Topological Mapping for Sentinel Mobile Robots

Miguel Anjos
Instituto Superior Técnico / UTL, Lisbon, Portugal
miguelferreiraanjos@gmail.com

I. Introduction

Mobile autonomous robots are one of the most researched topics in recent years with commercial designs already available for both industrial settings, (e.g. Waypoints's Vector), and for home settings (e.g. the vacuum Roomba from iRobot). The most common sensor deployed for accurate navigation is the LiDAR sensor [1] since it provides accurate depth information which facilitates estimating the robot's pose. The excellent quality of LiDAR sensors implies, however, very high hardware costs.

Nowadays, the availability of low cost cameras and inertial measurement units (IMU) means developing autonomous robots based on this sensor suite is very promising. If an accurate and real-time performing platform can be developed at a low cost, it might open the door for a future in which most people have mobile robots that do surveillance, carry objects from room to room autonomously and more. In a world where most people deal with robots, it seems imperative that the robots possess a human-like understanding of the world.

A. Related Work

A low-cost mobile system capable of navigation using a camera and an inertial measurement unit (IMU) has been explored previously in the literature. Fusing these sensors started with only estimating the local positioning of the platform without loop closure, i.e. visual-inertial odometry (VIO), as in the works [11] and [7] using filters.

For a more complete system with global consistency, and potentially multi-session mapping, SLAM is the next step. Filter-based solutions have been proposed in [18][12] that include loop closure and as such are capable of global consistency. Optimization-based algorithms seem to be the current trend of the art, with [21], [15], [16], [22] and [3] all developed since 2017 and with increasing performance. In particular, [3] can be said to be representative of the state of the art for visual and visual-inertial SLAM systems, capable of multi-session navigation, with a multiple map system with short, medium and long term recall capabilities.

These state-of-the-art algorithms have very high metric accuracy, however no semantic meaning of places is attempted [10]. For a robot to navigate homes using the same paradigm of humans, assignment of semantic meaning to places has to occur. The first step towards meaning assignment is the creation of topological maps, which has been explored for many years. Topomap [2] uses visual information to create topological maps and navigate them. From these topological maps, semantic meaning can be assigned to nodes of the map with research on this topic conducted recently in [1] and [4].

[1] e.g. Velodyne's LiDAR sensors are considered market leaders having been part of the Waymo/Google project autonomous car.

B. Problem Formulation and Thesis Approach

In this thesis we aim to address the first step in our view to enabling mobile robots to communicate with humans on the same abstract level when it comes to navigation, which is topological mapping.

As an example, humans navigate home environments not with reference to a frame, but by assigning meaning to locations, such as 'kitchen', and recognizing they are there or if they are in another location. For an autonomous system to be able to assign semantic meaning to locations it must first be able to differentiate them in some way.

Topological maps are a valuable tool for this since they provide discrete states that can be assigned meaning.

We propose a topological mapping framework that takes the output of a SLAM system that provides 3D structure and creates a scene graph. This scene graph is composed of nodes that map to 3D clusters of the environment. To evaluate if node recognition in this framework is successful we present several experiments where the robot navigates for a second time in a particular scene and matches the nodes being created for the current navigation to the first one.

A robot platform similar to the one developed in [14] was developed and used in these experiments. It uses a monocular camera and an inexpensive IMU.

II. Background and State of the Art

In this section we cover the mathematics behind the SLAM system chosen to use in combination with our work. We also provide background on the techniques used in the literature for visual topological mapping and a discussion on some advantages and disadvantages for each. We relate topological mapping to semantic labeling while also defining semantic labeling.

A. Unscented Kalman Filter on Lie Groups

The Brossard et al. [12] filter is a foundation of our project as the chosen SLAM system utilized. It is an Unscented Kalman filter on Lie groups that fuses camera and IMU measurements together in order to estimate the system's state.

1) Filter Implementation:

a) State Space: The state space used for this work is an extension of the $SE(3)$ Lie group. This Lie group is defined in $\chi \in SE_{2+p}(3)$ space.

