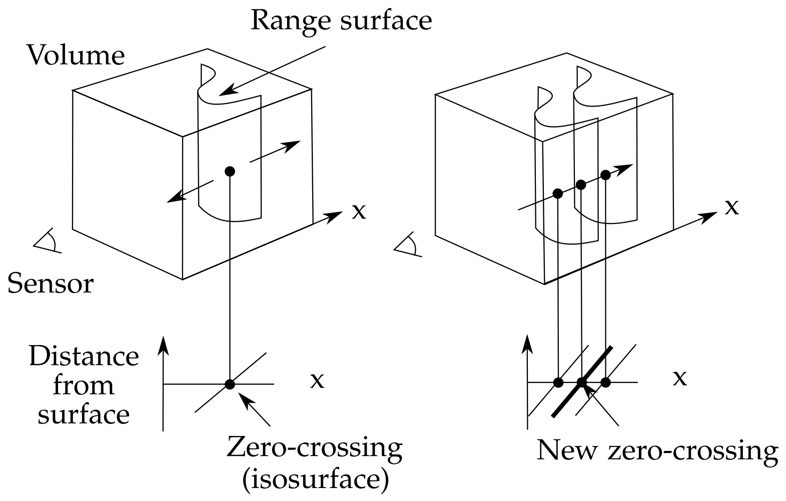


FIGURE 2.11: A volumetric grid is used for integrating and fusing multiple noisy 3D measurements using a TSDF. On the left, a new TSDF is inserted corresponding to a range measurement of the sensor. On the right, a second noisy measurement represented by another TSDF is fused with the original by summing both functions. The figure has been extracted from [22].

one order of magnitude. Voronoi diagrams can also be used for determining the fair value of k that preserves the connectivity between neighbors, while it still avoids redundancy in the graph.

2.6 SURFACE RECONSTRUCTION

The last step of a SLAM pipeline is responsible for creating the map of the scene. This map is usually a sparse or semi-dense point cloud that is not triangulated in a mesh. This makes the result not suitable for most applications. Some SLAM approaches include in their pipeline a post-processing step to convert the sparse point cloud into a convex volumetric surface.

The reconstruction process can be performed either online or offline, with computing requirements varying from one approach to another. On one hand, online reconstruction is useful in robotics applications and AR/VR gaming. The point cloud is iteratively refined and increased. On the other hand, offline reconstruction achieves more accurate results. In the end, the resulting point cloud is noisy and incomplete (e. g., holes). These problems pose significant challenges to conventional static 3D reconstruction algorithms.

It is possible to create Piecewise Linear (PL)-surfaces by triangulating the hull of the map. This reconstruction of surfaces is usually implemented on a data structure that integrates and fuses all the noisy partial meshes. A volumetric grid of voxels coupled with a TSDF [22] is the most popular alternative (see Figure 2.11). A powerful GPU is required to update the map in real-time when it is coupled with a dense approach as in KinectFusion [64].

More recently, Steinbrücker, Sturm, and Cremers [116] introduce an approach able to run in real-time in a CPU. They proposed a volumetric multi-resolution mapping system for RGB-D images. The textured 3D mesh is generated using an octree (instead of the conventional regular volumetric grid) that represents the scene at multiple scales. Furthermore, the reconstruction volume grows dynamically on demand. Only those voxels in a narrow band around the observed surface are allocated to save memory and computation time. The system maintains a consistent triangle mesh even though neighboring cells in an octree are more difficult to relate. They solve this challenge by keeping track of the dependencies between cells. As a result of this, the system can run in real-time on large RGB-D sequences on systems with limited computational resources.

The TSDF of each voxel can be stored and processed using sparse voxel hashes [92]. This approach scales well in large environments due to the sparsity of the grid achieved by removing empty spaces. This approach was extended in [23] to add integration and de-integration of frames (i. e., adding and removing frames from the reconstruction). Furthermore, these operations are symmetric, i. e., one inverts the other, to allow the system to update the model when the camera pose changes. In such case, the frame is de-integrated with its original pose and integrated with the updated pose.

The original work developed in KinectFusion serves as inspiration for other publications more focused on the 3D reconstruction than in the VO. For example, MonoFusion was introduced two years later in [103]. Instead of using a RGB-D camera, they use a single-lens, off-the-shelf camera as input sensor to build a 3D model in real-time. Therefore, this solution can manage robustly natural outdoor lighting, something that RGB-D cameras cannot handle. The system combines a sparse VO method with a volumetric fusion scheme. Camera poses are used for efficient dense stereo matching between the input frame and the best keyframe. The resulting depth maps are then fused in the volumetric grid using a computationally inexpensive method. In addition, the system can recover from tracking failures and filter out the noise from the 3D reconstruction. The system requires a GPU to compute the stereo map and merge them in real-time. The resulting 3D models are visually comparable to those obtained with KinectFusion.

Later, an approach similar to MonoFusion was introduced in [67]. The system utilizes a sparse tracker for camera pose estimation combined with a dense patch-based tracker for dense reconstruction. Feature matching in the sparse tracker is guided by a costly dense optical flow algorithm. The system determines if the model is completely covered before refining the reconstructed patches into denser and more accurate patches. The implementation is heavily based on the parallel architecture of the GPU.

MobileFusion [67] targets devices with limited resources like a smartphone by introducing the first pipeline for real-time surface reconstruction and camera tracking running on standard commodity mobile phones. They take advantage of a hybrid GPU/CPU pipeline that tracks the camera at 25 Hz. At the same time, the RGB input is registered in a volumetric grid to incrementally build a fully-connected 3D model by minimizing a noise-aware photo-consistency error metric. The reconstructed models show an average error of 1.50 cm w. r. t. the baseline models captured with KinectFusion. It should be noted that the scale of the reconstructed objects is small (e. g., human faces, shoes or pineapples).

It is common that these techniques are imported from other areas closer to Computer Graphics. One of these approaches is VolumeDeform [61]. This

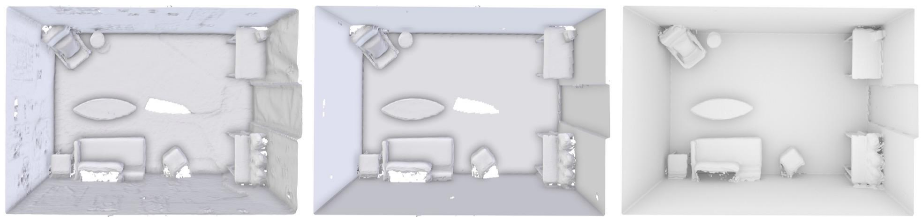


FIGURE 2.12: Usual 3D reconstructions suffer from noise and holes (left). The noise can be reduced by applying a plane prior (middle). The detected planes can be extended into unobserved regions to fill holes (right). The images have been extracted from [31].

work introduces an innovative approach to reconstruct at real-time the surface of an arbitrary non-rigidly deforming scene using an RGB-D sensor. The 3D model is built from scratch without a pre-defined shape template to start with. A common volumetric representation is used to encode a distance field of the surface geometry and the non-rigid space deformation. The tracking uses an indirect sparse VO approach combined with a dense depth constraint to reduce drift in model-to-depth alignment. The system finds the optimal deformation of space as a non-linear TV optimization that enforces local smoothness and proximity to the input. A data-parallel flip-flop optimization strategy allows to tackle the problem at 30 Hz in the GPU. The resulting system is robust to challenges such as fast motion and high-frequency textures in dynamically evolving scenes. The 3D reconstructed surface of the objects is iteratively refined in a conventional volumetric grid to include the latest deformations.

Some authors have proposed new improvements and optimizations over the core algorithms. For instance, in [31], the authors propose a novel stage in the 3D reconstruction pipeline to work with planar surfaces, such as walls, floors, and ceilings. Their contributions reduce the noise in individual mesh vertices and fill in the occluded and unobserved regions. The plane detection algorithm fits local planes and merges global planes. Local plane candidates are estimated with least-squares fitting on the volumetric grid. Global clustering is implemented with a simple one-point RANSAC over all planes. On each RANSAC iteration a plane candidate is chosen. Then, the other candidates are compared in terms of the angle of the plane normal and the Euclidean distance between the chosen plane and the volume of the candidate. Vertex jitter in flat surfaces is solved by omitting re-meshing whereas holes are filled in by extending planar regions into unobserved and occluded parts of the scene. The original reconstructions are augmented adding an average 40 % of mesh area. Their system runs in real-time (around 60 FPS) on CPU to generate more clean and visually appealing reconstructions (see an example of these results in Figure 2.12). Also, the algorithm supports a large number of natural environments, despite the presence of significant amount of occlusion, clutter, and noise.

2.7 Building Information Modelling

The Architecture Engineering and Construction (AEC) ecosystem is composed of technicians (e. g., architects, civil engineers, topographers), public administration (e. g., city halls, regional planners) and enterprises. This industry has been transitioning for the last ten years from the conventional manufacturing approach to an industrialized approach. This movement has been wrapped in the initiative called Industry 4.0 (see [113]), which comes after Industry 3.0 where computer and automation were the keys. In the new Industry 4.0, connected cyber-physical systems are part of a new set of technologies that intend to improve the industrial processes. The relevance of this new trend has motivated the creation of multiple European projects concerning the Industry 4.0 They have emerged from the Public Private Partnerships of Factories of The Future[1].

There exist two different ways of representing building models: entity-based modeling and object-based modeling. Despite the social, cultural and process factors affecting the wider adoption of object-based modeling (see e. g., [126]), the global trend is to support the latter. Nowadays, the object-oriented modeling approach is commonly known as BIM, which is one of the key components of the industry 4.0. The methodology proposed by BIM aims at collaborating around a single shared design of the building. This allows the team to analyze and visualize design decisions long before a project even breaks ground. At its core, BIM offers a digital documentation of the building, including function systems (e. g., electrical, air conditioning) and aesthetics (e. g., walls, roof, windows).

Semantics refers to the conceptual meaning behind an object. It can be described by an ontology (i. e., a representation of a knowledge domain [36]). Semantics and geometry in BIM are included in a common object-based data model: the IFC [78]. It is a platform-neutral, vendor-neutral open file format specification developed by buildingSMART[2], an international organization to foster interoperability between software applications in the construction industry. The IFC data model represent static building information, which includes structures, attributes, utilities, processes, actors, etc. However, environmental conditions or constructive processes are not just static and occur at different time scales. For example, small deformations happen frequently, whereas catastrophes occur suddenly. A more advanced framework would be necessary to represent these dynamic aspects too (see Section 2.7.3).

In BIM the same information can feed multiple applications based on AR, VR and Computer-Aided Manufacturing (CAM). Indeed, the same model can be used for the infographic design that advertises the building project (see e. g., Figure 2.13). Collaboration around the same model involves both AEC stakeholders (e. g., constructors, project managers, facility providers, etc.), but also any virtual agent or machine that require information from the model to accomplish their goals.

The combination of AR and the Internet of Things supposes the biggest transformation. This means that any manufacturing machine can directly ac-

1 http://ec.europa.eu/research/industrial_technologies/factories-of-the-future_en.html
2 https://www.buildingsmart.org/

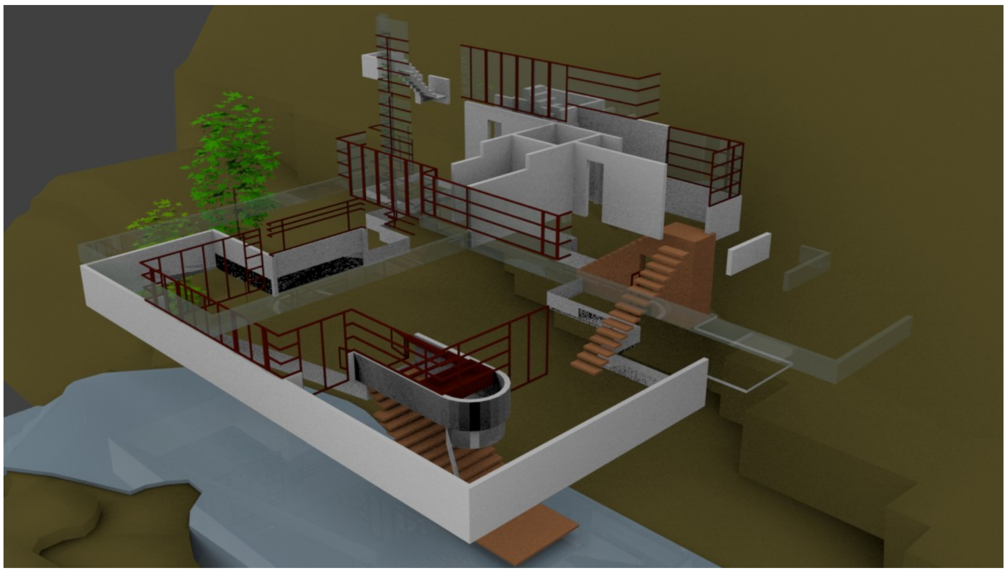


Figure 2.13: A rendered image of the Fallingwater house following an axonometric projection that highlights the internal structures of the building (stairs, windows, walls and doors). Image have been generated by rendering the Fallingwater house model with Blender. The model has been designed for the evaluation of 3D-SIMOS (see Section 4.2.1)

cess the shared BIM model so it can automatically start to craft the object without human interaction. Machines can directly talk to each other to collaborate in the manufacturing process, which usually involves 3D printing too.

The adoption of BIM lends itself to a more cost and time-efficient process, as well as a significant reduction in errors. The design-to-construction workflow has been re-engineered by the AEC industry in a similar way as the transition from 2D CAD to 3D solid models in the nineties. CAD was created to design and document objects and buildings via computer technology. Designers use CAD in their complex projects with multiple components that fit with precision within a larger assembly. The user can efficiently generate 2D drawings and 3D models to be manufactured with the corresponding materials. The 3D approach had a widespread adoption by many manufacturers that adapted their processes to meet the strict guidelines imposed by aerospace and automotive industries. This approach allowed the industrial market to create unique, high-quality products with a lower time-to-market.

The AEC industry is being progressively abandoned the CAD methodology and transitioning to the BIM methodology. This movement has been led by the users seeking real-time model analysis and 3D visualization of their designs. They also want to exchange the same model between different applications and stakeholders. The information in the design must be represented following the same specification. This is the key of BIM to ease several tasks, such as file-sharing, interference checking, and energy optimization.

Despite the apparent benefits of BIM, many companies still hesitate to invest mainly due to the complexity and the initial investment effort required by this new methodology. However, this investment has been shown profitable in the long term for enterprises. For instance, Oesterreich and Teuteberg [95] have assessed the economic impact of BIM at a corporate level using a cost-benefit analysis over a quantification model based on system dynamics. The assessment includes the subsystems of the organization affected by the adoption of BIM (e. g., personnel, customer, and finances), reflecting both indirect and direct cost. Results of the simulation show that the company's performance is improved in the long-term with 8 % return on revenue and a profit up to 15 times higher. The same conclusion can be extracted from other analysis [5, 32].

2.7.1 *Semantic Geographic Information System*

The main goal of GIS is to provide georeferenced support for conventional Information Systems. A GIS system represents 3D/2D information (e. g., buildings, infrastructures, underground installations or multiutility networks) anchored to a geographic location expressed in a Coordinate Reference System (CRS) where the terrestrial coordinates are projected. Georeferences support merging different information sources, such as aerial images, digital photogrammetry, and terrestrial 3D-laser scanning.

Conventional GIS representations only include the geometric primitives and the topological relations between them. Later, semantic information is overlayed and stored separately. However, current approaches advocate for a unified approach in which semantics, geometry, and topology are fused together. The main standard to represent and exchange semantic 3D-city models is City Geography Markup Language (CityGML) [71].

Conventional model representations only consider the structural aspects of the building, characterized by the objects and their relations. Dynamics can be incorporated over the static information provided by GIS models. Semi-destructive techniques (e. g., building refurbishment) or destructive techniques (e. g., underground excavation) require to simulate the effects of human interventions that may also affect their neighbors. This simulation was conventionally performed on separated layers but now it can be laid out over the semantic model, as in [94]. Furthermore, the temporal evolution of the static information allows the users to compare the results in scheduled tasks with their real execution.

There is a set of standard operations in GIS that have been traditionally performed over planar multilayer representations. In CityGML, volumetric *objects* are included to extend the scope of these operations. In general, any type of GIS must allow users to execute at least the following tasks:

- *Boolean operations*, such as intersection and union. They require masks and extended grids to manage the spatial data.
- *Identification of geometric primitives*, such as points, edges, and faces with their corresponding relations (e. g., incidence, adjacency, order).
- *Measurement* of distances, areas, and volumes.
- *Generation of thematic maps* related to the building and its environment, including available infrastructures, installations, and materials.

- *Topological tests* of spatial and temporal proximity. Spatial proximity depends on adjacency relations, whereas temporal proximity depends on the relations between components over time.
- *Queries* based on data, location, user-defined attributes, similarity, etc.
- *Recognition* of shapes by their role according to structural aspects, materials, and installations.

GeoBIM

The relation between BIM and GIS is studied in its own field of research, called *GeoBIM*[3] (see [4]). Both methodologies aim at different goals, different scales and, usually, different industries. Despite this apparent disconnection, an integration between both approaches is useful for task such as urban planning, disaster prevention or flooding simulation. The integration is technically feasible since any building can be georeferenced by selecting three anchor points. These points allow the system to compute the rigid transformation that can be applied to every coordinate of any building space. Additional anchor points enable to optimize the transformation to gain accuracy.

The semantic information from AEC environments is commonly integrated into GIS using their corresponding formats, i. e., IFC and CityGML. The Application Domain Extension (ADE) of CityGML allows any application to overlay domain-specific knowledge over CityGML (see the Unified Modeling Language (UML) examples in [128]). For instance, Hijazi et al. [57] proposed the UtilityNetworkADE to extend the semantics of CityGML to the utility networks within a building defined in IFC. Another example is the GeoBIM ADE developd by Laat and Berlo [74]. This extension appends to CityGML the information of the building that has no correspondence between IFC and CityGML. The ADE is part of BIMServer[4], a popular Open Source platform that allows the AEC stakeholders to collaborate in a building represented by the IFC. BIMServer also provides an open architecture based on plugins to foster further developments by the community.

Both CityGML and IFC can be interpreted as ontologies since they provide a knowledge representation for a specific domain. The reconciliation between different ontologies is studied in another research field called Ontology Alignment (see [21, 36]). Thus, techniques in this field are helpful to transfer information between IFC and CityGML. We surveyed and analyzed the utility of such techniques for automatic alignment of geospatial ontologies in [28]. Results show that an automatic alignment between geospatial ontologies is not recommended when high accuracy is required (above 90 %). In order to solve the problems with automatic ontology matching techniques, we have decided to align manually the schemas of the planning and the CAD model with the IFC that include concept from both schemas (see Section 4.1.3).

One of the main advantages of using ontologies is that they provide a natural language for spatial queries. For example, in [10] the authors developed a language for querying 3D-city models when they were labeled with a well-defined ontology. Unfortunately, most of the currently available datasets are

3 https://geo-bim.org
4 http://bimserver.org

still not labeled. Furthermore, there are not accurate techniques to implement automatic labeling of the structural elements of the building.

2.7.2 *Advanced Web3D visualization*

Web3D[5] was a term created at the end of the nineties linked to the navigation and visualization of 3D contents included in websites and rendered by a web browser. This basic idea has originated several proposals concerning the file format for representing the contents. The original Virtual Reality Modeling Language (VRML) language [15] to embed 3D contents in the browser implementation was extended later to X3D [11]. However, these specifications were never considered by the developers of web browsers, and their implementation were usually based on specific plugins.

On the other side, the browsers were not able to access the underlying power of GPU, which leads to the underperformance of any Web3D application w. r. t. its desktop counterpart. The WebGL specification [2] aims to fix this problem by providing a JavaScript API that allows applications to access the GPU for rendering advanced graphics. Furthermore, additional plugins like Flash are not required anymore since the WebGL API is implemented natively by the web browser.

The WebGL specification brings a new file format for exchanging graphics called GL Transmission Format (glTF) [107], which is a revamped version of COLLADA [3]. This new format was born with the same goal as COLLADA, i. e., to allow the applications in the 3D graphics industry to collaborate. However, glTF introduces some improvements like more efficient representations, embedded binary objects, and more complex materials. Despite COLLADA still exists and it is supported by many applications, it is being progressively replaced by glTF. Indeed, it never was popular among the targeted applications. The new glTF 2.0 specification includes Physically Based Rendering (PBR) materials following the trend in the Computer Graphics industry.

A widely used concept in 3D visualization is the *scene graph*, a general data structure that represents the logical and spatial relations between the objects of a graphical scene. A node of the graph may represent any transformation that affects its children, the geometry of the object, a material, a light source or simply a group of children. This transformation can be a geometrical transformation matrix that will be concatenated to determine the world coordinates of a leaf. The main advantage of this representation is the ability to group related objects and materials to ease its management as a single object. The idea of representing geometric transformations (i. e., rigid body transformations) is similar to the connectivity graph explained in Section 2.7.3.

The National Science Foundation defines the *Advanced Visualization* as a method of computing (see [50]). Visualization is focused on better understanding of reality. It transforms the symbolic into the geometric, enabling researchers to observe their simulations and computations. New scientific discoveries can be made to foster profound and unexpected insights. This knowledge discovery is supported by the representation of physical objects, which can be annotated with their corresponding semantics (i. e., the meaning in a

5 http://www.web3d.org

particular context). Semantics combined with smart interfaces, help to lighten the complexity of the visualization tool.

An Advanced Visualization system for BIM must be designed taking into account the following principles:

- *Needs and preferences of the users* refer to the integration, standardization, reliability, and performance of Advanced Visualization tools used to share, update, and manage the visualized information. Services provided to each agent must be defined with a flowchart. Furthermore, the roles of each agent must be identified according to their competences.
- The *generation of relationships* has two levels that concern components and collaborations between agents. In our approach (see Section 4.1.1), the first level follows a cellular structure where cells are defined by cuboids representing objects. Relations are determined by the allowed operations between cuboids. The second level corresponds to different tasks of constructive processes.
- The *generation of knowledge* is achieved through the visualization. Smart interfaces ease the generation of reports representing distances, thematic maps, or undiscovered associations between data.

These principles can be applied to Web3D technologies for providing insightful visualizations to the AEC stakeholders. This visualization should include useful functionalities like automatic animations of the elements of the scene. These animations can be performed by interpolating the position of the element along a trajectory path.

Spherical Linear intERPolation

Spherical Linear intERPolation (SLERP) [114] is a quaternion interpolation method to animate rotations with constant-velocity motion along a circular path. This path is the spherical equivalent of a conventional path along a line segment in the plane, i. e., a spherical geodesic. The main advantage is that the formulation of SLERP is independent of quaternions or the dimension of the space of the arc that describes the trajectory. It can be expressed as a weighted sum based on the fact that any point on the path is a linear combination of the beginning and the end. Let p_0 and p_1 be the unit vectors of the first and last point of the arc, and t the interpolation parameter with $0 \leqslant t \leqslant 1$, then any point at the trajectory is defined as

$$\boldsymbol{p}_t(\boldsymbol{p}_0, \boldsymbol{p}_1; t) = \frac{\sin((1-t)\Omega)}{\sin \Omega}\boldsymbol{p}_0 + \frac{\sin(t\Omega)}{\sin \Omega}\boldsymbol{p}_1 \tag{2.32}$$

where Ω is the angle of the arc so that $\cos \Omega = \boldsymbol{p}_0 \boldsymbol{p}_1$.

The interpolation can also be defined on the space of unit quaternions. In this space the interpolation executes a rotation with uniform angular velocity around the rotation axis defined by a quaternion. If the starting point is the identity quaternion, SLERP determine a segment of a subgroup of both $SO(3)$ and its universal covering group of unit quaternions (S3). The result maps to a rotation through an angle of 2Ω. However, the rotation path may be short ($\Omega < 180$) or long ($\Omega > 180$) since $\boldsymbol{q} = -\boldsymbol{q}$. This problem can be solved by negating the quaternion when the dot product is negative so $-90 \leqslant \Omega \leqslant 90$.

Quaternion exponentiation is the key to express SLERP in a compact way. Powers of quaternions are defined in terms of the quaternion exponential function

$$e^{\boldsymbol{q}} = 1 + \boldsymbol{q} + \frac{\boldsymbol{q}^2}{2} + \frac{\boldsymbol{q}^3}{6} + \cdots + \frac{\boldsymbol{q}^n}{n!} + \cdots \tag{2.33}$$

Let write the unit quaternion $\boldsymbol{q}$ in its versor form, i. e., $\boldsymbol{q} = \cos\Omega + \boldsymbol{v}\sin\Omega$ where $\boldsymbol{v}$ is a unit three-dimensional vector with $\boldsymbol{v}^2 = -1$. Then, $\boldsymbol{q} = e^{\boldsymbol{v}\Omega}$ and $\boldsymbol{q}^t = \cos(t\Omega) + \boldsymbol{v}\sin(t\Omega)$ lead to $\boldsymbol{q} = \boldsymbol{q}_1\boldsymbol{q}_0^{-1}$. Thus, the real part of q is $\cos\Omega$, which matches the geometric dot product of Equation 2.32. Hence, the SLERP expression in the quaternion algebra is:

$$\boldsymbol{q}_t(\boldsymbol{q}_0, \boldsymbol{q}_1; t) = \boldsymbol{q}_0(\boldsymbol{q}_0^{-1}\boldsymbol{q}_1)^t \tag{2.34}$$

2.7.3 *Dynamic information modelling*

Conventional BIM-based models represent the static view of the construction. They do not consider a representation of the dynamics of the processes affecting the building that are required by simulation or analysis (see [115]). This task is usually accomplished by third-party tools. However, the problem should be tackled from the lowest level by proposing a common symbolic representation in which embed the dynamic data concerning the building. It can be addressed by a spatial decomposition of the the building in a collection of linked cells, following a classical PL approach. These cells, which can be geometrically deformed, are the nodes of a directed acyclic graph (see [9]) that represents the building (see $S1, S2 \ldots S7$ in Figure 2.14). Such representation is adaptive since basic units (cells) are grouped in cubic complexes by simple operators (e. g., join and intersection). Beyond the formalism, this representation provides a unified approach to update, insert and evaluate changes in processes.

The life of a building includes several phases, such as design, planning, execution, maintenance and/or demolition. These stages, or any overall process, can be modeled as a workflow with a large number of inputs that evolve over time. These inputs can be managed by scalar, vector, and tensor fields (see e. g., [52]). Thus, the representation of static information (objects) and dynamic information (functionals) can be separated for a better management of the information.

Dynamical representations are motivated by changing environmental conditions. The graph-based representation can be adapted to different environments in topological terms by adjusting the relations between modules. Evolution of the environment is described by different types of fields acting over the building: scalar fields (e. g., functions), vector fields (e. g., evolving surfaces) or tensor fields (e. g., processes). Stakeholders intervening in the processes

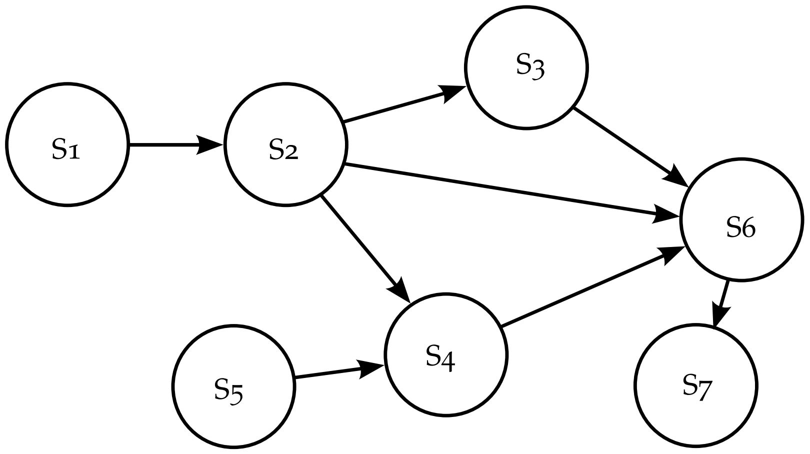


Figure 2.14: An example of a directed acyclic graph containing 7 nodes corresponding to 7 different building spaces

can alter these fields. Due to the broad diversity of objects and functionals to be evaluated, it is convenient to adopt a symbolic representation to treat the information as a flowchart.

Objects and functionals

The *topology* on a space X is defined as a collection of (open or closed) sets that cover X and verify closure properties w. r. t. simple set-theoretic operations (union and intersection). Alternately, the topology can be defined as a collection of functions on intersections of open sets U_i covering X. Both descriptions are useful since the first one refers to the managed *objects*, whereas the second one refers to the *functionals* evaluated along processes. They support the classical topological relations, such as order (for defining hierarchies), discontinuity (for degrouping), adjacency (for detecting the closest neighbors) or incidence (for aligning). Furthermore, they are compatible with metric constraints that can be helpful in some AEC tasks:

objects To assemble basic primitives during planning, to validate measurements or to evaluate characteristics that involve structural aspects (e. g., walls, slabs, pylons) and materials aspects (e. g., elasticity, tolerance under extreme conditions, porosity).

functionals To improve the execution of the tasks by applying the constraints to optimization procedures or to team evaluation.

The former constraints are represented as relations between objects in BIM, whereas the latter constraints are indirectly represented, for instance, as color scales. By combining both of them, we can establish feedback between the morphological and the functional aspects of the building using scalar, vector, or tensor fields.

In the symbolic representation, objects and functionals can be defined as follows

- Each *object* is represented by a finite collection of nodes (i. e., a geometric primitive). Nodes are connected by edges (i. e., adjacency relations). Expansion and contraction operations introduce a hierarchy for practical

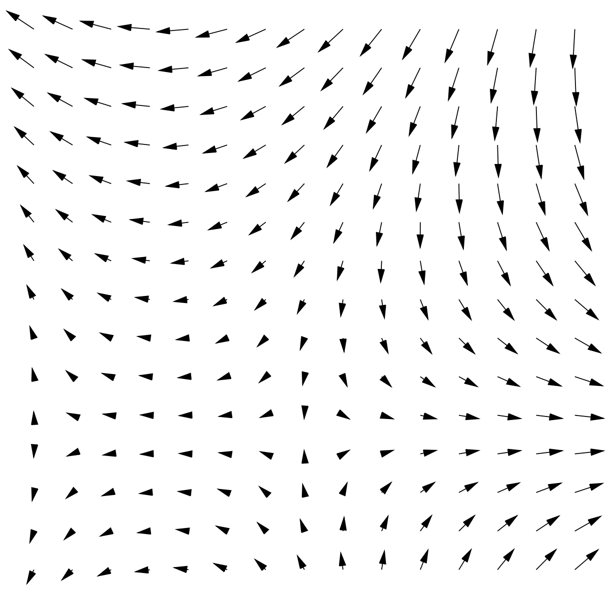


Figure 2.15: The airflow is visualized by a vector field where the vector at the center of each cell in the grid corresponds to the direction and magnitude of the flow.

visualization. In this way, any process $\mathcal{P}$ that involves objects and tasks is represented by a composition of *directed adjacency graphs* $\mathcal{G}_a(\mathcal{P})$.

- Each *functional* to be evaluated during a process is represented as a functional on the graph $\mathcal{G}_a(\mathcal{P})$. Its output returns a collection of weights for the nodes, so a weighted graph is achieved. This graph represents the processes to be accomplished for each task.

2.7.3.1 *Fields and flows*

Unexpected events (e. g., incidences, defective specifications, misrepresentations) may appear during constructive processes. These changes must be incorporated into the flowchart with the objects and functionals. In Physics and Mathematics changes are represented by fields. Any flow can be described and visualized by a field (see [52]). For example, any 2D/3D flow can be visualized with the vectors that represent the direction and magnitude of the flow at each point of the field. These points are usually the centers of the set of evenly-sized 2D/3D cells (i. e., squares or voxels) that subdivide the space in a grid. An example of conventional airflow is visualized in Figure 2.15.

As it has been mentioned in previous paragraphs, there are three types of fields:

scalar fields They represent scalar quantities such as geometric features (e. g., scale, height or depth), temporal features (e. g., time), costs of tasks (e. g., work intensity or wages).

vector fields They represent vector quantities, such as the oriented length of utility networks, attributes for areas of constructive elements, or evolving volume of the spaces of the building. Such quantities can be tracked with n-dimensional (nD) distributions of vector fields. These distributions can be integrated during a task to obtain trajectories for their control points. The work accomplished for each distribution $\mathcal{D}$ is evaluated with functionals $f : \mathcal{D} \to \mathbb{R}$ called differential forms. Such forms can be linear or non-linear depending on the coefficients that are linked to the characteristics of the process. Anyway, their evaluation provides scalar fields.

TENSOR FIELDS They represent the simultaneous evaluation of a finite collection of vector fields (representing workflows) or their dual differential forms (representing the performed work). For example, energy consumption can vary its representation during the lifecycle of the building. Different curvature tensors are formulated following the rigidity constraints (e. g., substitution of factors).

These fields fluctuate with time t, and they can be queried at different scales s. Hence, they can be evaluated at several scale/temporal slices with a parametrization (s, t). In practice, all fields are described in a discrete way, and all functions are defined locally, including their possible discontinuities.

The visualization of fields is implemented using PL-paths generated by sampling the field at critical values (e. g., unexpected events), control points (e. g., integral curves of vector fields) or hybrid milestones. Hence, the decoupled representation can be easily extended to a dynamical representation using distributions of fields defined on the source (ideally $\mathbb{R}^2$) and target (ideally $\mathbb{R}^3$) spaces.

Multidimensional modelling

Spatial 3D models can be extended to higher dimension for concurrent management of multiple building aspects (nD models). This model has been traditionally applied to GIS with $n \geqslant 4$ by means of a multilayer representation of different attributes/properties over the geometric primitives. However, this approach has not been applied to object-based modeling until the appearance of CityGML. Time is the most commonly included dimension, but other supplementary dimensions can be added, such as costs, environmental conditions, energy efficiency, etc. These additional dimensions can be interpreted as solutions of a field. Changes in the associated functionals can be visualized with color maps.

Each space of a building can be represented as a volume changing over time t that also depends on the scale s. This representation outlines a 5D model where each point has coordinates (s, t, x, y, z), as in [96].

Dependencies between objects are represented by adjacency relations. Spaces of the building can be assembled in a spatio-temporal surface by including additional scale-time coordinates that represent the coupling between spaces. The resulting 5D-model can be reformulated as the graph $G : \mathbb{R}^2 \to \mathbb{R}^3$. This decoupled representation enables to align static 3D models by performing transformations in the target space T of the map $G_{\mathcal{P}} : S \to T$ of process $\mathcal{P}$. Similarly, temporal alignment along different scheduled tasks is performed in the source space S. This graph can be queried within a specific time and scale.

Dependencies between functionals require the evaluation of functionals on the graph $\mathcal{G}_a(\mathcal{P})$. In the absence of critical points for these functionals, the topology between hypersurface levels of each functional is essentially the same. However, in the presence of a critical point $c \in \mathbb{R}$, qualitative changes appear in the topology of hypersurface levels. Let denote by $W\Gamma_{\mathcal{G}}$ the space of weighted graphs of decoupled volumetric representations G, then any particular graph $\mathcal{G}_a(\mathcal{P}_i) \in W\Gamma_G$ with different topology represents a qualitative change in constructive processes.

Both tasks and events can be interpreted as snapshots in a process. In a functional approach, tasks correspond to natural values, whereas events correspond to critical values. Additional constraints can be imposed with a combination of external forces (e. g., temporal availability or budget limitations) and internal forces (e. g., ground behavior or structural specifications). Such constraints are inspired by the Elasticity Theory, which distinguishes between tangential forces (i. e., do not modify the model) and internal/external forces (i. e., deform the model). The only difference is that this approach allows "fractures", i. e., qualitative changes in the topology of level hypersurfaces.

The sections of this chapter have been focused on the state of the art of the components included in our framework (see Section 1.1). We have conducted a broad review of the main concepts involved in this framework to provide a basic understanding of the background of our research. At the same time, it serves as a starting point for our contributions. The pipeline of the mentioned framework is susceptible to be improved at different levels and components. However, we have focused on the specific aspects of SLAM and BIM that allow us to accomplish the goals stated in Section 1.2. The next two chapters introduce the contributions to which we have largely dedicated our research. Chapter 3 covers the contributions implemented by KN-SLAM, whereas Chapter 4 comprises the contributions attributed to 3D-SIMOS. The final chapter (Chapter 5) summarizes the conclusions extracted from both contributions in particular, and from the thesis overall.

3 ACCURATE AND ROBUST SLAM

We have developed a SLAM system called KN-SLAM able to robustly track a monocular camera in different scenes. Our implementation is heavily inspired by ORB-SLAM, developed by Mur-Artal, Montiel, and Tardos [88]. It is a SLAM system based on indirect visual odometry, which is entirely driven by ORB features.

The accomplishment of our original SLAM goals requires to develop and integrate a set of existing libraries and components. We choose to reuse open-source components to the extent possible due to the scarcity of previous developments by our side. Our initial achievements at the beginning of the Ph. D. were exceeded by ORB-SLAM, whose main objectives were closer to our own goals. Later, the authors released their software under the GPLv3 licence[1]. Thus, we decided to not reinvent the wheel but to make a contribution over the excellent work performed by Mur-Artal, Montiel, and Tardos [88].

We put the focus in the evaluation of the influence of the topology of the connectivity graph in the accuracy of the trajectory. More concretely, we have analyzed the sparsity of the graph, since our main concern about ORB-SLAM is the choice of the threshold θ_{min}. This threshold represents the minimum number of observations required to connect two keyframes in the graph. We hypothesize that this threshold, combined with a KNN approach, may improve the topology of the graph. This hypothesis and the other ones are contrasted using an exhaustive experimental evaluation conducted along a set of more than 45 sequences belonging to 4 different challenging datasets. The idea of selecting only the K-best visual neighbors led us to choose KN-SLAM as the name for our implementation. Following the same initiative as ORB-SLAM, we release KN-SLAM as open source software for the benefit of the community[2].

Finally, our experimental results show that, even though KN-SLAM outperforms ORB-SLAM on average, it does not happen for all sequences. Thus, we consider an additional contribution in the form of a systematic procedure to decide between KN-SLAM or ORB-SLAM. Such procedure helps to discern which system would achieve a better accuracy regarding the characteristics of the input sequence. This is included as part of the analysis of the results of our experiments.

The rest of the chapter is structured as follows: Section 3.1 details our contributions to specific tasks in the SLAM pipeline exposed by ORB-SLAM; Section 3.2 describes the experimental setup, the datasets, the characteristic of the sequences and the evaluation metrics used to compare both approaches in terms of accuracy and robustness; Section 3.3 collects a quantitative evaluation and discussion of these results; Section 3.4 summarizes the conclusions extracted from our research and gives some guidelines and hints for further development.

1 https://github.com/raulmur/ORB_SLAM2
2 https://gitlab.com/fradelg/kn-slam

3.1 CONTRIBUTIONS

The advances in the state of the art performed by ORB-SLAM are inarguable. The system is able to automatically bootstrap and track the camera in challenging situations, outperforming other previous approaches. Besides this accurate results, the last version of the system also manages more complex setups, such as stereo and RGB-D cameras [90]. However, the performance and robustness of some stages of the pipeline can still be improved.

The accuracy achieved by ORB-SLAM thanks to the combination of local BA and global loop optimizations is remarkable. Its main limitation is inherited from its indirect approach to the problem: it only works in scenes with objects whose materials have low-frequency textures. However, the addition of a direct approach fallback would affect the real-time execution of the tracking. Moreover, the integration between both approaches in the SLAM pipeline is not straightforward. Other limitations addressed in our research include

- the excessive time required to load the vocabulary tree in the place recognition module;
- the non-exhaustive and greedy algorithm to match 3D points projected over new frames with the corresponding keypoints; and
- the low precision of the loop detector that discards some true positives and does not adapt to the state of the tracker.

In addition, we noticed the existence of a lot of hard-coded parameters in the code that do not adapt to the characteristics of the input sequence. Most of these parameters are thresholds in RANSAC procedures for assessing the quality of system operation. Besides, little explanation was given in the paper regarding the selected values, even if it might be obvious [88]. We have also modified the Graphical User Interface (GUI) to visualize the current frame, the keypoints of the features per frame and the three-dimensional point cloud with color information.

Next subsections are devoted to describing more precisely our contributions and improvements. They have been organizer taking into account their position on the processing pipeline from the system is bootstrapped until the trajectory is saved. More specifically: Section 3.1.1 describes our implementation of a faster vocabulary loading procedure; Section 3.1.3 pictures changes in 2D-3D matching algorithms used at performing a guided search; Section 3.1.4 dives into the algorithm to limit the number of keyframes linked in the connectivity graph; Section 3.1.5 delves into a smarter procedure for a more precise visual loop detection.

3.1.1 *Vocabulary loading*

Booting and loading times are aspects usually ignored in SLAM systems. In our experimental settings, we hold the input frames until the system is ready. However, there may exist configurations where a cold start is needed. In such cases, if the camera starts to move before the system has booted, the frames at the beginning of the sequence will be lost. In ORB-SLAM, system load settings and a vocabulary file at booting. The former takes a significant time whereas the later is almost unnoticeable.

The vocabulary plays an important role in the place recognition module embedded in ORB-SLAM. This module is based on the bag-of-words metaphor that abstracts an image with a set of features. Such features are equivalent to the most outstanding words of a document. This module intervenes in some important stages of the SLAM pipeline, such as the loop detection and the relocalization of the camera when the tracking is lost (see Section 2.4).

In the bag-of-word metaphor, the vocabulary is built as a discretized version of the features detected in the training scenes. This vocabulary is provided by ORB-SLAM and it has been built to be general enough to work in different types of scenes. However, its allocation in memory requires a significant amount of time. More precisely, the vocabulary downloaded by ORB-SLAM has 145 MB after decompressing and it requires around 9 s to parse and load in memory.

In order to solve this issue, we have developed a faster vocabulary loader based on Google's Protocol Buffers technology[3]. Protocol buffers are a flexible, efficient, automated mechanism for serializing structured data. In a first step, message types must be defined in separated `.proto` files. Each message represents a small logical record of information composed by (name, value) pairs. Then, the protocol buffer compiler generates data access classes for the selected programming language. This generated code provides accessors and serialize/parse methods for the whole structure to/from raw bytes.

