Isovist computation of outdoor environment with semi-dense line SLAM and monocular camera

Thomas Le Jan, Myriam Servières, Thomas Leduc, Vincent Tourre

▶ **To cite this version:**

Thomas Le Jan, Myriam Servières, Thomas Leduc, Vincent Tourre. Isovist computation of outdoor environment with semi-dense line SLAM and monocular camera. SAGEO 2021, 16th Spatial Analysis and Geomatics Conference, Alain Bouju; Christine Plumejeaud-Perreau, May 2021, La Rochelle, France. pp.77-88. hal-03368466

HAL Id: hal-03368466
https://hal.science/hal-03368466

Submitted on 6 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

Isovist computation of outdoor environment with semi-dense line SLAM and monocular camera

Thomas Le Jan[1], **Myriam Servières**[1], **Thomas Leduc**[2], **Vincent Tourre**[1]

1. *AAU, Ecole Centrale de Nantes, Nantes*
   thomas.le-jan@eleves.ec-nantes.fr,prenom.nom@ec-nantes.fr
2. *AAU, CNRS, ENSA Nantes, Nantes*
   thomas.leduc@crenau.archi.fr

ABSTRACT. *Isovists are powerful tools for the morphological analysis of urban spaces, allowing us to account for the characteristics of the visible space from a given point of view. However, computing isovist field from digital model is limited by the quality of the model, and computing isovist field of the real environment with LiDaR sensors is limited by its cost and the setup of cumbersome equipment (eg. on a car). To allow the use of lightweight devices, we propose a method to compute isovists from a single monocular camera, catching a path in an urban environment. This method uses SLAM (Simultaneous Localization and Mapping) and ray casting algorithms. Our results are compared against digital model's isovists.*

RÉSUMÉ. *Les isovists sont des outils performants d'analyse morphologique des espaces urbains permettant de rendre compte des caractéristiques de l'espace visible à partir d'un point de vue donné. Cependant, le calcul des champs d'isovists à partir d'un modèle numérique est limité par la qualité de ce modèle, et le calcul des champs d'isovists d'environnement réel à l'aide de capteurs LiDaR reste limité par son coût et la mise en place d'un équipement encombrant (par exemple, sur un véhicule). Pour permettre l'utilisation d'équipement portable, nous proposons d'établir une méthode d'obtention d'isovists à partir d'une caméra monoculaire captant un parcours dans un environnement urbain. Cette méthode est basée sur les algorithmes de SLAM (Simultaneous Localization And Mapping) et de lancer de rayons. Les résultats obtenus par notre méthode sont comparés à ceux obtenus sur un ensemble d'isovists calculés sur un modèle numérique 3D simplifié de la ville.*

KEYWORDS: *Isovists, camera, space reconstruction*

MOTS-CLÉS : *Isovists, caméra, reconstruction de l'espace*

1. Introduction

Isovists and isovist fields are conceptual tools used to carry out morphological analyses of architectural and urban spaces. They were developed and theorized in the 70s (Benedikt, 1979). An isovist determines the available space in a direct line of sight from a given position in an architectural or urban environment. As the visibility of the urban space is part of the urban design qualities identified by urban planners (Ewing, Handy, 2009), isovists are particularly relevant to evaluate the quality of an urban space. Depending on the dimension of the space, these isovists have a polygonal (2D) or polyhedral (3D) shape.

They're usually computed from a digital urban model extracted from a GIS, which is a simplified representation of the urban environment. Some methods can calculate isovists along a route caught with sensors, such as LiDaR (Schmid, Stülpnagel, 2018), but the use of these powerful sensors is not accessible to as many people as possible and this may be very inconvenient because of the cumbersome equipment that requires additional transport (e.g. on a car). Although there is more consumer LiDaR equipment available, here we focus on sensors that are embedded in everyone's smartphones. It is possible to reconstruct a map of the environment with lightweight sensors, for example monocular cameras, using SLAM algorithms (Simultaneous Localization And Mapping). SLAM addresses the computational issue of constructing or updating a map of an unknown environment while simultaneously keeping track of the observer's location. Time-of-flight camera are also available on several smartphones, but their actual depth performances and sensitivity to lighting conditions, among others drawbacks, prevent us to use them to reconstruct outdoor environments.