It contains the position $x \in \mathbb{R}^3$, $\mathbf{R} \in SO(3)$, linear velocity $\mathbf{v} \in \mathbb{R}^3$ and the 3D positions of landmarks $\mathbf{p}_1, ..., \mathbf{p}_n \in \mathbb{R}^3$ tracked in the scene. The state is formed by a square matrix χ with dimensions $(5+p) \times (5+p)$:

$$\chi = \begin{bmatrix} \mathbf{R} & \mathbf{v} & \mathbf{x} & \mathbf{p}_1 \cdots \mathbf{p}_n \\ & \mathbf{0}_{(p+2)\times 3} & & \mathbf{I}_{(p+2)\times(p+2)} \end{bmatrix} \tag{1}$$

To have an online estimation of the IMU biases we append these to the state. Consequently, the state has concatenated the IMU biases defined as the bias vector $b \in \mathbb{R}^6$:

$$b = \begin{bmatrix} b_\omega^\top & b_a^\top \end{bmatrix}^\top \tag{2}$$

with the accelerometer bias $b_a \in \mathbb{R}^3$ and gyroscope bias $b_\omega \in \mathbb{R}^3$. Finally, the state of the filter is expressed as the tuple (χ, b).

b) Dynamic Model: The system model chosen assumes that the robot is navigating a flat earth and it is equipped with an IMU:

$$\text{body state} = \begin{cases} \dot{\mathbf{R}} = \mathbf{R}(\omega - b_\omega + n_\omega)_\times \\ \dot{\mathbf{v}} = \mathbf{R}(a - b_a + n_a) - g \\ \dot{\mathbf{x}} = \mathbf{v} \end{cases} \tag{3}$$

$$\text{IMU biases} = \begin{cases} \dot{b}_\omega = n b_\omega \\ \dot{b}_a = n b_a \end{cases} \tag{4}$$

$$\text{landmarks} = \{\dot{p}_i = 0, i = 0, ..., p \tag{5}$$

where $(\omega)_\times$ portrays the skew-symmetric matrix related with the cross product with vector $\omega \in \mathbb{R}^3$ (as shown in equation **??**). The multiple noises are grouped as:

$$n = [n_\omega{}^\top \; n_a{}^\top \; n_{b\omega}{}^\top \; n_{ba}{}^\top] \sim \mathcal{N}(0, Q) \tag{6}$$

c) Sensors Model: The measurement model takes camera frames. The camera observes p landmarks tracked in the scene. Each landmark $\mathbf{p}_i$ is observed according to the standard pinhole model and the respective projection model, as such:

$$\mathbf{y}_i = \begin{bmatrix} y^i{}_u \\ y^i{}_v \end{bmatrix} + n^i{}_\mathbf{y} \tag{7}$$

where $\mathbf{y}_i$ is the result of the projection:

$$\lambda \begin{bmatrix} y^i{}_u \\ y^i{}_v \\ 1 \end{bmatrix} = K[\mathbf{R}^\top{}_{B\to C}(\mathbf{p}_i - \mathbf{x}) - \mathbf{t}_{B\to C}] \tag{8}$$

where γ is the scale factor, K is the camera intrinsics matrix, $\mathbf{x}$ is the IMU/Body position in the world frame, $\mathbf{R}$ is the IMU/Body orientation, $\mathbf{t}_{B\to C}$ and $\mathbf{R}^\top{}_{B\to C}$ are respectively translation and rotation from the IMU to the camera in the world frame.

With this projection a landmark $\mathbf{p}_i$ is converted from the world frame to the image plane and is now compared to its corresponding 2D feature that is being tracked frame to frame.