Protocol buffers introduce huge advantages over ASCII or XML representations. They are simpler and less ambiguous, and about 3 to 10 times smaller and up to 100 times faster. This is achieved thanks to the protocol buffer binary format, which encodes messages using base 128 varints, a method for serializing integers using one or more bytes. In this encoding scheme each byte, except the last one, has the most significant bit set to 1, which indicates that there are further bytes to come. The lower 7 bits of each byte are used to store the two's complement representation of the number, least significant group first. There are other wire types that encode different data types[4]

We have developed our vocabulary loader for KN-SLAM by including the code generated by the Protocol Buffer Compiler in our sources. We model each basic data structure of the vocabulary as a different `.proto` class. This means that visual words must be represented using messages of the protocol buffer. We have identified three related types of messages according to the meaning in the original publication [43]:

VOCABULARY . This is the main message since it represents the root element that contains the whole bag of words. It contains the branching factor, the depth levels, the weighting method based on conventional metrics from Information Retrieval (Term Frequency - Inverse Document Frequency (TFIDF), Term Frequency (TF), Inverse Document Frequency (IDF), or binary) and the scoring method employed to compare two vectors of words (L1, L2, Chi-Square, etc.). Moreover, it links with the list of first-level nodes of the tree and the list of words of the entire vocabulary.

3 https://developers.google.com/protocol-buffers
4 https://developers.google.com/protocol-buffers/docs/encoding

NODE A node in the tree abstract a set of related words. Each node contains its own identifier and the identifier of its parent. Also, it includes the weight and the feature descriptor linked to the node.

WORD . Each word is a metaphor to abstract a set of feature descriptors. A word in the vocabulary is represented as a leaf of the tree. A word also requires an identifier and the identifier of the linked node.

Once these classes are compiled, we include the resulting code in our implementation to develop new methods and tools. These tools include a converter that parses the original vocabulary in `.txt` format to export it in the `.proto` format. This task is only executed a first time. Once the vocabulary is converted the next executions will directly load it in the Protocol Buffer format, taking advantage of the performance benefits achieved with this data representation.

As a result of these changes, the ORB vocabulary is loaded almost 10 times faster than ORB-SLAM. More concretely, loading times have been reduced from around 9.04 s to around 0.78 s. Furthermore, the size of the vocabulary has been trimmed to 64 MB from the original size of 145 MB for the ASCII file (42.50 MB in the compressed version).

3.1.2 *Robust bootstrapping*

Bootstrapping is an important matter for any SLAM system. It is the first stage of the pipeline so that the accuracy of the camera poses greatly relies on a good initialization. The policy of ORB-SLAM is to bootstrap the system only when the initial pair of keyframes has enough quality. Such quality is measured in terms of the number of features of the keyframes, the number of features matched between both keyframes and the number of points in the initial map. The rules of this policy are fixed in the code and do not vary according to the characteristics of the sequence. This leads ORB-SLAM to detect false negatives if there are not enough matches to support the homography model described in Section 2.4.1. This happens even though most of the sequences usually have a good pair of keyframes from which the system can bootstrap.

Our proposal relaxes the thresholds imposed by ORB-SLAM as the previous thresholds are exceeded. We use a simple exponential decay function with the number of failures. Obviously, the function is clamped to a minimum or maximum to avoid degeneracies. Our goal is to robustify the bootstrapping procedure, i. e., to initialize sooner in more sequences. After analyzing the evolution of the number of matches in the sequences in which ORB-SLAM refuses to bootstrap, we have modified the bootstrapping procedure.

The minimum number of allowed matches between keyframes has been reduced from 100 to 50. However, this reduction is not fixed. It is proportional to the number of failures t in the bootstrapping procedure for the input sequence. We have modeled this reduction with an exponential decay function using a decay constant of -0.1 and the initial value of 100. The following function has been used to determine the threshold after the t-th failure:

$$\theta_{min}(t) = max(50, 100e^{-0.1t}),$$

where $\theta_m in$ represents the adaptive threshold for the minimum number of matches θ.

The size of the search window μ has been accordingly modeled in an inversely proportional way since wider windows allow to find more matches (although they also create more outliers). We start with a window size of 50 and this value is increased up to 100. We use again an exponential decay function with an initial value of 50 and a decay constant of 0.1:

$$\mu(t) = min(100, 50e^{0.1t}).$$

The decay constant $\lambda = 0.14$ has been selected according to the maximum number of allowed failures before reaching the clamping threshold. We have experimentally determined that in the worst sequences 5 failures have reached. Therefore, we need to solve the equation

$$100e^{\lambda 5} = 50 \tag{3.1}$$

to determine that $\lambda = 0.14$.

Our implementation allows KN-SLAM to test more and better pairs of keyframes due to the lower parallax imposed by the small size of the search window. Our hypothesis is that it is better to test more pairs of keyframes with a short baseline than fewer pairs with a long baseline. In this last case, it is more likely that the tracking was lost and the bootstrapping procedure resets the system. Moreover, the probability of finding bad matches increases with the baseline since some objects of the scene may appear between both frames. In addition, the longer the baseline, the wider the search window. This affects directly to the performance of the procedure since more keypoints must be compared.

The minimum number of points in the initial map is determined by the minimum number of matches. This effectively decreases the fixed threshold of 100 defined by ORB-SLAM and it is coherent with the reduction in the number of matches. After all, these points are created by triangulating the matched keypoints. Indeed, this threshold should be lower than the minimum number of matches since some map points are discarded if they do not fulfill some of the following conditions:

- Some coordinate x, y or z is at infinity.
- The depth of the point is negative, i. e., the point is behind both cameras.
- The reprojection error over both keyframes is above some threshold.
- The parallax between rays connecting the camera centers with the map point is under 20°.

We have not reduced this threshold because in the practice all matches are triangulated. A significant difference between the number of matches and points means that the homography or the fundamental matrix do not explain well the relative transformation between keyframes. Thus, we prefer to mark the intent as a failure and try to achieve a better model with a new pair.

These contributions to the bootstrapping procedure help KN-SLAM to bootstrap in challenging situations where ORB-SLAM discards both the homography and the fundamental matrix, like in the sequence `tum/fr1_floor`.

3.1.3 *Guided matching*

The advantages of sparse indirect methods reside in the abstraction of the images in terms of geometric features. These methods heavily rely on the tracking of this points (not pixels) along the frames of a sequence. Thus, the most important algorithm consists in searching the keypoints of the current frame that match the keypoints of the previous frame. The search algorithm may vary slightly depending on whether the system recovers from a bad pose estimation or the system is tracking consecutive frames.

The steps of any guided search are the same ones. In the first step, the queue of map points is initialized with the points of the scene observed in the previous frame. Outliers are filtered out from the original set of points. Then, the following sequence of actions is performed for each map point in the head of the queue:

1. The 3D map point is projected in the new frame using the camera pose that is estimated assuming a constant velocity model.

2. A local search is performed in a region of the new frame centered at the projection of the map point. The radius of the search window is 15 times the scale factor of the pyramid level in which the ORB feature was detected in the last frame. The radius of the search is increased to 30 times the scale factor if in the first search fewer than 20 features are found.

3. The keypoints of the current frame that lies inside the search window are selected.

4. Each keypoint is compared with the 3D map point using the descriptor distance as a metric. This distance is computed as the hamming distance between both descriptors of 32 B. The descriptor of a map point is defined as the median of the descriptors of its observations. In each comparison, we save the keypoint with the minimum distance to the map point.

5. If the closest point has a distance below a threshold, the map point is linked to the keypoint. We use the threshold of 100 used by ORB-SLAM. Here we are assuming that the keypoint is the projection of the linked map point, even though this fact has not been geometrically checked yet (it will be checked later).

6. In order to link each map point with its closest keypoint, we allow replacing a previous association. The replaced map point is appended to the end of the queue of map points. This point will be processed in a

posterior iteration. Then, it will be linked to the closest available keypoint, i. e., those keypoints that still remain without a link.

The addition of the last step implies a slight overhead in the computation cost of the algorithm. However, it is able to increase the number of valid matches, i. e., those matches that are not rejected in a posterior geometric validation of the model. We have achieved up to 20 % more matches than ORB-SLAM in our experiments. Our searching algorithm increases this number of valid matches between keypoints and map points (i. e., 2D to 3D) to achieve a more robust tracking. The system should be less prone to lose the tracking of the camera in challenging conditions, such as in sequences with fast camera movements or strong rotations.

3.1.4 *Connectivity graph*

The pose graph (or connectivity graph) is an abstract representation of the relations between the keyframes of a trajectory. Each keyframe is linked to other keyframe taking into account their shared portion of the scene. This portion can be quantified with the number of shared observations (map points projected on the image plane of the keyframe). The connectivity graph is used for searching spatial neighbors of some keyframe. Most of the procedures to optimize the camera pose of such keyframe depends on the connections in the connectivity graph. So the accuracy of a trajectory is heavily influenced by the structure and density of this graph.

The pose graph is incrementally updated when a new keyframe is detected by the mapping thread. We have maintained the same keyframe spawning policy than ORB-SLAM to handle strong rotations with more keyframes. These keyframes will be culled later if the camera revisits the same frames. The last frame will be converted to a keyframe and inserted in the connectivity graph if it shares less than 90 % observations with the previous keyframe. In addition, one of the following conditions must be true:

- More than one second has passed since the creation of the last keyframe.
- The local mapping is idle, i. e., there is no triangulation or optimization underway.

Thus, the ability of the tracking thread to insert new keyframes depends on the time required by the mapping thread to perform optimizations. Indeed, the performance of the whole system degrades if the frequency of arrival of new frames exceeds the time required to process such frames. This could happen if the image resolution of the frame were too big or the framerate was too high. In such cases, the mapping thread would be unable to triangulate and adjust the points of the scene.

The recently created keyframe is inserted in the pose graph. At the time of the insertion, the connectivity of the new keyframe is determined by accounting the number of map point shared with other keyframes. One of the main questions is when these shared observations are representative enough to create a new edge between the new keyframe and the other keyframe. The connectivity of the keyframe is exploited in several SLAM tasks, such as relocalization, loop detection, keyframe culling and bundle adjustments. Thus, the number of edges does not only impact the accuracy of the estimated trajectory but also the memory and performance of these procedures.

The more links between keyframes, the more constraints are imposed on the optimization procedures, which increase the computational cost. Besides, more dense graphs do not always achieve better accuracy. Actually, our experimental results show quite the opposite. Smaller pose graphs, i. e., a graph with a constraint set of edges or connections for each node, improve the accuracy of the estimated trajectory depending on the scene.

As mentioned above, the algorithm to create connections between two keyframes is based on the number of keypoints (pixels with a maximum in their intensity gradient) shared between both keyframes. Every time a new keyframe is created by the mapping thread, a new node is inserted in the graph. The 3D map points observed in the current keyframe are the key to determine the connections of the corresponding node in the connectivity graph.

Let denote the set of keyframes by $\mathcal{K}$ such that $K_i, K_j \in \mathcal{K}$ are the i-th and j-th keyframes in the connectivity graph. The number of shared observation between K_i and K_j is denoted by θ_{ij}. A map point viewed from the keyframe K_i is represented by $\boldsymbol{p}$. An observation $\boldsymbol{o}$ is the projection of a map point $\boldsymbol{p}$ over a keyframe K with $\boldsymbol{o} = \pi(\boldsymbol{p})$ (more details in Section 2.1). The number of keyframes in the graph is represented by $M = |K|$ and the number of observations by $N = \sum_{i,j}^{N} \theta_{ij}$. The Cumulative Distribution Function (CDF) of the observations shared between K_i and the rest of $K_j \in \mathcal{K}$ is given by ϕ_i.

The procedure described in Algorithm 1 updates the connectivity of the current keyframe K_i w. r. t. the rest of keyframes $\mathcal{K}$ in the connectivity graph. Thus, the output degree $\deg^+(K_i)$ of the keyframe K_i is determined with this algorithm too. The following steps are performed:

1. For each map point $\boldsymbol{p}$ observed from the camera pose of K_i:
   a) Counters of shared observations $\{\theta_{ij} \mid j \in [0..M-1\}$ are initialized to zero.
   b) For each observation of $\boldsymbol{p}$ in the keyframe K_j the value of the counter θ_{ij} is increased by one.
2. Counters $\boldsymbol{\theta}_i$ are sorted in descending order and filtered by selecting those θ_{ij} with a significant number of observations, i. e., $\theta_{ij} > \theta_{min}$.
3. For each counter $\theta_{ij} \in \boldsymbol{\theta}_i$:
   a) An update notification is sent to every K_j. The keyframe K_j updates its connections starting from the step 2 of this procedure.
   b) The number of shared observations θ_{ij} is accumulated in the CDF of the current keyframe with $\phi_i = \phi_i + \theta_{ij}$.
4. A new connection between K_i and K_j will be created if one of the following conditions is true:
   a) The output degree of the keyframe $\deg^+(K_i)$ is lower than 5.

b) The output degree of the keyframe $\deg^+(K_i)$ is lower than the Maximum Vertex Degree (MVD) and the number of shared observations represented by the linked keyframes is still below 95 % of ϕ.

In addition to these steps, the keyframe K_j with the highest θ_{ij} is parented to the current keyframe K_i. This connection will be used later to propagate the accumulated error along the minimum spanning tree during a possible graph optimization. Such optimization only takes place when a visual loop is detected. Thus, the detection of loops impacts the accuracy of the final trajectory, as it is shown in Section 3.1.5.

Algorithm 1 Update the connections of keyframe K_i in the connectivity graph

```
1:  procedure UpdateConnections($K_i$)
2:      $\theta_i \leftarrow 0$
3:      $\phi_i \leftarrow 0$
4:      for each p in $K_i$ do
5:          for each o in p do
6:              $j \leftarrow \mathrm{KeyFrame}(o)$
7:              $\theta_{ij} \leftarrow \theta_{ij} + 1$
8:          $\phi_i \leftarrow \phi_i + \theta_{ij}$
9:      $\theta_i^* \leftarrow \{\mathrm{highest}(\theta_{ij}) \mid \theta_{ij} > \theta_{min}\}$
10:     $\theta_{acc} \leftarrow 0$
11:     for each $\theta_{ij}$ in $\theta_i^*$ do
12:         UpdateConnections($K_j$)
13:         if $\deg^+(K_i) < 5$ or $(5 < \deg^+(K_i) < MVD \;\&\; \theta_{acc} < 0.95\phi_i)$ then
14:             Connect($K_i, K_j$)
15:             $\theta_{acc} \leftarrow \theta_{acc} + \theta_{ij}$
```

Maximum Vertex Degree

The connectivity graph is used for searching visually similar keyframes in a local 3D region. Our main hypothesis is that the connectivity of the graph can be limited to the best keyframes. Then, we reduce the complexity of the graph while it preserves a valid topology, as explained in Section 3.1.4. A keyframe must be connected enough to maintain the topology of the connectivity graph. This requires a lower and upper bound for the output degree of a keyframe in Algorithm 1. On one hand, we have picked 5 as lower bound since a minimum number of connected keyframes is required for the system to operate. On the other hand, the upper bound of the output degree is restricted to 20. This upper bound has been determined by studying the experimental degree on the connectivity graph generated by ORB-SLAM and the theoretical bound for the natural neighbors of any 3D vertex.

Firstly, the topology of the graphs created by ORB-SLAM shows an average output degree of 40, ranging from 20 in the least connected sequences to 60 in those with a lot of redundant information. We hypothesized that we could dramatically reduce the number of connections while maintaining the most significant ones by keeping track of the most promising connections sorting the candidate connections according to the number of shared observations.

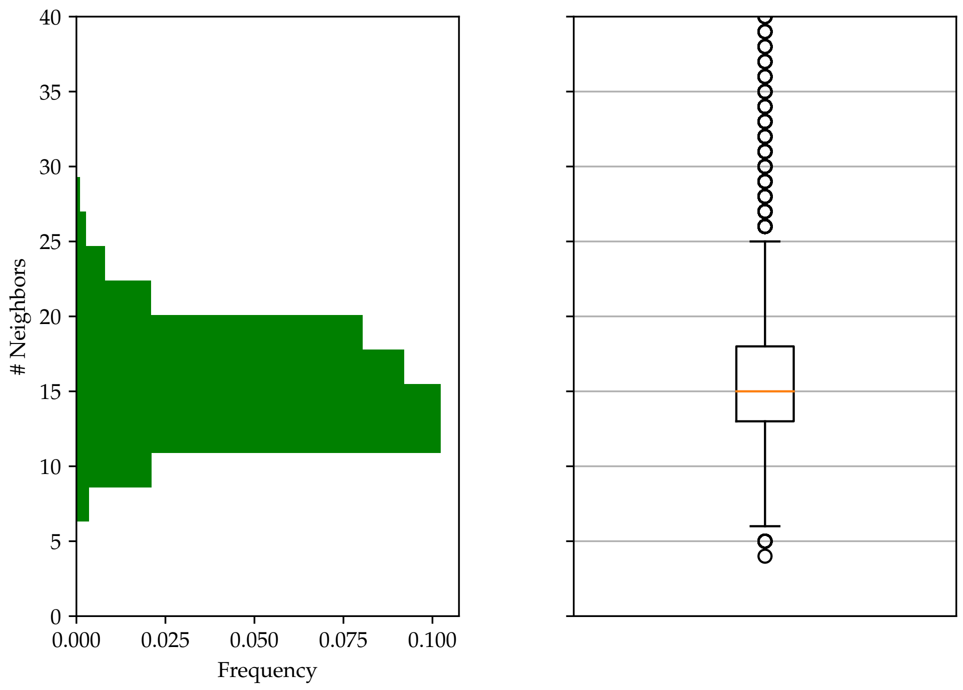


FIGURE 3.1: Distribution of the number of neighbors (i. e., connections) in the 3D Delaunay triangulation of one million random vertices which are uniformly distributed.

We have studied [5] the connectivity graph generated at the end of the sequence tracked by ORB-SLAM and guessed that many connections could be avoided without losing accuracy.

Secondly, each keyframe can be interpreted as a vertex in a 3D-space that originates a Voronoi cellular decomposition. Each keyframe is then a *natural neighbor* of all keyframes whose Voronoi cells share some side with it. In this natural context, we can evaluate how "well surrounded" a vertex is depending on the number of neighbors its cell has. If the original 3D-points are uniformly distributed at random, it has been proven that the average number of neighbors of a vertex is 15.5 (see [130]). This value is a good estimator of the minimum number of neighbors that a keyframe would require. We have also implemented [6] a simple program that calculates the 3D-Delaunay triangulation of a random set (dual to its Voronoi diagram) in order to better analyze the natural degree of a vertex.

The distribution of the vertex degrees is illustrated by the plots in Figure 3.1. Our experimental results agree on an average of 15.5 neighbors per vertex. They also show that the median is 15, that 50 % of the nodes have a degree between 13 and 18, and that 90 % of the nodes have less than 20 incident edges (i. e., 20 is the 90-percentile of the distribution).

5 The graph stats have been obtained with the Python module networkx https://networkx.github.io/

6 The Delaunay triangulation has been implemented using the package spatial https://docs.scipy.org/doc/scipy/reference/spatial.html of the scipy distribution, which provides spatial algorithms and data structures.

Vertex degree can also be studied from the point of view of Algebraic Topology. In this case, irregular tessellations can be viewed as PL-deformations of the spatial regular tessellations. Following Euclid, regular tessellations are given by tetrahedra, hexahedra (or their dual octahedra), and dodecahedra (or their dual icosahedra). In the presence of increasingly complex configurations, it is convenient to take into account the largest number of symmetries to obtain more stable and adaptive configurations. Among the regular configurations, the dodecahedron (20 vertices and 12 faces) or the icosahedron (12 vertices and 20 faces) tessellations show the largest number of symmetries.

Vertices of regular polyhedra provide a discretization of the 2-dimensional sphere $\mathbb{S}^2$ given as the boundary $\partial\mathbb{B}^3$ of the 3D ball. Any vertex $\mathbf{P}_0$ in the space can be taken as the center of a sphere connected to a finite set of surrounding vertices. Thus, in the most complex regular case, we constrain to local discrete neighborhoods with a maximum number of vertices N between 12 and 20 that surround the pivot point $\mathbf{P}_0$. The affixes of the connecting vectors provide a discrete representation of the Euler or radial vector field $\xi_E = x\partial_x + y\partial_y + z\partial_z$ whose integral curves are lines through the origin; this set is called a "star" in Geometry.

There is not a theoretical general principle for selecting the optimal N. In case of irregular behaviors along radial propagation, 20 vertices are recommended. In case of irregular behaviors along planar regions corresponding to wavefronts, 20 faces are preferred (corresponding to the icosahedron with 12 vertices).

The duality between propagation directions and wavefronts arises in any sort of propagation phenomena. Using a discrete representation from a geometric viewpoint, this duality is provided by the duality between vector fields and differential forms. For a 2D sphere, radial (Euler) vector field is the dual of the closed differential 2-form, which is locally described by

$$\sigma := \frac{1}{4\pi}(x dy \wedge dz - y dx \wedge dz + z dx \wedge dy)$$

on the 2-dimensional unit sphere $\mathbb{S}^2 := \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 = 1\}$. The normalization factor $1/4\pi$ is irrelevant; it guarantees that the area of the unit sphere is equal to 1

This 2-form can be interpreted as the contraction along the radial vector field ξ_E to the sphere of the trivial spatial orientation class $dx \wedge dy \wedge dz$. Its discretization supports inscribed triangles, which are the basic pieces of the icosahedron on the unit sphere; a discrete version for faces can be obtained by duality, where these faces correspond to the regular pentagons of the dodecahedron.

Neighboring cells share a face. In regular tessellations, it corresponds to the face of a regular polyhedron (i. e., regular triangles, squares or pentagons). In practice, these cells are deformed through a PL-map preserving incidence and adjacency relations. However, normal vectors to faces do not pass through the center of the original sphere wrapping the polyhedron. Furthermore, the vanishing of symmetries can cause the overlapping of vertices, which could collapse edges and/or faces (giving vanishing cycles in the Algebraic Topology framework). A local radial deformation with variable radius generates a totally ordered configuration of triangles with non-radial orientation. Despite the change in orientation and the relative arrangement between triangles, the topology remains the same.

As a consequence to the facts stated above, all patterns for generic configurations of points in a local neighborhood can be modeled by PL-deformations of the vertices of regular polyhedra with the maximum number of vertices or faces. This maximum number is 20 in both cases. Since this theoretical value also matches our experimental results, we will consider $MVD = 20$ as a fair upper bound for the number of connected keyframes.

3.1.5 *Loop detection*

One of the main uses of the connectivity graph in SLAM systems is to find loops in the trajectory. The presence of loops allows the system to optimize the trajectory beyond the local bundle adjustments performed in small regions. One of the main benefits of the connectivity graph is that it manages the spatial information about the sequence. This information enables spatial queries, such as the covisible keyframes of a given keyframe. In this way, the existence of a loop can be validated with the neighbors of the current keyframe.

A loop closure can be determined based on pure visual information, i. e., the image of the current keyframe is visually similar to the image of a previously seen keyframe. Other approaches use the estimated trajectory to detect the loops. However, the presence of camera drift in monocular SLAM makes this approach unreliable. In general, the detection of false positives should be avoided since it leads to increase the error of the trajectory. In KN-SLAM we have improved the hierarchical strategy implemented by ORB-SLAM. In this strategy, we use adaptive varying thresholds that depend on the results of the previous step.

The loop detection thread checks every new keyframe inserted in the queue of the mapping thread. When a new keyframe is inserted by the mapping thread, the loop detection thread performs the following steps to decide whether a loop has been created or not:

1. The bag-of-words similarity score between the current keyframe and a covisible keyframe is used to detect the closest keyframe, i. e., the keyframe with the minimum score.
2. The minimum score serves as the lower bound to query the keyframe database. A set of keyframes is returned as the loop candidates since they are visually similar to the current keyframe.
3. Each candidate is augmented with its covisible keyframes to create the candidate group. Then, the system search for a previously existent consistent group that shares at least one keyframe with the candidate group. If there exists such keyframe we say that both groups are consistent. In such case, the consistency of the group is increased by one to account for the number of consecutive keyframes in which the consistent loop has been found. If this consistency surpasses a threshold (currently 3), then the current keyframe is added to the set of enough consistent candidates. These candidates will be evaluated in the next steps. If the keyframe is

not consistent with any previous group, then it is added to the set of consistency groups with a consistency of 0. This group will be considered for the next keyframe.

4. The current keyframe is added to the database and the procedure continues only if there is at least one consistent candidate.

5. A set of matches is searched between each surviving candidate and the current keyframe. If the number of matches surpasses a threshold (around 20), then we find a $\mathrm{Sim}(3)$ transformation with a RANSAC scheme to handle the noise. The RANSAC solver will use 300 iterations with a minimum required number of 10 inliers. The iterations are performed in batches of 5 alternatively for each $\mathrm{Sim}(3)$ until one of them is successful or everyone fails.

6. Once the initial $\mathrm{Sim}(3)$ model is found, the set of initial matches is augmented with a guided matching between the candidate and the current keyframe. Then, the estimated $\mathrm{Sim}(3)$ is refined with the whole set of matches. A Levenberg scheme is used with the camera pose free and the map points fixed. The number of inliers supporting the estimated pose must be above a threshold again (around 20).

7. If the candidate passes every previous check, the estimated $\mathrm{Sim}(3)$ transformation is applied over the $\mathrm{Sim}(3)$ transformation of the candidate keyframe. A scale of 1.0 is used to convert from $\mathrm{SE}(3)$ to $\mathrm{Sim}(3)$. The resulting transformation will be used to distribute the error through the loop to the rest of keyframes.

8. Finally, map points seen from the loop keyframe and its covisibles are collected. Then, these points are used to increase the number of matches between map points and keypoints of the current keyframe. The search is guided by the $\mathrm{Sim}(3)$ transformation that projects the map points over the keyframe. If the number of matches is greater than a threshold (around 50) the loop is finally accepted. Otherwise, the candidates and the current keyframe are erased.

Over this pipeline, we have inserted our contributions to implement an adaptive threshold scheme. This decision has been taken after performing a detailed analysis of the output received at each one of the steps described above. Indeed, the range of values for the new thresholds has been chosen taking into consideration the evolution of these data in the loop detection thread across a wide range of sequences. More specifically, the following changes have been performed:

1. The minimum number of keypoints shared by the current keyframe and the loop candidate has been lowered from 20 to 15. We detected that many candidates were erroneously rejected in this step, so we decided to lower this threshold slightly. Most of the false negatives that pass this test are rejected in the next steps.

2. The minimum number of inliers to validate a $\mathrm{Sim}(3)$ transformation in a RANSAC scheme has been modified from 20 to $\mathrm{max}(10, \lambda/3)$, where λ

represents the number of 2D-3D matches. The use of $\theta/3$ as an estimator for the threshold avoids many false positives in candidates with a big number of shared observations. We clamp the value to 10 since it is the minimum value to distinguish something meaningful from noisy measurements.

3. The minimum number of inliers in the LM optimization that adjusts the $\mathrm{Sim}(3)$ transformation has been approximated using $\max(10, \beta/2)$, where β represents the number of matches found in the search guided by the initial $\mathrm{Sim}(3)$ transformation. Our guess is that the matches rejected by the optimization represent between 20 %-30 % of the original matches. Thus, a threshold around 50 % is reasonable. As in the previous threshold, we clamp the value to 10, the minimum number of matches to consider the model as meaningful.

4. The minimum number of map points to validate the final $\mathrm{Sim}(3)$ alignment, i. e., the product of the estimated $\mathrm{Sim}(3)$ transformation and the previous pose of the current keyframe, has been increased from 40 to 50. This set of points also includes the points of the neighboring keyframes of the current keyframe so the size has been significantly increased. Indeed, we have checked in our experiments that this counter is beyond 100 in 90 % of the sequences. However, to support the most challenging sequences we have lowered the threshold to the half.

A loop is considered as detected when the current keyframe passes all the tests. Then, the $\mathrm{Sim}(3)$ alignment hypothesis is considered as valid. Once the loop is detected, the loop detection thread corrects the trajectory drift by propagating the error along the poses of the keyframes. This involves the correction of the position of the map points, the fusion of many map point duplicates, the update of the connectivity graph, the optimization of the essential graph and, finally, the refinement of the poses and map points at the same time by means of a global bundle adjustment. We use a fixed number of 20 iterations for this final adjustment. Even though the increase in computing time does not worth the gain in accuracy, we have detected that this increment in time does not affect the overall performance of the SLAM system.

Our improvements to the loop detection procedure allow the system to detect more loops with fewer matches, without increasing the number of false positives. It must be taken into account that these contributions benefit from the improvements in the connectivity graph detailed in Section 3.1.4. A less connected graph simplifies the structure of the consistent groups and it makes easy to find candidates. In this way, KN-SLAM is more robust to situations in which the number of consistent candidates is limited, like in the ninth sequence of the KITTI benchmark `kitti/09`. This is an especially challenging sequence with a loop between at the end of the sequences, just when the system stops the tracking. In our experiments KN-SLAM detects the final loop in 100 % of the executions, whereas ORB-SLAM only detects that loop in 20 % of the executions.

3.2 METHODOLOGY

Our research has been conducted following a fully experimental methodology, in which we have compared the obtained results w. r. t. ORB-SLAM in terms of the robustness and the accuracy. Besides these metrics, we have evaluated the impact of our approach in terms of the number of keyframes and edges in the connectivity graph.

The experiments have been carried out in four different datasets with more than 45 sequences and different challenging situations for monocular SLAM. To the best of our knowledge this is the first VO or SLAM approach that compares the most widely used datasets. We have developed a unifying framework for testing the sequences of input images coming from each benchmark. This task has required to normalize file formats and ground truth between the different datasets for adopting the same format. As a result, the same set of tools and a single executable can be used to evaluate any sequence.

The rest of the section is organized as follows: Section 3.2.1 describes the traits and origin of the datasets included in our evaluation; Section 3.2.2 enumerates the challenging characteristics to be considered for each sequence; Section 3.2.3 provides the normalization procedure to unify the format of the files and data for all sequences of every dataset; and Section 3.2.4 defines the evaluation metrics to measure the performance and to compare the evaluated SLAM systems.

3.2.1 *Datasets and benchmarks*

Our goal is to evaluate the robustness and accuracy of SLAM for a wide range of paths, scenes and cameras. To accomplish this goal, we have gathered the sequences provided by publicly available datasets and benchmarks. We have to restrict the evaluation to those sequences with an associated ground truth of the trajectory. This ground truth is considered reliable since it has been captured using a high-precision device that determines the pose of the camera at a high frame rate usually around 100 FPS).

Each sequence in a dataset is intended to introduce a different level of complexity to test SLAM algorithms. Such complexity can be described in terms of a collection of challenges that make it difficult to achieve an accurate and robust tracking. Each challenging characteristic of a sequence can be represented by a label. These characteristics are gathered in Table A.1 and Table A.2. By considering these characteristics of the sequences in our evaluation we are able to understand the differences between the evaluated SLAM approaches. In this way, a classification can be created taking into account where a system outperforms another system in a new sequence.

Moreover, each dataset has been recorded using different types of cameras equipped with different lenses. The main traits of all datasets are summarized in Table 3.1. The diversity of settings and challenges posed by these sequences enriches our evaluation.

DATASET	FEATURES	SHUTTER	RESOLUTION	FPS
KITTI	Outdoor Car-mounted High-resolution Low framerate	Global	1241×376	10
EuRoC	Indoor MAV-mounted Fast movements Low lighting	Global	752×480	20
TUM RGB-D	Indoor Hand-held Shaking camera Dynamic scenes	Rolling	640×480	30
ICL-NUIM	Indoor Raytraced with noise High-frequency textures	Global	640×480	30

TABLE 3.1: Traits of the datasets included in our evaluation

KITTI

The KITTI Vision Benchmark Suite [44] is composed of multiple sequences to evaluate algorithms in different computer vision tasks applied to autonomous navigation, such as stereo matching, SLAM, semantic object recognition, etc. The benchmark was created in 2012 in the scope of a project developed by a partnership between the Karlsruhe Institute of Technology and the Toyota Technological Institute at Chicago. The goal of the benchmark is to reduce the bias between conventional datasets and real-world scenarios by providing novel difficulties to the community. It is the most widely used dataset for evaluating algorithms in autonomous navigation tasks.

The whole suite is composed of several annotated benchmarks to evaluate algorithms in several Computer Vision tasks. Its authors also provide the metrics, the scripts and the website to evaluate and compare the results achieved by these algorithms. The odometry benchmark [7] is intended to evaluate monocular or stereo VO, laser-based SLAM or even more advanced algorithms that combine visual and Light Detection and Ranging (LIDAR) information. It is composed of 22 stereo sequences saved in Portable Network Graphics (PNG) format. However, the ground truth is only available for the first 11 sequences, which were originally intended for training purposes. Only these sequences are included in our evaluation. In the rest of the document and for the sake of simplicity, we will refer to the odometry benchmark as the KITTI dataset.

The sequences were recorded with standard station wagon equipped with two high-resolution color and grayscale video cameras. The datasets were

7 http://www.cvlibs.net/datasets/kitti/eval_odometry.php

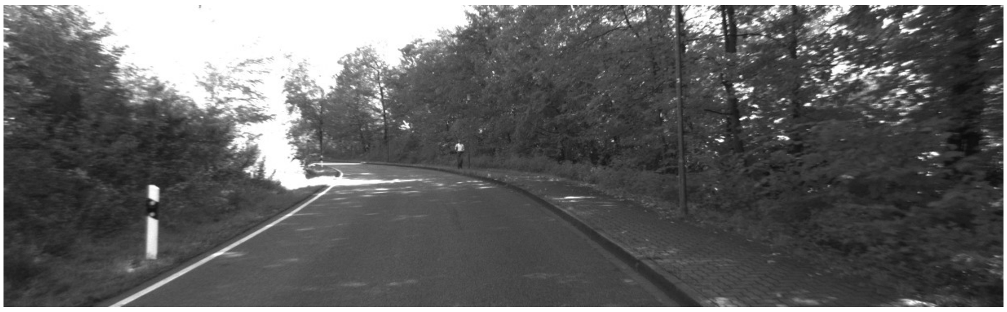


FIGURE 3.2: A sample frame of the sequence `kitti/09` in the KITTI benchmark.

captured by driving around the mid-size city of Karlsruhe, in rural areas and highways. Each video camera captured frames with a 1241 × 376 resolution equipped with a global shutter (see Figure 3.2). The ground truth of the trajectory was recorded with a GPS device and a Velodyne laser scanner.

The sequences of this dataset are characterized by long trajectories in scenes of high dimensions. In the sequences, there is a predominance of translations towards the vanishing point of the scene against rotation. Moreover, some dynamic objects, such as pedestrians and cars, move towards the camera occasionally. The length of the trajectory makes these sequences especially challenging for monocular SLAM due to the accumulated drift along the trajectory. Besides, relocalization procedures are examined to deal with moving objects in front of the camera. On the other side, the aspect ratio of the images (width is twice greater than height) eases the tracking of the camera pose (see an example in Figure 3.2).

TUM-RGB

The TUM RGB-D dataset [121] includes a set of hand-held indoor sequences recorded at full frame rate (30 FPS) with a Microsoft Kinect[8]. Sensor resolution of RGB images is 640 × 480 pixels. The color and depth images are provided along the highly accurate and time-synchronized ground truth camera poses. This ground truth was captured from a motion-capture system with eight high-speed tracking Raptor-E cameras running at 100 Hz.

The low-cost nature of the cameras used for collecting data introduces the typical rolling shutter artifacts in the frames of the sequence, such as motion blur and spatio-temporal aliasing (see for example Figure 3.3). This is the only dataset included in our evaluation that uses a camera with this kind of shutter, which is especially challenging for direct VO approaches.

The dataset was recorded in 2011 by the Computer Vision Group of the Technical University of Munich. It was published under a Creative Commons 4.0 Attribution License (CC BY 4.0) license with a total number of 39 sequences, which were recorded in two different indoor environments:

- a typical office environment with a 6 × 6 area, as depicted in Figure 3.3, whose sequences are labelled starting with `fr1`;
- a large industrial hall with a 10 × 12 area, whose sequences are labelled starting with `fr2`.

8 https://vision.in.tum.de/data/datasets/rgbd-dataset

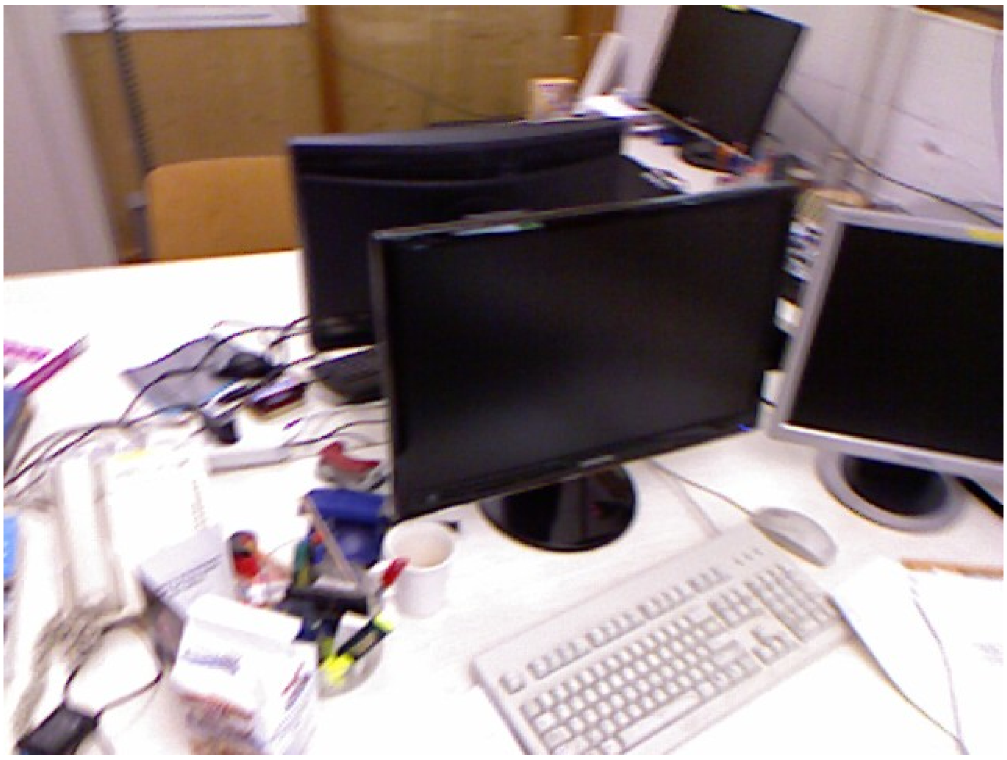


FIGURE 3.3: A frame for the sequence `tum/fr1_desk` in the TUM RGB-D dataset. Note the motion blur generated by the combination of a fast camera movement and the rolling shutter of the camera.

The dataset covers a large variety of scenes and camera motions, including sequences for debugging with slow motions as well as longer trajectories with and without loop closures. The authors also provide source code for automatic evaluation of drift of VO systems and the global pose error of SLAM systems. Indeed, this is the code and file format to which we have adhered for developing our custom scripts mentioned in Section 3.2.

Later, the dataset was expanded with a third group of sequences identified by a title starting with `fr3`. These sequences were recorded with an Asus Xtion camera in two different environments: a piecewise planar scene textured with posters and a dynamic scene moving the camera around a desktop in an industrial facility. The specifications of the Asus Xtion show no differences w. r. t. Kinect, at least regarding the characteristics of the lenses. This new group is intended to evaluate more specific challenges for SLAM algorithms. Three new categories were created to include these sequences: structured vs. textured scenes, dynamic objects, and 3D reconstruction.

This dataset has been thoroughly evaluated by the community in several VO and SLAM algorithms (e. g.,, see [88, 90]). The sequences are grouped according to the challenging feature that they expect to evaluate: calibration, debugging, handheld SLAM and robot SLAM. These features are properly represented in the sequence characterization of Table A.2 and Table A.1. Our evaluation includes only the 15 sequences that we considered more suitable for monocular SLAM. We have also added some other interesting sequences not evaluated in [88].

European Robotics Challenges Micro Aerial Vehicle

The European Robotics Challenges MAV dataset [13] is a recently published dataset (2015) composed of 11 stereo-inertial sequences with a total length of 19 minutes of video organized in two batches[9]. The dataset was collected with a set of devices placed on-board of a MAV. Each sequence includes pairs of synchronized stereo images, the IMU measurements and the corresponding ground truth with a 6 DoF pose for each frame. Each batch was recorded in a different scene with a different purpose. Different devices were used to determine the ground truth of the sequences, providing even a 3D scan of the environment. All the information is spatio-temporally aligned. The calibration sequences are available too. They are used for computing the extrinsic and intrinsic parameters of the sensors, which are also provided for convenience.

The first batch of sequences, whose labels start with `MH`, is intended to evaluate visual-inertial odometry algorithms on real flight data. It was recorded in a machine hall, where the ground truth was captured by a laser tracking system Leica MS50 with millimeter accurate position. The second batch, whose sequences are labeled starting with `V`, is aimed at 3D reconstruction tasks. It was recorded in a room equipped with a Vicon motion capture system to determine the ground truth of the trajectory. The same motion capture system was also used to record the ground truth of the TUM RGB-D. This batch has two groups of sequences according to the two different room setups in which they were recorded. The sequences are labeled starting with `V1` or `V2`.

Inside each batch, the type of sequences ranges from slow flights under good visual conditions to dynamic flights with motion blur and poor illumination. This enables a thorough evaluation of the SLAM approaches.

We have included the whole video sequence is our experiments. This includes the shaky movements of the camera at the beginning of the sequence to initialize the IMU of the MAV. Since we are evaluating only monocular SLAM approaches, only the frames from the first camera of the stereo pair have been used.

Imperial College London and National University of Ireland Maynooth

The Imperial College London and National University of Ireland Maynooth dataset [51] is composed of a collection of eight sequences recorded with a handheld RGB-D camera. These sequences have been generated by rendering the scene viewed from a virtual camera along different camera trajectories. Images are ray-traced using POVRay[10] based on the ground truth acquired by Kintinous in real scenes [132]. Even though the dataset was originally aimed at evaluating the accuracy of surface reconstruction algorithms, it can also be used to evaluate the trajectory estimated by SLAM algorithms, which is our main purpose.