In this article, we propose a method to delineate an isovist field from a video caught on a simple monocular camera (video-based isovists), using SLAM algorithms. To check the consistency of our results, we will confront them to the isovists generated from a digital urban model (mockup-based isovists).

2. Previous Works

SLAM algorithms provide a fairly accurate estimation of the trajectory of a camera while reconstructing the environment in which the camera evolves. Usually, SLAM algorithms return a point cloud mapping the environment, so-called *reconstructed map*. Depending on the SLAM method, the point cloud can be sparse (e.g. ORB-SLAM2 (Mur-Artal, Tardos, 2017), PTAM (Klein, Murray, 2007)), dense (e.g. DTAM (Newcombe *et al.*, 2011)) or semi-dense (e.g. (Engel *et al.*, 2014)). The sparse methods detect feature points on keyframes and they are assigned 3D coordinates in the *reconstructed map*. The dense methods rather give depth to almost every pixel of a keyframe. The map is then reconstructed by overlapping the different depth maps. Finally, semi-dense methods fall midway between sparse and dense methods. They detect

areas of interest in keyframes and assign a depth to every pixel of these areas. Unfortunately, these methods each present some weaknesses. Indeed, a sparse point cloud may be fairly accurate but, due to its lack of density, it does not permit to reconstruct real obstacles (there are not enough feature points detected). Dense and semi-dense methods can return much denser point clouds but the returned maps will be very noisy.

Some recent methods (He *et al.*, 2018) can reconstruct a map of the environment with lines instead of points, which makes it easier to reconstruct accurate surfaces using Delaunay triangulation. The method implemented in this paper uses a semi-dense version of ORB-SLAM2 (Mur-Artal, Tardos, 2015) to generate a semi-dense point cloud and extract lines from it.

SLAM methods often include loop closure which improves tracking and mapping by recognizing places already visited by the user. Then the associated map points and camera poses are adjusted according to their previous values.

There are some methods that allow isovist computation from point clouds (Díaz-Vilariño *et al.*, 2018). These methods are based on space discretization. Once space is subdivided into voxels, we determine if the voxels are occupied or empty by the number of points they contain. Then by ray-casting, we determine if the voxels are visible or obstructed which gives us our isovist. There are limits to these kinds of methods. The first is that the returned isovists won't be a vector object, as it is computed in a discretized space. The second is that the method expects a dense and accurate point cloud to clearly model the environment, which is not the case of the previous common visual SLAM methods.

The other simple way to compute isovists as vector objects is to apply ray-casting on surfaces, instead of points. That is possible with semi-dense-line SLAM (He *et al.*, 2018) and its surface reconstruction method.

3. Proposal – LIghtweight Real Isovist: LIRI

Our method's workflow is shown in Fig. 1. It is based on semi-dense-line SLAM algorithm (He *et al.*, 2018), which can reconstruct surfaces from semi-dense point cloud obtained by video processing (Fig. 2). We use a smartphone monocular camera but this method is standard enough to be applied to any type of monocular camera provided that we know its intrinsic parameters, which can be computed by camera calibration. So we have a 3D model of the environment made of triangles and an estimation of the sensor trajectory in the *reconstructed map*. The different poses returned by the algorithm are located in the coordinate system of the first detected keyframe. The global orientation of the *reconstructed map* then depends on the orientation of the camera lens during the detection of the first keyframe. In order to ease the comparison between video-based isovists and mockup-based isovists and to stabilize the