2) Filter Summary: The filter architecture can finally be presented as

$$\text{state} \begin{cases} \chi = \mathbf{exp}(\xi)\overline{\chi} \\ b_n = \overline{b}_n + \tilde{b} \end{cases}, \begin{bmatrix} \xi \\ \tilde{b} \end{bmatrix} \sim \mathcal{N}(0, \mathbf{P}_n), \tag{9}$$

$$\text{dynamics} \left\{ \chi_n, b_n = f(\chi_{n-1}, u_n - b_{n-1}, n_n) \right. , \tag{10}$$

$$\begin{cases} \mathbf{Y}_n = [\mathbf{y}_1{}^\top \; ... \; \mathbf{y}_p{}^\top] := \mathbf{Y}(\chi_n, \omega_n) \\ \mathbf{y}_i \quad \text{according to eq.7}, \quad i = 1, ..., p \end{cases}, \tag{11}$$

with $(\overline{\chi}_n, \overline{b}_n)$ as the mean estimate of the state at time n, $\mathbf{P}_n \in \mathbb{R}^{(15+3p)\times(15+3p)}$ as the covariance matrix that represents the state's uncertainties $(\xi, \tilde{b})$, and vector $\mathbf{Y}_n$ that contains the observations of the p landmarks with associated Gaussian noise $\omega_n \sim \mathcal{N}(0, \mathbf{W})$.

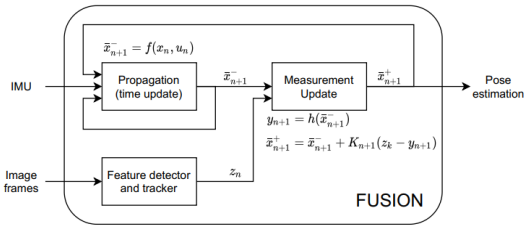


Fig. 1: Block diagram of Unscented Kalman filter on Lie groups used in our work. From [14]

B. Topological Mapping and Semantic Labeling

Topological maps are graph-like spatial representations. Nodes in such a graph often represent states in the agent's state space and edges represent system actions and trajectories that take the agent from a state to another.

In a mobile robot context, topological map's nodes most often represent a position in space with edges representing trajectories and connections between such positions.

Semantic labeling of images is the categorization of visual information according to an abstract meaning, often related to how humans perceive such information. These meanings often imply specific properties which are useful for the application at hand.

Topological mapping has been a research topic for many decades with recent advances in computing power and SLAM enabling progress [2] [8]. Whereas semantic labeling is more recent in the literature [1] [5] , with real advances being a result of the foundational work done with machine learning, more specifically, in deep learning and SLAM.

There have been recent works that combine both approaches in a complete system using sensor suites which directly observe depth [1][4] as opposed to ours which does not.

Libraries like [17] present an opportunity for accurate object-recognition which was not possible a decade ago, by making use of neural networks like this, it seems that semantic meaning can be assigned to places that a robot navigates, which in turn could enable a more seamless interaction between humans and robots.

1) Topological mapping: Topological mapping from visual information has been studied extensively in recent years [2][4][8][20].

In order to perform mapping using vision, it is necessary to describe the acquired images and be able to compare these descriptions. Consequently, the quality of the map will directly rely on the method used for visually describing the different environment locations.

The method used for representing the image can be classified into four main categories according to the authors of [8]:

- Methods based on global descriptors, with the image represented by a general descriptor computed using the entire frame as input.
- Methods based on local descriptors, where interest points are found in the image and then a patch around this point is described in order to identify them in other similar images.
- Methods based on the Bag-of-Words (BoW) algorithm [2], where local features are quantized according to a set of feature models called visual dictionary, representing images as histograms of occurrences of each word in the image;
- Methods based on combined descriptors, where several techniques described above are used together as a new solution.

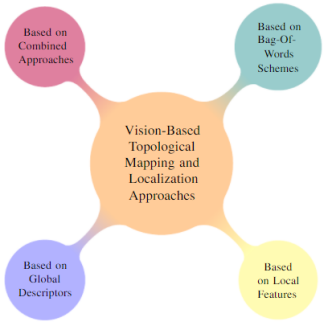


Fig. 2: Classification of vision-based topological schemes according to their image representation method. From [8]

There are advantages and disadvantages to each of these representation schemes. In table I a breakdown of these can be seen.