The eight ray-traced sequences have a total video length of 4.50 min[11]. The simulated camera has an image resolution of 640×480 pixels, a FOV of 90° and a focal length of 480 pixels. One of the main traits of this dataset is that the camera is inverted w. r. t. to the Y-axis. Instead of changing every camera

9 https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets
10 http://www.povray.org
11 https://www.doc.ic.ac.uk/~ahanda/VaFRIC/iclnuim.html

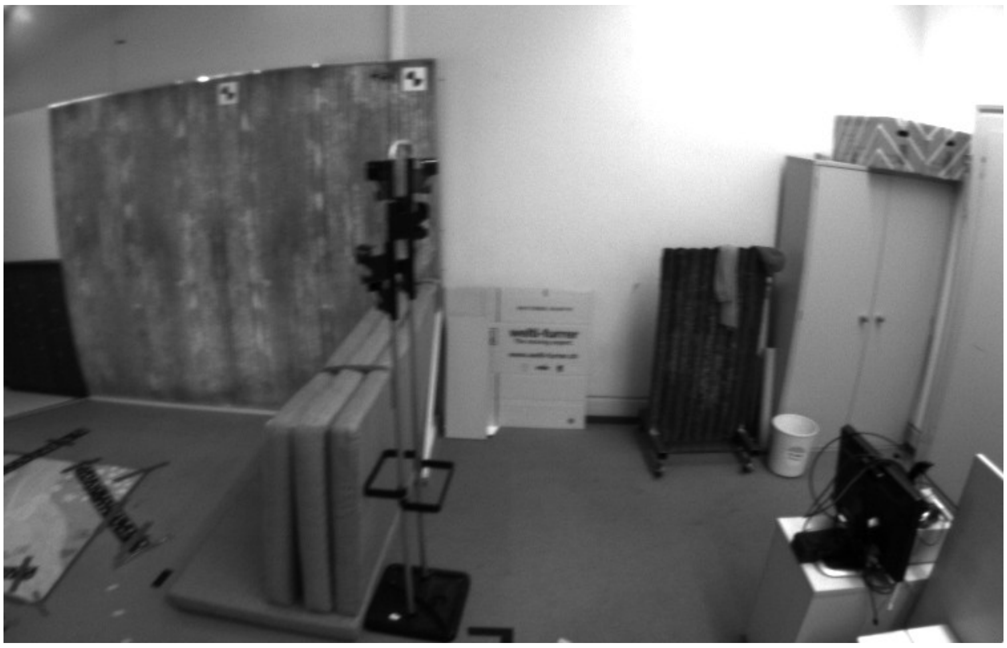


FIGURE 3.4: A frame of the sequence `euroc/V103` in the EuRoC dataset. It has been acquired by a MAV travelling around a room at 20 FPS.

pose in the ground truth, we have modeled such difference with a negative focal length in the intrinsic parameters of the camera.

As usual, the dataset provides the ground truth with the camera poses recorded by Kintinous. Each sequence also has a ground truth surface model to evaluate quantitatively the accuracy in the reconstructed surface of the scene. Images are rendered in a realistic way since they also simulate real-world artifacts like sensor noise. This noise is modeled using a linear CRF obtained by taking real images with different exposure times [48]. This CRF maps the irradiance perceived by the camera to the digital n-bit brightness values stored in the digital image. It provides an analytical model of the noise levels for pixels to determine some noise parameters. These parameters are used to simulate synthetic noise in the rendered image.

There are two different indoor environments: a living room and an office room, with four sequences for each environment. These sequences are labelled as `lr_kt*` for the living room, and as `of_kt*` for the office room, where $*$ ranges from 0 to 3. In order to ensure that images look as photo-realistic as possible, the rendering process includes real-world lighting phenomena, e. g., specular reflections, shadows and color bleeding. These artifacts greatly affect direct SLAM approaches but they should be handled by indirect methods like KN-SLAM and ORB-SLAM.

The sequences of the living room aim at evaluating three-dimensional reconstruction accuracy, whereas the sequences of the office room are intended to evaluate VO. This is due to the fact that the office scene is rendered procedurally without an explicit 3D model. However, both groups of sequences are included in our evaluation to evaluate the robustness of our system when dealing with challenging characteristics, such as fast camera motion, a strong rotation, and high-frequency textures. A single of the first sequence in the living room group (`icl/lr_kt0`) can be seen in Figure 3.5.

FIGURE 3.5: A single frame extracted from the sequence `icl/lr_kt0` in the ICL-NUIM dataset. These frames have been synthetically generated by POVRay at a simulated frequency of 30 Hz.

3.2.2 *Sequence characterization*

A fair comparison between different SLAM approaches must take into account the features, characteristics or traits that define each sequence. Some of these characteristics (e. g., dimensions, length, average speed, etc.) are defined in quantitative terms, but the characteristics describing challenges are purely qualitative. Thus, they must be qualitatively assessed from the point of view of an expert with experience and knowledge in SLAM.

Our aim is to determine what sequences are more suitable for KN-SLAM, i. e., in which sequences KN-SLAM outperforms ORB-SLAM. The characteristics of a sequence provide the criteria to analyze our results. Indeed, the heterogeneity of the achieved accuracy in different sequences can be explained principally by these characteristics, although we are conscious that they are not the unique factor to consider. In our evaluation we describe a sequence by the set of characteristics that make that sequence challenging:

- fast movements,
- shaking movements,
- close range,
- rotational movements,
- translational movements,
- dynamic events,

- visual loops,
- high-frequency textures, and
- lighting.

Each characteristic poses a different challenge for VO and SLAM algorithms, making an accurate and robust tracking more difficult to achieve [14]. Usually, a sequence introduces a subset of these challenges. We will say that a sequence has been recorded in *natural conditions* if none of these challenges has been purposefully added to the sequence. Even though SLAM algorithms should be able to deal with these special situations, they are actually uncommon in real-world scenarios.

Next paragraphs dive into a more detailed description of these challenges, providing the criteria to quantitatively assess their relevance in a video sequence.

FAST MOVEMENTS Passive sensing is greatly affected by image artifacts like blur and noise (see e. g., [34]). These artifacts are especially common in cameras equipped with a rolling shutter when the camera is moving faster than the readout time of the sensor. Thus, fast movements are challenging for tracking pixels between consecutive frames, especially for direct methods. Indirect methods are able to deal with blur as long as the borders in the image are clear.

Usually, the best approach to deal with these issues is losing the tracking and waiting until a non-blurred frame arrives. Then, this new frame is compared w. r. t. the last keyframe and if enough matches are found, the pose is estimated w. r. t. the existent keyframe. Actually, fast movements have a twofold influence in tracking. On one hand, blur and distortions remove many keypoints of the image. On the other hand, wide baselines between consecutive frames difficult the match between keypoints and map points.

In the assessment of this characteristic, we give a valuation according to the frequency of fast camera movements in the sequence. For example, if a fast camera movement occurs every 100 frames or less, the characteristic will have the highest possible value. Otherwise, if there are no fast movements, the characteristic will take the lowest value.

SHAKING MOVEMENTS This is a challenge similar to fast camera motion. Unlike fast movements, the camera pose after a shaking movement does not change, i. e., the final and the starting pose are the same. In some sequences, shaking can be useful for bootstrapping the system, but it is dangerous since the map optimization can degenerate in local optima. There are good examples of this challenge in the sequences of the EuRoC dataset, where the shaking is used to initialize the IMU of the MAV [13]).

Camera poses of the shaking frames are prone to errors since a lot of correspondences between keypoints and map points are lost. Moreover, the map degrades since bundle adjustment performs the optimizations within a local area with the points in the center of the frame. The situation is especially more difficult to handle in combination with close range.

Following the same criteria than with the characteristic for fast movements, we have assessed the shaking movements according to their frequency in the sequence. If camera shakes around once every 25 frames, the highest value is assigned. For instance, this happens with the sequences of the TUM RGB-D dataset. Then, the value diminishes proportionally to the number of occurrences. Obviously, the lowest valuation is reserved for sequences without shaking like in the ICL-NUIM dataset.

CLOSE RANGE This characteristic refers to the distance between the objects of the scene and the camera. In general, the wider the FOV of a lens, the greater the accuracy achieved with VO. This is caused by the fact that a higher degree of overlap between consecutive frames increases the number of correspondences in indirect methods. Moreover, a longer range usually increases the variance in the depth of the map, which increases the robustness of the tracking. In general, an overlap between consecutive frames below 90 % of the area of the image is prone to tracking loss.

A robust SLAM system must be able to run with lenses with a conventional FOV around 60°, which is the usual value for a commodity camera. Direct SLAM methods require higher angles such as those delivered by fish-eye lenses (see e. g., [34])). Sequences closer to the object plane are more challenging to track since the shared image area between consecutive frames is vastly reduced. On the contrary, sequences with a higher range are easier to track since many frames provide redundant information to iteratively minimize the error of the geometry.

In our assessment, we have reserved the highest values for sequences where the camera is placed near the object and there are no points in the horizon. Lowest values are attached to sequences where the background fills a significant portion of the image or where there are vanishing points, such as the sequence of the KITTI dataset. A combination of close-range long-range images is represented with a middle value.

TRANSLATIONAL OR ROTATIONAL MOVEMENTS The type of camera movement must be taken into account to evaluate the complexity of a sequence. We have defined two characteristics to represent the relevance of these movements in a sequence: translation and rotation. The value for each characteristic has been assigned taking into consideration relative transformations between consecutive poses. The predominance of the rotational or the translational component in these transformations determines the valuation of the corresponding characteristics. For instance, a sequence where the camera is moved with pure rotations along the optic axis has the highest value for rotation and the lowest value for translation. On the contrary, a sequence with a pure translational movement has the lowest value for rotation but the highest value for translation. If camera movements along the trajectory combine rotation and translation, both characteristics are valued in the middle of the range.

In real life operation, any sequence presents a combination of both. Pure translational movements are easy to track, especially in piece-wise planar scenes, as it was already stated by Faugeras and Lustman [38]. On the contrary, pure rotational movements are difficult to track since short baselines between two consecutive frames make the triangulation unfeasible. This is partially mitigated thanks to a combination of a generous keyframe spawning policy and a pose optimization with a motion-only BA on $SE(3)$. A combination of rotational and translational movements makes the sequence a fair trade-off for most of the SLAM systems.

DYNAMIC EVENTS There are a big number of unexpected and uncontrolled situations when a camera travels freely in a scene. These situations include abrupt movements, camera occlusions, failure in the delivery of the frame, moving objects, etc. Not all SLAM systems are able to deal with sequences with dynamic events, i. e., events that force the system to determine the camera pose without a prior guess. When the object is placed between the scene and camera, the overlap between consecutive frames is reduced. This usually leads to losing the camera tracking, depending on the size of the occluded area.

Whatever the cause, the loss of the camera pose takes the system to a new state. Usual strategies consist in launching a relocalization procedure that seeks a visually similar keyframe in the trajectory. Until this frame is found, the input frames are discarded so the estimated trajectory will have fewer keyframes. Although theoretically the accuracy of the estimated trajectory should not be affected, it is affected in practice. The main reason is that the PnP alignment algorithm, which estimates the camera pose after a relocalization, is not as accurate as the estimator based on a constant linear velocity model. In summary, the presence of dynamic events in a sequences leads to a relocalization, which makes the sequence more challenging in terms of accuracy and robustness.

This characteristic is difficult to assess since it comprises multiples challenging situations that lead to the same outcome: a relocalization. Our criterium is to assign the highest value to the sequences where dynamic actors play an important role, i. e., they are present in many frames and they occupy a significant region of the frame. Moreover, we consider also the speed of the actor such that the characteristic is valued lower if actors are moving slowly. In addition, we use high values for special situations where some frames are lost due to complete camera occlusion or communication failures. The lowest value is reserved for static sequences.

VISUAL LOOPS A loop in a trajectory is given by a closed simple path which is topologically equivalent to a circle; it appears when a keyframe matches another keyframe. In the case of visual loops, the comparison between keyframes is made using their corresponding images. Even though their detection represents a challenge for the system, their existence in a sequence must be viewed as something positive since they improve the overall accuracy of the trajectory.

Due to the purely projective nature of a single camera, monocular SLAM approaches are only able to estimate the geometry of the scene up to scale. In fact, the scale of the trajectory is estimated using the baseline between the pair of keyframes from which the system bootstraps. The main issue is that this scale is liable to drift over time. Conventional solutions to this issue are based on prior assumptions about the inter-frame distance or the well-known dimensions of some object in the scene (see e. g., [47]). Without this prior knowledge, the best strategy to correct the scale drift consists in closing the detected loops in the trajectory [119].

The loop detection thread of the system seeks the first and last keyframes of a loop. Then, the accumulated error is distributed along the trajectory. The efficacy of this module is critical to reducing the error in long trajectories, such as in the sequences of the KITTI dataset. Obviously, the system is unable to estimate this drift in open trajectories so these trajectories will exhibit a worse accuracy. It should be noted that although the drift of the pose is corrected thanks to the loop closure, monocular SLAM is still unable to determine the scale of the trajectory. This is considered when poses are compared with the ground truth since they are aligned with a $\mathtt{Sim}(3)$ transformation (see Section 3.2.4).

Some sequences have several local loops (see e. g., EuRoC) whereas other sequences have few loops but very significant (see e. g., `kitii/09`). Such sequences have the highest values for this characteristic since what we are assessing is the relevance of the loop in the accuracy of the trajectory. Sequences whose loops have a low number of keyframes are valued lower since these loops are irrelevant for the final accuracy. Finally, sequences with open trajectories have the lowest possible value.

high-frequency textures Sequences composed of objects with high-frequency textures, such as asphalt, grass, marble, etc, are especially challenging for indirect SLAM methods. The abstraction of pixels into geometric points fails when the keypoints cannot be detected with enough confidence. The system should fall back to a direct approach when a high-frequency texture is being tracked. This is the base principle of hybrid SLAM. However, the most probable outcome would be the loss of tracking due to a degenerate map, followed by the execution of the relocalization procedure. This trait makes the sequence more difficult to bootstrap and tracks for the system, as it can be seen in the sequences of the ICL-NUIM dataset.

We have assessed this characteristic according to the percentage of frames with a high-frequency texture. Thus, a high value is assigned to sequences where the presence of these textures is remarkable, i. e., they occupy a significant region of the image in most of the frames. On the contrary, if every frame views objects with a rich texture, the characteristic will have a low value.

lighting Passive sensing devices only rely on the light intensity received by the sensor of the camera. This is a significant limitation of passive sensing w. r. t. active sensing. Keypoints are more difficult to detect and track in the absence of the appropriate lighting. This issue affects both indirect and direct approaches. However, indirect approaches like ORB-SLAM and KN-SLAM handle these situations thanks to the abstraction of the image. Direct methods are more sensitive to illumination changes while keypoints only requires contrast between regions of the image to work.

Even though indirect approaches are more robust to illumination changes, if the lightning of the environment is reduced too much then the accuracy of the trajectory will be reduced. This is due to the absence of keypoints, which makes impossible to estimate the relative transformation between keyframes. Therefore, the effect is equivalent to the presence of high-frequency textures. Only the most difficult sequences of the EuRoC dataset introduce this issue when some parts of the sequences are recorded with the lights off.

The values of this characteristic have been assigned taking into account the length of the affected portion of the sequence. If more than half of the sequence is affected, the highest value is assigned. Otherwise, if the whole sequence is well illuminated, then it takes the lowest value.

Assessment of the characteristics

The characteristics described in Section 3.2.2 must be assessed thoroughly for each sequence. To the best of our knowledge, there not exists a characterization of this kind in the state of the art. Thus, in the absence of a previous work we have decided to tackle this problem in an agnostic way, i. e., without linking our assessment to any specific SLAM system or approach.

The assessment has been conducted through a detailed manual revision of each sequence involved in our experiments to determine the most appropriate value for its characteristics. The valuation has been decided by visually inspecting the frames of the sequence. The concrete values have been chosen using our own personal knowledge and experience following the same criteria along with all datasets. This characterization is provided as an additional contribution of our research in the hope that it can boost future developments in more robust SLAM approaches.

The relevance of a characteristic in a sequence is assessed using a Likert scale with five levels ranging from 1 to 5. In general, the level 1 represents the absence of the challenge in the sequence, i. e., the accuracy of the trajectory will not be affected. On the contrary, the level 5 means a strong influence of the challenge in the accuracy of the trajectory, i. e., if the challenge were not correctly handled, the final accuracy would be greatly diminished. We choose a scheme with five ordered response levels since it is proven that they may produce slightly higher mean scores relative to the highest possible attainable score [26]. Initially, we use a binary scale (0/1) but we realized that this scale was not discriminative enough to represent subtle aspects required to discriminate characteristics like rotational and translational movements. Section 3.2.2 provides more details about the meaning of a high/low score in each characteristic.

The valuation of each sequence allows us to analyze which characteristics make a sequence more suitable for KN-SLAM or ORB-SLAM. With this valuation, we are able to determine in which kind of sequence our approach is more accurate. The characteristics of the sequences in which KN-SLAM outperforms ORB-SLAM are assessed in Table A.1. On the contrary, the assessment of the sequences where ORB-SLAM is more accurate than KN-SLAM is provided in Table A.2. The sequences have been ordered by the value obtained from the SVM classifier (see Section 3.3 for further details).

3.2.3 *Dataset normalization*

The datasets included in our evaluation use different data structures and file formats for representing their sequences. This data must be homogenized by using the same representation conventions that simplifies posterior evaluations. For instance, the rotation of the camera pose is usually represented with quaternions using the conventional notation (w, x, y, z). Nevertheless, some authors and datasets prefer the notation (x, y, z, w). In addition, some reference frame transformations must be applied when the camera pose is living in a different coordinate system than the ground truth is expressed. For example, EuRoC measures the ground truth w. r. t. to a marker at the top of the MAV but the camera is slightly displaced and the Y-axis is inverted.

The homogenization of the sequences simplifies their evaluation across datasets and SLAM systems. In such case, the same benchmark could be applied, reducing the number of errors and misconceptions when the results are compared. This would also help to compare some state-of-the-art results that are difficult to reproduce because of the differences in the underlying computer architecture. Due to its simplicity and wide adoption by the community, we have chosen the file formats[12] and data structures proposed by the TUM RGB-D benchmark [121]. In this benchmark, each sequence is represented by the following set of mandatory files and folders:

RGB/ The directory with the colored or grayscaled images in a lossless compressed format like PNG. In the case of RGB images, they use 3 channels and 8 bits per channel. If the images are represented in grayscale, only 1 channel is required. Usually, the filename of the frame matches its corresponding timestamp in the ground truth, although this is not mandatory.

DEPTH/ This folder contains the depth images of the sequence. Only RGB-D or ToF cameras provide these images. If they are provided, each image is stored in PNG format with a single channel of 16 bits. Each pixel of the image represents the distance-to-camera. It is stored in meters scaled by a factor of 5000.

RGB.TXT The text file that includes the list of `(timestamp, filename)` associations. Each correspondence is represented in a separated line where the timestamp of the frame in the ground truth is linked to the image of the frame in the `rgb` folder.

DEPTH.TXT The text file that associates the depth image of a frame in the `depth` folder with its corresponding time-aligned timestamp in the ground truth. It follows the same conventions than `rgb.txt`.

GROUNDTRUTH.TXT The text file that contains the list of poses of the ground truth trajectory. Each pose is associated to a timestamp placed at the beginning of the line. Such pose is described in terms of a 3D translation vector x, y, z and a unit 4D quaternion x, y, z, w. Each line uses the format `(timestamp tx ty tz qx qy qz qw)`. The image corresponding to the pose is saved in the file `rgb.txt` using the same timestamp.

12 https://vision.in.tum.de/data/datasets/rgbd-dataset/file_formats

The homogenization task would require a huge amount of work if it were tackled from scratch for each dataset. Fortunately, every dataset usually provides a toolbox to manipulate its data. When these toolboxes are not released by the publisher of the dataset, they are provided by third-party developers, like the `pykitti` package[13] that handles the sequences of the KITTI benchmark. Taking these tools as starting point, we have developed more advanced scripts to implement a custom pipeline that automatically downloads, parses and converts the sequences from the original format of each dataset to the TUM RGB-D format. We have released these scripts as open source code for the benefit of the community[14].

Several adjustments have been applied to the original datasets in order to homogenize them for our monocular SLAM evaluation. These modifications are disseminated across different scripts of our toolbox. Among the major changes accomplished, we highlight the following:

- The ground truth provided by KITTI represents rotations in camera pose with a 3×3 matrix in $SO(3)$. As we have mentioned above, the TUM RGB-D format requires quaternions so a conversion between quaternions and 3×3 matrices is required[15]. We can switch between matrix and quaternion representations of the same rotation. However, two quaternions are possible for the same rotation matrix since rotations do not uniquely specify quaternions but quaternions uniquely specify rotations. So we choose the best quaternion in terms of smoothness.

- The `rgb.txt` file for KITTI sequences has been built from the contents of the `times.txt` files provided by the dataset.

- In the sequences of EuRoC the association between timestamps and images of a sequence is in the file `mav0/cam0/data.csv`. However, EuRoC timestamps are represented in nanoseconds and image filenames are not relative to the folder containing the sequence. The former is solved by dividing by 1×10^9 and the latter by adding the folder name at the beginning of the path of the image.

- The ground truth in the EuRoC sequences is stored in the file `mav0/state_groundtruth_estimate0/data.csv`. It includes raw sensor information in the Autonomous Systems Lab (ASL) format that must be converted to the TUM RGB-D format. This requires 1) parse and extract the translation and rotation of each timestamp, 2) change the timestamp to seconds, 3) apply to each pose the transformation defined by the extrinsic calibration of the camera sensor `cam0` in file `mav0/cam0/sensor.yaml`, 4) write the resulting poses in the `groundtruth.txt` file.

- The ICL-NUIM dataset uses a camera formulation whose Y axis is inverted. Instead or rotating every ground truth pose, we have defined a negative focal length f_v on the Y axis in the intrinsic calibration of the camera, as the authors explicitly recommend [16].

---

13 https://github.com/utiasSTARS/pykitti
14 https://gitlab.com/fradelg/tum-toolbox
15 Quaternions are managed with the numpy-quaternion module at https://github.com/moble/quaternion.git
16 https://www.doc.ic.ac.uk/~ahanda/VaFRIC/codes.html

- The ICL-NUIM benchmark requires a set of timestamps that simulates a frame rate of 30 Hz, i. e., a temporal separation of 33 ms between consecutive frames. This change affects both the `(timestamp, frame)` associations in `rgb.txt` and the ground truth in `groundtruth.txt`. Although the original association includes a number for each frame, this is just an integer representing the order of the image. However, it does not correspond to the time at which the frame was captured. These timestamps are required to perform a real-time simulation of the sequence.

These modifications have been applied to the datasets provided by their original publishers. We have also released[17] our converted datasets to support the research in more robust monocular SLAM approaches. This will allow future evaluations to focus on their contributions and not on these cumbersome normalization tasks. Original licenses are preserved in any case.

3.2.4 *Metrics*

Previous sections have described the datasets, its characteristics and the procedure to make uniform their data. But we still need to define how results can be quantitatively compared. Our evaluation is intended to consider both the accuracy and the robustness of the system. A sequence is labeled as an outlier if any SLAM system does not track enough frames in such sequences or does not bootstrap. Such sequences are discarded from the evaluation of the accuracy since they distort the comparison. We extend the definition of a sequence outlier in Section 3.2.4.1. We aim at analyzing the results from a multi-dimensional viewpoint since we are interested in sequences where a good trade-off between accuracy and robustness is achieved. This includes an accurate trajectory and a high number of tracked frames, which bootstraps the sequence as soon as possible in the most challenging environments.

We propose to evaluate the accuracy using the ATE (see Section 3.2.4) and the robustness using the PTT (see Section 3.2.4.1). Besides these two fundamental metrics, we have also considered other comparisons that provide more insight into the differences between KN-SLAM and ORB-SLAM.

Trajectory accuracy

The accuracy of the trajectory is evaluated using the Absolute Trajectory Error (ATE), which measures the global consistency of the tracking as the Root Mean Squared Error (RMSE) between the positions of the frames of the estimated trajectory and the ground truth (see the definition in [121]). Let define a trajectory as the set of camera positions $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}$ for each keyframe $k_i \in \mathcal{K}$, the ATE between the trajectory $\mathbf{P}$ and the ground truth $\mathbf{G}$ is

$$\mathrm{ATE}(\mathbf{P}, \mathbf{G}) := \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} \|\mathbf{p}_i - \mathbf{g}_i\|^2}\,, \tag{3.2}$$

17 https://fradelg.mobivap.es/datasets

where N is the length of both trajectories measured as the number of keyframes.

From our viewpoint, the main limitation of this metric is that it ignores the length of the tracked trajectory. This length can be measured as the total number of keyframes in which the camera pose has been estimated. Intuitively, it can be seen that this penalizes trajectories with more keyframes. The additional keyframes are usually acquired at the cost of a worse initialization. Another factor to be considered is the accumulated drift along the keyframes. Thus, the RMSE always tends to be higher in longer trajectories, as we have confirmed in our experiments.

The ATE only considers the positional error of the keyframes matched by their corresponding timestamp in both trajectories. The ground truth includes more frames, which are discarded since their timestamps do not match with timestamps of keyframes in the estimated trajectory. The time alignment between trajectories is performed before computing the RMSE.

A time alignment between trajectories is not enough to use the ATE in monocular SLAM approaches. Camera positions cannot be compared directly since the scale of the trajectory is unknown. In addition, the coordinates of every pose are relative to the first keyframe. Thus, both trajectories must be spatially aligned to make them comparable. They have been aligned using a similarity transformation $\mathrm{Sim}(3)$ with seven DoF (three DoF for the translation, three DoF for the rotation and one DoF for scale). The alignment between a trajectory and the ground truth is performed using the closed form computed with the method of Horn [58]. This method finds the best transformation $\mathbf{T}$ that maps the estimated trajectory $\mathbf{P}$ onto the ground truth trajectory $\mathbf{G}$. Then, Equation 3.2 is reformulated as

$$\mathrm{ATE}(\mathbf{P}, \mathbf{G}) := \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} \|\mathbf{g}_i - \mathbf{T}\mathbf{p}_i\|^2}\,. \tag{3.3}$$

Rotational accuracy

The rotational error of a pose is the difference between the estimated rotation and the rotation established by the ground truth. Although this metric is insightful for sequences with strong camera rotation, it is usually ignored in SLAM evaluations. Furthermore, In the cases where it is included, the rotational error is evaluated per frame instead of at the trajectory level. This is due to the difficulty of aligning the rotations of two trajectories. Per-frame evaluations only consider the drift at each frame of the trajectory (see e. g., [39]). Thus, they do not include in the metric the benefits of the pose graph optimization over the final trajectory. We have evaluated the rotation error at a trajectory level by considering the first frame of both trajectories as a common reference. Therefore, we measure the accumulated rotational error along the rest of the keyframes.

At a trajectory level, the rotational error can be formulated following the same idea than ATE. We define the ATRE between two sequences as the RMSE between the rotations of the poses of both sequences. The main difference w. r. t. the ATE dwells in the way that the rotational error between two keyframes is computed. We use the quaternion error measured in the space of quaternions instead in the space of rotations. In other words, we measure the error between two quaternions instead of the rotations they represent.

The differences between quaternion and rotational errors are caused by the fact that quaternions are spinorial. For example, a rotation of 0° over a quaternion does not deliver the same quaternion than a rotation of 360°. Indeed, the latter is the inverse of the former although they represent the same rotation. In general, any rotation can be expressed by two quaternions that differ just by a negative sign. Indeed, they might be multiplied by any nonzero number to obtain the same rotation again if the quaternion is normalized.

The choice of one quaternion depends on the function where quaternions are used. For example, smoothness is preferred in a differentiable function because discontinuities or jumps between signs would cause indetermination issues in the derivatives of the function. Another example is a spherical interpolation, where the discontinuities in the quaternion barely affect the effective rotation, but they greatly affect the stability of the algorithm.

The distance e between two quaternions $\mathbf{q}_1, \mathbf{q}_2$ measured in the rotation manifold $SE(3)$ is defined by

$$e(\mathbf{q}_1, \mathbf{q}_2) = 2\|\log(\frac{\mathbf{q}_1}{\mathbf{q}_2})\|_2 ,$$

where the $\log$ of a single quaternion $\mathbf{q}$ is another quaternion

$$\log(\mathbf{q}) = [\frac{\log(\mathbf{q}_w^2 + b^2)}{2}, f\mathbf{q}_x, f\mathbf{q}_y, f\mathbf{q}_z] ,$$

where

$$b = \sqrt{\mathbf{q}_x^2 + \mathbf{q}_y^2 + \mathbf{q}_z^2} ,$$

and

$$f = \frac{\operatorname{atan}(b, \mathbf{q}_w)}{b} ,$$

where $\operatorname{atan}(x, y)$ is the function that computes $\arctan(\frac{y}{x})$ as the angle in $[0, 2\pi]$ formed by b and $\mathbf{q}_w$ and the positive semiaxis x_+.

Finally the quotient between two quaternions $\mathbf{q}_1, \mathbf{q}_2$ is computed according to the following expression:

$$\frac{\mathbf{q}_1}{\mathbf{q}_2} = \frac{1}{\|\mathbf{q}_2\|_2^2} \begin{bmatrix} \mathbf{q}_{1w}\mathbf{q}_{2w} + \mathbf{q}_{1x}\mathbf{q}_{2x} + \mathbf{q}_{1y}\mathbf{q}_{2y} + \mathbf{q}_{1z}\mathbf{q}_{2z} \\ -\mathbf{q}_{1w}\mathbf{q}_{2x} + \mathbf{q}_{1x}\mathbf{q}_{2w} - \mathbf{q}_{1y}\mathbf{q}_{2z} + \mathbf{q}_{1z}\mathbf{q}_{2y} \\ -\mathbf{q}_{1w}\mathbf{q}_{2y} + \mathbf{q}_{1x}\mathbf{q}_{2z} + \mathbf{q}_{1y}\mathbf{q}_{2w} - \mathbf{q}_{1z}\mathbf{q}_{2x} \\ -\mathbf{q}_{1w}\mathbf{q}_{2z} - \mathbf{q}_{1x}\mathbf{q}_{2y} + \mathbf{q}_{1y}\mathbf{q}_{2x} + \mathbf{q}_{1z}\mathbf{q}_{2w} \end{bmatrix} \quad (3.4)$$

In our formulation of $e(\mathbf{q}_1, \mathbf{q}_2)$ the second quaternion $\mathbf{q}_2$ is replaced by its opposite $-\mathbf{q}_2$ when the chordal distance $e_c(\mathbf{q}_1, \mathbf{q}_2)$ is greater than $\sqrt{2}$. This makes the implementation more robust to huge numerical differences between both quaternions that, in fact, represent small rotations. Here, the chordal distance e_c is defined by

$$e_c(\mathbf{q}_1, \mathbf{q}_2) = \|\mathbf{q}_1 - \mathbf{q}_2\|_2 .$$

As in the computation of the ATE, rotation of both trajectories must be aligned before applying the RMSE formula. The reason is that the camera pose of the current keyframe is estimated by SLAM algorithms relative to the first keyframe of the trajectory. In the absence of a global reference system the first keyframe is assigned to the identity matrix, whose translation is $0\,\mathrm{m}$ and rotation is $0°$. However, the rotation of the corresponding frame in the ground truth is expressed w. r. t. the coordinate system of the tracking system. It is impossible to determine the error in the first keyframe so we assume this keyframe has no error w. r. t. the ground truth. According to this assumption, the rotations of the other keyframes must be aligned before computing the rotational error per keyframe. This task is accomplished by determining the relative rotation $\mathbf{q}_\alpha$ between the first matching keyframe of the ground truth $\mathbf{q}_0^{GT}$ and the estimated trajectory $\mathbf{q}_0^{T}$. This relative rotation is defined by the quaternion

$$\mathbf{q}_\alpha = \mathbf{q}_0^{GT} \overline{\mathbf{q}}_0^{T} , \tag{3.5}$$

where $\overline{\mathbf{q}}$ is the conjugate of a quaternion $\mathbf{q} = q_w + q_x\mathbf{i} + q_y\mathbf{j} + q_z\mathbf{k}$, which is defined by

$$\overline{\mathbf{q}} = q_w - q_x\mathbf{i} - q_y\mathbf{j} - q_z\mathbf{k} .$$

The conjugate term in Equation 3.5 would not be necessary if the first keyframe of the estimated trajectory had always a match in the ground truth. However, the ground truth of some sequences (especially in the sequences of the EuRoC dataset) does not always contain the poses of the first frames. So there are images for some frames whose timestamp is not in the ground truth. So, unfortunately, this assumption is not necessarily true.

Finally, $\mathbf{q}_a lpha$ is applied over the rotation of the pose of each matched keyframe in the estimated trajectory. This aligns both trajectories from a rotational viewpoint. In this way, the first keyframes of both trajectories have the same rotation. Rotations in the quaternion space are performed by non-commutative multiplications. We choose the rotation of the ground truth as the reference rotation in the comparison. Then the rotation of the i-th pose in the estimated trajectory is aligned with

$$\mathbf{q}_i^{T} = \mathbf{q}_\alpha \mathbf{q}_i^{T} . \tag{3.6}$$

It is easy to prove that this alignment makes the first keyframes of both trajectories equivalent. Replacing $\mathbf{q}_\alpha$ with the equation Equation 3.6, we have

$$\mathbf{q}_0^{T} = \mathbf{q}_0^{GT} \overline{\mathbf{q}}_0^{T} \mathbf{q}_0^{T} = \mathbf{q}_0^{GT} . \tag{3.7}$$

3.2.4.1 *Robustness*

There is a lack of agreement in the literature regarding the best metric to evaluate the robustness of SLAM [14]. We have decided to measure robustness using twofold criteria in our evaluation.

Firstly, we take into account the number of executions in which the system refuses to initialize, i. e., no keyframe is selected. As mentioned in Section 3.1.2, such result can be caused by a low-parallax configuration, a bad homography or an erroneous initial estimation of the fundamental matrix. Also, the system does not bootstrap in sequences with a low number of featured points like in the ICL-NUIM. This is a logical consequence of the indirect nature of our SLAM approach.

Secondly, we have employed the Percentage of Tracked Trajectory (PTT) as a metric for robustness. This metric is defined as the total accumulated time that the system tracks the camera pose w. r. t. the total duration of the sequence. The PTT is computed by looking at the timestamps of the first and last keyframes in the trajectories. However, the metric should also consider eventual losses of track in the middle of a sequence. These special situations arise only in some sequences that combine dynamic events with loops (e. g., the sequence `fr3_camera_kidnap` in the TUM RGB-D dataset). We define a non-tracked period as the time interval between consecutive keyframes whose difference between timestamps is above a given threshold. In order to make this metric more robust to the temporal distribution of keyframes in a sequence, this threshold is determined using the inter-quartile range:

$$t_\delta = Q_3 + 1.5(Q_3 - Q_1)\,, \tag{3.8}$$

where Q_i is the i-th quartile of the time intervals between consecutive keyframes of the trajectory. The time intervals above t_δ are discarded because the tracking is considered lost. So the PTT is defined by

$$\mathrm{PTT} = \frac{\sum_{i \in \mathcal{N}} t_i - t_{i-1}}{t_{n-1} - t_0}\,, \tag{3.9}$$

where $\mathcal{N}$ is the set of indices of time intervals above the threshold t_δ

$$\mathcal{N} = \{i \in 0..N \mid t_i - t_{i-1} > t_\delta\}\,. \tag{3.10}$$

3.3 EXPERIMENTAL RESULTS

This section is aimed to discuss the experimental results gathered in the 47 sequences of the evaluated datasets. Each experiment has been executed eleven times alternatively for each sequence and SLAM approach in the same machine. In order to provide a fair comparison of all approaches use the same parametrization for the ORB detector and the same calibration for the intrinsic parameters of the camera. The median has been used as the estimator of the ATE for each sequence. This reduces the variance introduced by the outliers created in those executions where the system does not initialize or a bad pose estimation drifts the trajectory away from the optimum. This randomness is caused by the RANSAC iterations that remove the outliers. This problem is even more notorious at the bootstrapping stage since in some sequences the system only bootstrap inside a specific range of frames. If the system passes over these frames then it will not bootstrap later.

The experiments and the evaluation have been performed with our custom benchmarking scripts, included in the same toolbox to convert datasets in the TUM RGB-D format[18]. This tool launches the executable, saves the map and the trajectory, and compares this trajectory w. r. t. the ground truth. Since the scale is unknown in monocular SLAM, we have to align the trajectory w. r. t. the ground truth by computing a $\mathrm{Sim}(3)$ transformation before comparing both trajectories as in [88].

3.3.1 *Experimental settings*

The experiments have been performed in a computer equipped with a processor Intel i5 2550K manufactured in 2012 with 4 cores running up to 3.80 GHz. The memory hierarchy includes both 8 GB of Random Access Memory (RAM) and 6 MB of cache inside the processor. Despite providing more computing power than mobile devices, this processor is far behind the latest releases. Benchmarks confer a global score three times lower than an Intel 8700K manufactured in 2017. The computer is governed by an Ubuntu distribution based on Linux Operating System (OS) without additional dependencies like Robot Operating System (ROS), in contrast to most of the proposals in the state of the art. In this way, the stack of dependencies of the system is reduced. Besides, third-party projects are linked as submodules in the same SLAM project.

The software has been compiled using GNU Compiler Collection (GCC) 7.3.0 with the compiler flags adjusted to achieve the highest level of optimization. This includes

- `-O3` to activate loop unrolling and inline procedures,
- `-march=native` to use the set of optimizations defined for the specific processor architecture,
- `-msse4.2` to enable the fourth version of the Streaming SIMD Extensions (SSE), a processor extension to execute floating point computations in parallel over arrays of data.

The SLAM systems included in our evaluation have been compiled using the same set of compiler flags. All tests have been achieved using these flags in release mode. Unless stated otherwise, any result reported in this documents has been obtained in this mode. The algorithms are unable to run in real-time without the optimization procedures applied by the compiler targeting the architecture of the machine. Obviously, this time depends on the total number of pixels of the image, as Table 3.2 shows. Algorithms should achieve a framerate of 25 Hz or higher to be considered as running in real-time. This is

18 https://gitlab.com/fradelg/tum-toolbox

DATASET	RESOLUTION	TIME PER FRAME
KITTI	1241×376	40 ms
TUM RGB-D	640×480	30 ms
EuRoC	752×480	28 ms
ICL-NUIM	640×480	29 ms

TABLE 3.2: Average time per tracked frame in the sequences of the different datasets.

equivalent to a time per tracked frame of 40 ms. All sequences are below this threshold in the evaluated datasets, even in KITTI where frames are received at 10 Hz.

3.3.2 *Robustness analysis*

The PTT by KN-SLAM and ORB-SLAM are compared in different tables, one for each dataset: Table A.4, Table A.3, Table A.5, and Table A.6. The first two columns of each table contain the mean of PTT achieved in the 11 executions for each system. Cells with a blank gap indicate that this sequence has been labeled as an outlier, i. e., it is not comparable since a system is not able to bootstrap. A pair (sequence, SLAM) is marked as an outlier if it fulfills that

- the system has not been able to bootstrap the sequence at least in 3 executions, or
- the mean PTT in all executions is below 30 %.

An outlier in the set of sequences is a special sequence not suitable for comparing two monocular SLAM systems. Outliers are excluded from the rest of the evaluations since we consider that comparison as unfair. For example, a trajectory with 50 % more keyframes is prone to be less accurate due to the accumulated error. Some sequences, such as `kitti/01`, have not even been included in our evaluation since it is evident that they are not intended for monocular SLAM, as it was explicitly stated in [88]. Other challenging sequences, such as `tum/fr1_desk` or `tum/fr3_nstr_tex_far`, have been included to evaluate the improvement in robustness of KN-SLAM w. r. t. ORB-SLAM.

In the TUM RGB-D benchmark, KN-SLAM shows better robustness than ORB-SLAM since it tracks an average of 12 % more frames than ORB-SLAM (see Table A.4). This difference is illustrated by the sequence `tum/fr1_desk`, whose trajectory is represented by Figure 3.6. It shows a comparison of the trajectories estimated by KN-SLAM and ORB-SLAM w. r. t. the ground truth of the sequence. It can be seen that KN-SLAM (top) bootstrap sooner and also tracks more frames than ORB-SLAM (bottom). These plots are created by linking consecutive positions of the keyframes projected over the plane XY, except trajectories of the KITTI dataset that are projected over the plane XZ.

Sequence `tum/fr1_floor` is also a special case where ORB-SLAM refused to initialize in all executions whereas KN-SLAM achieves a robust 78 % in

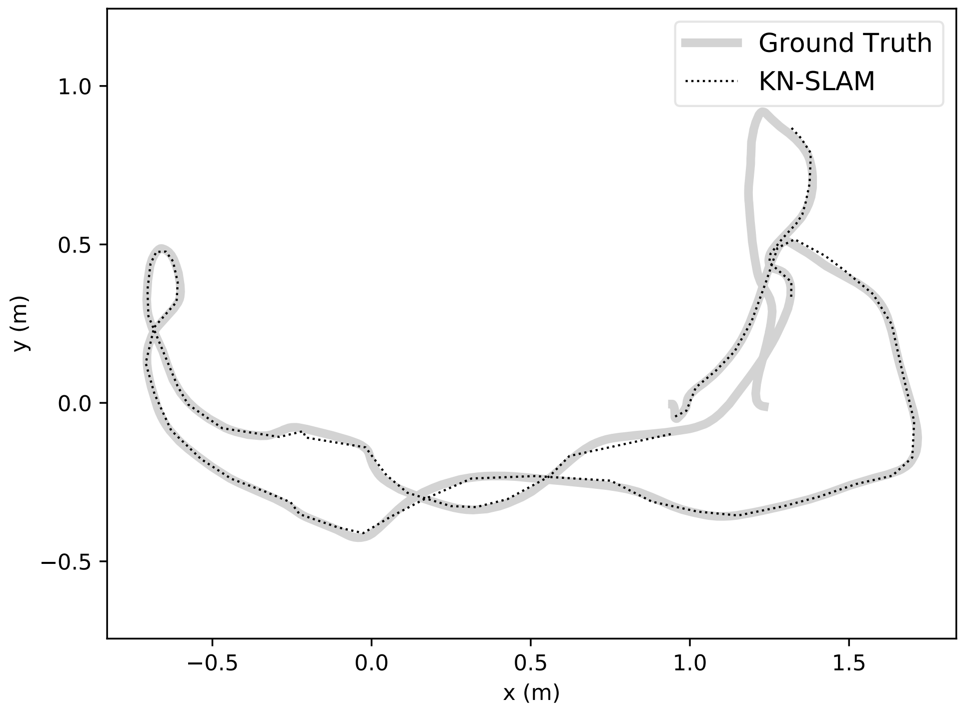
PTT. Since ORB-SLAM fails to track the sequence, it is considered an outlier. Both systems refuse to initialize in sequence `tum/fr3_nstr_tex_far`. This is a special sequence where the movement of the camera shows the well-known twofold ambiguity described by [82]. Despite this problem that affects the selection of the homography or the fundamental matrix, our bootstrapping method is able to track some frame at least in 5 executions, but it only achieves an average 47 % in PTT. Thus, the sequence is labeled as an outlier too.