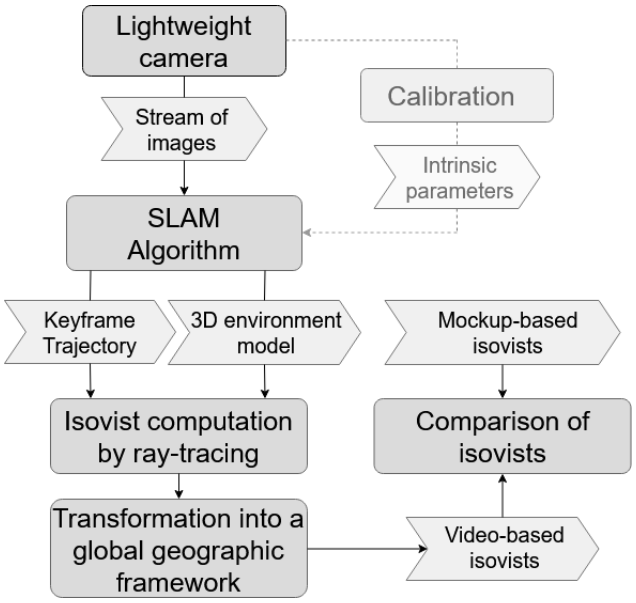

Figure 1. Workflow of our method.

skyline, we use an additional phone stabilizer, minimizing the camera orientation error. The phone stabilizer doesn't impact the standard and lightweight setup hypothesis. The resulting model has some outlier triangles that don't represent real assets so we apply a filter upon the area and the perimeter of the triangles to eliminate the outliers. This filter was set up experimentally.

We use a ray-tracing based method to compute isovists from the *reconstructed map* of the environment and estimated trajectory. We scan the environment using a panoptic approach in the horizontal plane of vision. To compute ray-tracing, we use the Möller-Trumbore intersection algorithm (Möller, Trumbore, 2005) that can quickly calculate the intersection point between a triangle and a ray in 3D space, without using plane equations. For each ray, we look at every triangle and determine the intersection point, if it exists, and we only keep the nearest intersection point from the ray origin. By connecting the intersection points obtained in the different directions we can reconstruct the isovist for a given position. Finally, if we aggregate the different isovists obtained by the positions along a trajectory, we have the isovists field related to that trajectory.

To compare our method to the topographic-based model, we have to transform our data in a geographical context. So we have to apply scaling, translation, and homothety (dilation) but also extract geographical coordinates of

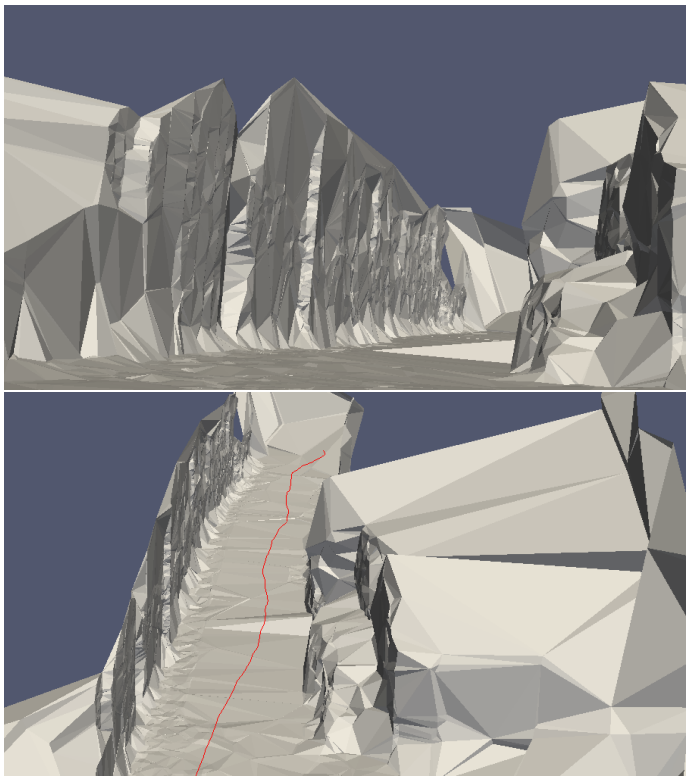

Figure 2. Surfaces reconstructed with semidense-line SLAM method. The red line represents the trajectory of the sensor.