Factor	Gl. Descriptors	Loc. Features	BoW
CPU Load	Low	High	Medium
Storage Need	Low	High	Medium
Matching Complexity	Medium	High	Low
Discrimination Power	Low	High	Medium
Perceptual Aliasing Effect	High	Low	Medium
Large-Scale Operation	Medium	Low	High
Spatial Loss Information	Medium	Low	High
Pose recovery complexity	High	Low	Medium

TABLE I: Advantages and disadvantages for each representation scheme. From [8]

[2]https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision

These image representations serve the purpose of allowing for fast and accurate image matching which concerns topological mapping in terms of aggregating similar frames to the same place node.

However, a completely different approach which concerns itself with the geometry of the environment has also been studied, for example in [2]. From the 3D structure of the environment, occupancy information can be retrieved and it is possible to create a set of convex free-space clusters, which form the vertices of the topological map.

This approach lends itself more to sensors that provide depth information such as RGB-D cameras, stereo cameras and LiDAR. However, there have been good results in estimating the geometry of an environment through RGB only video [13] which in turn could result in this approach constructing accurate topological maps using information from a simple RGB camera alone.

III. Geometric Scene Labeling and Place Registration

Figure 3 illustrates the proposed solution of our work. Given a mobile ground robot doing landmark-based SLAM that loses metric tracking for any reason, is it possible to answer the question "What was the nearest *Place*?".

In order to accomplish this, we generate a topological framework that maps the scene with nodes representing *Places* via a previous navigation that generates a database for comparison.

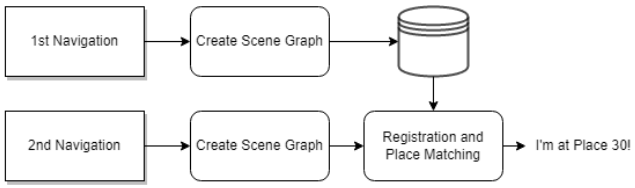


Fig. 3: Basic diagram of the proposed methodology. Given a sentinel robot navigating a previously mapped environment, can the robot recognize its current location based on the notion of *Places*.

A. Topological Mapping

Topological ideas allow for recognition of locations without a need for accurate metric information, providing a potential approach for addressing the hijacked robot problem. In our proposal, location recognition is based on the overall structure of landmarks tracked by the SLAM system. This structure can be robust to slight metric errors.

B. Place registration as a solution

As discussed, an advantage of topological mapping is that robot localization does not require highly precise metric estimation and navigation. If the mobile robot system can traverse from node to node and identify at which node it currently is, topological mapping and navigation is a success.

We propose a mapping strategy that combines information from a metric SLAM system with topological mapping such

that to recognize a *place* the robot must only navigate near it and capture it with its camera, as long as the scene has been previously mapped.

The feature type chosen to represent *places* in our topological map is the geometry relation of the 3D landmarks, that are being tracked in the SLAM system. Estimation of structure of the 3D landmarks has robustness to errors in the metric estimation and does not depend on the coordinate system used.

We have built a software system to evaluate the proposal.

1) Place definition: The methodology used will be based on *places*. During operation, a SLAM algorithm outputs the 3D landmarks that it is tracking. These 3D landmarks form a pointcloud of the environment. Clusters of landmarks shall be regarded as *places* and will be the basis for recognizing locations in our proposal. *Places* hold the geometric information of every point of the cluster. It holds the cluster's centroid information as well as the images that are associated to each 3D point, if available. The algorithm chosen for the clustering of landmarks is DBSCAN [6].

2) 3D pointcloud information: The chosen SLAM algorithm for providing 3D structure information in our work is based on the work of Brossard et al.[12]. We provide both the IMU information and the camera frames, and receive as output the 3D information of the landmarks and the trajectory taken by the robot.

In some experiments we utilize structure from motion software to create the 3D structure of the environment as if it was the output of a SLAM system. This is because initialization is important for the filter's stability and no initialization parameters exist for our own datasets.

3) Clustering 3D landmarks:

a) DBSCAN for place creation: We utilize the DBSCAN algorithm for 3D clustering. DBSCAN is an acronym that stands for Density-Based Spatial Clustering of Applications with Noise. It is an algorithm for clustering data based on density.