The results in the EuRoC dataset are mixed, with the sequence `euroc/V202` displaying the highest difference. In the second half of this sequence, ORB-SLAM loses the track of the camera due to the fast movements of the MAV. Our adaptive bootstrapping procedure handles the challenges introduced by the sequence `euroc/MH05`. In this sequence, the camera is moved fast by the MAV in a local environment to initialize the IMU. The system will usually bootstrap a map of this local environment without including any point beyond. When the camera leaves this environment, the tracking is lost after a few frames. Unfortunately, the system cannot recover from this situation. This is a special case since the situation is difficult to replicate in a real-world scenario. However, KN-SLAM is able to wait for a few frames until enough points with high depth are available. The scale of the trajectory changes depending on the selected pair of keyframes. However, this scale will be adjusted when the ATE is computed, as explained in Section 3.2.4.

Results in the KITTI benchmark for both SLAM systems show almost no differences regarding the PTT, except for the sequences `kitti/02`, `kitti/08`, and `kitti/09`. In `kitti/08`, ORB-SLAM needs more time to bootstrap due to the fast camera motion in the first frames of the sequence. The sequence `kitti/09` is the sequence most difficult to track for ORB-SLAM. It has been completely tracked in 5 of 11 executions. In the other executions, the tracking has been lost at some point at the last quarter of the sequence. These executions offset the PTT of ORB-SLAM to a value significantly lower than KN-SLAM (16 % lower). Finally, ORB-SLAM lost the tracking in the middle of the sequence `kitti/02` in 2 executions. This implies a lower mean for the PTT, which explains the difference of almost 9 % between both systems.

Among all the evaluated datasets, ICL-NUIM clearly highlights as the most challenging for an indirect SLAM approach. The sequences of this benchmark belong to scenes characterized by high-frequency textured objects. The challenges are clearly reflected in the PTT of Table A.6. It can be seen that ORB-SLAM does not even initialize in 4 of the 8 sequences, whereas KN-SLAM bootstrap in all sequences. However, 5 of these sequences (`icl/lr_kt0`, `icl/lr_kt1`, `icl/lr_kt3`, `icl/of_kt0`, and `icl/of_kt3`) are labelled as outliers for ORB-SLAM. The PTT achieved by KN-SLAM is greater than 90 % in 4 out of 8 sequences, which is above the average PTT achieved for all the evaluated sequences. On average, and accounting the 8 sequences of the dataset, KN-SLAM tracks 77 % of the trajectory compared with 55 % tracked by ORB-SLAM (considering only the sequences where it initialized at least once).

A false positive loop is detected by KN-SLAM in sequence `icl/of_kt1`. This has been the only noticeable false positive detected by our method. The combination of a degenerate map and two visually similar frames drives the method to choose a poor candidate for loop closing. These two frames are



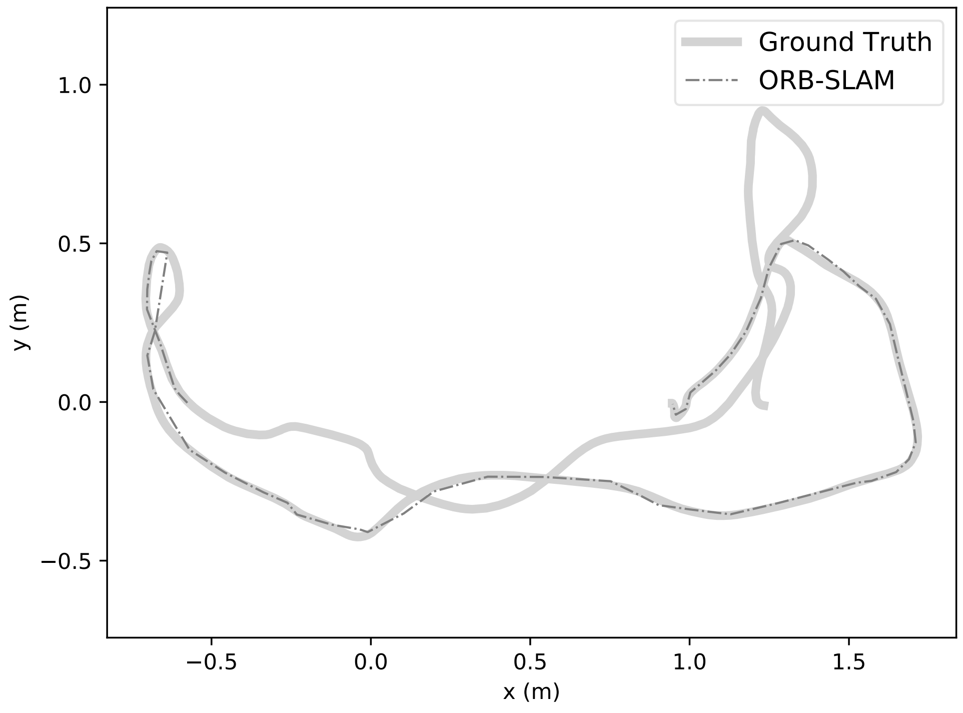


FIGURE 3.6: Comparison between the ground truth and the estimated trajectory for KN-SLAM (top) and ORB-SLAM (bottom). The trajectory corresponds to the median ATE for each SLAM system in sequence `fr1_desk`. The thin, discontinued line represents the tracked trajectory whereas the thicker line represents the trajectory of the ground truth.

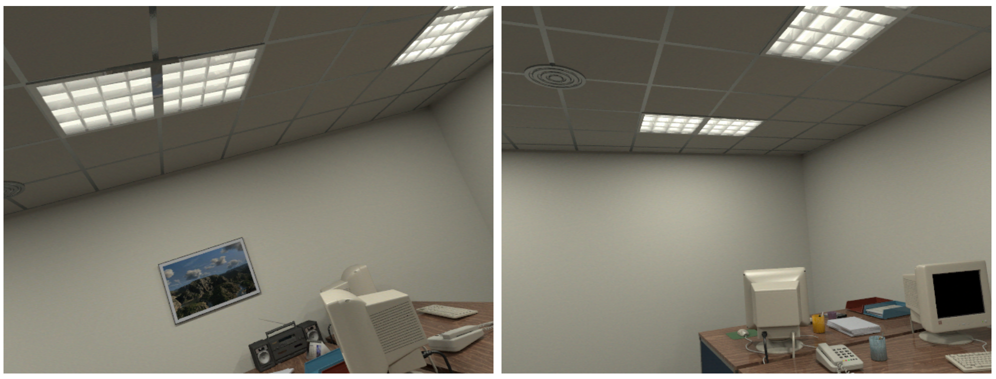


FIGURE 3.7: These two visually-similar frames (175 and 370) of the sequence `icl/of_kt1` induce KN-SLAM to detect a bad loop.

shown in Figure 3.7. Even though this candidate is supported by few inliers (around 15), it is not rejected by the PnP test nor the $\mathrm{Sim}(3)$ test. Besides, the final loop is supported by more than 200 points of the map.

3.3.3 *Trajectory accuracy*

The accuracy of the trajectory is evaluated with the ATE introduced in Section 3.2.4. First of all, we shall highlight that the ATE has been reduced for ORB-SLAM in all sequences by an average of 25 % w. r. t. the original results published in [88]. We attribute this improvement to the underlying performance of our CPU and the enhancements introduced in our testing environment composed mainly of the compiler and the OpenCV library.

We have split the set of sequences according to the difference between the ATE achieved by ORB-SLAM and KN-SLAM. Results reveal that ORB-SLAM outperforms the accuracy achieved by ORB-SLAM in 25 out of the 47 evaluated sequences. However, in other 14 sequences ORB-SLAM estimates more accurate trajectories. We cannot compare 8 sequences since they are labeled as outliers (see the definition in Section 3.3.2).

Results are represented by two figures: Figure 3.8 and Figure 3.9. The former corresponds to the sequences in TUM RGB-D, KITTI and ICL-NUIM datasets, whereas the latter corresponds to the sequences in the KITTI dataset. The plot has been split due to the differences in the dimensions of the trajectories, which lead to differences in the scale of the ATE: While the ATE in the KITTI dataset is above 1 m, the ATE in the other datasets is below 1 m.

The numerical values of the ATE are also split in different tables, one for each dataset: Table A.7, Table A.8 Table A.9 and Table A.10. This separation enables a fair comparison since sequences in the same dataset present common characteristics. Each table shows the identifier of the sequence in the first column. The ATE achieved by KN-SLAM and ORB-SLAM are placed in the second and third columns respectively. The fourth column corresponds to the percentage of improvement w. r. t. the higher ATE. The ATE represented for a sequence is the median of the ATE achieved in each one of the eleven executions. The number of successful executions (i. e., the executions where the system has bootstrapped) is represented between parentheses next to the ATE.

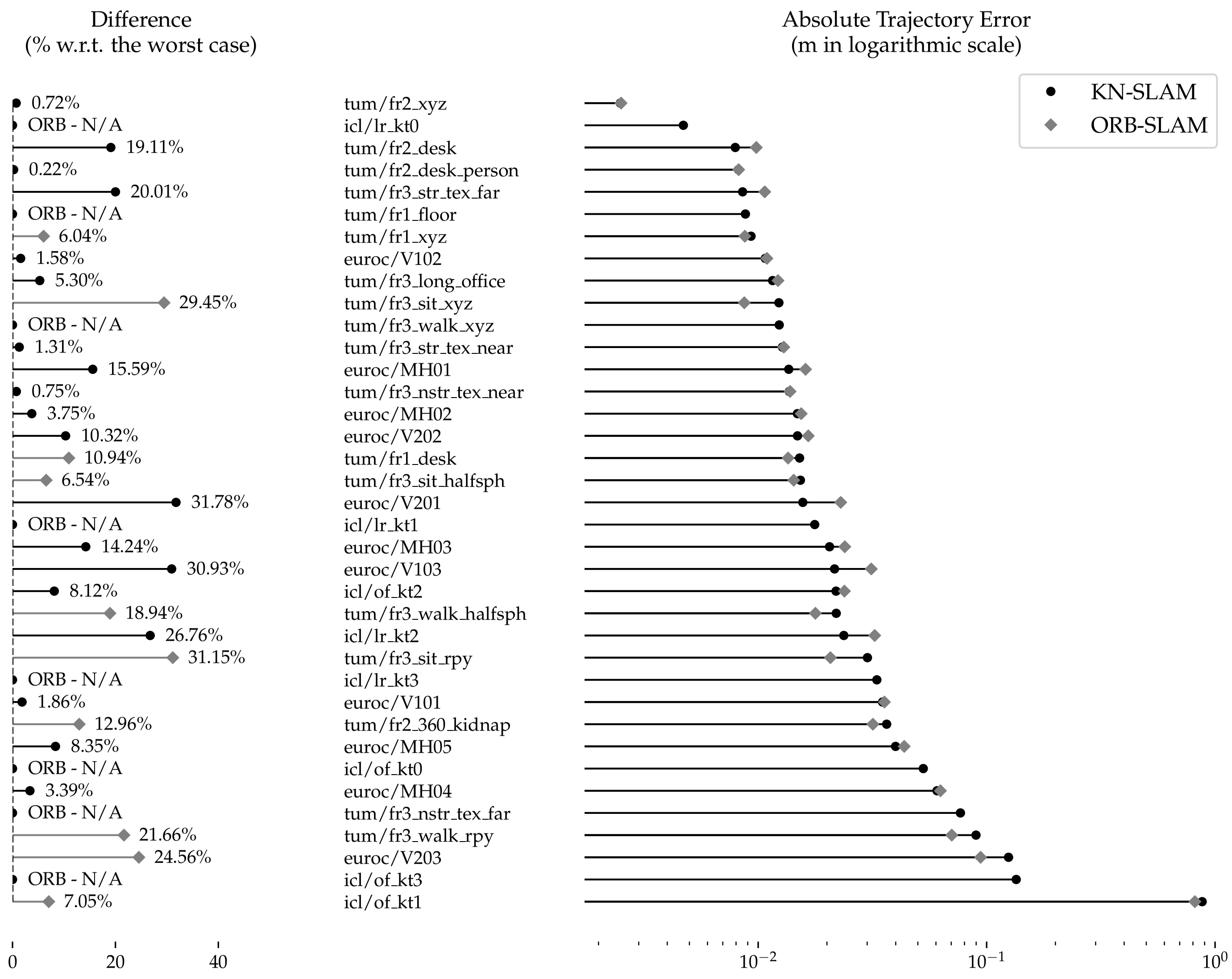


Figure 3.8: Comparison between the ATE of KN-SLAM and ORB-SLAM. Left side of the graph represents the difference between both systems measured as the percentage of improvement of the ATE w. r. t. ORB-SLAM. The marker type signals which implementation has a lower ATE. The right side of the figure compares the values of the ATE using a logarithmic scale to highlight the differences.

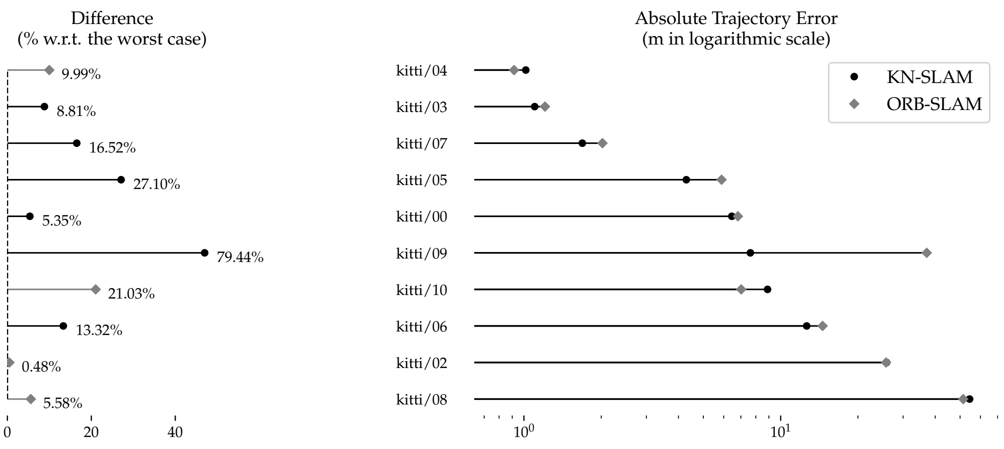


FIGURE 3.9: ATE of KN-SLAM and ORB-SLAM for the sequences in the KITTI dataset. This figure follows the same structure as Figure 3.8, but with the sequences in KITTI since they are in a different scale. The average ATE of all the KITTI sequences is 12.15 m whereas the average ATE for the rest of the sequences is 0.07 m, a difference of three orders of magnitude.

In the KITTI dataset, KN-SLAM reaches a superior accuracy in six of ten sequences (see Figure 3.9 and Table A.7), with an estimated ATE an average 11 % lower. This difference is almost unnoticeable in the first four sequences. However, the ATE of the trajectory estimated by KN-SLAM in the sequence `05` is 27 % lower. This is due to the existence of various loops that allows KN-SLAM to correct the drift with a less-connected graph. These loops also exist in sequences `06`, `07` and `09`, so KN-SLAM estimates more accurate trajectories. The sequence `10` does not have any loop (i. e., it is an open trajectory) and, then the accuracy delivered by ORB-SLAM is 21 % better than KN-SLAM. The sequence `08` does not introduce any loop either, but the ATE of KN-SLAM is just 6 % higher than ORB-SLAM. Such differences can be caused by the long section of the trajectory where the camera drifts due to a straight-line movement.

The main outlier in the KITTI dataset is the sequence `kitti/09`, where ORB-SLAM experiences several difficulties to detect the last loop. This difficulty was already declared in [88], although the authors claim that ORB-SLAM detects the loop in some executions. However, in our experiments, ORB-SLAM has never detected such loop. Indeed, we have tested this sequence thoroughly more than fifty times. In these tests, ORB-SLAM has detected the loop only once. Furthermore, as we also mentioned in Section 3.3.2, the tracking has been lost before the last frame of the sequence in six of eleven executions. We discard that this problem is related to the testing environment since both ORB-SLAM and KN-SLAM have been compiled in the same machine, with the same compiler flags, and the same dependencies. This can be acknowledged as one of our main achievements. We are proud of the robustness of our method in this sequence since the last loop has been detected in eight of the executions. In the other three executions, KN-SLAM has tracked around 87 % of the sequence.

The TUM RGB-D dataset includes a collection of diverse sequences (see Figure 3.8 and Table A.8). There are sequences affected by different types of challenges among them. However, they can be arranged in small groups to explain the differences in the ATE. In summary, KN-SLAM delivers better results in seven of seventeen sequences, whereas three sequences are not comparable.

Sequences fr1_xyz and fr2_xyz are used for debugging purposes since the camera is displaced sideways only. Differences in the ATE between ORB-SLAM and KN-SLAM are very subtle.

A second group is composed by desktop sequences: fr1_desk, fr2_desk and fr2_long_office. In the first one, the ATE delivered by KN-SLAM is 11 % higher than ORB-SLAM due to the fast camera movements without a final loop closure. Both of them are present in the other two sequences, where the ATE is up to 19 % lower than ORB-SLAM.

The third group of sequences is formed by fr3_nstr_tex_near, fr3_str_-tex_far, and fr3_str_tex_near. In this group, KN-SLAM achieves better ATE than ORB-SLAM, where fr3_str_tex_far has the biggest difference. The sequence has been recorded with slow camera movements and a good balance between rotations and translations, which benefits KN-SLAM. The last group of sequences includes six sequences with dynamic scenes and pure rotations and translations. The sequences fr3_sit_* and fr3_walk_* are pieces of a long sequence where two persons move in front of the camera. It is clear that KN-SLAM does not achieve good trajectories since the ATE is an average 22 % lower. Pure translations and rotations are not satisfactorily handled by KN-SLAM. Furthermore, moving objects in front of the camera affect the bootstrapping in some executions where it creates a corrupted map.

The camera in the sequence fr2_360_kidnap is occluded in the middle, forcing the system to relocalize the camera in the next frames. Experiments show that KN-SLAM trajectory is less accurate after relocalization since the ATE is 13 % higher than ORB-SLAM. It seems that a less-connected pose graph includes fewer keyframes in the PnP optimization, affecting the accuracy of the estimated pose.

The sequences fr1_floor and fr3_walk_xyz are not compared since ORB-SLAM does not initialize on them. In fr3_nstr_tex_far both methods refuses to initialize due to the twofold ambiguity configuration.

The results shown in the EuRoC dataset are very promising (see Figure 3.8 and Table A.9). Here, KN-SLAM outperforms the ATE achieved by ORB-SLAM in ten out of eleven sequences. The best results are in sequences V103 and V201, which introduce multiple loops in the trajectory with several rotations and translations among them. In both sequences, the ATE of the trajectory is more than 30 lower than ORB-SLAM. The exception is the sequence V203 where the error of KN-SLAM is 25 % higher than ORB-SLAM. This sequence introduces shorter distances between the camera and the scenes, besides some dynamic object, which influences negatively to KN-SLAM. In the rest of the sequences, KN-SLAM performs on average 7 % better than ORB-SLAM, which increases slightly for MH01 and MH03.

Finally, it is worth to comment the results in the ICL-NUIM dataset (see Figure 3.8 and Table A.10). This dataset is not intended for monocular indirect SLAM due to the intensive presence of high-frequency textures. For this reason, ORB-SLAM only tracks three of the eight sequences. In the three comparable sequences, KN-SLAM achieves lower ATE in two of them: lr_kt2 and of_kt3. However, the ATE of the trajectory for of_kt1 is 7 % higher than ORB-SLAM, due to the false loop detected by KN-SLAM (more details in Section 3.3.2). The result for this sequence must be carefully interpreted since only half of the frames have been tracked by both systems. In the other two sequences, more than 90 % of the frames have been tracked.

As a general remark, it should be reflected that the ATE of a sequence tends to be higher in long trajectories due to the accumulated error in the pose of new keyframes. Thus, the longer the sequence, the higher the error. This effect is partially alleviated in the presence of loops at the end of the trajectory.

We have executed an additional analysis of the dispersion of the results achieved by the eleven executions for each sequence. Results of this analysis confirm that the ATE is spread around the median, so it is a good estimator for the average ATE of the system. More concretely, in our analysis 50 % of sequences have their five best executions in an interval of 5 % around the mean. Another way of measuring this dispersion is by computing the standard deviation of the results within the inter-quartile range for each sequence. The mean of this standard deviation in all sequences is 7.31 % for KN-SLAM and 6.85 % for ORB-SLAM. This analysis also includes the most challenging sequences where systems can fail at bootstrapping, such as fr3_nstr_tex_near.

3.3.4 *Other metrics*

Besides the PTT and the ATE, we have estimated other interesting metrics concerning both the performance and the effectiveness of both SLAM systems. These metrics are not usually included in other evaluations since they are less meaningful than ATE or PTT. However, they provide important highlights about KN-SLAM. The rest of this section briefly described the values achieved for these metrics.

Number of keyframes

Connected keyframes in the connectivity graph help to cull those deemed redundant. Our approach in KN-SLAM limit the creation of links between keyframes in the connectivity graph, which avoid the culling procedure to reach them. This has a side effect: the mapping thread inserts more keyframes than ORB-SLAM when the camera travels around the same region of the scene. In such cases, a more dense graph (i. e., with more edges) helps to find a previous keyframe with a significant overlap with the current frame. However, if the past keyframes are not connected, the mapping thread would likely create a new keyframe. This consequence is more evident in the sequences where the camera sweeps the scene multiple times (e. g., tum/fr3_walk_rpy). This increment also affects the memory footprint of the process since keyframes are the central data structure preserved in memory besides the map.

The number of keyframes has been analyzed using the mean for each dataset (see Table 3.3). Overall, the number of keyframes is an average 15 % higher in KN-SLAM than ORB-SLAM. It is seen that the increment is most noticeable in the sequences of TUM RGB-D or ICL-NUIM. The former is expected due to the number of loops in some sequences. The latter is caused by the of_kt1 sequence and the false loop detection. In the sequences of KITTI

Dataset	KN-SLAM	ORB-SLAM	Difference
KITTI	901	822	12 %
TUM RGB-D	57	55	17 %
EuRoC	191	169	13 %
ICL-NUIM	79	62	19 %

Table 3.3: The mean number of keyframes for each sequence in the evaluated datasets. The third column represents the mean of the percentage increment per sequence between KN-SLAM and ORB-SLAM.

Dataset	KN-SLAM	ORB-SLAM	Difference
KITTI	15 948	29 763	−47 %
TUM RGB-D	1087	3379	−41 %
EuRoC	3732	11 656	−67 %
ICL-NUIM	1411	1930	−34 %

Table 3.4: The mean number of edges in the connectivity graph generated by KN-SLAM and ORB-SLAM for the sequences of the evaluated datasets. The third column represents the mean of the percentage decline per sequence between KN-SLAM and ORB-SLAM.

and EuRoC, KN-SLAM inserts an average 13 % more keyframes than KN-SLAM. These sequences combine translation with rotations, and the camera rarely passes across the same region.

Number of edges

One of our central goals is to reduce the number of connections between the keyframes of the graph. This is accomplished by limiting the number of edges in the connectivity graph. Hence, the total number of edges in the graph must be significantly lower. Whereas ORB-SLAM only imposes a minimum number of shared observations to create a link, KN-SLAM constrains the number of links per keyframe to 20. We have analyzed the total number of edges at a dataset level, as it is shown by Table 3.4.

Overall, the number of edges created by ORB-SLAM is an average 49 % higher than the same number for KN-SLAM. This decrease is more extreme in the EuRoC dataset, where 67 % fewer edges are created by KN-SLAM. The lowest reduction is observed in the ICL-NUIM dataset due to the few sequences that are comparable. In the other two datasets, the reduction is slightly below 50 %.

The number of edges does not impact the memory consumption as the number of keyframes since they are stored with pointers in a hash table. However, they impact the NLLS optimization procedures that are performed to optimize the map, the camera pose, and the loops.

Dataset	KN-SLAM	ORB-SLAM	Difference
KITTI	1.12	1.19	−7.71 %
TUM RGB-D	2.40	1.12	4.29 %
EuRoC	1.19	1.07	5.00 %
ICL-NUIM	1.38	1.54	1.67 %

TABLE 3.5: The mean ATRE introduced by KN-SLAM and ORB-SLAM grouped by dataset. The third column represents the mean of the percentage difference per sequence between KN-SLAM and ORB-SLAM.

Rotational error

The rotational error has been defined in Equation 3.2.4. This error is not usually analyzed in the evaluation of SLAM systems since it is relevant only in sequences with pure rotational movements. However, these sequences are not suitable for monocular SLAM, due to the concerns mentioned in Section 3.2.2.

At a dataset level, results show that the ATRE is reduced by KN-SLAM in the KITTI dataset. The ATRE has imperceptible differences in the ICL-NUIM dataset, whereas KN-SLAM exhibits an error around 5 % higher than ORB-SLAM in the other two datasets. There are 21 sequences where KN-SLAM achieves lower error and 19 sequences where the ATRE of ORB-SLAM is lower. Overall, the ATRE is always below 3°, except for the sequences `fr3_walk_rpy` and `fr3_nstr_tex_far` of the TUM RGB-D dataset. In these two sequences, KN-SLAM generates a higher error. The former is a sequence where KN-SLAM has difficulties to track the camera due to its pure rotational movement. The latter is not comparable (see Section 3.3.2), due to the twofold ambiguity detected at bootstrapping. However, in the sequences where KN-SLAM bootstraps the ATRE is slightly higher than ORB-SLAM, with an average of 1.67° compared to 1.15°, respectively.

Our evaluation has included eight sequences with pure rotational movements that can be analyzed separately. Among them, the most meaningful ones are the subsets of sequences identified by `tum/fr3_*_rpy` and `tum/fr3_*_halfsph`. In the former, the ATRE created by ORB-SLAM is 65 % lower, whereas in the latter it is 25 % higher. In the `rpy` sequences the camera rotates around the optical axis so that a complete pose graph is created by ORB-SLAM. The constraint of the number of links in the graph increases the ATRE in KN-SLAM.

In the rest of the sequences, the difference between KN-SLAM and ORB-SLAM in the ATRE ranges from −35 % to 65 %. In absolute terms this difference is always below one degree except for the previously mentioned sequences and the sequence `V203` of the ICL-NUIM dataset. This sequence contains several fast rotations that complicate the tracking so that the ATRE in the trajectory obtained by KN-SLAM is 1.13° higher than ORB-SLAM.

3.3.5 *Comprehensive approach for Absolute Trajectory Error*

We devote this section to dive into what challenging characteristics make our system more suitable for a sequence. Our first idea was to seek correlations between the characterization of the sequences and the accuracy of the trajectory. We delve into this core idea to outline a more systematic approach based on Machine Learning. This idea consists in classifying the sequences of the evaluated datasets according to the accuracy w. r. t. ORB-SLAM. Each sequence is represented in terms of the characteristics described in Section 3.2.2.

There is no doubt that the most commonly used metric to evaluate SLAM is the ATE [33, 88, 121]. As it was explained in Section 3.2.4, it measures the positional error in camera poses and accounts for the drift accumulated in the estimated trajectory. Although it provides a summarized view of the accuracy of the system in a sequence, we honestly think that it does not reflect other relevant aspects that should be taken into account in a fair comparison of SLAM systems. Other authors also share this viewpoint (see [81, 91]) since they also include metrics like the starting ratio or the tracking success after the initialization in their evaluations. We have adopted a more systematic and reproducible approach based on classification algorithms using the sequence labeling introduced in Section 3.2.2.

At the beginning of our research, we assume that our contributions would KN-SLAM perform better in sequences with specific challenges such as shaking and fast movements. However, our experimental results reveal that a single feature is not enough to explain the differences w. r. t. ORB-SLAM. Indeed, the SLAM pipeline includes tasks whose operation is affected by different challenges. Thus, a combination of several challenges is what truly makes a scene hard to track. This combination can be easily interpreted if it is modeled as a weighted sum of the characteristics. Fortunately, some classifiers are already base on this linear model.

The first step consists in merging the results described in Section 3.3.3 with the characteristics assessed in Section 3.2.2. Our goal is to determine what characteristics are more favorable to one system or another. We reformulate the task as a standard classification problem with a binary *category* assigned to each sequence. This category is labeled as KN-SLAM if the ATE achieved by KN-SLAM is lower than the ATE achieved by ORB-SLAM. Otherwise, the category is labeled as ORB-SLAM. Therefore, this label is chosen according to the sign of the difference between the ATE of KN-SLAM and the ATE of ORB-SLAM[19]. In the machine-learning argot, the characteristics of the sequences are represented as *features* and the sequences as *instances*. These instances are classified by the algorithm according to the assigned category.

Once the dataset is ready for training, a machine learning algorithm must be chosen. There exist several alternative machine learning algorithms (see [106] and the references therein). We seek for an algorithm based on a linear model since its relation with the characteristics of the sequences is easier to understand. In this kind of models, the coefficients or weights linked to the characteristics are representative of the influence or relevance of such characteristics. Thus, they indicate what challenges in a sequence might make it more suit-

19 The error is computed using six decimal positions for floating point.

able to run KN-SLAM or ORB-SLAM. For instance, a weight close to 0 means that the linked characteristic is irrelevant to decide whether to choose one system or the other. On the contrary, a higher absolute value for a weight emphasizes that the challenge should be taken into account at the time of choosing a SLAM system.

There are several machine learning algorithms such as SVM [12], logistic regression [59] or neural networks [110] that fit our linear model requirement. Among them, we have chosen SVM with a linear kernel because it combines simplicity, speed, and flexibility with a model easy to interpret: the absolute value of the coefficients of this model is directly proportional to the discriminatory power of the corresponding characteristic. Besides, the algorithm is extensively referred in the literature for similar purposes, where its good properties have been highlighted. Next section reviews the fundamental concepts and the mathematical background of this algorithm. Furthermore, we will dive into the reason why the classifier has been chosen.

Support Vector Machine

Classification methods play an important role in data analysis in a wide range of scientific applications. There are multiple classification algorithms in the supervised machine learning landscape (see the comparison introduced in [41] and the references therein). Obviously, the approached classification problem influences the choice of the proper algorithm. According to Caruana and Niculescu-Mizil [17], the most important aspects to consider include the desired accuracy, the training time, the number of features, the size of the training space or the linearity of the model. The simpler the algorithm, the easier the interpretation of the classifier in the space of features.

One of the most widely known classification algorithms is SVM [12]. This algorithm determines the hyperplane that split a collection of instances by their corresponding category. In other words, given a training dataset composed of the sequences labeled with their characteristics, the algorithm finds the optimal hyperplane that separates the instances in the space of the characteristics. This hyperplane is based on a combination of several support vectors that maximize the distance between classes. To illustrate the meaning of the hyperplane, think about the simplest case: a two-dimensional space of features. In this case, the hyperplane can be visualized as a line dividing the plane into two parts, so each class lay on one side. Higher dimensions require to perform transformations in the space of features or the addition of more synthetic features (i. e., as a function of the already existing features). Indeed, SVM is widely used for high-dimension data classification.

The algorithm has several parameters to fit the resulting hyperplane to our needs and the classification problem. These parameters influence especially over the linearity of the underlying model, the accuracy of the classifier and the training time. The most meaningful parameters in this broad set can be reduced to the following:

- Margin. The separation between the hyperplane and the closest class points. Good margins correspond to large separation between the estimated hyperplane and the classified instances. Indeed, the main goal of SVM is to achieve the best margin.

- PENALTY. Sometimes called regularization and referred by C, it quantifies how much the algorithm should penalize misclassified instances in the training dataset. A large value indicates that a small-margin hyperplane is preferred if the classification accuracy is improved. Conversely, a small value means that a large-margin hyperplane is desired, even if that hyperplane misclassifies more instances.

- GAMMA. The distance between the hyperplane and the instances considered for its computation at training time. In other words, a high gamma means that only those instances closest to the hyperplane are considered for the computation of the hyperplane, while a low gamma tells the algorithm to include also instances far away from the hyperplane.

- KERNEL. The hyperplane is learned by transforming the space of features for instances. This transformation requires to establish an algebra, which is done by the kernel. Given a training dataset $\{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\}$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, each instance is defined by its features $\boldsymbol{x} = (x_1, ..., x_d)$ and its class label y. The kernel predicts the class y of a new input $\boldsymbol{x}$ using the support vectors (a subset of the inputs $\boldsymbol{x}_i$). There are three main different types of kernels: linear, polynomial and exponential.

  The linear kernel separates classes by a linear boundary:

$$K(\mathbf{x}) = \sum_{j=1}^{d} w_j x_j + b\,,$$

  where $\boldsymbol{w} = (w_1, ..., w_d)$ are the coefficients of the hyperplane and b denotes the intercept. Linear SVM is commonly used for referring to SVM with a linear kernel.

  Other non-linear kernels split the space of instances with a separation line in a higher dimension. Two examples are the polynomial kernel

$$K(\mathbf{x}) = 1 + (\sum \mathbf{x}\mathbf{x}_i)^d$$

  and the exponential kernel

$$K(\mathbf{x}) = exp(-\gamma \sum \mathbf{x} - \mathbf{x}_i^2)$$

The class or category for a new instance $\boldsymbol{x}' = (x_1', ..., x_d')$ is determined by the sign of the distance computed by the kernel. The optimal hyperplane with maximal margin is found by a convex optimization procedure. The maximization of the margin can be achieved by solving

$$\arg\max_{b,w} \sum [1 - y_i K(\boldsymbol{x})]_+ + C_\lambda(w)\,, \tag{3.11}$$

where the penalty term $C_\alpha(w) = \lambda ||w||_2^2$ uses the L_2 norm ("ridged penalty"), which shrinks the coefficients without making them zero.

Interpreting the hyperplane

The algorithm for linear SVM matches perfectly with our two main requirements: a short number of features for each instance and linearity of the model. In this model, the coefficients $\boldsymbol{w}$ of the hyperplane represent the coordinates of the vector orthogonal to that hyperplane. The direction of the vector represents the predicted class y for instance $\boldsymbol{x}$. The absolute value of these coefficients determines the importance of a feature for the data separation task. In this way, covariates playing an important role in discrimination can be identified according to their contribution to the classifier. Therefore, the classifier trained with our dataset will have a set of coefficients that represent the relevance of each characteristic in the classification result.

A wide number of *feature selection* methods have been proposed in the literature to take advantage of this property [131]. They have been applied in other fields like bioinformatics. These methods can be subdivided into two classes: filter and wrapper methods (see the classification proposed in [8]).

Filter methods discard irrelevant features before the learning algorithm constructs the prediction rule. An example of a filter method for feature selection is Recursive Feature Elimination (RFE) [49]. It is employed to determine the most expressive genes in a DNA sequence segmented with microarrays. This method is based on the dual formulation of the hyperplane for linear SVM. It uses the square of the coefficients as a ranking metric for deciding the relevance of a particular feature.

Feature selection is performed within the optimization procedure by wrapper methods, which avoids overfitting and increases the predictive power of a model [84]. One example of these methods is Least Absolute Shrinkage and Selection Operator (LASSO)čitepbradley1998feature. It adopts a L_1 penalty function for SVM, which has the form

$$C_\alpha(\boldsymbol{w}) = \lambda \|w\|_1 = \lambda \sum_{i=1}^{d} |w_i|$$

In contrast to the conventional L_2 penalty, this can shrink the small coefficients of the hyperplane to zero. Other SVM-based example is Smoothly Clipped Absolute Deviation (SCAD) [135]. This method uses a non-convex penalty based on a quadratic spline function. This function behaves as L_1 for small coefficients. However, it applies a constant penalty for large coefficients, whereas it increases it linearly in L_1. In this case, the maximum penalty reduces the probabilities of estimating large coefficients. The SCAD penalty has better theoretical properties than the L_1 function in large spaces of features [37].

We have performed our analysis using the implementation of SVM provided by [101]. Every parameter uses the default value of the implementation, except for the linear kernel and the variable penalty (or regularization) C. This assesses the influence of the penalty over the tradeoff between the accuracy and the margin of the hyperplane. The trained dataset includes only comparable sequences (39 of 47) according to the definition of outlier provided in Section 3.3.2. The coefficients achieved for different penalties are shown in Figure 3.10. It can be shown that the weights assigned to each characteris-

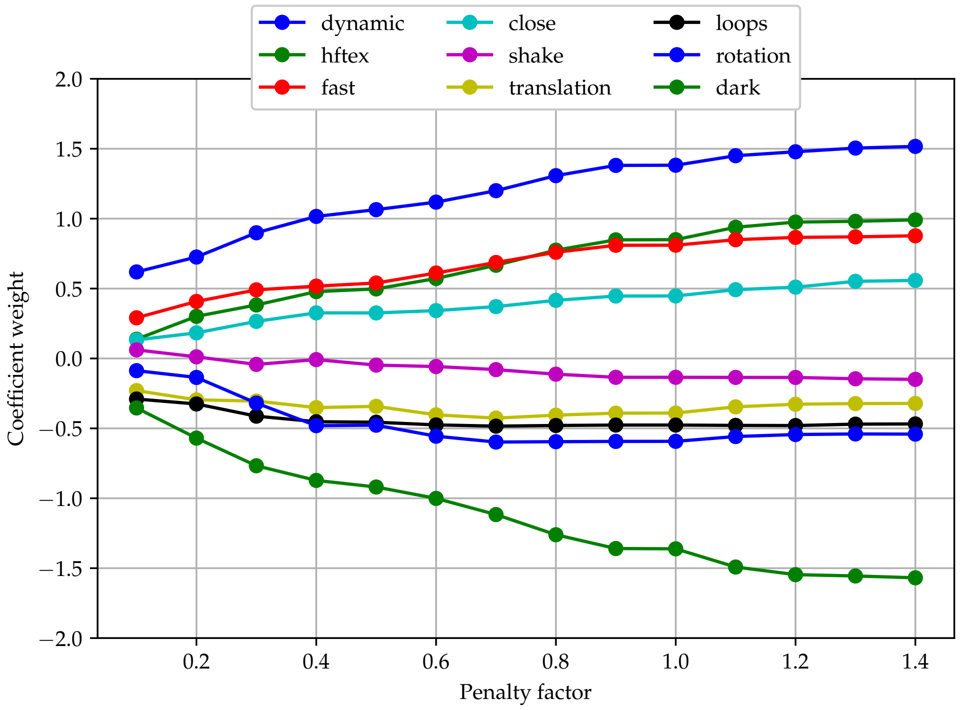


FIGURE 3.10: Evolution of the coefficients of the hyperplane for different values of the penalty C in linear SVM. The accuracy and the margin of the hyperplane are shown in the background.

tic diverge when the penalty value is increased. However, their order is the same starting from a penalty value of 0.25.

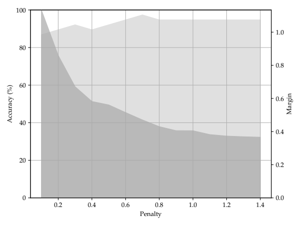


FIGURE 3.11

We prioritize the accuracy of the classifier over a large-margin hyperplane. We seek a classifier that explains the maximum number of sequences, although that implies a lower margin. This tradeoff between accuracy and margin can be visualized in Figure 3.11. We finally choose the coefficients for the classifier trained with a penalty of $C = 0.7$. This classifier has a final accuracy above 95 % with a margin of around 0.50.

The coefficients of the hyperplane estimated by SVM are represented in the second column of Table 3.6. The sign of the coefficient indicates whether the characteristic benefits KN-SLAM (negative sign) or ORB-SLAM (positive sign). Their absolute value is linked to the relevance of the characteristic in the comparison. This relevance must be interpreted as how much the characteristic contributes to bias the result to KN-SLAM w. r. t. ORB-SLAM in terms of accuracy. However, a lower coefficient by itself does not necessarily mean that one system manages that characteristic worse than the other; it simply means that the characteristic does not explain the differences between both systems. The hyperplane created by SVM indicates that a weighted combination of several challenges is what makes KN-SLAM more suitable than ORB-SLAM for a sequence, or vice versa. Certain characteristics influence more in that decision than others, according to the coefficients in the

CHARACTERISTIC	SVM	LOGREG	MLP
Fast movements	0.69	1.08	1.01
Shaking movements	-0.08	0	-0.01
Close range	0.37	0.56	0.49
Rotational movement	-0.6	-0.77	-0.56
Translational movement	-0.43	-0.71	-0.4
Dynamic scenes	1.2	1.81	1.41
Visual loops	-0.48	-0.64	-0.48
High-frequency textures	0.67	1	0.91
Low-lighting	-1.12	-1.75	-1.66

TABLE 3.6: Coefficients obtained by the SVM classifier trained with comparable sequences. Each row holds the coefficients associated to each characteristic of the sequences. In the training dataset, the class of a sequence is assigned according to the difference between the ATE of ORB-SLAM and the ATE of KN-SLAM. The table is ordered in ascending order.

weighted combination. Therefore, we can conclude that KN-SLAM achieves more accurate trajectories in sequences with

- a good balance between translational and rotational movements, with a strong presence of both of them;
- visual loops at some points of the trajectory (especially at the end);
- consecutive frames with low-lighting images and low contrast between pixels;
- highly textured regions, i. e., the materials of the objects have non-repetitive textures;
- a steady camera, exceptionally allowing shaking movements;
- slow motion between consecutive frames;
- no moving objects between the camera and the scene;
- a relatively large distance between the camera and the scene.