one of the points of the trajectory. In our case we focus on the first point which corresponds to the first moment a keyframe is detected. The position of the camera associated with this first keyframe is predominant; we determine it empirically with the help of orthophotographs. We use a similar method to get the scale factor: we determine manually the geographical location of two points of the trajectory, using a GIS and orthophotographs, and we calculate the distance between them in the *reconstructed map* and in the topographic-based model. By applying this factor to our data, it is now possible to implement the estimation of the trajectory in the digital model to compute mockup-based isovists[1]. We can also compare the isovists obtained by both models.

1. To calculate the mockup-based isovists, we use standard topographic data sets such as the ones provided by the IGN BD TOPO® database (June 2020 edition) and the t4gpd Python plugin (Leduc, Leduc, 2020).

4. Results

We present here the results on one dataset collected in the city of Nantes. We suppose that the user made a single one-way journey. We also implement an arbitrary limit distance for the ray-casting, based on the size of the *reconstructed map*. The computation has been conducted on a PC with 8GB RAM, 4x2.60 GHz Intel Core i7 CPU, using Ubuntu 16.04 64 bits. With these characteristics, semi-dense-line SLAM ran for half an hour to fully map the environment from a one minute video. Then our method takes about two minutes to generate the 164 isovists related to the video.

Figure 3. Comparison of isovists field. Left: video-based isovists. Right: mockup-based isovists.

A quick comparison (Fig. 3) of the video-based and mockup-based isovists shows the similarities and the differences between the two models. Thus, the video-based isovists from the first poses of the trajectory are very circular, because of the limited field of view of the camera that doesn't allow the SLAM method to reconstruct the environment behind the observer. As seen in Fig. 3 left, as the scene behind the observer is not reconstructed, there is no mask (i.e. no building facades) to stop the rays. This lack of mask casts the rays to the limits of the artificial horizon (which is here of very limited range) which explains the circularity of the resulting shape. The video-based and mockup-based isovists attached to the following poses seem pretty similar. Finally, the last poses show again some differences. This is caused by the fact that video-based isovists take into account walls delineating private plots while mockup-based isovists only consider buildings surfaces. The fences detected in the video are not present in the 3D model derived from standard datasets.

To go beyond visual comparison of the different isovists, we used four morphological indicators:

- Perimeter : P
- Area : A
- Ellipticity factor : $\epsilon = \pi \times L^2/4A$
- Convexity defect : $e_{conv} = A/A_{hull}$

With L, the largest distance between two vertices of the polygon and A_{hull} the area of the convex hull of the polygon.

The ellipticity factor reflects the compression of a circle to form an ellipsoid and the flattening of the isovist. The convexity defect reflects the convex aspects of an isovist, especially the presence of concavity.

All these four parameters give information upon the global form of an isovist. We then compute these parameters for one in ten of the isovists.

By observing the Fig. 4, 5 and 6 we can separate the isovists into three main groups whose limits are indicated by green dashed lines in the figures. The first group may contain the isovists corresponding to the first poses of the trajectory. In this group, the video-based and mockup-based isovists show some differences. Indeed the first video-based isovists of the trajectory are almost circular, because of the lack of information of the *reconstructed map* in the first poses of the trajectory. This lack of information is due to the camera's limited field of view. This period during which the SLAM algorithm has not reconstructed the environment with enough information yet could be considered as an initialization phase.

The second group contains the next isovists. Here, the video-based and mockup-based isovists show much more similarities than in the first group. This is because there is enough information in the video to fully map the environment and the digital model is similar to what we can see in the video.

Finally, the third group contains the isovists of the last poses of the trajectory. Here the video-based and mockup-based isovists show differences. This may be explained by the fact that the video shows the reality of the environment, with additional occlusions such as walls or fences delineating private plots, while the digital model only considers buildings as polygons. The environment around the latter part of the trajectory contains a lot of these additional occlusions.