DBSCAN searches for clusters by examining the ϵ-neighborhood of each datapoint. If the ϵ-neighborhood of datapoint p contains more than $MinPts$, a new cluster is created using point p as the core datapoint. DBSCAN collects directly density-reachable datapoints repeatedly from these core datapoints, which may include the merging of a few density-reachable clusters.

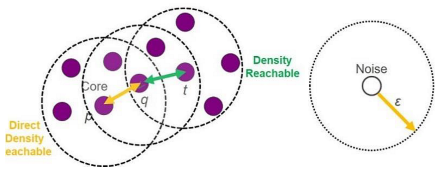


Fig. 4: DBSCAN visualization. From [9]. In our case, datapoints are 3D points that are clustered by density into *places* that represent nodes in the scene graph.

For our application, the clusters of landmarks found in the first navigation performed will represent *places* and in the second locations which are candidate *places*. The two parameters of the algorithm ϵ and *MinPts* were found experimentally.

Using these clusters of landmarks, we can construct a graph with nodes containing information about a cluster, and edges containing information about the relationship of two clusters.

For evaluating our results clusters from the first navigation, *places*, are matched to clusters from the second navigation, *locations*.

4) Graph Split into Places: Assignments of names to each cluster is done after *place* creation has happened. The methodology used for naming in our work is simple, using a sequential name assignment.

5) Registration: We consider the registration of the pointclouds obtained by a SLAM framework after each of the two navigations. We will use the resulting alignment-transformation to change the robot position from scene2's coordinate system to scene1's such that we can find the closest *place*. The chosen registration algorithm for our application is the matlab's implementation of Generalized-ICP which is based on Segal et al.'s work [19].

6) Generalized ICP: Standard ICP is a point-to-point method, which attempts to align all matched points precisely by minimizing their Euclidean distance. This does not take into account the fact that an exact matching is typically not possible due to the different sampling of the two point clouds, which leads to pairs that do not have perfect equivalence with one another.

We utilize a plane-to-plane variant by Segal et al. [19] that takes into account the surface normals of both the model pointcloud and also the data pointcloud.

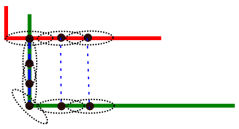


Fig. 5: Plane to plane matching. From Seagal et al. [19]

By applying a probabilistic model to the problem, it modifies the error function utilized in the Standard ICP and assigns a covariance matrix to each point.

This results in the new error function:

$$T \leftarrow \arg\min_{T} \sum_{i} {d_i^{(T)}}^{\top} (C_i^A + T C_i^B T^{\top})^{-1} d_i^{(T)} \qquad (1)$$

where $d_i^{(T)} = a_i - T b_i$, C_i^A and C_i^B are covariance matrices (assuming that all points which could not be matched were already removed from A and B and that a_i corresponds with b_i).

They key benefit of introducing information about the surfaces of both pointclouds is that if the matching of points reveals inconsistent surface matching, those correspondences will contribute less in the optimization problem. This improves the registration process and provides more robustness to incorrect correspondences in most indoor scenarios since human constructions are highly structured. This is preferred for our approach since we aim to construct topological maps of indoor home environments.

C. Software diagram and Summary of proposal

We have built a software system to evaluate the proposal. The system is implemented such that it can use any SLAM framework, provided they output 3D structure, such as landmarks being tracked.

We have implemented the use of an unscented kalman filter on Lie groups for fusion of IMU measurements and camera frames based on Barrau et al. [12].

The full software system's diagram is represented in fig. 7. The methodology it follows is:

1) Begin first navigation and get information of the scene from SLAM system and all the landmarks as a pointcloud.
2) Create clusters of landmarks using DBSCAN [6] clustering method on their 3D position.
3) Assign *places* label to each landmark having each landmark belonging to the same cluster considered the same *place*.
4) Begin second navigation and get information of the scene from SLAM system and all the landmarks as a pointcloud.
5) Match the second navigation's clusters landmark pointcloud to the prior clusters pointcloud based on plane-to-plane ICP [19].
6) Use the transformation obtained to transform robot's position, and *places* centroids from second navigation to first.
7) Compute euclidean distance to all *places* and localize the robot as being at the nearest *place*.