We have trained other classifiers based on linear models to evaluate the coherence and consistency of our analysis. The Multilayer Perceptron (MLP) [109] and Logistic Regression [59] have been examined with this goal. The coefficients achieved by these classifiers are shown in the third and fourth column of Table 3.6. The MLP classifier has been trained with $\alpha = 0.01$ to obtain an accuracy of 97 % when it is evaluated with all the sequences. The Logistic Regression classifier uses a L1 penalty function and an inverse regularization strength $C = 2.0$. This last parameter has the same meaning than the regularization or penalty parameter C of SVM but on a different scale. Although the order of the coefficients differs slightly between the three models, they agree in the relative significance. This significance is created by grouping the characteristics according to the absolute value of their respective coefficients in three different categories:

ALMOST IRRELEVANT shaking movements of the camera;

QUITE RELEVANT fast movements, high-frequency textures, close-range scenes, translational and rotational movements, the presence of visual loops; and

VERY RELEVANT dynamic scenes, low-lighting environments.

The sign of these coefficients must be interpreted as follows: in those sequences where the challenges corresponding to the negative coefficients have a higher weight than the challenges corresponding to the positive coefficients, KN-SLAM achieves more accurate trajectories than ORB-SLAM. According to the coefficients shown in Table 3.6, the amount of shaking movements of the camera is nearly irrelevant to discriminate between KN-SLAM or ORB-SLAM. In other words, the differences in the accuracy between both approaches are not determined by this characteristic.

The low relevance of **shaking movements** means that quick and short camera movements affect similarly to both systems. This is logical because the map points supporting the tracking between consecutive frames are well-established by both systems, i. e., they are seen at least by more than two keyframes. Our more exhaustive strategy for guided matching does not introduce any improvement in the accuracy of sequences where the camera shakes.

There are three challenging situations where KN-SLAM achieves more accurate results than ORB-SLAM: the presence of strong rotations and translations in the camera motion and the presence and detection of visual loops.

The presence of strong **rotational/translational movements** benefits KN-SLAM. Besides, the fact that both movements have a similar weight means that the best results are achieved in sequences where both kinds of movements are performed by the camera. The main cause is the edge-pruning mechanism introduced in the connectivity graph by KN-SLAM. Strong rotations are, in general, difficult to track by monocular SLAM, especially when the camera rotates around the optical axis. To deal with this challenge, ORB-SLAM uses a spawning policy that introduces keyframes as fast as possible. Thanks to our limitation in the number of keyframes linked to the reference keyframe, the local BA achieves better results. The keyframes involved in this optimization share most of the observations viewed in the reference keyframe, but the keyframes with fewer observations are not included. The optimization is less constraint to previous keyframes that could be possibly bad estimates. Moreover, the side effect of the accumulated drift can be corrected later in a possible loop closure.

Precisely, the presence of visual loops also benefits KN-SLAM. As explained in the previous paragraph, this is caused by the inclusion of a more limited local map in the local BA optimization. However, our contributions to the loop detection procedure are key to this improvement. The use of smarter thresholds for searching more candidates without including false positives allows KN-SLAM to detect more loops than ORB-SLAM. The `kitti/09` sequence of the TUM RGB-D dataset is a good example. This sequence appears as an outlier in Figure 3.8 because ORB-SLAM is not able to detect the final loop closure (in some cases it even does not completely track the whole sequence), whereas KN-SLAM detects the final loop in all executions where the tracking is not lost. What is more important, in sequences where a loop is detected by both systems so their ATE can be considered comparable, KN-SLAM outperforms the accuracy of ORB-SLAM. This can be explained again by the removal of some connections in the connectivity graph. It simplifies the optimization procedure in long trajectories where the number of shared features is high enough[20]. Visual loops are prone to appear in sequences recorded in natural conditions, or they can be imposed to the trajectory.

On the contrary, there are other three challenging situations where KN-SLAM behaves worse than ORB-SLAM: the camera travels fast around the scene, the objects of the scene have homogeneous textures and the distance between the camera, and the scene is too short.

Each feature should not be interpreted in an isolated way. Instead, we must consider the combination of these three features to analyze why KN-SLAM achieves worse results. Despite our improvements in the guide matching procedure, the combination of fast motion with close ranges in low-texture environments is managed better by ORB-SLAM.

When the camera performs **fast movements** between keyframes the tracker has fewer observations to accurately determine the map point position. Thus, the remaining map points must have the higher possible accuracy. Indeed, the accuracy of the estimated pose is directly related to the number of tracked features. Thus, if the number of map points included in the local BA is too low, it is likely that the estimated pose had a higher error. If this number degrades below a threshold, the tracking can be lost. Even though the minimum threshold is not reached, the accumulated drift affects the accuracy. The effect is reinforced in the absence of loop closures. We can validate this assumption by looking at the average number of edges in sequences with these characteristics, which is always above the limit of 20 imposed by our algorithm.

The argument stated in the previous paragraph is also valid for **closed-range** sequences and **high-frequency textures**. In the first case, the overlap between consecutive keyframes decreases and fewer inliers are available to estimate the camera pose, so they should have high quality. In the second case, it is difficult that any indirect SLAM system detects a set of tractable keypoints. The absence of enough keypoints to track leads to lose the tracking between consecutive keyframes. Recall that direct approaches bypass this issue by treating direct pixel intensities with the inconvenience that they are more sensitive to image noise and blur. Therefore, again fewer map points are detected and, if the remaining points are not well-optimized, the pose estimate is less accurate. In general, we can affirm that a lower number of points in the local map (i. e., the map corresponding to the reference keyframe) deteriorate the results achieved by KN-SLAM.

20 Note that the essential graph used in loop optimization takes the edges with $\beta_{obs} > 100$

We have to take into account that sequences with a shorter distance between the object and the camera (**close range**) are also affected by our more restrictive policy in the connectivity graph. The lower overlap between consecutive keyframes increases the pace of insertion of new keyframes. However, the low number of shared observations usually discards the edge between keyframes. Although this also affects to ORB-SLAM, it can be counterbalanced by linking more than 20 keyframes. These keyframes are usually spatially close enough to the current keyframe, but they are far away in terms of the time of insertion. The rejection of these edges between spatially-similar keyframes adds an accumulated drift in the keyframes created later when the camera moves away from the close-range subsequence. This is an additional cause of underperformance of KN-SLAM in this sort of sequences.

Finally, the most meaningful characteristics to discriminate the results for a sequence are dynamic scenes and low-lighting environments. For the first one, ORB-SLAM outperforms KN-SLAM whereas for the second one KN-SLAM outperforms ORB-SLAM.

The presence of **dynamic objects** affects negatively to KN-SLAM. Any dynamic object causes that a region of the frame cannot be tracked w. r. t. the previous keyframe. This implies a lower number of points in the local map and the argument stated in the paragraph above can be applied. This partially explains the differences between both systems in sequences with dynamic objects, but the accuracy is also affected by the performance of the relocalization procedure. This procedure is launched after the loss of the tracking. It actively searches an existing keyframe visually similar to the current frame for establishing the camera pose. The neighboring keyframes in the covisibility graph are included in this search, and they also participate in the PnP optimization to align both keyframes. Since we are connecting fewer keyframes than ORB-SLAM, it is more difficult to find the correct alignment. Thus, the higher the connectivity of the graph, the more accurate the estimated pose. Furthermore, there are two possible causes for relocalization: the loss of one or more frames, or the loss of most of the keypoints due to the intersection of an object between the camera center and the map point. Our experimental results have shown that KN-SLAM is more affected by the former one.

The presence of **low lighting** environments increases the probability to achieve a better tracking with KN-SLAM than with ORB-SLAM. Even though ORB detector is really discriminative and works in low contrast environments, features detected in these environments are more difficult to discriminate. Our contributions to the guided matching algorithm ensure a higher quality of the matchings between keypoints and map points. These high-quality matches allow the system to preserve an acceptable accuracy, even though the number of map points is reduced. At least the accuracy is preserved better by KN-SLAM than ORB-SLAM, which discards low-quality matchings in RANSAC iterations. They are labeled as outliers by the optimization procedure when the matching is not well-determined. Although this is the most discriminative characteristic, it must be interpreted beside the other ones. There are only 15 sequences introducing this challenge, all of them in the EuRoC and KITTI datasets. The results show that KN-SLAM performs better then

ORB-SLAM in 11 out of 15, which gives an idea of how this characteristic is significant but it does not determine by itself the best SLAM system.

3.4 CONCLUSIONS

We have implemented KN-SLAM, an indirect monocular SLAM system able to track the trajectory of a single camera while a sparse map of the scene is being reconstructed. Our system is based on the work developed in ORB-SLAM. However, KN-SLAM introduces some novelties:

- It uses a connectivity graph where the connectivity is limited by the number of represented observation already included. This limit is also clamped between a minimum and maximum vertex degree. The number of edges is reduced up to a 75 % depending on the sequence.
- It loads a reduced version of the ORB vocabulary, which is also loaded faster to reduce the bootstrapping times of the system. Loading times have been reduced from 8 s to 0.80 s.
- It tracks more frames thanks to adaptive thresholds in the bootstrapping procedure that decay with the number of failures.
- It detects more visual loops without increasing the number of false positives, using a set of lower thresholds for inliers that vary depending on the results in the previous step.

We have included in our evaluation sequences from four datasets developed for different purposes. Sequences have been normalized following the file structure and directory layout of the TUM RGB-D dataset to be able to reuse them for testing several SLAM systems. Ground truths have been also transformed when the camera coordinate system does not match the coordinate system of the tracker. These sequences are left publicly available for the benefit of the community.

Our system has been evaluated in the 47 sequences of the four datasets that are suitable for monocular SLAM. Our evaluation compares several metrics with the implementation provided by ORB-SLAM, focusing mainly on the robustness of the tracking and the accuracy of the trajectory. Sequences with an insufficient number of keyframes or multiple tracking failures in consecutive executions of one SLAM system are considered incomparable. The number of sequences tracked without failure by KN-SLAM is 46, whereas ORB-SLAM tracks 39. Thus, a total of 39 sequences are considered comparable in our evaluation.

It can be argued that the number of sequences is not high enough to obtain reliable results. However, most of the state of the art evaluations have been conducted using a specific dataset or indeed the subset of sequences that better adapt to the purpose of the evaluated algorithm. We have included some sequences not appropriate for monocular SLAM to evaluate the robustness of KN-SLAM. Every included sequence introduces particular aspects beyond its challenges. Such aspects are displayed in our characterization since we see interesting to evaluate the extents of their effect on the accuracy of the trajectory. Moreover, the number of sequences is not a problem to apply the SVM classifier since the underlying mathematical model suits perfectly to a low number of instances.

Our results show that KN-SLAM outperforms the robustness of ORB-SLAM in 29 of the 47 sequences with an average 86 % of tracked trajectory. If the 47 are considered, KN-SLAM tracks 86 % of the frames while ORB-SLAM tracks the 75 %. In terms of accuracy, KN-SLAM achieves a lower ATE in 25 of the 39 comparable sequences, with an average 14 % of improvement in the ATE. In this subset of sequences KN-SLAM also tracks an average 6 % more frames than ORB-SLAM.

Each sequence presents multiple challenging characteristics for monocular SLAM that enrich its evaluation. We have labeled the sequences on a five-level scale for each characteristic, in order to discriminate the result achieved in our evaluation. These results show that the accuracy of the trajectory is improved depending on the characteristics of the sequence. We have used the characterization of the sequences in a SVM classifier to decide what type of sequence is better handled by KN-SLAM w. r. t. ORB-SLAM in terms of accuracy. We have concluded that KN-SLAM achieves better accuracy in sequences

- with a good balance between camera translations and rotations, several visual loops and recorded in low-lighting environments; and
- without moving objects, high-frequency textures, fast movements of the camera or short distance between the camera and the scene.

3.4.1 *Compliance with objectives*

The objectives of our research were initially stated in Section 1.2. This section is devoted to evaluate and review the degree of fulfillment of these initial objectives according to the final results shown in Section 3.3.

Increase the Percentage of Tracked Trajectory

Our experiments show an average 11 % increase in PTT for all the sequences (86 % for KN-SLAM, 75 % for ORB-SLAM). Furthermore, the difference in the PTT is also positive in those sequences where the ATE is also improved by KN-SLAM (84 % for KN-SLAM, 77 % for ORB-SLAM).

There are some specially challenging sequences, such as `tum/fr1_floor` or `icl/kt3`), where ORB-SLAM refuses to initialize in all executions. However, KN-SLAM is able to bootstrap in these sequences too, including in the challenging case of `tum/fr3_nstr_tex_far`. Sequences where the bootstrapping procedure of ORB-SLAM usually fails are characterized by fast motions, high-frequency textures and low variance in the depth of the points for the initial map, which leads to corrupted maps. Actually, 5 out of 8 of these sequences belongs to the ICL-NUIM dataset.

As a general remark, we have observed that KN-SLAM bootstrap sooner in 5 out of 10 sequences of the KITTI dataset. If we dive into the differences between the first frame tracked by each system, we see that it is only one frame. The only exception is the `kitti/03` sequences, where the KN-SLAM does not bootstrap until the ninth frame. This is due to an outlier in one of the 11 experiments. Nevertheless, we can say that KN-SLAM start sooner or at the same time as ORB-SLAM.

Bootstrap as soon as possible

The frame at which a system starts to track a sequence can be evaluated by looking at the frame - timestamp associations of the sequence. We search for the index of the first association corresponding to the first keyframe of the resulting trajectory. We rely on these associations since the ground truth usually includes a different set of frames.

The first frame of a sequence is defined as the mean of the starting keyframe for all the executions in that sequence. The first frame is not comparable when one of the evaluated systems does not bootstrap the sequence. In our experiments, we have checked that KN-SLAM starts sooner than ORB-SLAM in 22 sequences. On the contrary, ORB-SLAM initializes sooner in 8 sequences. There are 9 sequences where both systems start at the same frame and 8 incomparable sequences.

Adaptive strategy for visual loop detection

This objective is considered fulfilled thanks to the broad variety of sequences in which KN-SLAM has been evaluated. In these sequences, KN-SLAM detects more loops than ORB-SLAM without inserting false positives. It also adapts to the environment of the scene, as it is shown by the sequence `kitti/09`. This is the most representative example of our improvements due to the relevance of the longitude of the trajectory and the amount of accumulated drift.

Reduce the trajectory error in natural conditions

A sequence recorded in natural conditions is characterized by slow motion without shaking, the presence of translational and rotational movements (with an eventual loop) and the absence of artifacts like high-frequency textures or low-lighting environments. As it has been explained in Section 3.3, these sequences are better tracked by KN-SLAM in terms of ATE and PTT. Good examples of these sequences are `tum/fr2_desk`, `euroc/MH01` or `icl/kt2`.

In general, a sequence recorded in natural conditions is a sequence free of challenging characteristics, i. e., special situations introduced for testing the robustness of a SLAM system. Although every dataset includes a set of "easy-to-track" sequences for debugging purposes, the existence of a dataset with sequences recorded in natural conditions would be helpful. In this sense, the closest dataset is KITTI, since its sequences have been recorded in real-world conditions.

Deal with more sequences

We aim at building a system that is robust against exceptionally challenging situations in order to handle any sort of sequence. These sequences include a challenging bootstrapping due to high-frequency texture or low-parallax

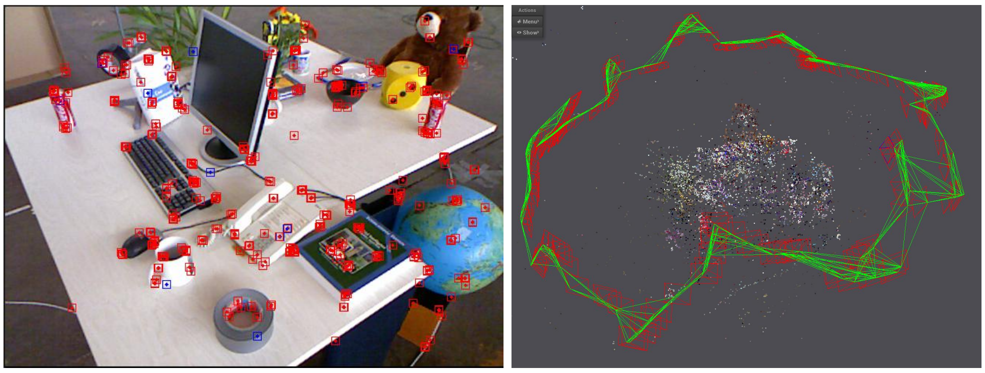


FIGURE 3.12: The colored map (sparse point cloud) generated for the sequence `tum/fr2_desk` with 2964 frames and 7868 points.

movements of the camera. Our experiments show that KN-SLAM tracks 46 of 47 sequences, while ORB-SLAM only tracks 39. Furthermore, KN-SLAM tracks more than 50 % of the frames in 43 out of 47 sequences, whereas ORB-SLAM does the same in 37 of 47 sequences. Therefore, taking into account the diversity of datasets and the sequences contained in them, we can affirm that KN-SLAM deals with more diversity regarding scene environments and sequences.

Colored map

The goal is to include RGB information of the map points tracked by the system. This information allows a viewer to distinguish between the objects of the scene by the color of their points. The color of the images is not stored by ORB-SLAM in the tracked keyframes. Nevertheless, KN-SLAM implements an optional OpenGL-based GUI that displays information about the current frame, the keyframe, the trajectory and the points in the map. This map is represented as a colored point cloud where each point is rendered with a size of one pixel.

We save the image corresponding to each keyframe to recover the color of a map point. This is accomplished by averaging the color of the observations of the map point. A median filter allows us to deal with noise and blur in the original keyframes where the map point is observed. The Figure 3.12 shows a screenshot of the GUI with information about the last frame of the sequence `tum/fr2_desk` and the point cloud. The information about the frame is overlayed at the bottom of the window that renders the current frame. The colored point cloud representing the map is rendered in the background of the window with a point size of one pixel.

Reduce booting times

One of the main problems of ORB-SLAM is the time required to load the ORB vocabulary, which is internally used for relocalization and loop detection. In our contributions, we have implemented a faster loader that uses a serialized version of the visual vocabulary using protocol buffers (see Section 3.1.1). Loading times in our experiments have been reduced from 8.96 s to 0.79 s, which represent a reduction in one order of magnitude. These times

	KN-SLAM	ORB-SLAM
At the map creation	717	503
At the end of tracking	869	541
Number of keyframes	65	60
Average / keyframe	2.30	0.65

TABLE 3.7: Mean memory footprint at different moments for the sequence `tum/fr1_desk` (in MB). The average per keyframe can be extrapolated to other sequences.

correspond to the mean of 11 consecutive executions of each system. They do not depend on the sequence, so we considered this objective as fulfilled.

Reduce the memory footprint

Our principal goal is to promote the adoption of SLAM in devices with moderate computing resources. Thus, we aim at working with the minimum CPU consumption and a small memory footprint. A side effect of reducing the number of edges in the connectivity graph is the increase in the number of keyframes. When the culling procedure search for redundant keyframes it uses the connectivity graph to determine the neighbors of the current keyframe. Therefore, a less-connected graph usually implies the insertion of more keyframes, although it finally depends on the trajectory.

The increase in the number of keyframes in the connectivity graph impact on the memory footprint. This impact depends on the memory required for each keyframe. One contribution in KN-SLAM is the preservation of the color of the map. As we have mentioned a few paragraphs above, the color of a map point is determined by interpolating between its observations. This requires to maintain the RGB image in memory for each keyframe, which increases significantly the memory per keyframe.

According to our experimental results shown in Table 3.7, the average memory required per keyframe have increased from an average of 0.65 MB per keyframe in ORB-SLAM to an average of 2.30 MB per keyframe in KN-SLAM. These results have been obtained for the sequence `tum/fr1_desk`, but they can be extrapolated to the rest of the sequences since the information stored for each keyframe is independent of the sequence. Thus, it is also likely that longer sequences would produce a higher memory footprint due to the increased number of keyframes to describe it.

The additional 215 MB of memory required when the system starts is due to the additional libraries included for the visualization of the frame, the connectivity graph and the map of the scene.

3.4.2 *Future work*

This section includes some of the ideas and contributions left aside due to our limited resources to carry on the required research.

Firstly, our evaluation could be improved by considering the impact of removing connections in the connectivity graph at each step of the SLAM pipeline, instead of the system as a whole. New metrics may be defined, and the results can be interpreted at a stage level and at a global level. We have specified some of the observed effects in Section 3.3, but a thorough study would be required.

As our results have shown, an accurate position of the map point influences the accuracy of the pose and the robustness of the tracker. Although local BA is usually enough to create an accurate map in few iterations, a probabilistic mapping framework might achieve faster convergence in the estimated depth of several observations. This idea can be applied in the mapping thread of KN-SLAM to increase the robustness of the tracking.

The problem of tracking low-textured environment is inherent to indirect SLAM. However, direct methods could be combined with indirect methods as a fallback when the tracking is lost. This increases the complexity of the system but makes it more robust to every type of environment. Furthermore, it does not necessarily increase the computational cost since the fallback procedure would be invoked only when the tracking is lost. The tracking would be performed w. r. t. the reference keyframe, which must have a short baseline with the current frame.

Our evaluation could be easily extended to decide which system would achieve better accuracy in an unknown sequence. To achieve this, the characteristics of the new sequence should be assessed. This is the intended application of machine learning algorithms used for supervised classification. However, we have not evaluated the accuracy of the classifier outside the training dataset. Besides, our training dataset is too small and contains few sequences recorded in natural conditions to generalize our results in other sequences. Nevertheless, it would be nice to have a model capable of deciding which is the best SLAM system to use. The required data can be collected if the system were used in real-world applications.

One of the main drawbacks of our approach is the increment of the memory footprint. The culling procedure does not include the second level neighbors when searching for redundant keyframes so in a less-connected graph some keyframes are maintained. This could be fixed by modifying the culling procedure. However, this also reduces the accuracy reached with our approach. The tradeoff between accuracy and performance is one of the keys to designing a SLAM system. A deeper study of the relation between the number of keyframes and the ATE is left as future work.

The trajectory and the map retrieved by SLAM are useful for many applications (e. g., VR/AR). Furthermore, the keyframes and the point cloud allows us to augment the original model with an additional 3D reconstruction that provides refined information about the scene. This information is required as input for the basic tasks and applications of several industries such as architecture, video games, audiovisual production, etc. Next chapter is devoted to exemplifying this application to the AEC industry.

4 BUILDING INFORMATION MODELLING

The Architecture Engineering and Construction (AEC) industry assembles a big number of stakeholders coming from different fields and professions. In any building project, they have to collaborate across the different phases of the lifecycle of the project, from design to retrofitting. The best way to collaborate is working on a set of standards for representing the structural parts and the constructive processes. Then, stakeholders can exchange information about the building to enrich the corresponding tasks assigned to each stakeholder.

The BIM methodology surged in an attempt to facilitate the collaboration between different stakeholders. This methodology encourages the stakeholders to share the building information across its entire lifecycle. This information includes the processes involved in the scheduled planning of the construction. Scheduled tasks can be tracked in time (i. e., checking the current date and the scheduled deadline) and space (i. e., matching the current 3D reconstruction against the original 3D model). The planning can also be used to simulate the state of the building at a specific time as a snapshot. Furthermore, the advance of the construction can also be simulated by transitioning between consecutive stages and tasks.

Despite the benefits stated in the previous paragraph, the adoption of BIM is being slow. Indeed, its main promoters are the public governments, in an effort to reduce the maintenance costs of the building project along its lifecycle. The main reasons for this lazy adoption are the initial investment and the complexity of the tools to implement this new methodology. This barrier can be reduced by developing applications within a limited scope of BIM. An intuitive visualization and a simple creation of BIM projects would help to accelerate the adoption. Ubiquity is another important trait, so the proposed solution should be accessible from different devices too. The visualization of the building model can be addressed with the new Web3D technologies (see Section 2.7.2).

In this chapter, we dive into 3D-SIMOS, the third component of our reference framework (see Section 1.1). A subset of the results introduced here have been previously published in [29]. This component is an advanced visualization software that integrates the geometry of the CAD model of the building with its planning in a single project following the BIM methodology. The resulting project can be visualized in any modern web browser without additional plugins. The software has been designed to be easy to use in order to foster the BIM adoption among the AEC stakeholders. The use of standard technologies, open source software and standard file formats facilitates the extension of the system to support future functionalities.

The rest of the chapter is organized following the same scheme as Chapter 3. Section 4.1 introduces our contributions concerning the representation, alignment, and simulation of tasks on buildings. It also details the symbolic representation that supports both processes and building spaces. In addition, the design of 3D-SIMOS is described at the end of the section. Section 4.2 specifies the experimental methodology followed to evaluate our results. Next, Section 4.3 announces, analyzes and discusses the results. Finally, Section 4.4 summarizes the conclusions of this part of our research and enumerates open challenges to be researched in the future.

4.1 CONTRIBUTIONS

The representation and management of 3D information is a meaningful problem concerning computer graphics and 3D reconstruction. Highly repetitive spaces can be modeled with cuboids as the most basic constructive blocks. Irregularities in the spatial organization can be modeled with cuboidal maps (i. e., the image of a cuboid). These cuboids can be patched and recombined along cuboidal chains. These chains can be evaluated by functionals, which are attached to each chain using combinatorics on the faces of the cuboids. This contribution is outlined in Section 4.1.1.

The underlying model of BIM is represented by the Industry Foundation Classes (IFC). It allows us to design new procedures that take advantage of combining constructive elements and processes around the building. We have formulated a new procedure to manage the geometry and the tasks of the building using a graph-based symbolic representation. Structural (i. e., objects and relations) and functional (i. e., generalized functions) aspects must be combined to achieve this goal. Processes of the AEC industry can be validated and tracked using this representation. Further details about this procedure are provided in Section 4.1.2, which is based on the properties of the graph introduced in Section 4.1.1.

The traditional workflow using CAD defines the planning and the geometry of the building in separated files. As we have mentioned in Section 2.7, the AEC stakeholders are progressively transitioning from CAD to BIM. However, the adoption of BIM is still limited, and there are no easy ways to migrate a project from CAD to BIM. To alleviate this problem, we have developed an alignment between the tasks of the planning and the geometry of the building to track the construction advances. The results of the alignment are represented using the IFC to make them compatible with the rest of BIM models. We explain this process in Section 4.1.3.

We have implemented our contributions in 3D-SIMOS, a Web3D application to track, showcase and simulate the geometry and the planning of the building in any WebGL-capable web browser. Real-time information about the current state of the building can be merged with the designed information for accurate reporting. The application also provides the user with advanced visualization functionalities to inspect the building. We dive into the details concerning the design and implementation of this application in Section 4.1.4.

4.1.1 *Cuboid-based modeling*

Object-based models can be managed using symbolic representations linked to the morphology of the objects. The evolution of these objects (i. e., their changes over time) can be represented by a flow, which is discretized with a scalar, vector or tensor field. In this work we have focused on the attributes or properties linked to the processes, which are represented by scalar fields.

Three-dimensional models of buildings can be segmented in a collection of interconnected spaces. These spaces can be modeled in terms of basic geometric primitives like cuboids and their constituent components. Indeed, it is possible to use any topologically-equivalent variation of these primitives, which can be created using a continuous map instead of a conventional linear map. The spaces of the building can be viewed as cells, which are composed by a set of connected cuboids whose boundary forms a cuboidal complex (see e. g., [117]). This representation is also compatible with curved surfaces and with holes in the surface. Cuboids and their possible degenerations (i. e., prisms and tetrahedra) are assembled along their boundaries to represent built spaces in AEC environments.

One of the main advantages of cuboidal representations is that cuboids can be managed using contraction/expansion transformations defined by a combinatorial kernel. This is a particular case of the discrete version of blow-ups and blow-downs used in Algebraic Geometry. These transformations will be detailed in the next section.

A *PL-cuboid* is the image of a cube by an affine map, whereas a *Piecewise Smooth (PS)-cuboid* is the image of a cube by a continuous map. For the sake of simplicity, we constrain ourselves to structures given by PL-cuboids and their adjacency and incidence relations. Indeed, both types of structures can be managed using the same combinatorics.

A symbolic representation

Graph-based representations are commonly used in GIS and BIM. As we have stated above, any constructive space can be represented by an adjacency graph of cuboidal decompositions of its indoor and outdoor spaces. This graph has associated a *dual adjacency graph* $\mathcal{G}$, where each node $n_i \in \mathcal{G}$ corresponds to the centroid of each cuboid or cubical cell. Edges $e_{ij} =< n_i, n_j >\subset \mathcal{G}$ correspond to adjacent cubical cells that bound the filled workspace. Finally, faces $f_{ijk...m} = \{e_{ij}, e_{jk}, \dots, e_{mi}\} \subset \mathcal{G}$ correspond to circular paths of adjacent cubical cells in any spatial direction representing open spaces[1]. The main advantage of using this representation is that classical topological tools can be applied for information management.

Any cubical representation supports recursive grouping and subdivisions, which can be efficiently managed with an octree [86]. There is a finite number of levels of depth corresponding to successive subdivisions. The boundary operator links each cubical cell with its faces (e. g., slabs, roof, walls), faces with edges (columns), and edges with vertices (corners or joints). This operator can be extended by linearity to cubical complexes representing other objects. Subdivision procedures can be used to add elements such as doors or windows

1 Outdoor spaces in walls, floors, and roofs are included in CityGML, but not in IFC.

for faces, or installation networks for enlarged edges. Any functional can be defined over a cuboid to model specific attributes of the represented element.

Cubical representations have already been proposed by other authors. For instance, utility networks have been designed with graphs in [57], where the features of the network are modeled as junctions. In this case, a pipeline is described by a graph with the form of a polygonal chain. In this work, certain nodes include embedded components as subgraphs that can be expanded or contracted. With this representation, the manager can manage the network and the processes associated with physical elements.

Moreover, the topology of the representation is flexible enough to adapt to evolving coarse models, as it has been shown by Zlatanova, Rahman, and Shi [138]. The main advantage is that well-known topological tools are suitable for set-theoretical, functional and symbolic representations. Hence, they can be adapted to objects, fields, and graphs. Furthermore, topology provides invariants preserved in a large number of graph modifications.

The representation can be extended to annotated graphs to include labels and pointers. Labels can represent construction metadata linked to the corresponding ontology. Pointers refer to the localization of the physical elements represented by nodes and edges. Each element is associated with its adjacent elements by the following inclusion relation:

$$p_i^{(0)} \in s_{ij}^{(1)} \subset f_{ijk\ldots}^{(2)} \subset V_{ijk\ell\ldots}^{(3)},$$

where the superindex denotes the dimension of points, segments, faces, and volumes of a cuboid. They verify the following incidence conditions:

- Two cuboids share a face: $V_{ijk\ell\ldots}^{(3)} \cap V_{ijkm\ldots}^{(3)} = f_{ijk}$
- Two faces share a segment: $f_{ijk}^{(2)} \cap f_{ijh}^{(2)} = s_{ij}^{(1)}$
- Two segments share a common vertex: $s_{ij}^{(1)} \cap s_{ik}^{(1)} = p_i^{(0)}$

These elements can be represented by a triple connected list with additional attributes/labels related to embedded information.

4.1.2 *Process management with graphs*

Symbolic representations described in the previous section are useful for managing the processes concerning the building. From the topological viewpoint, the optimal design of the graph outlines two problems concerning the tasks to represent:

- Planning tasks are solved by converting the graph in a tree. The Minimum Spanning Tree (MST) algorithm can be used, for instance.
- Execution phases are solved by representing the pipeline of tasks with a flowchart (see [66] and references therein). Tasks are interpreted as evolving multi-paths $\Gamma = (\gamma_1, \ldots, \gamma_a(t))$ submitted to different kinds of constraints $G = (g_1, \ldots, g_s)$.

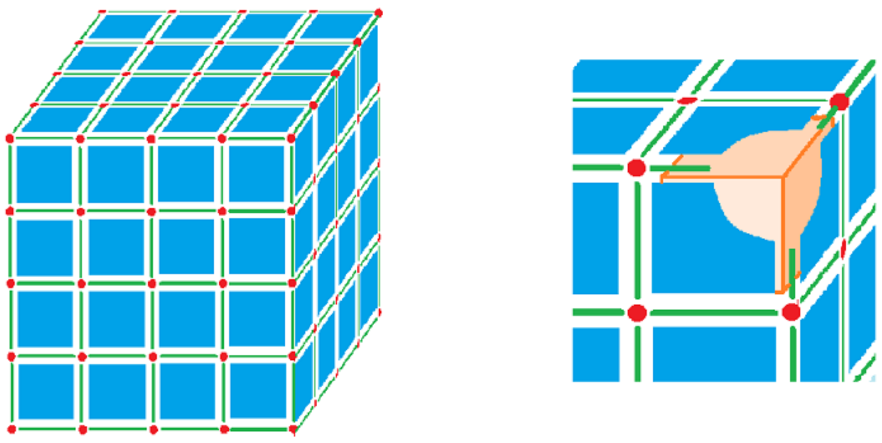


FIGURE 4.1: A cubical complex is defined by different cell types: 0-cells or vertex (red), 1-cells or edge (green), 2-cells or pixel (blue), 3-cells or voxel (orange).

The graph becomes non-static when it is used along the building lifespan. An evolving model must be developed to manage incidences as changing graph features. The expansion or contraction of subgraphs at different levels of depth lift these subgraphs to a hypergraph for lowering redundancy in symbolic representations. Adjacency graphs can be embedded at different Level of Detail (LoD) to provide a spatio-temporal representation of cuboidal maps. Thus, construction processes are represented as embedded graphs that vary along time.

A scheduled task can be inserted at each level in graph nodes (involving structural elements, materials, installations) and edges (connecting tasks linked to adjacent cells). Then, ordered subgraphs corresponding to layers can be selected by different criteria (e. g., processes, tasks, activities), and changes like identification/insertion of events can be managed. Again, they are represented as operations and transformations acting on the graph.

The cubical complex

Cubical complexes represent quasi-regular spatial decompositions (see Figure 4.1). A *cubical complex* is composed of a weighted chain of cuboids with embedded components, coupled with the boundary operator (see [1]). Such weights can be associated with the relevance of attributes, such as materials properties, structural behavior, indicators of utility networks, energy efficiency, etc. Processes can be classified by the topological invariants imposed on cubical complexes. Indeed, the management of processes as symbolic information was already presented in [7].

A *volumetric segmentation* of a 3D mesh defines a PL-model. The basic cell of this model is represented by a cuboid. The model is the adjoint of multiple basic cells e^d_α along boundary components ∂e^d_α). In this representation, processes correspond to fields defined on objects where the simplest case is given by functions (i. e., a scalar field). This formulation comes from Algebraic Topology, where a chain (weighted combination of cells) is replaced by a co-chain (weighted combination of functions defined on cells).

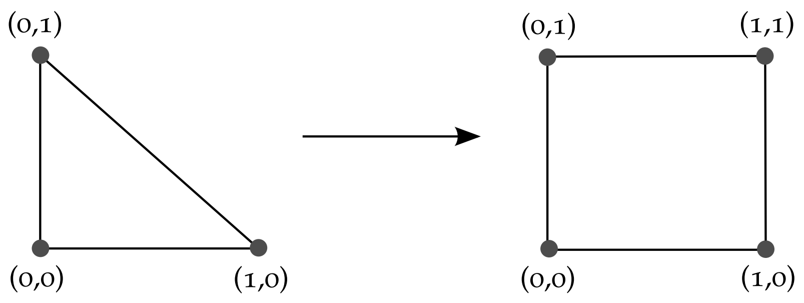


FIGURE 4.2: The 2D simplex can be converted to a unit square through expansions.

Taking into account the wide variety of decoupling strategies and their resulting decompositions, we shall restrict ourselves to very simple objects defined by cuboids. Each basic cell e^d_α (where d is the dimension) is the image by an application φ_α of the standard d-dimensional unit cube C^d, where $0 \leqslant d \leqslant 3$; in particular C^0 is a point, C^1 is the segment $[0,1]$, C^2 is the unit square $[0,1]^2$ whose edges are C^1, and C^3 is the unit cube $[0,1]^3$ whose faces are C^2. In other terms, each d-dimensional unit cube C^d is the convex hull of 2^d vertices having binary coordinates in $\mathbb{R}^d$. This binary scheme induces an incremental ordering to efficiently determine relations. This ordering induces a counterclockwise orientation on each cell when the appropriate sign is selected. This is useful to evaluate operators on more complicated representations.

Hexagonal/triangular prisms can be achieved as simple subdivisions of cubes by diagonal planes. This subdivision operation associates cubic cells with simplicial cells. Hence, these prisms can be understood as degenerated cuboids. Inversely, adjunction of simplicial cells returns cubical cells by passing through the prisms.

Less known is that a d-dimensional simplex σ^d can be converted to a unit cube C^d through a set of controlled expansions for $d \geqslant 1$. Contraction and expansion transformations modify cuboids using morphological operations on graphs, which accumulate components along the boundary of the original cuboid. An expansion operation replicates the opposite side according to the dimension of the pivot element, which is considered the center of the transformation. For instance, the expansion of a regular tetrahedron with the pivot point at its barycenter gives a triangular prism. The expansion of this prism pivoting along its largest line returns a cube (see the whole process in Figure 4.3). Inverse transformations are defined by contraction operators. These transformations allow us to manage scheduled tasks by supporting the update, insertion, and evaluation of any cuboidal complex.

Next paragraphs dive into the details of these transformations applied to two basic cases: the triangle and the tetrahedron.

THE 2D CASE The required steps to perform the transformation in the two-dimensional case are illustrated in Figure 4.2. The standard triangle $\sigma^2 = \mathcal{H}(p_0, p_1, p_2)$ with center $p_2 = (0,1)$ can be expanded by replacing such vertex by a new edge $e_{23} =< (1,0),(1,1) >$ (opposed to $e_{01} =< p_0, p_1 >$) with the opposite orientation to the source edge $e_{01} =< (0,0),(0,1) >$. Here e_{ij} denotes the segment $< p_i, p_j >$ and i, j are written in binary form. Note that

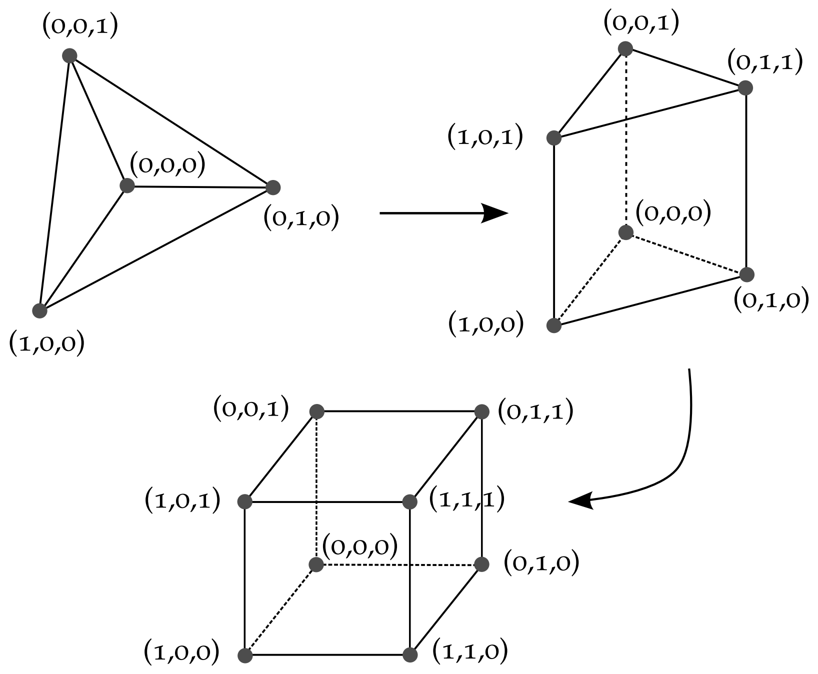


FIGURE 4.3: The 3D simplex (a tetrahedron) can be converted to the unit cube C^3 in two steps.

the coordinates of the new edge are obtained by adding 1 to the coordinates of the original edge. A globally coherent orientation requires the cancellation of the shared edge e_{12}. Thus, we take the oriented triangle T_{123} with the opposite orientation, i. e., $-<p_1, p_2, p_3>$. Then, the adjunction of triangles gives the globally well-oriented unit square.

The final square is decomposed into two triangles by the diagonal d_{12} connecting vertices $p_1 = (1, 0)$ and $p_2 = (0, 1)$. These triangles are positively (i. e., counterclockwise) oriented to cancel the shared edge corresponding to the diagonal. If the diagonal is removed, the coherent orientation is still preserved. The expansion transformation inverts the original orientation.

THE 3D CASE The three-dimensional case describes the conversion of the 3D standard simplex (i. e., a tetrahedron) into the unit cube. It is a natural generalization of the 2D case described in the paragraphs above. The conversion is performed in the two steps illustrated in 4.3:

1. *Conversion of the* 3D *standard simplex to a triangular prism*: The expansion of the standard 3D simplex $\sigma^3 = \mathcal{H}(p_0, p_1, p_2, p_3)$ pivoting at $p_3 = (0, 0, 1)$, replaces this vertex by a new 2D face

$$f_{345} =< (0, 0, 1), (1, 0, 1), (0, 1, 1) >,$$

which is opposite to the source

$$f_{012} =< (0, 0, 0), (1, 0, 0), (0, 1, 0) > .$$

The coordinates of the face can be determined by adding 1 to the vertices of the original face f_{012} where f_{ijk} denotes the segment $< p_i, p_j, p_k >$

and i, j, k are written in binary form. Compatibility between orientations can be achieved by making an elementary transposition.

2. *Conversion of the triangular prism to a unit cube*: The triangular face f_{012} (respectively f_{345}) is expanded at vertices $p_0 = (0, 0, 0)$ (respectively, $p_3 = (0, 0, 1)$). Reusing the argument provided for the 2D case, a vertex $p_6 = (1, 1, 0)$ (respectively $p_7 = (1, 1, 1)$) must be added to transform the triangle T_{012} in the square C_{0126} (respectively the triangle T_{345} in the square C_{3457}).