It can also be seen from Fig. 7 that the convexity defect graph of video-based isovist is less constant and overall higher than the mockup-based isovist convexity defect graph. This indicates the presence of more occlusion leading to more concavity, which corresponds to the less smooth real environment.

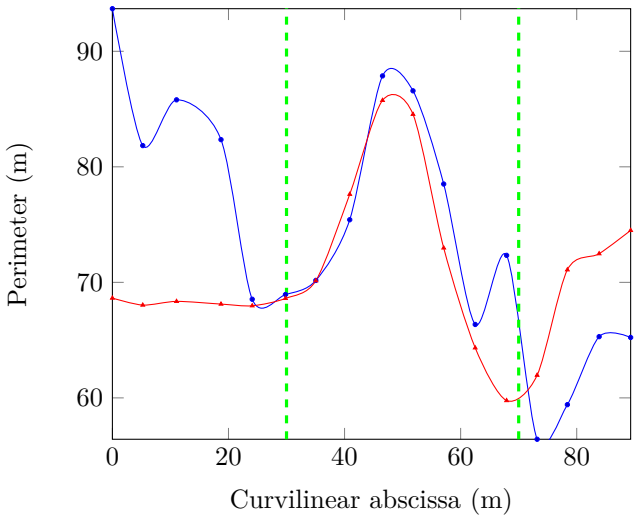

Figure 4. Perimeters (m) of video-based (blue) and mockup-based (red) isovists.

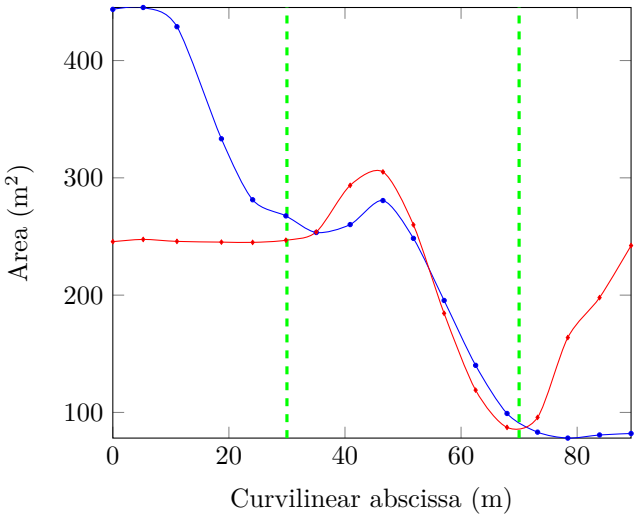

Figure 5. Areas (m^2) of video-based (blue) and mockup-based (red) isovists.

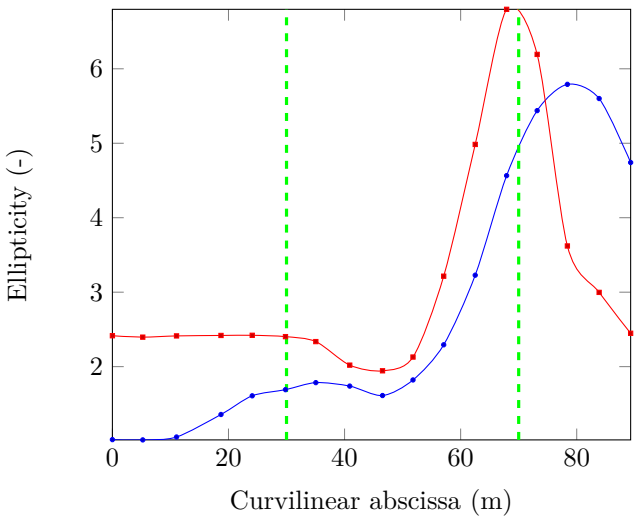

Figure 6. Ellipticity of video-based (blue) and mockup-based (red) isovists.