Using this methodology a graph scene composed of *places* as states is created and matched to a previous state.

We postulate that the structure of the clusters created is similar enough that a good matching can be created between navigations.

To evaluate each experiment success, we consider images from the matched clusters using the process depicted in figure 6. We plot the images that generated the 3D points of a *place* from Navigation1 and its correspondent from Navigation2.

We plot both a reference image for the *place* and the mean image. The representative image i_r of *place* X with images i and mean image i_m was chosen as such:

$$i_r = \min_{\forall i \in X} |i - i_m| \tag{12}$$

A complete diagram of the implemented software design can be seen in figure 7. It illustrates each of the main steps implemented characterized as a rounded block.

IV. Experiments

In this section we will present three types of experiments that were conducted to evaluate the *place* registration methodology for solving topological location recognition.

For each type of experiment, pairs of robot navigations are performed. The first represents the scene acquisition and *place* generation for topological mapping. The second represents scene navigation from which we can verify if current *place* recognition was successful.

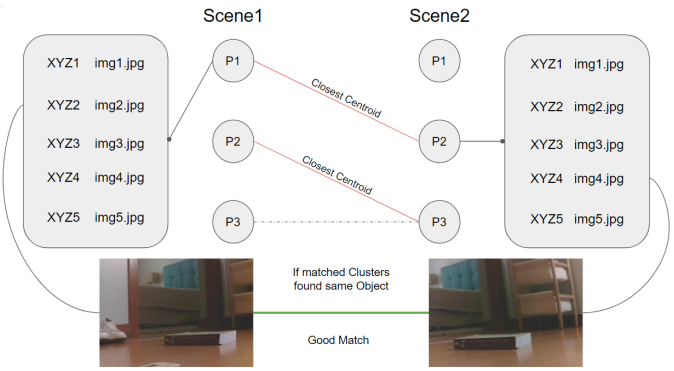


Fig. 6: To evaluate matching success we employ the strategy depicted above. Each *place* holds information about each 3D point it contains as well as the associated image from which the 3D point was first generated. We compare *place* matches by comparing a representative image of each matched cluster and the average image of each matched cluster.

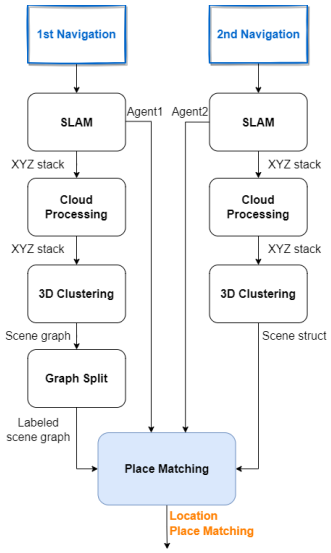


Fig. 7: Diagram of complete software implementation

We utilize the public EuRoC MAV Dataset Vicon Room 1 02 dataset for one of the experiment setups. All other datasets were acquired by us in a regular house setup.

Since achieving SLAM is not the focus of our work, we utilize a structure from motion software package, VisualSFM [3], to acquire a 3D pointcloud for home datasets. We then apply our proposed methodology on this 3D structure as if it was the output of a SLAM system.

[3] VisualSFM is a GUI application for 3D reconstruction using structure from motion (SFM).
http://ccwu.me/vsfm/index.html

A. Experimental Setups and Implementation Details

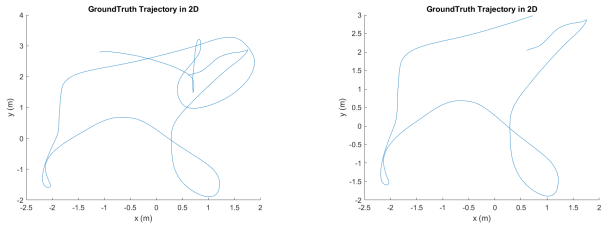
a) Setup I: For this experiment Navigation2 will be a subset of the dataset used for Navigation1. This experiment serves as a proof of concept for the methodology used. It is the only setup for which we use the public Euroc dataset.

b) Setup II: For this experiment Navigation1 and Navigation2 observes the same scene with very similar trajectories, Navigation2 being a re-acquisition of Navigation1. The second navigation starts and ends at a different, but similar, location to the first navigation.