The unfolding procedure is similar to the planar case but harder to describe. The unfolding process reproduces in an inverse order the computation of the volume of a tetrahedron from its diagonal planes. More concretely, this process decomposes the original cube in two triangular prisms and each prism in three tetrahedra, which gives the well-known formula of the volume of the tetrahedron as $1/6$ of the volume of a parallelepiped.

Representation of building spaces

In structured environments, a multilinear map L_α converts a standard cube in a cuboid (e. g., by modifying the scale for each direction). The basic case corresponds to a cuboid, which is the image of a cube (see Section 4.1.1); depending on the geometric framework, we shall distinguish Euclidean, affine or projective cuboids. Composition rules and elementary operations adapt cuboids to arbitrary shapes since incidence conditions between linear elements are projectively invariants.

Cuboids can have curvilinear elements in non-structured environments. These *curved cuboids* are the image of a cube by a multi-algebraic map φ_α. From the topological viewpoint, the combinatorics of PL-cuboids are the same as the combinatorics of curved cuboids. Furthermore, both of them can be managed with octrees for adaptive subdivision. The tangency conditions (and consequently, traversal properties) are preserved by projective transformations, such as in PL-cuboids.

For example, the product of k low degree algebraic curves is a continuous map: snakes for $k = 1$, B-splines for $k = 2$ or T-splines for $k = 3$ (T as in threefolds). Another example corresponds to quadratic (respectively cubic) arcs, which can be parameterized by the segment whose image is the affine map $t \mapsto (1, t, t^2)$ (respectively $t \mapsto (1, t, t^2, t^3)$). This parametrization only requires three (respectively four) control points and it can be sampled in an arbitrary way depending on the construction needs. This example can be extended to a quadratic (respectively cubic) surface, which is locally parameterized by the product of two affine maps $(u, v) \mapsto (1, u, v, u^2, uv, v^2)$ (respectively $(u, v) \mapsto (1, u, v, u^2, uv, v^2, u^3, u^2v, uv^2, v^3)$). This image provides quadratic (respectively cubic) patches that can be stitched together to provide complex volumetric representations.

Cuboids can be matched along boundaries of cells (2D faces) with opposite orientation to achieve PS-structures like B-spline surfaces. The matching can be performed using flows whose essential elements are topologically equivalent to 2D disks $\mathbb{D}^2$. The result is a cellular cuboid complex, i. e., a collection of cubic cells that are topologically equivalent to e^k_α.

The *boundary map* $\partial : C_k(X) \to C_{k-1}(X)$ links the faces (with the induced orientation) of the k-dimensional cuboids of a chain X. The boundary map is extended by linearity to the *boundary operator* $\partial : C_k(X; \mathbb{Z}) \to C_{k-1}(X; \mathbb{Z})$, where $C^d(X; \mathbb{Z})$ represents all combinations with integer coefficients of d-dimensional basic cuboids on the building X. It can be verified that $\partial^2 = 0$ by choosing the appropriate orientations. This defines topological invariants for X, which are especially useful for controlling the flows interchanged between spaces, such as electricity, heat, water, ventilation, etc.

A *chain of* k*-dimensional cuboids* is a weighted combination of k-dimensional cuboids $\sum_{i=1}^{r} w_i c_i^k$, where $w_i \in \mathbb{R}$ are the weights determined by a path in the space or by natural operations like the boundary operator.

The matching of cells is performed over their boundaries. In the cubical complex a face of a cube is shared by two spaces, an edge is shared by at least two faces, a vertex is shared by at least three edges. Each one of these elements admits different attributes related to processes. Common components of adjacent cells in boundaries ∂e_α^d are identified by the opposite local orientations to preserve the global coherence of the resulting object.

Representation of constructive processes

Cuboids can be labeled as full or empty along the constructive process since the spaces of the building can be temporally empty, but they can be filled at different stages of the construction. Consequently, the topology of the representation changes dynamically.

Each geometric element of a cuboidal cell can represent different attributes: (0) *structural efforts* (compression, traction, torsion) at control points; (1) *distribution of charges* supported by structural elements like columns or arcs; (2) state of *embedded installations* of utility networks at 2D faces like walls, slabs, etc; (3) *comfort performance* measured according to the energy efficiency for non-empty indoor 3D cuboids. All these attributes can be managed with the same representation using combinatorics of k-chains and functionals defined on them (k-cochains for $0 \leqslant k \leqslant 3$). Functionals can have different meanings related to space: geometric (relative to design), material (materials of the objects), analytic (quality control) or economic (inputs and outputs). Their evolution is modeled with flows that evolve over time.

In construction processes, it is essential to evaluate operations from the design phase to the execution phase. This evaluation involves operations that are also modeled as functionals on the chain complexes. This is implemented with a numerical evaluation on the chains, which requires to introduce two basic concepts:

- A *cochain* is a functional $f : C^k(X) \to \mathbb{R}$ defined on any d-dimensional chain $c^d = \sum_{i=1}^{r} w_i C_i^k$, where w_i are the weights for each k-dimensional cuboid C_i^k and r is the length of the chain.
- The dual of the boundary map is the *coboundary map* $\delta : C^{k-1}(X) \to C^k(X)$ between cochains. Thus, the attributes defined as functionals on $(k-1)$-dimensional chains (e. g., faces of cuboids) can be topologically extended to functionals defined on k-dimensional cells, independently of the selected propagation model.

The *coboundary map* δ in the discrete case plays the same role as the differential map in a smooth framework. Thus, the set of cochains supports differential and integral calculus for PL-structures too. More concretely, it provides a discrete version of integral calculus to aggregate planar or volumetric flows.

Advantages of graph representations

Transformations between triangular meshes and quadrilateral meshes connect two different representations arising from unordered vs ordered cases. Quad-based representations are friendlier to use since their edges can be locally parameterized as the product of rational curves. Hence, they can be regularly sampled, supporting the simulation or the evaluation of processes. These transformations are usually applied to improve the topology of 3D meshes resulting from 3D reconstruction. In graph-based representations, these transformations provide a structural link between the basic cells of the tetrahedral and the cuboidal approaches.

The natural orientation induced by the original ordering of vertices defines a route (i. e., an oriented path) for each chain of basic cells. Different attributes can be linked to this route to be evaluated at the workplace with the functionals defined by cochains. Routes can be inspected with the opposite orientation to verify whether the insertion of an attribute is appropriate or not.

In addition, this symbolic representation supports natural operations on graphs such as the insertion or deletion of vertices. The application of the graph transformations is optional. For instance, if we insert a door on a face of a cuboid, this additional element is not necessarily expanded. Furthermore, cuboids support CAD operations (e. g., extrusion), preserve Euclidean properties (e. g., parallelism or perpendicularity), and are compatible with perspective projections.

Note that the graph transformations are a simplification of shuttering work for tied structures employed in some sensitive construction processes. Thus, any refinement of such transformations with additional control elements could be also applied to the original structural workflow.

The graph-based representation described along this section is underneath the IFC, which represent a building in BIM. Taking advantage of this underlying representation we can overlay additional properties and operations as the previously mentioned. Unfortunately, some meaningful information of the building is still isolated in CAD formats. Next section concerns the reconciliation of planning and geometry of the building in the IFC framework, which allows us to exploit the graph transformations over the IFC data structure.

4.1.3 *Alignment between planning and 3D model*

Information systems do not represent conceptual schemas in the same way. Instead, they use different taxonomies and hierarchies that must be reconciled manually. This process is commonly known as ontology alignment (see

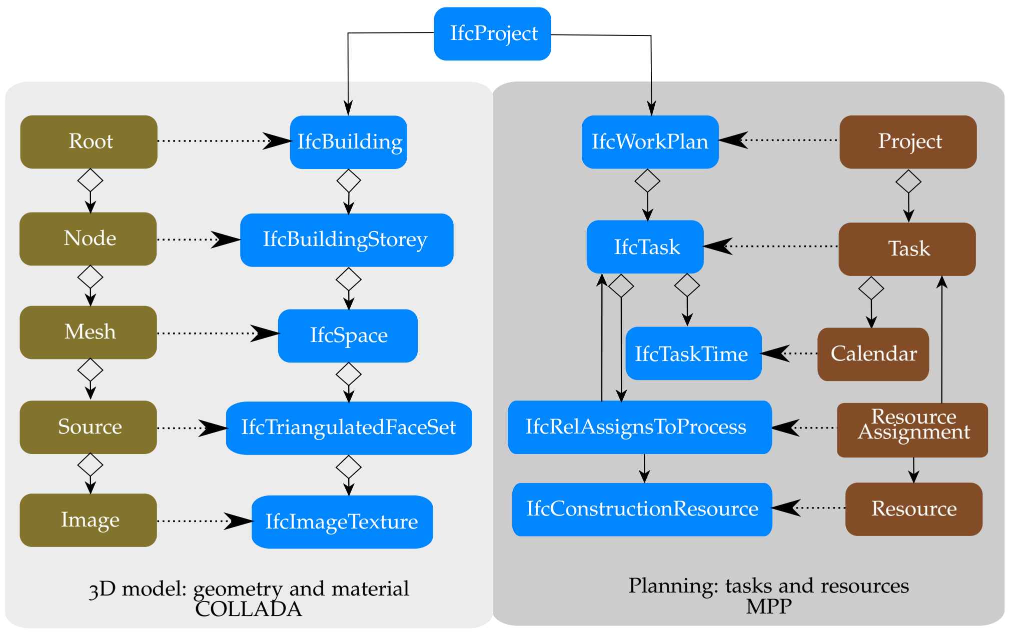


FIGURE 4.4: Planning and 3D model schemas can be aligned with the IFC model entities. The concepts from the COLLADA model are represented at the left side of the figure, whereas the right side shows the concepts from the MPP model.

Section 2.7.1). Its goal is to be able to bridge data represented in one format to the other. The term "alignment" in this section refers to the set of mappings established between semantics or concepts of two different schemas.

In order to ease the migration from CAD to BIM, usual CAD data sources must be aligned (e. g., the planning of the construction, and the CAD design of the building). Fortunately, the IFC provides a unified framework to include information from different data sources. Correspondences between entities from sources and IFC should be manually established since automatic alignments are not recommended when accuracy is required. We have taken advantage of our previous experiences in Ontology Alignment (see [28]) to map the geometry represented in the COLLADA schema and the planning represented in the MPP schema into the IFC. These models have been selected according to the actual requirements of the AEC industry and their condition of open standards (see more details in Section 4.1.4).

The proposed alignment is illustrated in Figure 4.4. Both data sources are gathered to create a new building project. Then, this information is aligned (or mapped) to the entities of the IFC v4 standard[2]. More concretely, we have selected the small subset of 766 IFC entities that represents the information required by 3D-SIMOS. This way, the IFC allows us to extend this information with future data sources. Furthermore, the representation is IFC compliant, which means that the information can be exchanged with other tools following the BIM methodology.

Any building project created in 3D-SIMOS is represented by the root concept of IFC called `IfcProject`. Then, we map the COLLADA and MPP models

2 http://www.buildingsmart-tech.org/ifc/IFC4/final/html/index.htm

to different concepts in the IFC model. Next, we describe the rationale behind each mapping created from COLLADA and MPP to IFC.

The COLLADA schema comprises a wide set of concepts to represent the geometry and the appearance of the model, such as vertex, normal and texture coordinates, animations, materials, etc. Such concepts can be directly mapped into IFC 4, since this new version of the standard supports boundary representation of geometries. In addition, we have linked the conceptual hierarchy of the meshes with the corresponding classes defined by the IFC. More specifically, we have defined the following mappings:

GEOMETRY Triangles of meshes (`Source` in COLLADA) can be converted to `IfcTriangulatedFaceSet` that supports triangular tessellations. A set of `IfcTriangulatedFaceSet` entities forms an `IfcShapeRepresentation`, which corresponds to the geometry linked to any IFC semantic concept. Eventual textures (`Image`) linked to a mesh can be mapped to `IfcImageTexture` that represents a 2-dimensional image-based texture map applied to the geometry of a surface.

SEMANTICS The `Root` of COLLADA is mapped to the `IfcBuilding` since we assume that the 3D model represents a complete building entity. The `Node` elements are mapped to `IfcBuildingStorey` that represents a (nearly) horizontal aggregation of vertically-bound spaces. The last level of the COLLADA tree (`Mesh`) is mapped to `IfcSpace` since the requirements of the application demand a separated mesh for each space. Meshes can be easily merged with any 3D design tool.

The MPP schema includes semantic concepts for planning a building construction. The 3D-SIMOS application only considers tasks with their recursive hierarchy. However, we also provide the mappings for the core concept included in MPP. So the following mappings have been established:

- The root entity of the project file is mapped to `IfcWorkPlan`, which represents a set of scheduled works in construction or facility management project. It also supports references to activities (`IfcTask`) and resources used in the work plan.

- `Task` in MPP is mapped to `IfcTask`, i. e., a unit of work to be carried out in a construction project. An `IfcTask` can nest another task, procedure of event.

- `Resource` in MPP is mapped to a subtype of `IfcConstructionResource`. For example, if the resource is a user, it is mapped to `IfcActor`, which represents human agents involved in a project during its full lifecycle. Then, the actor is linked to the task by the `IfcLaborResource`, which inherits from `IfcConstructionResource`.

- `ResourceAssigment` is mapped to the `IfcRelAssignsToProcess` class. It represents the assignments from one `IfcTask` to the objects where the task operates on or the resources it needs. More specifically, this concept allows us to relate `IfcConstructionResource` with `IfcTask`.

- Finally, `Calendar` is mapped to `IfcTaskTime`, which models the time-related information about a task. It also includes the different types (actual or scheduled) of starting and ending times. This information is useful to determine offsets in the execution of the construction activities.

4.1.4 *3D-SIMOS*

The BIM methodology in AEC environments aims at boosting real-time collaboration around a shared 3D model. Taking into account this goal, we have designed 3D-SIMOS, an easy-to-use application based on the BIM methodology that supports advanced visualization of the building elements. The application allows users to import a non-semantic 3D model designed with CAD and its scheduled planning, which can be merged as it has been explained in Section 4.1.3. Then, the building can be visually inspected by involved stakeholders, such as visitors, constructors or project manager. To make the visualization more appealing, a three-dimensional real-time renderer has been developed. In addition, the application is ubiquitous since it can be accessed through any web browser. In this way, the same model can be used for selling the building and surveying/tracking the evolution of the construction.

We have developed 3D-SIMOS using an iterative and incremental development methodology (see [75]). The whole development has been split into three iterations that have been validated by AEC stakeholders. At the beginning of a new iteration, we fix the bugs and inconsistencies detected at the end of the previous one. New functionalities are developed at each new iteration. The final version is used to evaluate our research, as it will be detailed in Section 4.3.

Requirements

We have listened to the stakeholders in the AEC industry to determine the requirements of 3D-SIMOS (functional and not functional). In a nutshell, they require a simple tool that allows the project manager to visualize the construction progress at the same time that any potential customer can inspect the building to take an overview of the final result. Furthermore, the activities of the construction can be tracked and monitor in real-time supported by a single shared model. General requirements have been enumerated in Section 1.1. In this section, we provide a more detailed description from the perspective of the design. We have identified the following functional requirements:

- The manager can create a new project by importing the planning and a 3D CAD model of the building. The system allows the user to link geometries and tasks.

- The system provides a Graphical User Interface (GUI) for the interactive association (drag and drop) of the different planning stages with the layers of the 3D model.

- The 3D model can be navigated with the mouse and the keyboard to simulate an orbit camera. Mouse movements are used to define rigid transformations on the model (i. e., zoom, translation, and rotation).

- The simulation of the construction is based on the spatio-temporal representation (4D) provided by the planning-geometry associations. The simulation tracks the time required for each task to render progressively the geometry linked to such tasks. The trajectory of the camera can move around the building, animated by spherical linear interpolation.

These requirements are complemented by the following non-functional requirements and constraints:

- The planning of the building is specified in a Microsoft Project file[3]. This format is generally accepted in conventional workflows (not BIM) based on entity modeling.
- The design of the building is represented as a compressed COLLADA file[4]. The system detects automatically the identifiers of the layers, the geometric groups, and the objects.
- The system renders the building in three dimensions in a web browser supporting the WebGL API. Thus, the system is independent of the operating system and the operating devices, although it is dependant on the browser.
- The evolution of the construction tasks can be simulated at 30 Hz.

The functional requirements stated above allow us to identify the following meaningful use cases:

PROJECT CREATION : The user uploads the planning and the 3D model of the building designed following the classical CAD methodology. Then, each planning task is linked to the corresponding layer of the 3D model. The project will be saved by the system for future queries.

INTERACTION WITH THE BUILDING The user navigates and inspects the building with the mouse and/or the keyboard.

VISUALIZATION OF THE MODEL The system displays a real-time render of the 3D building model in the screen, according to the capabilities of the (mobile or desktop) device. Texture information is also included when it is provided with the CAD model.

SIMULATION OF THE CONSTRUCTION : The system lifts at each instant t the geometry of the layers linked to the tasks that should be completed at that instant according to the planning.

Since the binding between tasks and geometries is a vital step for the rest of the application, the most significant use case is "Project creation". The BIM model generated at this use case allows the user to simulate, track, and evaluate constructive processes. This use case can be subdivided into four additional use cases:

3 We use the MPP file format, which is the Microsoft's proprietary way of storing project data. However, the library MPXJ supports other planning formats too (more details in Figure 4.1.4

4 `https://www.khronos.org/collada`

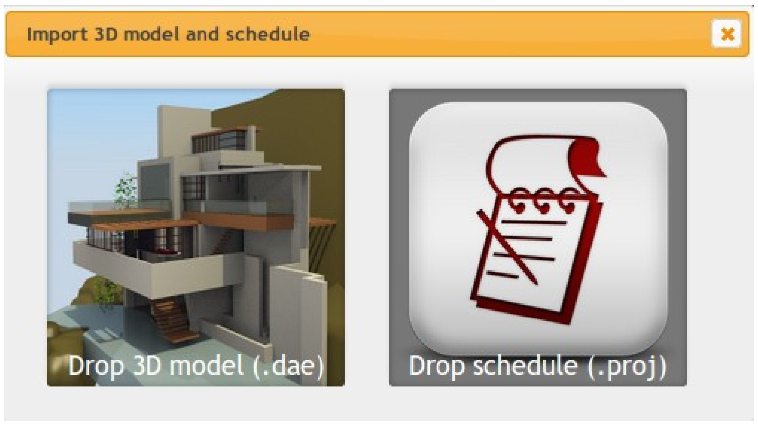


FIGURE 4.5: First dialog shown when a new project is created. It allows the user to import the 3D model and the project planning by dropping the data sources on each corresponding box.

IMPORT THE 3D MODEL : The COLLADA file is parsed to extract the geometry and the hierarchical structure of the model. The geometry and the appearance are converted to a scene graph, which is rendered by the underlying graphics engine.

IMPORT THE PLANNING : The system parses the MPP file to obtain the planning tree composed of a set of nested tasks.

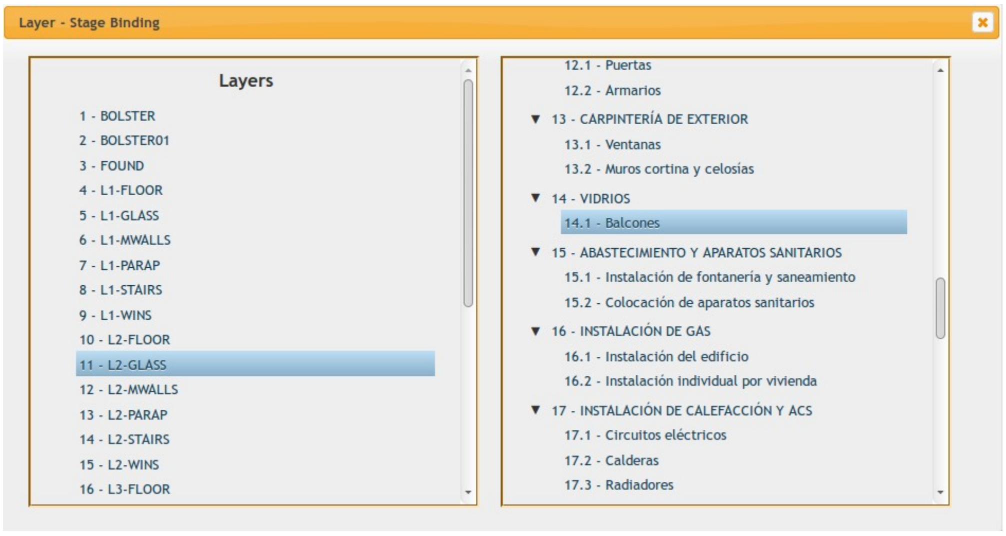
LAYER-TASK BINDING : The user interactively determines which layer correspond to a specific task. The system provides a drag & drop interface where the user drops the layer of the 3D model over the associated task. Both the layer and the linked task will be highlighted with identical color to facilitate the association of multiple layers.

SAVE THE PROJECT : The user chooses a name and provides a description to save the project for future sessions. Hence, the previous three use cases are not mandatory if the user needs to retrieve the same project again.

Event scenario

The most likely event scenario comprises a set of interactions between a user and 3D-SIMOS. These interactions can be described in the following big steps:

1. In an external tool for 3D CAD modeling, the user designs the building with its components grouped in different layers. Each layer must be created to separate the building elements by the task where they are built. Any existing model can be modified following this requirement and the cost of this change is not prohibitive. Then, the model is exported to a COLLADA file (there are several exporters for the most outstanding 3D modeling tools since it is an open standard). The planning of the construction processes is created separately and exported to a MPP file (this option is available in many tools used in architecture planning).
2. Once both the planning and the 3D model are ready, they can be imported into 3D-SIMOS. The user authenticates into the system and selects the "Create project" option. The MPP and COLLADA files must be


Layer - Stage Binding

Layers

1 - BOLSTER
2 - BOLSTER01
3 - FOUND
4 - L1-FLOOR
5 - L1-GLASS
6 - L1-MWALLS
7 - L1-PARAP
8 - L1-STAIRS
9 - L1-WINS
10 - L2-FLOOR
11 - L2-GLASS
12 - L2-MWALLS
13 - L2-PARAP
14 - L2-STAIRS
15 - L2-WINS
16 - L3-FLOOR

12.1 - Puertas
12.2 - Armarios
▼ 13 - CARPINTERÍA DE EXTERIOR
13.1 - Ventanas
13.2 - Muros cortina y celosías
▼ 14 - VIDRIOS
14.1 - Balcones
▼ 15 - ABASTECIMIENTO Y APARATOS SANITARIOS
15.1 - Instalación de fontanería y saneamiento
15.2 - Colocación de aparatos sanitarios
▼ 16 - INSTALACIÓN DE GAS
16.1 - Instalación del edificio
16.2 - Instalación individual por vivienda
▼ 17 - INSTALACIÓN DE CALEFACCIÓN Y ACS
17.1 - Circuitos eléctricos
17.2 - Calderas
17.3 - Radiadores

FIGURE 4.6: The dialog for linking the 3D-model layers with the corresponding tasks of the planning of the building.

dropped over the corresponding boxes of the starting dialog shown in Figure 4.5. Next, the system parses the files to display the layers along with the nested tasks in a new dialog (see e. g., Figure 4.6). Now, the user determines which task is linked to each layer of the 3D model. The drag & drop metaphor is also used to establish these associations. Finally, 3D-SIMOS saves the project in its database using the model described in Section 4.1.3 that includes tasks, geometries, and materials. The current user is assigned as the owner of the project with the appropriate rights granted.

3. The project created in the previous step can be retrieved later at any time. For example, the user can request 3D-SIMOS to show the current state of the construction. The user can also simulate the progress of the construction tasks up to a particular timestamp or up to the last scheduled task.

4. The user can inspect the building by picking a geometric layer or a specific task. They are presented in tree structures that contains the nested tasks of the planning and the layers of the geometry. Each task can be enabled/disabled to show/hide the associated 3D meshes or selected to highlight the geometry of the building affected by such tasks. The 3D model in the viewport and the trees are synchronized in both ways, i. e., picking a geometry in the model selects the task in the planning; on the contrary, the selection of any task highlights the geometry of the linked layer with the color specified by the user.

System architecture

The main non-functional requirement of 3D-SIMOS is the ubiquitous availability of the application. Taking into account this requirement, we have designed 3D-SIMOS as a distributed system with a *client-server architecture*. In this architecture, the software components are distributed in two different computing

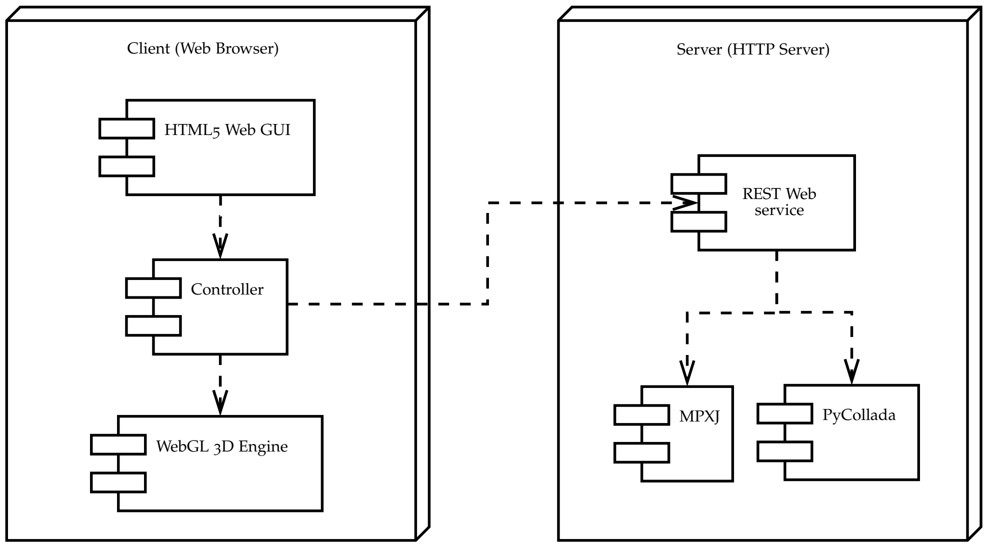


FIGURE 4.7: Deployment schema for the distributed components of 3D-SIMOS. The client is executed in a WebGL-capable web browser, whereas the server is deployed as a virtual machine running on a remote host. Connections are proxied through a lightweight HTTP server.

devices: the server and the client. The server is responsible for performing the complex processes and managing the persistence, whereas the client only requires a WebGL-capable web browser. This architecture is depicted by the UML diagram of Figure 4.7, which includes the distribution of the components and their relations.

The client side of the architecture (or front-end) is executed in a web browser. It includes the GUI and the 3D graphics engine to render the building model in real time. It is developed following the classical Model-View-Controller (MVC) in which the model role corresponds to the 3D model and its planning (e. g., rotations/translations, state of the tasks, name or author of the project, etc); the view role is played by the GUI and the WebGL 3D engine; the controller role is played by the "controller" component. These components interact following the MVC pattern: changes in the model are detected by the controller, which requests an update to the view that displays the latest modifications; any information entered in the view by the user (e. g., in a dialog) passes to the controller that updates the underlying model consequently. The controller is the proxy that manages the local transformation of the model and launches the appropriate requests to the server.

The server side of the architecture (or back-end) is allocated in a remote server that collects the information about projects, which is mainly composed of a planning and a 3D model. The components of the server are not distributed. In order to access to the main functionalities of the service, a REpresentational State Transfer (REST) interface is provided. Requests received through this interface are executed by the service, which depends on other technologies to perform certain operations, such as an SQLite database (to manage the persistence of structured data) or a Java Virtual Machine (to parse the planning of a building project). Thus, the service is an entry point that orchestrates with other components to perform project management, data parsing, and user authentication. Section 4.1.4 includes more details concerning the Open Source technologies adopted to implement the functionalities of the services.

The communication workflow of each component has been described in the previous paragraphs. However, the responsibilities (i. e., the request that each component must attend) of each one have not been detailed. Such responsibilities bound the scope of the components as follows:

HTML5 WEB GUI This is the front-end of 3D-SIMOS, which allows the users to interact with the building model and its related information (tasks and layers). It includes the widget to render the building model on the screen through the WebGL 3D engine and the widgets to manage the scheduled tasks.

WEBGL 3D ENGINE This is the widget of the GUI to render the 3D models, and implement the navigation controls with the mouse and the keyboard. The scene components are managed through the scene graph, which includes not only the geometries and materials but also the lights and rigid body transformations. It also listens to mouse events to allow the user to inspect the building geometry. Picked geometries are immediately synced with their corresponding element in the tree.

CONTROLLER It handles user interactions to create links between the layers of the building model and the tasks of the planning. Furthermore, it is also responsible for authenticating the user, loading a preexisting model, managing calls to the Web 3D Engine and performing the appropriate requests to the Web Service through its REST interface. In summary, this component commands the orchestration between other components of both client and server.

WEB SERVICE This component is allocated in the server, and it is the entry point of any requests coming from the client. It parses 3D models of building in the COLLADA format to generate the scene graph visualized by the WebGL 3D Engine. The MPP files with the planning are also parsed by this component and returned to the client. Furthermore, this component is also responsible for user authentication and project management. These functionalities are exposed through a REST interface with Create, Read, Update and Delete (CRUD) operations.

MPXJ AND PYCOLLADA These components are wrappers to adapt the corresponding libraries that parse the 3D models and the construction planning. They will be detailed in next section.

Implementation

Once requirements, functionalities, and the architecture of 3D-SIMOS have been identified, it is necessary to implement each component described in Figure 4.7, even the wrappers of MPXJ and PyCollada. Several Open Source third-party technologies have been adapted and integrated to accomplish this goal. In this way, 3D-SIMOS is independent of proprietary technologies, avoiding the risk of vendor lock-in. In addition, it can be easily adapted and expanded if further developments were required. More concretely, the following technologies are involved in the development of the components:

- The task and schedule of the planning are parsed from the MPP file using the MPXJ[5] library. The library parses the file to convert the data in a list of tasks following a tree structure. This structure is returned to the client as a JavaScript Object Notation (JSON) in response to a GET request.
- The COLLADA file is parsed by the PyCollada[6] library. The parsed 3D model follows an object-oriented structure that can be easily traversed to find the interesting scene nodes. Then, the scene graph of the building is created and returned as a JSON file. This scene graph represents the hierarchical relations between the elements in the scene. It is provided to the Web3D engine to render the building model. Indeed, layers are extracted by the browser from this scene graph.
- The Web Service has been developed using Bottle[7], a Python Web Framework to create portable, extensible, cross-platform and lightweight services. In addition, Bottle provides a simple routing mechanism that supports CRUD operations as Python decorators.
- The communication with the Java Virtual Machine (JVM) that parses the plannings is performed using the Py4J[8] library. This communication is implemented through a UNIX socket to reduce the overhead. Java objects and methods can be dynamically accessed as if they reside in the Python interpreter. Py4J also enables Java programs to call back Python objects.
- The HTML 5 Web GUI has been developed using the widgets provided by the jQuery UI[9] library, which provides theme customization and multi-purpose widgets. The layout is implemented with a combination of the HTML5 and the CSS3 languages. The design of the interface is pretty straightforward since it only includes the top menu, modal dialogs, and a tree widget to represent the nested tasks from the planning. This last widget is included in an accordion along with the layers of the building.
- The controller implements the logic of 3D-SIMOS operations at the client side. It has been developed in the CoffeeScript[10] language, which translates into the JavaScript application executed by the web browser. Advanced interactions with the Document Object Model (DOM) are implemented with the jQuery[11] library in combination with the latest features of the JavaScript API. In this sense, we have used specific HTML5 features, such as the File API, Drag and Drop events and WebGL API, among others.

5 http://www.mpxj.org
6 http://pycollada.github.io
7 https://bottlepy.org
8 https://www.py4j.org
9 https://jqueryui.com
10 https://coffeescript.org
11 https://jquery.com

- The WebGL 3D engine has been developed over the SceneJS API[12] and, in turn, is supported by the WebGL API. This library has been chosen due to the simple management of the scene graph it offers. Furthermore, it achieves high framerates at visualizing large and complex meshes. The new version of the library called xeogl[13] has been specially tailored for the AEC industry .

We have implemented several functionalities in the WebGL 3D engine, such as the navigation of the 3D model using the rolling bar metaphor, the two-way synchronization between selected tasks and picked meshes, or the simulation of the construction following the tasks of the planning. The simulation can be animated using SLERP (more information at the end of the Section 2.7.2) for the trajectory of the camera. This type of interpolation allows us to create a smooth animation for the three-dimensional rotation of the vector that connects the origin of the camera with the center of the building model.

Each frame is rendered taking into account the camera position estimated in the interpolation procedure at the same time that the material opacity (alpha) is linearly interpolated from zero to one too. The evolving transparency of the layer corresponding to the current task in the simulation gives the user the impression that the execution of such task is moving forward.

The choice of the rolling ball metaphor to navigate the 3D model instead of a simpler tracking ball metaphor is motivated by the fact that some CAD models have been designed with the Y and Z axis swapped. Furthermore, we also want to leave the user the choice of inspecting certain model layers of the building using a different angle around the optical axis of the camera (i. e., in the depth direction). This type of rotation is neglected in the tracking ball metaphor, where the rotation with the mouse is simulated using just two Euler angles, which are the solid angles over the spherical surface of the trackball. With this decision, we are reducing the usability to improve the options that the system offers to the user. In addition, the system will be able to manage challenging models where rotations cannot be limited to the plane XZ.

4.2 METHODOLOGY

In this part of our research, we have followed a combined methodology that connects a top-down and a bottom-up approach. The top-down approach intends to split the overall problem into its parts, which leads to the components of 3D-SIMOS described in Figure 4.1.4. On the contrary, the bottom-up approach starts with the underlying symbolic representation of processes and spaces described in Section 4.1.2 to create the system. Both approaches are joined to design 3D-SIMOS.

The bottom-up approach is fed with the models described by CityGML and IFC, which can be embedded in the proposed symbolic representation. These models describe both the geometry and the semantics of the building, which constitute the static information. However, they lack the dynamic information concerning the processes associated with the building along its entire

12 http://scenejs.org
13 http://xeogl.org

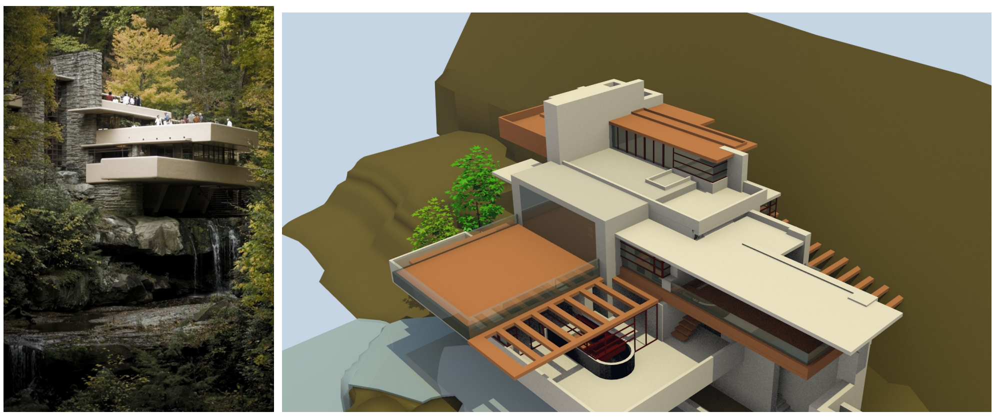


FIGURE 4.8: An original picture of the Fallingwater house and the computer-generated image of the 3D model designed for our use case.

lifecycle. This problem has been addressed with the proposed representation of Section 4.1.2.

The top-down approach is based on understanding the requirements posed by the stakeholders of the AEC industry in order to propose a solution that complies with their demands. We have decomposed the constituent parts of the system to create a reusable architecture based on open standards and technologies. This solution is described in Section 4.1.4.

In the end, both approaches converge into a shared model based on the IFC that sustains the operations performed by the system. The feedback between both approaches allows us to iteratively polish the model while the offered functionalities are being extended.

On the one hand, the symbolic representation is considered valid since it is formally based on already-proven concepts and results. On the other hand, the system can be experimentally evaluated by its stakeholders. We have evaluated the fulfillment of our starting goals following an experimental methodology that assesses both qualitative and quantitative aspects. Some aspects have been evaluated with the execution of the functionalities of 3D-SIMOS in two different use cases. Other aspects have required expert validation from several stakeholders of the AEC industry.

4.2.1 *Use cases*

In order to evaluate 3D-SIMOS in different types of building projects, we have conducted the evaluation over two use cases:

1. The Fallingwater house or Kaufmann Residence (see Figure 4.8) was designed by Frank Lloyd Wright in 1935[14] and it is considered one of the most beautiful jobs of this architect. The house is located in Pennsylvania, and it is placed over a waterfall. The total cost of the building project was $155,000\$$, but it was donated as a museum by its owner in 1964. We have designed the 3D CAD model of the house from scratch, taking advantage of the documentation freely available. A computer-generated

14 http://www.fallingwater.org

image of such model is shown in Figure 4.8, next to the original picture of the house.

2. The church of “Santa Maria La Antigua” is located at the city of Valladolid (Spain)[15]. It was likely built in the 11th century but its oldest parts date to the late 12th century. It follows a Romanesque style, and it is one of the principal symbols of the city. We have also designed a simple 3D model of the main envelopes of the church to evaluate 3D-SIMOS on a cultural heritage case.

4.2.2 *Aspects*

According to our initial objectives, we are interested in evaluating the following aspects of 3D-SIMOS:

USABILITY Whether the system is easy to use or not, compared with the available alternatives to execute the same task. This aspect requires expert evaluation.

USEFULNESS Whether the system is useful for showcasing and simulating plannings over buildings. This aspect also requires expert evaluation.

VISUAL APPEAL Whether the system results visually attractive to the user or not. We have tested that the materials are appropriately rendered, including the textures of the object (available only in the second use case). This evaluation can be initially performed without experts, but we ask them for confirmation.

PERFORMANCE Whether the system runs smoothly with low computer resources. We have evaluated this aspect with the average framerate at which the system runs the simulation of the construction on different hardware setups (including both mobile and desktop platforms). This aspect is evaluated without experts.

CROSS-PLATFORM OPERATION : Whether the system runs on mobile devices. This aspect does not require expert evaluation.

4.2.3 *Procedure*

Firstly, we have evaluated the aspects described in previous section that do not require an expert validation. Secondly, we have evaluated the other aspects from personal interviews with the experts. In this case, the evaluation has been performed with the following steps.

1. The stakeholder or the potential user of 3D-SIMOS receives a small training session to learn how to execute the main functionalities of the system, which includes: create a project, navigate the building, interact with layers and tasks, and simulate the advance of the construction.

15 `https://www.valladolid.com/la-antigua`

2. We give ten minutes to the user to get comfortable with the 3D-SIMOS GUI interface. This improves the engagement of the user with the system and helps us to achieve more accurate impressions.

3. After that, we give the user a script with a set of tasks to execute in a specific order. Tasks are performed using a dataset composed of the planning and the 3D model of the Fallingwater use case (see Section 4.3). In this way, the user can detect possible pitfalls and drawbacks of our approach. The provided script contains the following steps:
   a) Create a new project by importing the CAD model and the planning into 3D-SIMOS.
   b) Align the tasks and the layers of the model until every layer has a corresponding task.
   c) Orbit the camera around the model using the orbit camera controls (i. e., the left button, the middle button and the wheel of the mouse)
   d) Disable the first layer of the model and select the 7.2 task of the planning.
   e) Play the simulation of the construction. Observe the achieved framerate.
   f) Change the animation framerate to 10 FPS in the preferences dialog.
   g) Play the simulation again. Check that the camera rotates more slowly around the model.
   h) Save the project, reload the application and load the same project.

4. Once the tasks have been finished, we evaluate with an interview whether the user has accomplished the tasks or not, and what sort of problems have been found. Then we extract our conclusions from this interview.

This evaluation has been performed with a set of five stakeholders involved in the AEC, due to the difficulties of finding experts that can validate our system. These users include two architects, two engineers and one topographer. They have tested 3D-SIMOS in their own web browsers in mobile and desktop environments, following the instructions specified above. Then, we gather the results concerning the evaluated aspects and outline the conclusions.

4.3 EXPERIMENTAL RESULTS

The system 3D-SIMOS has been evaluated according to the methodology proposed in Section 4.2. We asses the results accomplished by 3D-SIMOS taking into account the aspects described in the methodology. Then, we determine the level of compliance with our starting goals. Each aspect has been evaluated in a different section.

The experiments have been executed using Mozilla Firefox and Google Chrome web browsers running on a Nvidia GTX 285 with an Intel Xeon CPU (E5630 Quad-core 2.53 GHz) and 8 GB of RAM.

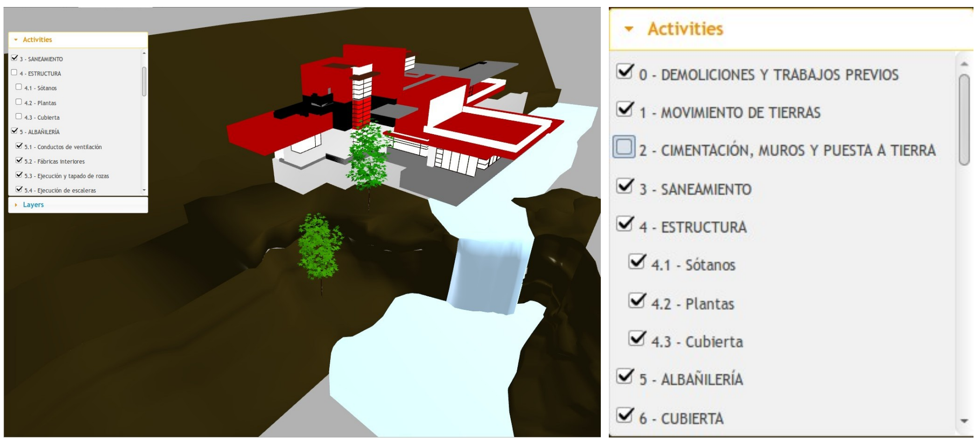


FIGURE 4.9: The two main components of the 3D-SIMOS's GUI: the viewport where the 3D model is visualized and the tree where the nested tasks of the planning are grouped. The information shown in both of them corresponds to the use case of the Fallingwater house.