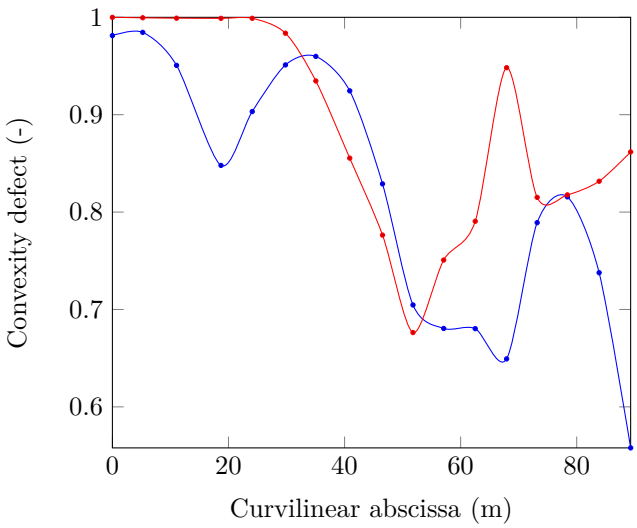

Figure 7. Convexity defect of video-based (blue) and mockup-based (red) isovists.

In this way, video-based isovists generated by our method more accurately reflects the reality of the environment, as long as there are enough data provided by the video.

5. Conclusion, Discussion and Perspectives

In this paper, we presented a process to generate isovists from a video recorded with a monocular camera, in an urban street. If the video shows enough information, the video-based isovists are pretty similar to the ones generated using standard topographic datasets, provided the viewing distance. There is even evidence to suggest that our method gives additional details upon the real environment, compared to the digital urban model, by taking into account some obstacles that are not houses or buildings.

Our method can be used to further improve the digital model by taking into account real assets obstructing the user's view represented by video-based isovists; thus tackling the well known updating problem in GIS. Moreover, the video-based reconstruction of the shape of the immediate surroundings could help several applications, eg. the display of geolocalised information in augmented reality.

Our method cannot run in real-time due to the important amount of calculation during the mapping of the environment in the semi-dense-line SLAM algorithm. This forced us to use short videos to test our method and thus to work on small environments, and this is why we limited the range of the isovists. This issue could be solved using a more powerful computer and longer videos of the environment. We could also use the loop closure of the SLAM algorithm to optimize the *reconstructed map* by recognizing places already visited by the user.

Another axis of improvement would be to automatize the acquisition of world coordinates and scale factor of the *reconstructed map* of the environment and the estimated trajectory. This may be done by getting the camera's metadata giving GPS coordinates and orientation of the sensor. This would also allow an automatic correspondence of video-based and mockup-based isovists.

Our method is not dependant on any kind of device, and any monocular video – if we know the intrinsic parameters of the camera – can be taken as an entry. Considering that a limit of the video-based isovists is the limited field of view of a simple monocular camera, we may wonder about the possibility to extend this method to omnidirectional cameras. Indeed omnidirectional cameras can catch videos with a very large field of view and adapting this method to this kind of camera only demands to adapt the mathematical model of the SLAM algorithm from pinhole model to unified model (Caruso *et al.*, 2015). Using omnidirectional cameras implies processing more pixels which

may make the calculations heavier but we would have complete video-based isovists rather than half video-based isovists for the first poses of the trajectory.

Furthermore, it would be interesting to add semantic treatment to our video in order to optimize the surface reconstruction. It's an approach which is already used by MIT-SPARK (Rosinol *et al.*, 2020) but with stereo cameras. This approach may soon be extended to monocular cameras which could also be inserted in our method.

Another very recent method (Yang *et al.*, 2020) allows user to reconstruct in real-time a dense surface mesh of the environment with a mobile device equipped with an embedded monocular camera. This method has given interesting results in indoor environments. We could imagine applying it to outdoor environments to replace the semidense-line SLAM.