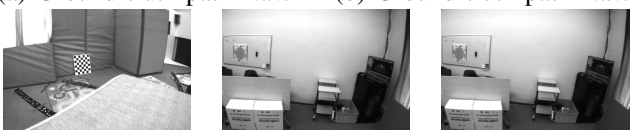
c) Setup III: For this setup, the robot takes a different trajectory observing the same scene. A estimation of the groudtruth for both navigations can be seen in figure 15. This is a more complex case than the previous experiments. With different perspectives on the scene and a different trajectory, it is more challenging to obtain a 3D structure that is comparable between navigations. We postulate that clusters will still form according to the objects seen, permitting correct matches.

B. Experiment I: Place registration results and analysis

This experiment uses Setup I. The navigations used for this are taken from the EuRoC MAV Dataset Vicon Room 1 02 dataset. We use the SLAM system from Barrau et al. [12] and utilize this public dataset since it is the only one for which the chosen SLAM system provides the initialization parameters for. We use this SLAM filter to compute both the trajectory of the robot and the 3D structure of the scene. We then use this output and perform our proposed methodology of *place* clustering and *place* matching.



(a) Ground truth path Nav.1 (b) Ground truth path Nav.2



(c) Nav.1 fr. 1 (d) Nav.1 fr. 141 (e) Nav.1 fr. 368

(f) Nav.2 fr. 1 (g) Nav.2 fr. 214 (h) Nav.2 fr. 414

Fig. 8: Representation of Navigation1 and Navigation2 for Experiment I. From left to right, first frame, early middle frame, late middle frame, last frame.

Navigation1 initiates at the same time as Navigation2. Navigation1 composes 6000 IMU iterations and 600 frames. Navigation2 composes 4000 IMU datapoints and 400 frames. A representation of Navigation1 and Navigation2 can be seen in 8.

From running the Barrau et al.[12] software we receive as output information about both the trajectory of the mobile robot and the 3D structure of the environment. The filter only holds thirty landmarks at all times, removing old ones when a new 3D landmark is required. We require more that thirty 3D points to evaluate our proposal and so we externally save all landmarks the first time they are added to the filter. This means they are never updated inside the filter after we save them which, in turn, means there is less accuracy.

Even with less 3D geometric accuracy we postulate that as long as there can be a match between scenes, clusters will form around the same locations and *places* can be extracted and matched.

For registration using plane-to-plane ICP we utilized an inlier-ratio of 0.75 for this pair of navigations.

The root average squared error of the registration was 0.0023 m. It is understandable that registration is done very successfully since we utilize a subset of the same dataset in Navigation2 compared to Navigation1 and the filter created an identical pointcloud up to the cutoff point of Navigation2.

For this experiment the amount of clusters created in Navigation1 was ten and for Navigation2 was six. Between navigations we can observe that there are some clusters that Navigation1 could produce due to more data points that Navigation2 could not.

a) Matching Results: - From the previously shown alignment of scenes we have available the rigid transformation from scene2 to scene1. By applying the registration to the clusters and matching them using smallest euclidean distance we obtain the matching shown in figure 10.

To evaluate how good the matching process was, we show in figure 13 a representative image for each *place* and the average image associated to the *place*, with the standard brightness deviation shown below.

Since images are taken in motion and a specific place is observed from several close but different perspectives it is normal for average images to present some blur. Because this dataset is from a drone that moves at a relatively high velocity, it is expected that this will happen.

In *Place1* we note that the average image has a very high standard deviation and appears very blurred. This seems to be due to the fact that the cluster associated a lot of 3D points and their related image. While a lower clustering distance might have solved this problem, some *places* could become unrecognizable by the system due to having their core points farther away.

Despite these problems, all place matches seem to have been made correctly, with representative images depicting close locations of the room.