The access to 3D-SIMOS is open[16] to allow interested readers to reproduce the results described in this section (at least those results that do not require expert validation). We have created a demo account with the following credentials: *mobivap@simos3d.com* and *mobivap*. This authenticated user owns several use cases that include the two cases used for this evaluation. The source code of the project is publicly available in our git repository[17]. The project included the client and the server components ("srv" folder), and also the resources corresponding to the use cases.

A complete video[18] has been recorded to illustrate how to retrieve a previously created project and simulate the evolution of the construction with an animation. The current task is highlighted during the simulation. The video also shows how tasks can be selected/enabled to highlight/hide the geometry of the 3D model. This selection can be performed both in the 3D model and the planning. The animation runs smoothly, despite the low framerate perceived in the video since the recording application only captures the screen at 10 FPS.

4.3.1 *Usability*

We start the section by illustrating the GUI of 3D-SIMOS when the user performs the main operations. The visual feedback received by the user is illustrated by means of several figures corresponding to screenshots of the two evaluated use cases.

When a model is retrieved, 3D-SIMOS shows two widgets to the user: the viewport, where the 3D model of the building is rendered, and the tree, where the tasks of the planning can be selected/enabled (see the left and the right side of Figure 4.9, respectively). Both widgets are used for inspecting and navigating the building project once it has been created.

16 `https://simos3d.mobivap.es`
17 `https://gitlab.com/fradelg/simos3d`
18 `https://bit.ly/2KnIQvp`

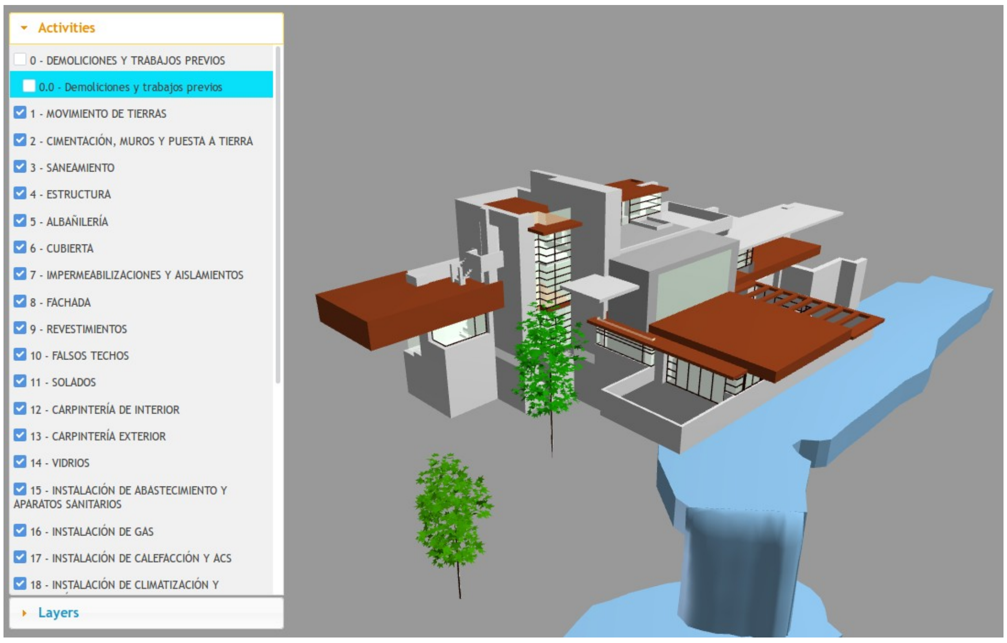

Activities
0 - DEMOLICIONES Y TRABAJOS PREVIOS
0.0 - Demoliciones y trabajos previos
1 - MOVIMIENTO DE TIERRAS
2 - CIMENTACIÓN, MUROS Y PUESTA A TIERRA
3 - SANEAMIENTO
4 - ESTRUCTURA
5 - ALBAÑILERÍA
6 - CUBIERTA
7 - IMPERMEABILIZACIONES Y AISLAMIENTOS
8 - FACHADA
9 - REVESTIMIENTOS
10 - FALSOS TECHOS
11 - SOLADOS
12 - CARPINTERÍA DE INTERIOR
13 - CARPINTERÍA EXTERIOR
14 - VIDRIOS
15 - INSTALACIÓN DE ABASTECIMIENTO Y APARATOS SANITARIOS
16 - INSTALACIÓN DE GAS
17 - INSTALACIÓN DE CALEFACCIÓN Y ACS
18 - INSTALACIÓN DE CLIMATIZACIÓN Y
Layers

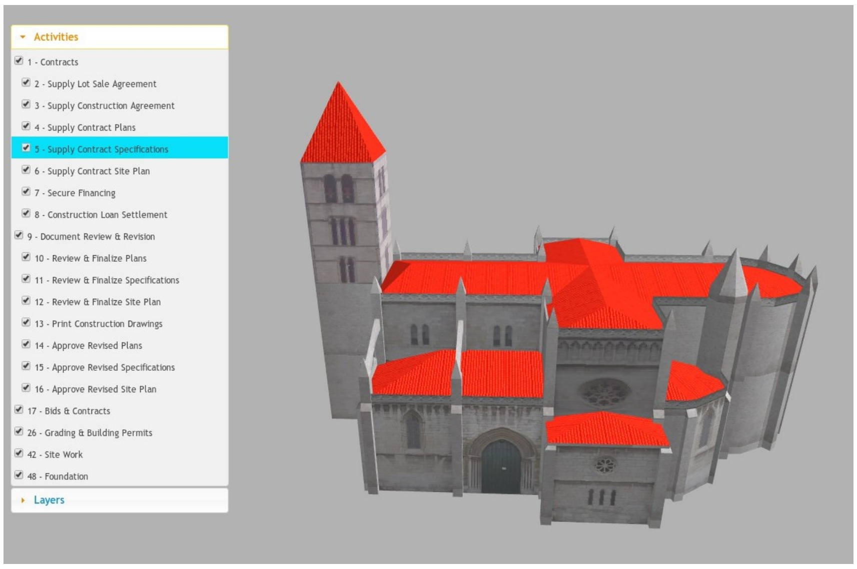
FIGURE 4.10: The Fallingwater house model when the first task is disabled. Each task contains nested tasks that can be individually disabled affecting the children.

Operations over nested tasks are performed recursively. For example, a disabled task disables its subtasks too. This allows the user to clean the visualized model, as it is illustrated in Figure 4.10.

The user can select any task in the tree to highlight the material of the geometry linked to the task. For example, Figure 4.11 shows how the selection of the fifth task highlights the material of the roofs of the building. Tasks can be selected by clicking over their name in the tree or picking the 3D mesh of the model whose layer is linked to the task.

The usability aspect refers to the ease of use and learnability of 3D-SIMOS. In our interviews with the five experts, all of them acknowledge that the simplicity of the system considerably reduces the learning curve. It also eases the execution of the operations on the script. After all, there are only four functionalities to test that can be reached even with a trial and error methodology. They also remark that there would be desirable more control over the operations. For example, they want to change the parameters of the simulation to speed up/slow down the pace at which the layers appear. They would also want to alter the color of the highlighted material. Some of these settings, along with other rendering options, have been added to 3D-SIMOS in a separated dialog (see Figure 4.12). They can be accessed on the top menu.

Overall, the usability of 3D-SIMOS is the most appreciated aspect by the users. It is undeniable that the minimalist design of the application forces the user to focus on the particular functionalities. This fact allows 3D-SIMOS to concentrate on a specific niche of the AEC industry that demands affordable and simplistic solutions, in contrast to the all-in-one solutions offered by GRAPHISOFT ArchiCAD or Autodesk Revit.

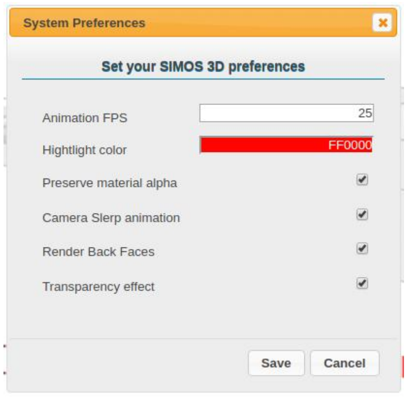


Activities

- ✓ 1 - Contracts
  - ✓ 2 - Supply Lot Sale Agreement
  - ✓ 3 - Supply Construction Agreement
  - ✓ 4 - Supply Contract Plans
  - ✓ 5 - Supply Contract Specifications
  - ✓ 6 - Supply Contract Site Plan
  - ✓ 7 - Secure Financing
  - ✓ 8 - Construction Loan Settlement
- ✓ 9 - Document Review & Revision
  - ✓ 10 - Review & Finalize Plans
  - ✓ 11 - Review & Finalize Specifications
  - ✓ 12 - Review & Finalize Site Plan
  - ✓ 13 - Print Construction Drawings
  - ✓ 14 - Approve Revised Plans
  - ✓ 15 - Approve Revised Specifications
  - ✓ 16 - Approve Revised Site Plan
- ✓ 17 - Bids & Contracts
- ✓ 26 - Grading & Building Permits
- ✓ 42 - Site Work
- ✓ 48 - Foundation

Layers

FIGURE 4.11: The selection of the fifth task highlights in red the material of the geometry linked to such task on the use case of "Antigua".



System Preferences

Set your SIMOS 3D preferences

Animation FPS	25
Hightlight color	FF0000
Preserve material alpha	✓
Camera Slerp animation	✓
Render Back Faces	✓
Transparency effect	✓

FIGURE 4.12: The dialog to configure the settings of a session with 3D-SIMOS. The user can change the opacity/framerate of the animation, the color of highlighted materials, and rendering options like the backface culling or the material transparency.

4.3.2 *Usefulness*

The utility or usefulness of 3D-SIMOS is the most subjective aspect included in our evaluation. It depends on the expectations of the user and the known approaches. In our interviews with the experts, we have detected that the system is simple for effective. Three of five think that the system accomplishes its purpose, but it is limited to set of functionalities that they would want to extend. The other two users opine that the system does its best and it is enough according to the initial requirements.

The simplicity of 3D-SIMOS is a double-edged sword. On the one hand, the system focus on a very limited subset of functionalities to cover specific requirements of the AEC industry. This makes the system lightweight and simple to maintain. On the other hand, these functionalities are commonly linked to other advanced functionalities demanded by some stakeholders. The main advantage of 3D-SIMOS is that the underlying model is based on the IFC, which eases the manipulation of the project with other tools. The system can be also extended to include the advanced functionalities with the same argument.

4.3.3 *Visual appeal*

Images rendered in the viewport are generated by the WebGL 3D engine using a conventional Phong shader, which supports textures for the materials of the model if they were available. Qualitative results are illustrated for the Fallingwater house (no textures) in Figure 4.9, and for the "La Antigua" (basic textures) in Figure 4.11. It can be seen that materials provide more visual grepping when the textures do not have enough resolution (as it is the case of the 3D model of "La Antigua". However, better textures can improve the quality of the rendering, although they harm the performance of the system, specially in non high-end smartphones.

In the view of the experts, the system quality is poor. Note that they have evaluated only the use case of the Fallingwater house that does not include textures. Four of five believe that the visualized image should have a quality similar to the computer-generated images that they use to showcase the construction project (see e. g., Figure 4.13). This way the client would have a better impression of the building they are going to buy, which would be also easy to recognize since both images would match. The remaining user is aware of the difficulties to provide in real-time an image with the same quality as another image that has been rendered off-line.

It is well-known that real-time visualization engines do not achieve the same quality as an off-line rendering engine. These engines are based on physically-accurate simulations of the light path, normally achieved by tracing the ray of light. However, new Physically Based Rendering (PBR) models allow the system to achieve more visually appealing results with a moderate increase in the hardware requirements. We have left this implementation as future work.

FIGURE 4.13: A ray-traced image of the Fallingwater house generated with Blender using the Cycles render engine. The underlying 3D model is based on the one imported in 3D-SIMOS.

4.3.4 *Performance*

The simulation functionality is the most resource-intensive functionality of 3D-SIMOS. This process is performed at the maximum allowed framerate to render smooth animations that provide a pleasant experience to the user. A single frame of the animation generated with SLERP is in Figure 4.14. Moreover, it can be seen in the same figure that the opacity of the mesh corresponding to the current task is also animated as several polygons are translucent in the current frame. The blur of the figure has been caused by using the screenshot tool at the same time that the animation is running.

We have measured the performance of the rendering by measuring the framerate of the simulation performed on the Fallingwater use case. The geometry of this use case is complex enough to extrapolate the differences between devices to other use cases. These results are shown in Table 4.1. The WebGL 3D engine is capable of rendering frames at 60 FPS on desktop machines equipped with a decent GPU. On a modern smartphone like the Samsung Galaxy S6, it runs stable at more than 55 FPS, with an slight improvement in the UCL web browser. In an older device like the Nexus 4, the simulation runs at an average framerate of 21 FPS due to the limitations of the old Adreno 320 GPU. Animations rendered below 30 FPS are not smooth since the eye catches the skipped frames.

It should be noted that the maximum framerate depends on the number of callbacks to the underlying rendering function. This number is limited by the web browser to the display refresh rate, following the W3C recommendation. This rate is usually 60 FPS, and hence the maximum framerate delivered by all the evaluated devices.

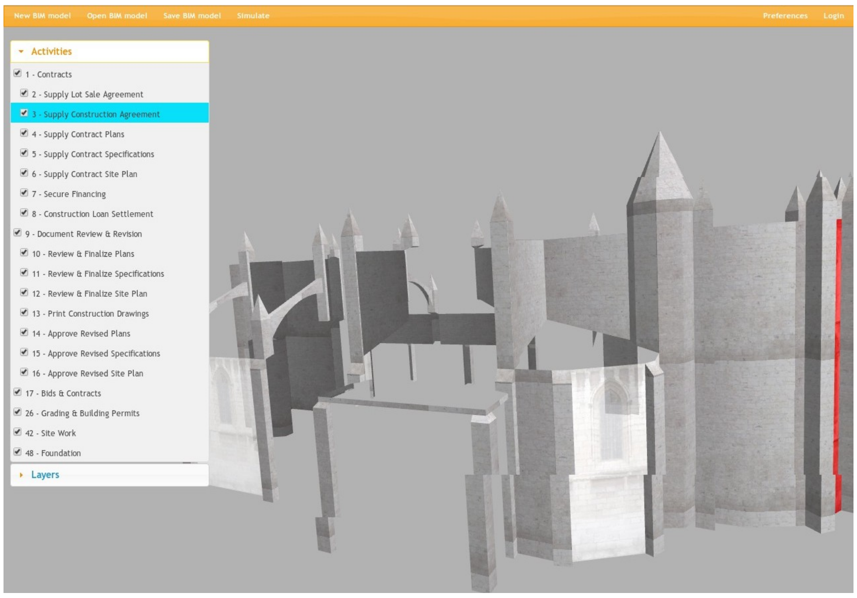

New BIM model
Open BIM model
Save BIM model
Simulate
Preferences
Login
Activities
1 - Contracts
2 - Supply Lot Sale Agreement
3 - Supply Construction Agreement
4 - Supply Contract Plans
5 - Supply Contract Specifications
6 - Supply Contract Site Plan
7 - Secure Financing
8 - Construction Loan Settlement
9 - Document Review & Revision
10 - Review & Finalize Plans
11 - Review & Finalize Specifications
12 - Review & Finalize Site Plan
13 - Print Construction Drawings
14 - Approve Revised Plans
15 - Approve Revised Specifications
16 - Approve Revised Site Plan
17 - Bids & Contracts
26 - Grading & Building Permits
42 - Site Work
48 - Foundation
Layers

FIGURE 4.14: A single frame of the animation performed by 3D-SIMOS to simulate the building construction process. Blur is caused by the tool that captures the screenshot.

Device	Web Browser	Mean FPS
Nvidia GTX 1060	Google Chrome	60
Nvidia GTX 1060	WebKit	60
Nvidia GTX 265	Mozilla Firefox 60	
Samsung Galaxy S6	Google Chrome	55
Samsung Galaxy S6	UCL browser	57
Nexus 4	Google Chrome	21

TABLE 4.1: Framerates (FPS) achieved by 3D-SIMOS during the simulation of the constructive processes.

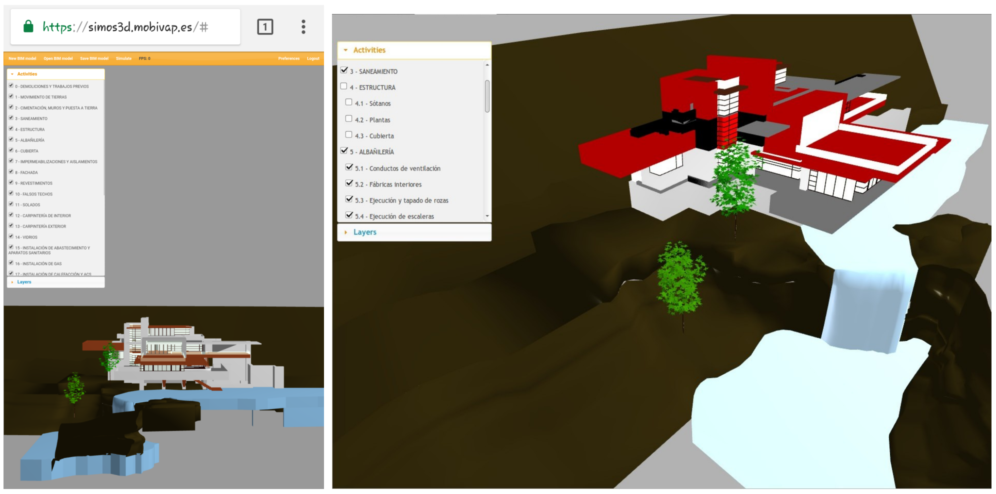

https://simos3d.mobivap.es/#
Activities
3 - SANEAMIENTO
4 - ESTRUCTURA
4.1 - Sótanos
4.2 - Plantas
4.3 - Cubierta
5 - ALBAÑILERÍA
5.1 - Conductos de ventilación
5.2 - Fábricas interiores
5.3 - Ejecución y tapado de rozas
5.4 - Ejecución de escaleras
Layers

FIGURE 4.15: Comparison of 3D-SIMOS in a mobile (Samsung Galaxy S6) and a desktop environment. The image has been rendered in real-time by the Google Chrome browser in its mobile and desktop versions.

4.3.5 *Cross-platform operation*

At its current state, 3D-SIMOS is compatible with any web browser that fulfills with the WebGL 1.0 specification. Figure 4.15 illustrates a comparison between the model rendered in the same web browser (Google Chrome) in its mobile and desktop version. Furthermore, the browser must implement other APIs of the HTML5 specification required by 3D-SIMOS, such as the File API. We have successfully run 3D-SIMOS on the following browsers:

- Microsoft Edge 17
- Mozilla Firefox 56
- Apple Safari 11.1
- Google Chrome 49
- Chrome for Android 66
- UC Browser for Android 11.8

4.4 CONCLUSIONS

The AEC industry is progressively evolving from the traditional CAD design to the BIM methodology. This methodology proposes a new form of collaborating around the building in a unique shared model represented by the IFC. This methodology proposes a different collaborating model built around a unique shared model of the building described by the IFC. Stakeholders share information between them about any process that affects the building. However, the pace of adoption of BIM is being slow. The costs of consultancy, software acquisition and training create a high entry barrier for many small and medium enterprises. The cost of change (i. e., the cost of modifying an existing project after the design phase) is the most meaningful. Furthermore, BIM is focused on representing the static information about the building, leaving aside the dynamics of the processes that are not executed by a stakeholder.

Simple 3D visualization of the building at its different stages allows stakeholders to evaluate constructibility of the different spaces. It also helps to improve the coordination and the understanding between these parts when collaborating in the same project. Visualization must be accessible from the office (for general contractors and final clients) and from the workplace (for construction managers). Ubiquitous visualization is part of the collaboration effort in BIM to offer safer, more secure and more efficient processes.

In this chapter, we have proposed a symbolic representation that integrates continuous information from external processes into the spatial representation of the building. This representation is based on graphs of cellular decompositions whose transformations (contraction/expansion) can be managed in combinatorial terms thanks to the notion of the cubical complex. Cubical complexes provide multilayered and multifunctional support for nD information systems. Dynamic information is managed with graph transformations and visualized as flows that correspond to variations in scalar, vector or tensor fields. The temporal evolution of these fields is sampled using spatio-temporal slices.

We have also developed an alignment (or mapping) between the two traditional sources of project data (the planning as a MPP file and the CAD 3D model as a COLLADA file) and the IFC schema used in BIM. The alignment takes into account both the geometry and the semantics of the building. This provides a reverse-engineering approach to recycle project designed without following the BIM methodology.

These ideas have been applied to design an uncomplicated system, called 3D-SIMOS, that allows the user to import the planning of a building project and simulate its execution. We have implemented 3D-SIMOS as a web application that reconciles the tasks and the geometry of a building project. Later, the building can be visualized and inspected in a three-dimensional environment using any WebGL-capable web browser. The application helps both project managers and visitors to monitor and track the progress of the construction.

The system has been evaluated using an experimental methodology that combines both qualitative and quantitative aspects. The qualitative aspects have been partially evaluated with interviews with a group of five experts in the AEC industry. Results confirm that the system is easy to use and efficient in the simulation process, operating at 60 FPS on most devices. Furthermore, it is useful for quickly creating and showcasing a building project without the need for complex and unaffordable software licenses. However, the approach is considered too naive for some operations, and the rendered 3D model is not as eye-catching as they would desire, especially when it is compared to an off-line rendered image.

The basis of the BIM methodology advocate for collaborating around a building model shared between the involved stakeholders. Despite the apparent benefits, the AEC industry has been reluctant to adopt this new methodology. In our research, we have checked that the visualization and simulation of constructive processes provide insightful information about the building. Furthermore, a simplification of the process of creating BIM projects is proved beneficial for a better understanding of BIM.

4.4.1 *Compliance with the objectives*

This section analyzes the degree of fulfillment of our original goals according to the results obtained from our evaluation. As it has been done in Section 3.4.1, we consider the goals individually and discuss the advantages and drawbacks introduced by 3D-SIMOS.

Demonstrate the benefits of the BIM methodology

Our central goal is to demonstrate the benefits of using the BIM methodology in the AEC industry. Even though the scope of 3D-SIMOS is very restricted due to its intended simplicity, our interviews with the stakeholders revealed that they understood the collaboration workflow. They also value the idea of working around the same shared model. These are the core propositions of the BIM methodology. Furthermore, stakeholders have found useful the functionalities of the application, although they have remarked an excessive degree of simplification. This should not be an inconvenience to recognize the benefits of BIM.

Planning and simulating the advances in construction

The system allows stakeholders to import buildings designed with CAD. After creating the project, tasks can be inspected/selected/enabled on the tree widget. The current state of the construction is represented by the task being executed according to the schedule. We consider the goal as achieved since it is implicit in the functionality provided by 3D-SIMOS and stakeholders have validated the utility of the system. Moreover, certain parameters of the simulation can be customized to suit the specific requirements of the user.

Enrich the CAD model with semantic information

The developed alignment between MPP and COLLADA with IFC is intended to reach this goal. This model provides 3D-SIMOS a unified representation of both tasks and layers of the 3D model. As it has been described in Section 4.1.3, the concepts have been manually mapped to achieve maximum accuracy. Hence, the resulting project contains the scheduled tasks linked to the geometry of the model they affect. The objective is fulfilled since any geometry can be selected/enabled with its corresponding task (i. e., non-geometric information) and vice-versa. The selection is synced between the tree of nested tasks and the viewport.

Simplify the creation of the BIM model

This goal has been addressed with the creation of a GUI for binding layers to tasks at the lowest level of the tree. This GUI hides this complexity of the underlying model under a simple drag and drop movement, which is more simple than a design of the BIM model from scratch. Interviews do not reflect any inconvenience with this method, except it might be too simplistic. The color scale with which created bindings are labeled helps the user to identify the correspondences. Stakeholders have evaluated the usability of 3D-SIMOS very positively, and this includes binding as one of the main steps executed during the evaluation. Therefore, we consider the goal accomplished.

Develop a light-weight cross-platform application

The use of the WebGL API for the visualization of the building is the key to fulfill this objective. It simplifies the development of the application since the GPU of the device can be accessed without considering specific details of the platform. The main drawback is that the GUI is not responsive, i. e., it does not adapt to the resolution of the screen. However, most operations are not intended to be performed on a mobile device. Except for visualization, the other functionalities are more suitable for a desktop environment. Stakeholders value positively that the visualization and model inspection functionalities are available in mobile devices too.

The simulation must run smoothly

A smooth animation must run at least at 25 FPS to provide a pleasant experience to the human eye. This was our main concern at the time of choosing SceneJS as the visualization engine since we have to deal with complex building models. We have evaluated this goal explicitly in the performance aspect described in our methodology (Section 4.2). Experimental results show that 3D-SIMOS achieves the maximum frame rate allowed in desktop displays (60 FPS) and in high-end smartphones. In middle-end smartphones like Nexus 4, this framerate drops up to an average 21 FPS. We evaluate this goal as accomplished since the simulation works on mobile devices although its smoothness depends on the complexity of the model and the GPU.

Render appealing models, visually enhanced by textures

Textures are included in the visualization of the model (e. g., see the use case of "La Antigua" church in Figure 4.11). We utilize these textures to represent the diffuse component of the material. However, stakeholders do not assess the quality of the visualization as appealing. At least, it is not as appealing as the computer-generated images they employ to sell the building project. This affects the showcasing capabilities of 3D-SIMOS since the potential customers may perceive the building as not attractive. Thus, this objective has been partially accomplished, and further improvements of the OpenGL Shading Language (GLSL) shader are left as future work.

4.4.1.1 *Support different types of building models*

The system has been evaluated in two distinct use cases, as it has been detailed in Section 4.2.1. The first one is the Fallingwater house, a particular house employed as a museum nowadays. The second one corresponds to "La Antigua" church, a cultural heritage building. Experiments show that the systems operate equally for both models. This is reasonable since the design of

3D-SIMOS is completely independent of the nature of the building. A building is represented as a collection of geometric primitives linked to a set of nested tasks.

4.4.2 *Future work*

During the evaluation of our research, we have identified additional points for improvement that have been left as future work. For instance, we would like to use the output generated by KN-SLAM (see Chapter 3) in 3D-SIMOS (see Chapter 4). The hardest challenge here involves the generation of a dense reconstruction using at least a PL-planar approach. This problem has been already addressed by several authors (see e. g., [6, 19, 42]). A further contribution on this topic would include the automatic detection and recognition of planes and volumes, that may be part of a more ambitious reverse BIM approach.

The problem of representing free/occupied spaces that evolve over different processes is also challenging. In this case, it seems convenient to use a discretization of a more abstract tensor approach. Differential forms represent the evaluation of functionals along lines (e. g., installations), surface elements (e. g., walls, roofs, and ceilings) or volumetric elements (e. g., free/occupied zones). Similarly, vector fields represent the spatio-temporal evolution of the flow generated by such evaluations. Any tensor is the product of a finite number of differential forms and vector fields, so Tensor Calculus provides a natural framework to manage both of them simultaneously.

Additional useful functionalities for the BIM stakeholders are left as future work too. These functionalities include the exportation of the BIM model using the IFC schema and the binding of the geometries associated with a task by picking on the 3D mesh instead of the name of the layer. Finally, a PBR-based shader to render the materials of the model would help to increase the visual appeal of the advanced visualization.

This chapter of the thesis has dived into the benefit of the 3D information produced by SLAM methods for the AEC industry. We have examined new alternatives to promote the adoption of BIM among the AEC stakeholders by showcasing the building with an advanced visualization tool. As in Chapter 4, we have provided the conclusions extracted from this part of the research in the last section of the chapter. However, we think it is useful to merge the conclusions from both chapters to give the reader a broader view of the whole research. The next chapter is devoted to summarizing our general conclusions concerning 3D reconstruction, SLAM, and BIM. Moreover, we will briefly discuss lessons learned during the Ph. D. and personal thoughts about the process.

5 CONCLUSIONS

In this thesis, we have addressed the problem of achieving affordable 3D reconstructions from a low-cost mobile camera, and its application in a specific sector like the AEC industry. In the first half, we have focused on developing a new approach for implementing more accurate and robust visual SLAM. We have proposed a simplification of the connectivity graph to reduce the complexity of the optimizations at the same time that the accuracy is boosted. We have also added adaptive bootstrapping to track more the frames of the sequence.

Several sequences from heterogeneous datasets have been thoroughly analyzed taking into account their challenging characteristics. Our contributions have been performed over the current implementation of ORB-SLAM, that has been used as the reference to compare our results in the evaluation. Results have shown that, despite the difficulties found in some of the sequences, KN-SLAM improves the accuracy and robustness in common sequences (i. e., sequences where camera moves with a good balance between rotation and translation).

The results generated by ORB-SLAM are composed of a collection of keyframes and 3D map points. In this way, it cannot be treated as a digital model of the scene. However, an additional step based on the principles of Multiple View Geometry can recover the surface of the objects to produce a complete 3D model. This 3D model can be of use for many applications and industries. In the second half of our research, we have centered on the AEC industry and its integration with the BIM methodology. In this context we have outlined the following contributions:

- A symbolic representation that incorporates the dynamics of constructive processes in the model using basic concepts from Algebraic Topology. The structure is represented as a graph of cellular complexes that can be exploded/shrunk using combinatorics. The processes alter the functionals linked to the attributes of the cells that can be visualized as flows moving through the spaces of the building. These flows can be sampled using different types of fields (scalar, vector, and tensor fields).

- An alignment between planning, stored in MPP file format, and a 3D model of the building, expressed in COLLADA file format. This alignment defines a mapping between concepts of both schemas and the IFC to integrate information from both data sources into a single IFC model.

- A visualizer that highlights by its simplicity and its simulation functionalities in any Web3D-capable browser. Building projects can be created from a planning and a 3D CAD model. The application allows different stakeholders in the AEC industry to collaborate around the same building, to simulate the advances in the construction and to track the evolution of the project.

The results of this second part of our research have been evaluated experimentally taking into consideration the informed opinion of a group of experts. Several aspects have been assessed from a qualitative and a quantitative viewpoint: usability, usefulness, performance, visual appeal and cross-platform operation. We can conclude that the system is simple and fast enough to fulfill its main goals (sharing, simulation, and tracking building information), but it would be also desirable to extend functionalities and to improve the rendering quality. The system is able to work across different devices, although usability isthe set of functionalities is limited by the size of the screen.

5.1 DISCUSSION

The interplay between 3D reconstruction and motion estimation is a long-standing problem in Computer Vision with multiple applications from the early nineties. Nowadays, the ubiquitous presence of digital cameras in urban or construction environments makes possible to achieve digital models with minimal effort. Often, most SLAM/VO approaches demand complex camera setups or powerful computing architectures to minimize the uncertainty or to complete the available information. Our proposal relies only on a single-lens single-camera setup, where specific keyframes are selected. These keyframes are inserted in the connectivity graph, where keyframes sharing redundant information are managed to perform spatial queries in certain stages of the pipeline. Our work highlights that a simplification of the connectivity graph is worthwhile since it can be beneficial in certain sequences both in terms of accuracy and robustness. Our results confirm that an excess of connections with non highly-redundant keyframes do not contribute to reducing uncertainty since the achieved accuracy is usually lower.

We have determined that KN-SLAM is more suitable for sequences recorded in natural conditions. These sequences are usually generated by robots working in controlled environments or by mobile devices if they are stated in the application requirements. In the absence of these conditions KN-SLAM still works, but they are better manages with a more dense graph like in ORB-SLAM. The increment in memory consumption is due to the coloring of the map points, but there are simple strategies to reduce it. Even though we have not evaluated our approach in a mobile device, we are certain that the computational resources demanded can be provided by a current middle-end device.

We honestly think that the future of 3D Reconstruction is on the mobile devices. The recent introduction of Google ARCore[1] and Apple ARkit[2] confirms this thought. Indeed, these toolkits already include a subset of the algorithms that are underneath any sparse SLAM system. The market trends also indicate that Augmented Reality and Virtual Reality applications are currently demanded although the underlying technology is not mature enough yet. We hope that our contributions to this research field can promote the adoption of this technology in the middle term.

The AEC industry struggles to spread the adoption of the BIM methodology around its stakeholders. Most of the professional solutions are not afford-

1 https://developers.google.com/ar
2 https://developer.apple.com/arkit

able for all of them. As we have attempted in the first half of our research, we have designed a simple system to allow these peers to share the planning tasks of a building. Such tasks have been aligned with the geometry of the designed 3D model, which can be compared with any 3D reconstruction of the built environment to track the progress. The alignment between different knowledge representations is studied in ontology alignment and it has been applied in different applications concerning KMS. Even though we have only completed a partial alignment between the three involved schemas, it serves to illustrate how the broad model described by IFC can be fed with data from other sources. Simulation and tracking of constructive processes have been illustrated with the Fallingwater house, a representative example of a very complex and natural environment.

Overall, we think that our research should help to spread the use of BIM in the AEC industry. Although the developed system is simple, it should be enough to highlight the goodness of collaborating around the same model. This will contribute to enhance the productivity and improve the quality of the results. Nowadays, many stakeholders use three-dimensional digitalization of the environment to track the advances in the construction. However, they lack a common framework to embed these results or a methodology to collaborate. This is a well-known problem with a specific line of research.

The principal purpose of BIM is to substitute the archaic CAD methodology in which any generated artifact was disconnected from the rest. Current efforts are in the integration of the existent information into BIM since most of the contemporary building projects were designed with CAD but they still can be managed with BIM. We expect that our research can shed light on this problem and help the industry with its progressive transition.

Furthermore, BIM in its current state only stores static information about the building. Simulations and analysis cannot be compared with real data linked to the building model. Such experimental data generated by the building processes must be managed by each application according to their specific requirements. Our proposal for dynamic processes can be efficiently integrated into BIM data taking advantage of the relations between the spaces of the building. The principal required addition would be the representation of functionals, which can be described analytically or discretely as fields. Complex simulations and analysis that models coupled processes would be feasible if this framework were adopted.

Overall, we have focused our research on studying affordable ways to bring powerful functionalities to devices with limited resources. We are aware that the mobile revolution has been left behind and nowadays smartphones are used as a swiss-knife tool where everything seems to fit. However, in practice, most of them fail to reach the target market. Indeed, some important initiatives like the project Tango developed by Google have been discontinued due to the problems found to engage developers. Nevertheless, we think there is a rationale for the adoption of 3D reconstruction in mobile devices as the information that surrounds us is mostly three-dimensional. Usually, the problem lies in the proposed applications of the technology and the use of 3D information. In our conversations with the stakeholders of the AEC industry, we notice a real interest in the technology. However, in the end, the adoption does not only depends on the innovation or technology behind a product but on the tradeoff between the cost and the usefulness for the final user.

The research on low-cost algorithms is also encouraged by the increasing penetration of smartphones in third-world countries. Although half of the population already have a smartphone, the number of devices is growing steadily as well as their computing capabilities. More powerful devices will foster the execution of SLAM algorithms like KN-SLAM, which will allow the users to feed affordable BIM solutions like 3D-SIMOS. Both proposals may represent an entry point to use the underlying technology for many users. Since the beginning of the research, our main motivation has been to make advanced technologies more accessible, ubiquitous and affordable to everyone. We should not forget that the population of these third-world countries growths faster than the population of occidental countries. Soon, these countries will be positioned among the largest economies in the world, and in order to get there, they are going to require affordable technologies that pave the way to be a developed country.

At a research level, I have discovered along these Ph. D. studies that all problems cannot be addressed at the same time. Instead, it is better to consider small and thoroughly evaluated contributions to be sure that the right steps are taken and to isolate the sources of changes. I started the Ph. D. trying to implement a complete 3D reconstruction pipeline from scratch, but in the middle, I discovered very compelling approaches released as open source software. Then, I pivoted around my original plan in order to understand these approaches as a solid background to append further contributions. Like this, many significant lessons have been learned in the path by means of mistakes. Profound knowledge of state of the art has been acquired through the extensive revision of the literature and the interaction with other researchers. We have understood the real importance of outlining a right methodology to validate the primary objectives and support the final conclusions. And, in the end, we think that these are the core foundations of a thesis.

The future of 3D reconstruction and mobile devices looks promising, and new applications are being developed for different industries. We hope that our contributions are meaningful for advancing in state of the art.

5.2 FUTURE WORK

In the foreseeable future we think that the following research areas are especially promising to advance in the current state of the art:

- A framework for a homogeneous treatment of degenerate and regular cases that arises from the relation between two cameras in multiple view geometry. This framework should be compatible with tensor fields created for fundamental and essential matrices. Indeed, we have already performed theoretical contributions in this sense. We have introduced a completion of the ambient space (as a locally symmetric space) to manage matrices for epipolar geometry (homography, essential and fundamental matrix) as an extension of the Linear Algebra. This contribution is currently under revision by the board of experts of an international peer-reviewed journal. A brief summary of the framework has been included in Section A.1.

- A framework to determine the location of a mobile camera that coarsely estimates the relative kinematics of other mobile agents. The problem can be tackled from a local approach with an intrinsic formulation of kinematics where the Phase Space provides the support for this framework. This is a locally symmetric space for multiagent systems with the evolving geometric and radiometric properties of the objects. Agents are no longer considered as points in space, but as mobile configurations with partial kinematic information about the scene. The geometry of the Moment Map allows us to insert these concepts into a single framework.

Contributions in these areas would provide a solid kinematic understanding for complex scenes. To esteem the complexity of the problem, we introduce two elements that play a fundamental role to connect both approaches in the dynamic case:

- Graphs provide a symbolic representation compatible with different LoD depending on the meaning of nodes. However, the large diversity of dynamic events to handle requires more flexible models and transformations rules than the ones described in this thesis. It is necessary a larger development in computational kinematics and dynamics.

- Finite differences in sparse data can be transformed to discrete operators over fields. In this work, we have only introduced scalar fields whose modifications are determined by differences beyond thresholds (i. e., events). It is necessary to go farther and manage these events with a hierarchical structure. Another choice is to represent these events as singularities of mappings in fields. However, this theoretical approach is not computational implementable, even for low-dimensional models or low-corank operators.

It can be seen that there exist many contributions to be made for advancing in the topics involved in our research. Indeed, we are developing some of them for publication in the near term. We left the ideas of this section just as a suggestion for anyone ready to advance in state of the art.

Part I

APPENDIX

A APPENDIX

This chapter gathers additional information about our results that has been moved here due to limitations in space. Besides, this separation contributes to increase the readability of the document. Tables are appropriately referenced where it corresponds to provide a more coherence document.

A.1 HOMOGRAPHIES AND FUNDAMENTAL MATRICES

As it has been mentioned in Section 2.4.1, there is a dichotomy between homographies and fundamental matrices. A SLAM system can bootstrap a sequence from both models. However, each model is more suitable depending on the motion described by the camera between the first two keyframes. The best model is selected by ORB-SLAM using a heuristic procedure that evaluates the quality of the map reconstructed from these keyframes. However, the dichotomy can be justified from the underlying topological structure of both models. This topological description understands fundamental matrices as "degenerated" cases of homographies and analyzes possible degenerations of fundamental matrices to be avoided. This section summarizes the contents of a manuscript that we have submitted and it is currently under revision [27].

Let consider the rank stratification of the space $\mathcal{E}$ of 3×3-matrices $\mathfrak{M}$ representing endomorphisms up to scale of a 3D vector space V. The action of the projectivized $\mathbb{P}GL(3;\mathbb{R})$ of the general linear group $GL(3;\mathbb{R})$ on $\mathfrak{M}$ delivers three orbits $\mathfrak{M}^c$ for $1 \leqslant c \leqslant 3$, where $c = 3 - rk(\mathbf{M})$ is the corank of the matrix $\mathbf{M} \in End(V)$:

- The regular orbit $\mathfrak{M}^0$ represented by the group of homographies $\mathbb{H}$ corresponding to regular matrices with rank 3 (non-vanishing determinant) up to scale.
- The singular orbit $\mathfrak{M}^1$ represented by the manifold $\mathcal{F}$ of rank 2 matrices (vanishing determinant), which are interpreted as fundamental matrices.
- The "supersingular" orbit $\mathfrak{M}^2$ represented by the variety of rank 1 matrices (vanishing determinant of every 2×2-minor of the matrix $\mathbf{M} \in End(V)$.

It should be noted that

$$\mathfrak{M}^2 \subset \overline{\mathfrak{M}^1} \subset \overline{\mathfrak{M}^0} = \mathfrak{M},$$

where the overline denotes the topological adherence or "closure" of the orbit. There exists a G-equivariant decomposition of $\mathfrak{M}$ by the rank, i. e., a description as a disjoint union of G-orbits with $G = \mathbb{P}GL(3;\mathbb{R})$:

$$\mathfrak{M}^0 \cup \mathfrak{M}^1 \cup \mathfrak{M}^2 .$$

In addition, $\mathtt{Sing}(\overline{\mathfrak{M}^0}) = \mathfrak{M}^1$ and $\mathtt{Sing}(\overline{\mathfrak{M}^1}) = \mathfrak{M}^2$.

Tangent spaces to these manifolds must be considered to achieve a geometrical interpretation of homographies (elements of $\mathfrak{M}^0$) or fundamental matrices (elements of $\mathfrak{M}^1$) in a common framework. The adjoint map ad provides a natural representation of the dual map, and assigns to each matrix $\mathbb{M}$ its adjoint matrix $ad(\mathbb{M})$. The elements of the adjoint matrix M_{ij} are the determinants of the complementary minor of $m_{ij} \in \mathbf{M}$. The array formed by the entries of the adjoint matrix is the normal to the tangent hyperplane $T_{\mathbf{M}}\mathfrak{M}$ to the space $\mathfrak{M}$ at the point $\mathbf{M} \in \mathfrak{M}$.

From a global viewpoint, the adjoint map defines an isomorphism with its dual space $\mathfrak{M}^\vee$ since $\mathfrak{M}$ is a vector space. Hence, there exists an element of $End(V^\vee)$ up to scale, where $V^\vee$ is the dual space of V for each element $\mathbf{M}$ corresponding to an endomorphism of the vector space V up to scale.