As future work, we could directly compare isovist field extracted from LiDaR point cloud with mock-up based isovist. Such a comparison could validate our method, taking the point cloud as a reference.

References

Benedikt M. L. (1979). To take hold of space: isovists and isovist fields. *Environment and Planning B: Planning and Design*, Vol. 6, No. 1, pp. 47–65.

Caruso D., Engel J., Cremers D. (2015, sep). Large-scale direct SLAM for omnidirectional cameras. In *2015 IEEE/rsj international conference on intelligent robots and systems (iros)*, pp. 141–148. IEEE. Retrieved from https://ieeexplore.ieee.org/document/7353366/

Díaz-Vilariño L., González-deSantos L., Verbree E., Michailidou G., Zlatanova S. (2018, 9). From point clouds to 3D isovists in indoor environments. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XLII-4, pp. 149-154. Retrieved from https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-4/149/2018/

Engel J., Schöps T., Cremers D. (2014). LSD-SLAM: Large-Scale Direct Monocular SLAM. In *European conference on computer vision (eccv)*, pp. 834–849. Retrieved from http://link.springer.com/10.1007/978-3-319-10605-2_54

Ewing R., Handy S. (2009). Measuring the unmeasurable: Urban design qualities related to walkability. *Journal of Urban Design*, Vol. 14, pp. 65-84.

He S., Qin X., Zhang Z., Jagersand M. (2018). Incremental 3D Line Segment Extraction from Semi-dense SLAM. *Proceedings - International Conference on Pattern Recognition*, Vol. 2018-August, pp. 1658–1663.

Klein G., Murray D. (2007, nov). Parallel Tracking and Mapping for Small AR Workspaces. In *IEEE international symposium on mixed and augmented reality (ismar)*. IEEE. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4538852

Leduc T., Leduc M. (2020). Minimum-area ellipse bounding an isovist: towards a 2D GIS-based efficient implementation. *International Journal of Geographical Information Science*, Vol. 00, No. 00, pp. 1–24.

Möller T., Trumbore B. (2005). Fast, minimum storage ray/triangle intersection. *ACM SIGGRAPH 2005 Courses, SIGGRAPH 2005*, No. 1, pp. 1–7.

Mur-Artal R., Tardos J. (2015, jul). Probabilistic Semi-Dense Mapping from Highly Accurate Feature-Based Monocular SLAM. In *Robotics: Science and systems xi.* Robotics: Science and Systems Foundation. Retrieved from http://www.roboticsproceedings.org/rss11/p41.pdf

Mur-Artal R., Tardos J. D. (2017, 10). Orb-SLAM2: An open-source SLAM system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, Vol. 33, pp. 1255-1262. Retrieved from http://IEEExplore.IEEE.org/document/7946260/

Newcombe R., Lovegrove S., Davison A. (2011). DTAM: Dense tracking and mapping in real-time. In *Proc. of the intl. conf. on computer vision (iccv), barcelona, spain*, Vol. 1. Retrieved from http://www.doc.ic.ac.uk/~ajd/Publications/newcombe_etal_iccv2011.pdf

Rosinol A., Abate M., Chang Y., Carlone L. (2020, may). Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In *2020 ieee international conference on robotics and automation (icra)*, pp. 1689–1696. IEEE. Retrieved from https://ieeexplore.ieee.org/document/9196885/

Schmid K., Stülpnagel R. von. (2018). Quantifying local spatial properties through lidar-based isovists for an evaluation of opinion-based vgi in a vr setup. In P. Kiefer, H. Huang, N. V. de Weghe, M. Raubal (Eds.),, p. 173-178. ETH Zurich. Retrieved from https://www.research-collection.ethz.ch/handle/20.500.11850/225613

Yang X., Zhou L., Jiang H., Tang Z., Wang Y., Bao H. *et al.* (2020). Mobile3Drecon: Real-time monocular 3D reconstruction on a mobile phone. *IEEE Transactions on Visualization and Computer Graphics*, pp. 3446-3456.