The isomorphism $V \simeq V^\vee$ induces a perfect pairing between V and $V^\vee$ in the following sense: $V \simeq (V^\vee)^\vee$. However, this claim is not true for $End(V)$ since the inverse is not defined for degenerate endomorphisms. To solve this problem, the graph of the adjoint map $ad : End(V) \to End(V^\vee)$ must be compactified with the closure of pairs $(\mathbb{M}, ad(\mathbb{M})$ in the space $End(V) \times End(V^\vee)$ to include the degenerate cases. This compactification extends the isomorphism $V \simeq V^\vee$ to a duality between their endomorphisms. This duality is compatible with the locally homogeneous structure of the orbits so it is a locally symmetric space. This mathematical framework supports "degenerations" or "regenerations" of fundamental matrices and homographies using secant spaces.

In addition, the estimation of the homography or fundamental matrix can be reinterpreted using the geometry of complete endomorphisms, i. e., pairs $(\mathbf{M}, ad(\mathbf{M}))$ where $\mathbf{M}$ represents a matrix and $ad(\mathbf{M})$ its adjoint.

When $\mathbf{M}$ is a regular endomorphism up to scale (i. e., a homography), only four points are needed since they provide a projective reference for the projective plane $\mathbb{P}^2$ representing the view. This can be justified by the fact that matrices $\mathbf{M}$ and $ad^2(\mathbf{M}) = ad(ad(\mathbf{M}))$ represent the same point in the projective space $\mathbb{P}End(V)$. In other words, the dual transformation represented by the $ad(\mathbf{M})$ (up to scale) operates in the same way on 4-tuples of dual lines of the above reference points. The dual endomorphism $ad(\mathbf{M})$ exists for a degenerate matrix $\mathbf{M}$ (i. e., corank is one) since at least one 2×2-minor has a non-vanishing determinant. However, the biduality disappears, i. e., the original $\mathbf{M}$ cannot be recovered from $ad^2(\mathbf{M})$ because $rk(ad(\mathbf{M})) = 1$. This problem is solved by "adding" a tangent direction corresponding to the spatio-temporal evolution of $\mathbf{M}$. This tangent direction is the projection on the variety $\mathfrak{M}^1$ of the chord connecting two close locations passing through the degenerate matrix.

The prolongation of each chord $\overline{\mathbf{M}\mathbf{M}'}$ of $\mathfrak{M}^1$ gives a projective line ℓ through $\mathbf{M}$ whose generic element is a regular matrix, i. e., each point of the line represents a homography. Hence, data corresponding to a generic ℓ is determined by two homographies, which can be viewed as the "recent history" of the evolving tensor represented by eventually degenerated homographies.

If degenerate maps represented by $\mathbf{M} \in \mathfrak{M}^1$ persist (e. g., in unstructured scenes without perspective elements), the chords passing through $\mathbf{M}$ can be used. They are part of the secant variety $\mathrm{Sec}(1, \mathcal{F})$ to the fundamental variety $\mathcal{F}$. The secant variety is a PL-approach to the tangent space since it is not well defined for the most degenerate case.

These arguments provide a geometric interpretation of the classical algebraic argument [82] and the defense of the eight-point linear algorithm [53]. In this case, the estimation is computed using the local projection on the fundamental variety $\mathcal{F}$ of the chordal variety $\mathrm{Sec}(1, \mathcal{F})$. Furthermore, the underlying smooth structure of the generic elements lying on chords of $\mathrm{Sec}(1, \mathfrak{M}^1$ allows us to update the model estimation in order to avoid tracking losses. The first-order "expected variation" is represented by tangent elements in the space of completed endomorphisms $(\mathbf{M}, \mathrm{ad}(\mathbf{M})) \in \mathrm{End}(V) \times \mathrm{End}(V)^{\vee}$ up to scale.

A.2 SEQUENCE CHARACTERISTICS

The characteristics of a sequence refers to the special traits that make that sequence specially challenging for monocular SLAM. Each characteristics is labeled with a value between 1 and 5 taking into account the presence of the challenge in the sequence. More information about these characteristics, including their meaning, is described in Section 3.2.2.

The table has been split in two parts according to the value assigned by the linear model of the SVM classifier described at Section 3.3.5. On the one hand, Table A.1 includes the sequences where the linear function returns a negative value (i. e., the classifier predicts that the ATE of KN-SLAM is lower). On the other hand, Table A.2 contains the sequences where the trained SVM classifier predicts that ORB-SLAM achieves a lower ATE in the trajectory of the sequence.

SEQUENCE	FAST	SHAKE	CLOSE RANGE	ROTATION	TRANSLATION	DYNAMIC	LOOP	HF TEXTURE	LOW LIGHTING	SVM
euroc/V103	5	3	2	5	4	1	5	1	5	−6.32
euroc/MH05	3	1	1	3	3	1	4	1	5	−5.80
euroc/MH04	3	2	1	3	3	1	3	1	5	−5.39
euroc/MH03	4	3	1	5	3	1	5	1	3	−4.72
euroc/V201	1	3	1	5	3	1	5	1	1	−4.54
euroc/V101	1	1	1	5	3	1	5	1	1	−4.38
euroc/MH02	3	2	1	2	4	1	5	1	2	−2.84
euroc/MH01	4	2	1	3	4	1	5	1	2	−2.75
tum/fr2_desk	1	3	2	3	3	1	4	1	1	−2.49
euroc/V202	5	5	2	5	5	1	5	1	1	−2.44
tum/fr3_nstr_tex_near	1	3	5	5	3	1	4	2	1	−1.91
icl/lr_kt0	1	1	1	5	2	1	3	3	1	−1.66
tum/fr2_xyz	1	1	3	1	5	1	4	1	1	−1.62
tum/fr3_str_tex_far	1	4	3	3	5	1	1	1	1	−1.60
tum/fr3_long_office	2	2	3	3	3	1	4	1	1	−1.36
icl/of_kt0	1	3	4	5	2	1	4	3	1	−1.19
kitti/9	5	4	2	4	4	2	4	4	4	−1
icl/lr_kt2	1	1	2	3	3	1	4	3	1	−1
kitti/7	2	2	1	3	5	3	4	2	2	−1
kitti/0	4	1	1	2	5	3	5	3	4	−1
euroc/V102	5	3	2	5	2	1	5	1	1	−1
kitti/6	3	2	1	2	5	1	4	2	1	−1
kitti/5	3	1	1	2	5	3	4	2	3	−0.75
tum/fr3_str_tex_near	1	4	5	3	4	1	1	1	1	−0.43
tum/fr2_desk_person	1	3	2	3	3	3	4	1	1	−0.09

Table A.1: Characterization of the sequences with a value for the linear model of SVM < 0. Sequences are sorted by that value in ascending order. Each characteristic represents a challenging trait for SLAM, valued on a scale with five levels to asses the results of a monocular SLAM system.

SEQUENCE	FAST	SHAKE	CLOSE RANGE	ROTATION	TRANSLATION	DYNAMIC	LOOP	HF TEXTURE	LOW LIGHTING	SVM
kitti/8	4	3	1	2	5	2	1	2	3	0.03
icl/of_kt3	1	2	2	4	2	1	4	5	1	0.08
icl/of_kt1	1	1	3	4	3	1	4	5	1	0.11
tum/fr1_xyz	3	2	5	1	5	1	4	1	1	0.42
tum/fr2_360_kidnap	1	2	1	5	2	4	3	1	1	0.53
kitti/2	5	2	1	2	5	3	4	4	4	0.76
tum/fr3_nstr_tex_far	1	5	5	2	5	1	1	3	1	0.99
icl/of_kt2	1	2	4	3	3	1	4	5	1	0.99
kitti/10	4	3	2	3	5	3	1	2	3	1
tum/fr3_sit_xyz	1	4	1	1	5	4	4	1	1	1
tum/fr3_sit_halfsph	1	4	4	5	2	4	4	1	1	1
tum/fr3_sit_rpy	2	4	1	5	1	4	4	1	1	1
euroc/V203	5	5	4	5	4	3	5	1	1	1.13
kitti/3	3	2	1	2	5	3	1	3	3	1.29
tum/fr1_desk	5	3	2	4	3	1	2	2	1	1.29
icl/lr_kt1	1	1	3	5	2	1	1	5	1	1.39
icl/lr_kt3	1	1	5	3	3	1	4	5	1	1.45
tum/fr3_walk_halfsph	2	4	2	5	2	5	4	1	1	2.14
tum/fr3_walk_xyz	3	4	1	1	5	5	4	1	1	3.57
tum/fr3_walk_rpy	4	4	1	5	1	5	4	1	1	3.57
kitti/4	4	1	1	1	5	3	1	3	2	3.77
tum/fr1_floor	3	1	4	3	3	2	2	4	1	3.95

TABLE A.2: Characterization of the sequences with a value for the linear model of SVM > 0. Sequences are sorted by that value in ascending order. Each characteristic represents a challenging trait for SLAM, valued on a scale with five levels to asses the results of a monocular SLAM system.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
00	99.10 %	**99.22** %	−0.12 %
02	**99.32** %	91.31 %	8.01 %
03	98.46 %	**99.55** %	−1.09 %
04	96.50 %	**99.63** %	−3.13 %
05	**99.97** %	99.46 %	0.51 %
06	97.11 %	**98.46** %	−1.35 %
07	93.27 %	**93.50** %	−0.23 %
08	**99.30** %	85.36 %	13.94 %
09	**95.99** %	89.70 %	6.29 %
10	96.16 %	**97.13** %	−0.97 %

Table A.3: Comparison of the Percentage of Tracked Trajectory (PTT) for each sequence in the KITTI dataset. The PTT is estimated as the mean value of all executions where the system bootstrap. The best value for each sequence (greater is better) is highlighted in bold. The last column shows the difference between KN-SLAM and ORB-SLAM.

A.3 Percentage of Tracked Trajectory

This section includes the complete results achieved in our evaluation of the robustness using the PTT as metric. They are separated by the dataset where the sequences belong to. The analysis and discussion of these results can be found in Section 3.3.2.

The trajectories generated by KN-SLAM and ORB-SLAM in the KITTI dataset (Table A.3) have similar length. Sequence `08` exhibits the highest difference since ORB-SLAM lost the tracking in the middle of the sequence three of eleven times. It can also be seen that the robustness of KN-SLAM w.r.t. ORB-SLAM is higher in most sequences of the TUM RGB-D dataset (see Table A.4). There are almost no differences between both systems in the sequences of the EuRoC dataset (see Table A.5). The main difference is observed in the sequence `V202`, where ORB-SLAM bootstraps erroneously in four of eleven executions. Finally, the results achieved in the sequences of the ICL-NUIM dataset are shown in Table A.6. Here, we can see that ORB-SLAM only bootstrap in three sequences. The differences between the PTT achieved by both systems are almost imperceptible.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
fr1_xyz	**87.53** %	68.02 %	19.51 %
fr2_xyz	**94.31** %	92.75 %	1.56 %
fr1_floor	**50.53** %		
fr1_desk	**82.63** %	60.94 %	21.69 %
fr2_360_kidnap	**44.40** %	39.02 %	5.38 %
fr2_desk	**81.41** %	74.31 %	7.10 %
fr2_desk_person	92.47 %	**93.34** %	−0.87 %
fr3_long_office	**98.47** %	94.59 %	3.88 %
fr3_nstr_tex_far	**47.67** %		
fr3_nstr_tex_near	95.64 %	**95.67** %	−0.03 %
fr3_str_tex_far	**93.37** %	91.87 %	1.50 %
fr3_str_tex_near	92.48 %	**92.92** %	−0.44 %
fr3_sit_xyz	**92.12** %	85.82 %	6.30 %
fr3_sit_halfsph	**94.31** %	86.62 %	7.69 %
fr3_sit_rpy	**75.79** %	65.04 %	10.75 %
fr3_walk_xyz	**70.43** %		
fr3_walk_halfsph	85.82 %	**86.50** %	−0.68 %
fr3_walk_rpy	84.13 %	**89.17** %	−5.04 %

Table A.4: Comparison of the Percentage of Tracked Trajectory (PTT) for each sequence in the TUM RGB-D dataset. The PTT is estimated as the mean value of all executions where the system bootstrap. The best value for each sequence (greater is better) is highlighted in bold. The last column shows the difference between KN-SLAM and ORB-SLAM.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
MH01	**87.28** %	84.38 %	2.90 %
MH02	90.38 %	**93.97** %	−3.59 %
MH03	94.96 %	**96.16** %	−1.20 %
MH04	**91.93** %	81.33 %	10.60 %
MH05	91.29 %	**92.83** %	−1.54 %
V101	93.31 %	**94.32** %	−1.01 %
V102	**91.20** %	90.00 %	1.20 %
V103	**89.23** %	86.82 %	2.41 %
V201	**90.86** %	87.98 %	2.88 %
V202	**84.33** %	59.15 %	25.18 %
V203	84.16 %	**85.75** %	−1.59 %

Table A.5: Comparison of the Percentage of Tracked Trajectory (PTT) for each sequence in the EuRoC dataset. The PTT is estimated as the mean value of all executions where the system bootstrap. The best value for each sequence (greater is better) is highlighted in bold. The last column shows the difference between KN-SLAM and ORB-SLAM.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
lr_kt0	**91.15** %		
lr_kt1	**46.78** %		
lr_kt2	90.29 %	**93.19** %	−2.90 %
lr_kt3	**93.47** %		
of_kt0	**91.07** %		
of_kt1	**47.45** %	42.60 %	4.85 %
of_kt2	93.02 %	**93.31** %	−0.29 %
of_kt3	**69.55** %		

Table A.6: Comparison of the Percentage of Tracked Trajectory (PTT) for each sequence in the ICL-NUIM dataset. The PTT is estimated as the mean value of all executions where the system bootstrap. The best value for each sequence (greater is better) is highlighted in bold. The last column shows the difference between KN-SLAM and ORB-SLAM.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
00	**6.4525** (11)	6.8172 (11)	5.35 %
02	25.8591 (11)	**25.7361** (11)	−0.48 %
03	**1.1012** (11)	1.2076 (11)	8.81 %
04	1.0167 (11)	**0.9151** (11)	−9.99 %
05	**4.2906** (11)	5.8852 (11)	27.10 %
06	**12.6292** (11)	14.5704 (11)	13.32 %
07	**1.6884** (11)	2.0226 (11)	16.52 %
08	54.4592 (11)	**51.4204** (11)	−5.58 %
09	**7.6201** (11)	37.0662 (11)	79.44 %
10	8.8903 (11)	**7.0203** (11)	−21.03 %

Table A.7: Absolute Trajectory Error (ATE) in the sequences of the KITTI dataset. Each cell contains the median ATE after eleven alternative executions of the system for each sequence. The number of executions where the system bootstraps the sequence is expressed between parentheses. The lowest value for each sequence is highlighted in bold. Empty cells corresponds to non-comparable sequences, i. e., sequences where a system tracks fewer than 30 % of the frames or bootstraps less than three of eleven times.

A.4 Absolute Trajectory Error

The ATE, as defined in Section 3.3.3, is the RMSE of the positions of the camera along the keyframes of the trajectory. The tables gathering the results for each sequence have been collected in this section, although the corresponding results for each sequence are analyzed in Section 3.2.4.

The results achieved in the KITTI dataset are mixed with a significant difference in the `09` sequence, as it can be seen in Table A.7. Results in the sequences of the TUM RGB-D dataset are shown in Table A.8. Here, ORB-SLAM finds problems to track the collection of sequences with moving objects. However, desktop-based sequences are tracked more accurately than KN-SLAM. Next table is Table A.9, which corresponds to the EuRoC dataset. In this dataset KN-SLAM outperforms the accuracy of ORB-SLAM in all sequences, except `V203` due to the fast camera movements that cause relocalizations. Finally, Table A.10 contains the results for the trajectories of the sequences in the ICL-NUIM datasets. Only three sequences are comparable since they are difficult to bootstrap with a sparse approach. In the `of_kt1` sequence KN-SLAM finds a false positive at loop detection that explains the higher error.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
fr1_xyz	0.0093 (11)	**0.0087** (11)	−6.04 %
fr2_xyz	**0.0025** (11)	0.0025 (11)	0.72 %
fr1_floor	0.0088 (11)		
fr1_desk	0.0152 (11)	**0.0135** (11)	−10.94 %
fr2_360_kidnap	0.0365 (11)	**0.0318** (9)	−12.96 %
fr2_desk	**0.0079** (11)	0.0098 (11)	19.11 %
fr2_desk_person	**0.0082** (11)	0.0082 (11)	0.22 %
fr3_long_office	**0.0116** (11)	0.0122 (11)	5.30 %
fr3_nstr_tex_far	0.0768 (5)		
fr3_nstr_tex_near	**0.0137** (11)	0.0138 (11)	0.75 %
fr3_str_tex_far	**0.0085** (11)	0.0107 (11)	20.01 %
fr3_str_tex_near	**0.0128** (11)	0.0129 (11)	1.31 %
fr3_sit_xyz	0.0123 (11)	**0.0087** (11)	−29.45 %
fr3_sit_halfsph	0.0153 (11)	**0.0143** (11)	−6.54 %
fr3_sit_rpy	0.0301 (11)	**0.0207** (11)	−31.15 %
fr3_walk_xyz	**0.0124** (11)		
fr3_walk_halfsph	0.0220 (11)	**0.0178** (11)	−18.94 %
fr3_walk_rpy	0.0898 (11)	**0.0704** (11)	−21.66 %

Table A.8: Absolute Trajectory Error (ATE) in the sequences of the TUM RGB-D dataset. Each cell contains the median ATE after eleven alternative executions of the system for each sequence. The number of executions where the system bootstraps the sequence is expressed between parentheses. The lowest value for each sequence is highlighted in bold. Empty cells corresponds to non-comparable sequences, i. e., sequences where a system tracks fewer than 30 % of the frames or bootstraps less than three of eleven times.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
MH01	**0.0136** (11)	0.0161 (11)	15.59 %
MH02	**0.0149** (11)	0.0154 (11)	3.75 %
MH03	**0.0205** (11)	0.0239 (11)	14.24 %
MH04	**0.0606** (11)	0.0628 (11)	3.39 %
MH05	**0.0399** (11)	0.0436 (11)	8.35 %
V101	**0.0350** (11)	0.0357 (11)	1.86 %
V102	**0.0107** (11)	0.0109 (11)	1.58 %
V103	**0.0216** (11)	0.0313 (11)	30.93 %
V201	**0.0157** (11)	0.0230 (11)	31.78 %
V202	**0.0149** (11)	0.0166 (11)	10.32 %
V203	0.1247 (11)	**0.0941** (11)	−24.56 %

Table A.9: Absolute Trajectory Error (ATE) in the sequences of the EuRoC dataset. Each cell contains the median ATE after eleven alternative executions of the system for each sequence. The number of executions where the system bootstraps the sequence is expressed between parentheses. The lowest value for each sequence is highlighted in bold. Empty cells corresponds to non-comparable sequences, i. e., sequences where a system tracks fewer than 30 % of the frames or bootstraps less than three of eleven times.

SEQUENCE	KN-SLAM	ORB-SLAM	DIFFERENCE
lr_kt0	0.0047 (11)		
lr_kt1	0.0177 (11)		
lr_kt2	**0.0237** (11)	0.0324 (11)	26.76 %
lr_kt3	**0.0331** (11)		
of_kt0	**0.0528** (11)		
of_kt1	0.8780 (11)	**0.8162** (11)	−7.05 %
of_kt2	**0.0219** (11)	0.0239 (11)	8.12 %
of_kt3	0.1348 (11)		

Table A.10: Absolute Trajectory Error (ATE) in the sequences of the ICL-NUIM dataset. Each cell contains the median ATE after eleven alternative executions of the system for each sequence. The number of executions where the system bootstraps the sequence is expressed between parentheses. The lowest value for each sequence is highlighted in bold. Empty cells corresponds to non-comparable sequences, i. e., sequences where a system tracks fewer than 30 % of the frames or bootstraps less than three of eleven times.

BIBLIOGRAPHY

[1] Madjid Allili, Konstantin Mischaikow, and Allen Tannenbaum. "Cubical homology and the topological classification of 2D and 3D imagery." In: *Proc. of the 2001 International Conference on Image Processing, Thessaloniki, Greece, October 7-10.* Vol. 2. IEEE. 2001, pp. 173–176.

[2] Edward Angel and Dave Shreiner. *Interactive Computer Graphics with WebGL*. Addison-Wesley Professional, 2014.

[3] Remi Arnaud and Mark C Barnes. *COLLADA: sailing the gulf of 3D digital content creation*. AK Peters/CRC Press, 2006.

[4] Ken Arroyo et al. *GeoBIM project*. June 30, 2018. URL: https://3d.bk.tudelft.nl/ken/files/18_geobim.pdf.

[5] Salman Azhar. "Building information modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry." In: *Leadership and management in engineering* 11.3 (2011), pp. 241–252.

[6] Caroline Baillard and Andrew Zisserman. "Automatic reconstruction of piecewise planar models from multiple views." In: *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* Vol. 2. IEEE. 1999, pp. 559–565.

[7] Hans-Jürgen Bandelt and Victor Chepoi. "Graphs of acyclic cubical complexes." In: *European Journal of Combinatorics* 17.2 (1996), pp. 113–120.

[8] Avrim L Blum and Pat Langley. "Selection of relevant features and examples in machine learning." In: *Artificial intelligence* 97.1-2 (1997), pp. 245–271.

[9] John Adrian Bondy. *Graph Theory With Applications*. Oxford, UK, UK: Elsevier Science Ltd., 1976. ISBN: 0444194517.

[10] A Borrmann. "From GIS to BIM and back again — A spatial query language for 3D building models and 3D city models." In: *Proceedings of the 5th International 3D GeoInfo Conference, Berlin*. Vol. XXXVIII-4/W15. 2010.

[11] Don Brutzman and Leonard Daly. *X3D: extensible 3D graphics for Web authors*. Elsevier, 2010.

[12] Christopher JC Burges. "A tutorial on support vector machines for pattern recognition." In: *Data mining and knowledge discovery* 2.2 (1998), pp. 121–167.

[13] Michael Burri et al. "The EuRoC micro aerial vehicle datasets." In: *The International Journal of Robotics Research* 35.10 (2016), pp. 1157–1163.

[14] Cesar Cadena et al. "Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age." In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332.

[15] Rikk Carey and Gavin Bell. *The annotated VRML 2.0 reference manual*. Vol. 15. Addison-Wesley Reading, MA, 1997.

[16] Luca Carlone et al. "Initialization techniques for 3D SLAM: a survey on rotation estimation and its use in pose graph optimization." In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 4597–4604.

[17] Rich Caruana and Alexandru Niculescu-Mizil. "An empirical comparison of supervised learning algorithms." In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 161–168.

[18] Javier Civera, Andrew J Davison, and JM Martinez Montiel. "Inverse depth parametrization for monocular SLAM." In: *IEEE Transactions on Robotics* 24.5 (2008), pp. 932–945.

[19] Alejo Concha and Javier Civera. "Using superpixels in monocular SLAM." In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 365–372.

[20] Eduardo Bayro Corrochano and Garret Sobczyk. *Geometric algebra with applications in science and engineering*. Springer Science & Business Media, 2011.

[21] Isabel F Cruz, William Sunna, and Anjli Chaudhry. "Semi-automatic ontology alignment for geospatial data integration." In: *Geographic Information Science*. Springer, 2004, pp. 51–66.

[22] Brian Curless and Marc Levoy. "A volumetric method for building complex models from range images." In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM. 1996, pp. 303–312.

[23] Angela Dai et al. "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration." In: *ACM Transactions on Graphics (TOG)* 36.3 (2017), p. 24.

[24] Yuchao Dai, Hongdong Li, and Laurent Kneip. "Rolling shutter camera relative pose: Generalized epipolar geometry." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4132–4140.

[25] Andrew J Davison et al. "MonoSLAM: Real-time single camera SLAM." In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 6 (2007), pp. 1052–1067.

[26] John Dawes. "Do data characteristics change according to the number of scale points used? An experiment using 5-point, 7-point and 10-point scales." In: *International Journal of Market Research* 50.1 (2008), pp. 61–77.

[27] Francisco Delgado and Javier Finat. "Complete endomorphisms in Computer Vision." In: *Submitted to the International Journal of Mathematical Imaging and Vision (Under revision)* (2018).

[28] Francisco Delgado, M. Mercedes Martinez-Gonzalez, and Javier Finat. "An evaluation of ontology matching techniques on geospatial ontologies." In: *International Journal of Geographical Information Science* 27.12 (2013). (**JCR, Q1**), pp. 2279–2301. DOI: 10.1080/13658816.2013.812215.

[29] Francisco Delgado et al. "Towards a client-oriented integration of construction processes and building GIS systems." In: *Computers in Industry* 73 (2015). (**JCR, Q2**), pp. 51–68.

[30] Sahibsingh A Dudani. "The distance-weighted k-nearest-neighbor rule." In: *IEEE Transactions on Systems, Man, and Cybernetics* 4 (1976), pp. 325–327.

[31] Maksym Dzitsiuk et al. "De-noising, stabilizing and completing 3d reconstructions on-the-go using plane priors." In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 3976–3983.

[32] Chuck Eastman et al. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. John Wiley & Sons, 2011.

[33] Jakob Engel, Vladlen Koltun, and Daniel Cremers. "Direct sparse odometry." In: *IEEE transactions on pattern analysis and machine intelligence* (2017).

[34] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM." In: *European Conference on Computer Vision*. Springer. 2014, pp. 834–849.

[35] Bernard Espiau, François Chaumette, and Patrick Rives. "A new approach to visual servoing in robotics." In: *IEEE Transactions on Robotics and Automation* 8.3 (1992), pp. 313–326.

[36] Jérôme Euzenat, Pavel Shvaiko, et al. *Ontology matching*. Vol. 18. Springer, 2007.

[37] Jianqing Fan and Runze Li. "Variable selection via nonconcave penalized likelihood and its oracle properties." In: *Journal of the American statistical Association* 96.456 (2001), pp. 1348–1360.

[38] Olivier D Faugeras and Francis Lustman. "Motion and structure from motion in a piecewise planar environment." In: *International Journal of Pattern Recognition and Artificial Intelligence* 2.03 (1988), pp. 485–508.

[39] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. "SVO: Fast semi-direct monocular visual odometry." In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 15–22.

[40] Christian Forster et al. "SVO: Semidirect visual odometry for monocular and multicamera systems." In: *IEEE Transactions on Robotics* 33.2 (2017), pp. 249–265.

[41] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York, 2001.

[42] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. "Piecewise planar and non-planar stereo for urban scene reconstruction." In: (2010).

[43] Dorian Gálvez-López and Juan D Tardos. "Bags of binary words for fast place recognition in image sequences." In: *IEEE Transactions on Robotics* 28.5 (2012), pp. 1188–1197.

[44] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[45] Ross Girshick. "Fast R-CNN." In: *IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2015, pp. 1440–1448.

[46] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. "Unsupervised Monocular Depth Estimation with Left-Right Consistency." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6602–6611. DOI: 10.1109/CVPR.2017.699.

[47] Johannes Gräter, Tobias Schwarze, and Martin Lauer. "Robust scale estimation for monocular visual odometry using structure from motion and vanishing points." In: *IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 475–480.

[48] Michael D Grossberg and Shree K Nayar. "Modeling the space of camera response functions." In: *IEEE transactions on pattern analysis and machine intelligence* 26.10 (2004), pp. 1272–1282.

[49] Isabelle Guyon et al. "Gene selection for cancer classification using support vector machines." In: *Machine learning* 46.1-3 (2002), pp. 389–422.

[50] Hans Hagen, Heinrich Müller, and Gregory M Nielson. *Scientific Visualization*. Schloss Dagstuhl, 1991.

[51] A. Handa et al. "A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM." In: *IEEE Intl. Conf. on Robotics and Automation, ICRA*. Hong Kong, China, May 2014.

[52] Charles D Hansen and Chris R Johnson. *The Visualization Handbook*. Academic Press, 2005.

[53] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[54] Kaiming He et al. "Deep residual learning for image recognition." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[55] Kaiming He et al. "Mask r-cnn." In: *IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2017, pp. 2980–2988.

[56] Johan Hedborg et al. "Rolling shutter bundle adjustment." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2012, pp. 1434–1441.

[57] Ihab Hijazi et al. "Initial investigations for modeling interior utilities within 3D Geo Context: Transforming IFC-interior utility to CityGML/UtilityNetworkADE." In: *Advances in 3D Geo-Information Sciences*. Ed. by T.Kolbe et al. Lecture Notes in Geoinformation and Cartography. Springer-Verlag, 2011, pp. 95–113.

[58] Berthold KP Horn. "Closed-form solution of absolute orientation using unit quaternions." In: *Journal of the Optical Society of America. A, Optics and image science* 4.4 (1987), pp. 629–642.

[59] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

[60] Ming Hsiao et al. "Keyframe-based dense planar SLAM." In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 5110–5117.

[61] Matthias Innmann et al. "Volumedeform: Real-time volumetric non-rigid reconstruction." In: *European Conference on Computer Vision*. Springer. 2016, pp. 362–379.

[62] Michal Irani and P Anandan. "About direct methods." In: *Vision Algorithms: Theory and Practice*. Springer. 2000, pp. 267–277.

[63] K Ismo et al. "Outlier detection using k-nearest neighbour graph." In: *null*. IEEE. 2004, pp. 430–433.

[64] Shahram Izadi et al. "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera." In: *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM. 2011, pp. 559–568.

[65] Olaf Kähler, Victor A Prisacariu, and David W Murray. "Real-time large-scale dense 3d reconstruction with loop closure." In: *European Conference on Computer Vision*. Springer. 2016, pp. 500–516.

[66] Sawako Kaijima et al. "Computational Fluid Dynamics for Architectural Design." In: *Architectural Design Special Issue: Computation Works: The Building of Algorithmic Thought* 83.2 (2013), pp. 118–123.

[67] Zhuoliang Kang and Gerard Medioni. "Progressive 3D model acquisition with a commodity hand-held camera." In: *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*. IEEE. 2015, pp. 270–277.

[68] Georg Klein and David Murray. "Parallel Tracking and Mapping for Small AR Workspaces." In: *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE. Nov. 2007, pp. 225–234.

[69] Donald Ervin Knuth. *The art of computer programming*. Vol. 3. Pearson Education, 1997.

[70] Mohammad Kolahdouzan and Cyrus Shahabi. "Voronoi-based k nearest neighbor search for spatial network databases." In: *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment. 2004, pp. 840–851.

[71] Thomas H Kolbe. "Representing and Exchanging 3D City Models with CityGML." In: *3D Geo-Information Sciences* (2009), pp. 15–31.

[72] Oliver Kramer. "K-nearest neighbors." In: *Dimensionality reduction with unsupervised nearest neighbors*. Springer, 2013, pp. 13–23.

[73] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[74] Ruben de Laat and Léon van Berlo. "Integration of BIM and GIS: The development of the CityGML GeoBIM extension." In: *Advances in 3D geo-information sciences*. Springer, 2011, pp. 211–225.

[75] Craig Larman and Victor R Basili. "Iterative and incremental developments. a brief history." In: *Computer* 36.6 (2003), pp. 47–56.

[76] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. "EPnP: An accurate O(n) solution to the PnP problem." In: *International Journal of Computer Vision* 81.2 (2009), p. 155.

[77] Stefan Leutenegger et al. "Keyframe-based visual-inertial odometry using nonlinear optimization." In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[78] T Liebich et al. *Industry Foundation Classes 4*. June 30, 2018. URL: http://www.%20buildingsmart-tech.%20org/ifc/IFC4/final/html.

[79] Fayao Liu, Chunhua Shen, and Guosheng Lin. "Deep convolutional neural fields for depth estimation from a single image." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5162–5170.

[80] H. Liu, G. Zhang, and H. Bao. "Robust Keyframe-based Monocular SLAM for Augmented Reality." In: *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Sept. 2016, pp. 1–10. DOI: 10.1109/ISMAR.2016.24.

[81] Haomin Liu, Guofeng Zhang, and Hujun Bao. "Robust keyframe-based monocular SLAM for augmented reality." In: *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE. 2016, pp. 1–10.

[82] H. Longuet-Higgins. "The reconstruction of a plane surface from two perspective projections." In: *Proc. Royal Soc. London Ser. B, Biol, Sci* 227.1249 (1986), pp. 399–410.

[83] Yi Ma et al. *An invitation to 3-d vision: from images to geometric models*. Vol. 26. Springer Science & Business Media, 2012.

[84] Florian Markowetz and Rainer Spang. "Molecular diagnosis." In: *Methods of information in medicine* 44.03 (2005), pp. 438–443.

[85] Daniel Martinec and Tomas Pajdla. "Robust rotation and translation estimation in multiview reconstruction." In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2007, pp. 1–8.

[86] Donald Meagher. "Geometric modeling using octree encoding." In: *Computer graphics and image processing* 19.2 (1982), pp. 129–147.

[87] Etienne Mouragnon et al. "Real time localization and 3d reconstruction." In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2006, pp. 363–370.

[88] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system." In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.

[89] Raul Mur-Artal and Juan D Tardós. "Probabilistic Semi-Dense Mapping from Highly Accurate Feature-Based Monocular SLAM." In: *Robotics: Science and Systems*. 2015.

[90] Raul Mur-Artal and Juan D Tardós. "ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras." In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

[91] Luigi Nardi et al. "Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM." In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 5783–5790.

[92] Matthias Nießner et al. "Real-time 3D reconstruction at scale using voxel hashing." In: *ACM Transactions on Graphics (ToG)* 32.6 (2013), p. 169.

[93] David Nistér, Oleg Naroditsky, and James Bergen. "Visual odometry." In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1. IEEE. 2004, pp. I–I.

[94] Romain Nouvel et al. "SimStadt, a new workflow-driven urban energy simulation platform for CityGML city models." In: *Proceedings of International Conference CISBAT 2015: Future Buildings and Districts Sustainability from Nano to Urban Scale*. EPFL-CONF-213437. LESO-PB, EPFL. 2015, pp. 889–894.

[95] Thuy Duong Oesterreich and Frank Teuteberg. "Looking at the big picture of IS investment appraisal through the lens of systems theory: A System Dynamics approach for understanding the economic impact of BIM." In: *Computers in Industry* 99 (2018), pp. 262–281.

[96] Peter van Oosterom and Jantien Stoter. "5D Data Modelling: Full Integration of 2D/3D Space, Time and Scale Dimensions." In: *Geographic Information Science*. Springer, 2010, pp. 310–324.

[97] Luc Oth et al. "Rolling shutter camera calibration." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2013, pp. 1360–1367.

[98] Rodrigo Paredes and Edgar Chávez. "Using the k-nearest neighbor graph for proximity searching in metric spaces." In: *International Symposium on String Processing and Information Retrieval*. Springer. 2005, pp. 127–138.

[99] Seonwook Park, Thomas Schöps, and Marc Pollefeys. "Illumination change robustness in direct visual SLAM." In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 4523–4530.

[100] Geoffrey Pascoe et al. "NID-SLAM: Robust Monocular SLAM using Normalised Information Distance." In: *Conference on Computer Vision and Pattern Recognition*. 2017.

[101] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[102] Pedro Pinies, Lina Maria Paz, and Paul Newman. "Dense mono reconstruction: Living with the pain of the plain plane." In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 5226–5231.

[103] Vivek Pradeep et al. “MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera.” In: *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*. IEEE. 2013, pp. 83–88.

[104] Joseph Redmon et al. “You only look once: Unified, real-time object detection.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[105] Shaoqing Ren et al. “Faster r-cnn: Towards real-time object detection with region proposal networks.” In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[106] Christian Robert. “Machine Learning, a Probabilistic Perspective.” In: *CHANCE* 27.2 (2014), pp. 62–63. DOI: 10.1080/09332480.2014.914768. URL: https://doi.org/10.1080/09332480.2014.914768.

[107] Fabrice Robinet and P Cozzi. *Gltf—The Runtime Asset Format for WebGL, OpenGL ES, and OpenGL*. 2013. URL: https://github.com/KhronosGroup/glTF/blob/master/specification/README.%20md.

[108] Ethan Rublee et al. “ORB: An efficient alternative to SIFT or SURF.” In: *IEEE International Conference in Computer Vision (ICCV), 2011*. IEEE. 2011, pp. 2564–2571.

[109] Dennis W Ruck, Steven K Rogers, and Matthew Kabrisky. “Feature selection using a multilayer perceptron.” In: *Journal of Neural Network Computing* 2.2 (1990), pp. 40–48.

[110] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors.” In: *nature* 323.6088 (1986), p. 533.

[111] Davide Scaramuzza and Friedrich Fraundorfer. “Visual Odometry. Part I: The First 30 Years and Fundamentals.” In: *IEEE Robotics & Automation Magazine* 18.4 (2011), pp. 80–92.

[112] Johannes Lutz Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo.” In: *European Conference on Computer Vision (ECCV)*. 2016.

[113] Klaus Schwab. *The fourth industrial revolution*. Crown Business, 2017.

[114] Ken Shoemake. “Animating rotation with quaternion curves.” In: *ACM SIGGRAPH computer graphics*. Vol. 19. 3. ACM. 1985, pp. 245–254.

[115] Ravi Srinivasan et al. “Preliminary research in dynamic-BIM (D-BIM) workbench development.” In: *Proceedings of the Winter Simulation Conference*. Winter Simulation Conference. 2012, p. 53.

[116] Frank Steinbrücker, Jürgen Sturm, and Daniel Cremers. “Volumetric 3D mapping in real-time on a CPU.” In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 2021–2028.

[117] JE Stoter and Siyka Zlatanova. “3D GIS, where are we standing?” In: *ISPRS Joint Workshop on Spatial, Temporal and Multi-Dimensional Data Modelling and Analysis', Québec*. 2003.

[118] Hauke Strasdat. "Local accuracy and global consistency for efficient visual SLAM." PhD thesis. Imperial College London, London, U.K, Oct. 2012.

[119] Hauke Strasdat, JMM Montiel, and Andrew J Davison. "Scale drift-aware large scale monocular SLAM." In: *Proceedings of Robotics: Science and Systems. Zaragoza, (Spain), Jun.* University of Zaragoza, 2010.

[120] Hauke Strasdat et al. "Double window optimisation for constant time visual SLAM." In: *Proceedings of the 2011 International Conference on Computer Vision*. IEEE Computer Society. 2011, pp. 2352–2359.

[121] Jürgen Sturm et al. "A benchmark for the evaluation of RGB-D SLAM systems." In: *Proc. of the International Conference on Intelligent Robot Systems (IROS)*. IEEE. 2012, pp. 573–580.

[122] Petri Tanskanen et al. "Semi-direct EKF-based monocular visual-inertial odometry." In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 6073–6078.

[123] Keisuke Tateno et al. "CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2017, pp. 6565–6574.

[124] Philip HS Torr and Andrew Zisserman. "Feature based methods for structure and motion estimation." In: *Vision Algorithms: Theory and Practice*. Springer. 2000, pp. 278–294.

[125] Roberto Tron and René Vidal. "Distributed 3-D localization of camera sensor networks from 2-D image measurements." In: *IEEE Transactions on Automatic Control* 59.12 (2014), pp. 3325–3340.

[126] Tao-chiu Kenny Tse, Kamdin Andy Wong, and KF Wong. "The utilisation of building information models in nD modelling: a study of data interfacing and adoption barriers." In: *Special Issue From 3D to nD modelling*. Vol. 10. ITcon, 2005, pp. 85–110. URL: http://www.itcon.org/2005/8.

[127] Benjamin Ummenhofer et al. "Demon: Depth and motion network for learning monocular stereo." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 5. 2017.

[128] Linda Van den Brink, JE Stoter, and Sisi Zlatanova. "Modelling an application domain extension of CityGML in UML." In: *7th International ISPRS Conference on 3D Geoinformation, The International Archives on the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume XXXVIII-4, part C26, 16-17 May 2012, Québec, Canada*. ISPRS, 2012, pp. 11–14. DOI: 10.5194/isprsarchives-XXXVIII-4-C26-11-2012. URL: http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XXXVIII-4-C26/11/2012/.

[129] George Vogiatzis and Carlos Hernández. "Video-based, real-time multi-view stereo." In: *Image and Vision Computing* 29.7 (2011), pp. 434–441.

[130] Fumihiro Wakai, Naoya Enomoto, and Hiroshi Ogawa. "Three-dimensional microstructural evolution in ideal grain growth—general statistics." In: *Acta Materialia* 48.6 (2000), pp. 1297–1311. ISSN: 1359-6454. DOI: https://doi.org/10.1016/S1359-6454(99)00405-X. URL: http://www.sciencedirect.com/science/article/pii/S135964549900405X.

[131] Jason Weston et al. "Feature selection for SVMs." In: *Advances in neural information processing systems*. 2001, pp. 668–674.

[132] Thomas Whelan et al. "Kintinuous: Spatially Extended KinectFusion." In: *Workshop on RGB-D: Advanced Reasoning with Depth Cameras, in conjunction with Robotics: Science and Systems*. 2012.

[133] Thomas Whelan et al. "ElasticFusion: Dense SLAM Without A Pose Graph." In: *Robotics: Science and Systems*. Vol. 11. 2015.

[134] Ramin Zabih and John Woodfill. "Non-parametric local transforms for computing visual correspondence." In: *European conference on computer vision*. Springer. 1994, pp. 151–158.

[135] Hao Helen Zhang et al. "Gene selection using support vector machines with non-convex penalty." In: *bioinformatics* 22.1 (2005), pp. 88–95.

[136] Tinghui Zhou et al. "Unsupervised Learning of Depth and Ego-Motion from Video." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2017, pp. 6612–6619. DOI: 10.1109/CVPR.2017.700.

[137] M Zeeshan Zia et al. "Comparative design space exploration of dense and semi-dense SLAM." In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1292–1299.

[138] Siyka Zlatanova, Alias Abdul Rahman, and Wenzhong Shi. "Topological models and frameworks for 3D spatial objects." In: *Computers & geosciences* 30.4 (2004), pp. 419–428.

COLOPHON

This document was typeset in LaTeX using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography *"The Elements of Typographic Style"*. Most of the graphics were generated using Open Source projects like `inkscape` and `matplotlib`. The bibliography was typeset using `biblatex`.