C. Experiment II: Place registration results and analysis

The navigations used for this experiment are a pair of navigations taken at home using the approximately the trajectory depicted in red in figure 11. This experiment uses setup II, with Navigation2 having a similar trajectory to Navigation1

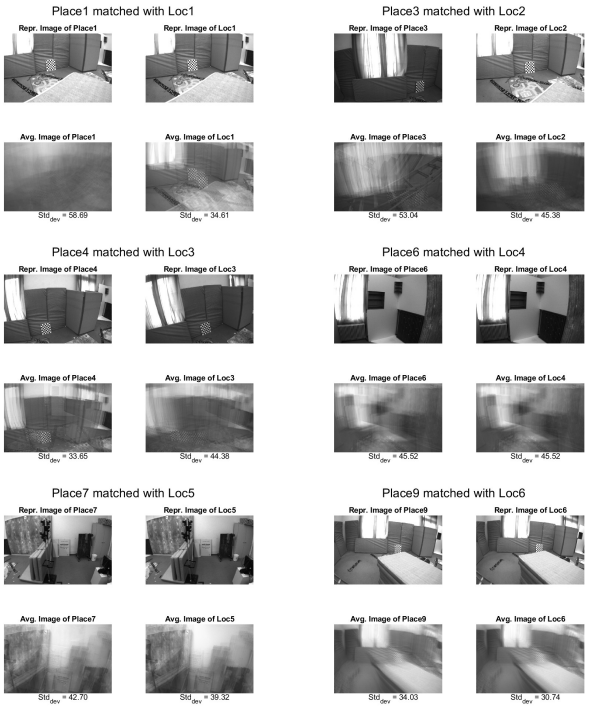


Fig. 9: Reference images for each *place* match for Experiment I. Each match presents two sets of images, images related to the *place* of Navigation1 that was matched (left) and images related to the matched cluster of Navigation2 (right). For each *place* and cluster, two images are shown. A representative image of the cluster, and the average image taken by using all images related to said cluster. The representative image is obtained by finding the closest image in the cluster set to the average image. Below each average image we show the brightness standard deviation found for the cluster.

but different starting point and ending point and thus observing the scene from a diffrent but very similar perspective. A representation of Navigation1 and a representation of Navigation2 can be seen in 11.

The SLAM framework utilised by us [12] could not produce reasonable results without the initialization parameters highly calibrated. As thus, and since SLAM is not the focus of our work, we produced the 3D structure of the scene by using a structure from motion software, VisualSFM [4]. We treat the output of this software as if it was the output of a SLAM system, and associate *places* from navigation to navigation. The localization module is, however, turned off since there is no trajectory or pose estimation output.

For registration using plane-to-plane ICP we utilized an inlier-ratio of 0.8 for this pair of navigations.

The root average squared error of the registration was 0.0083 in normalized distance units.

In figure 12 we show the pointclouds from a front view with the coloured clusters obtained from DBSCAN, note the axis' relative size is not kept.

[4]VisualSFM is a GUI application for 3D reconstruction using structure from motion (SFM).
http://ccwu.me/vsfm/index.html

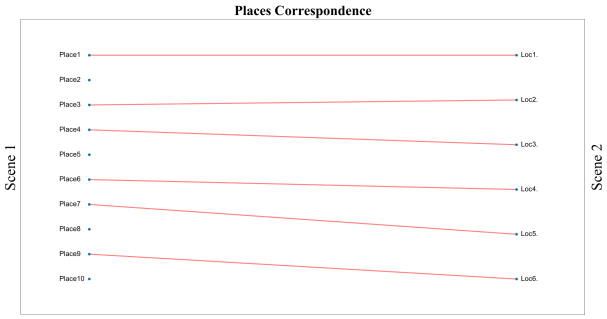


Fig. 10: Visual representation of *place* matching between navigations, according to closest *place* centroids for Experiment I. For this experiment all clusters found in Navigation2 found a *place* match in Navigation1. However, not all *places* from Navigation1 had a correspondence. This might happen if a different amount of clusters is found or if the closest *place* from Navigation1 is the same for two clusters found in Navigation2, in which case only the closest cluster of Navigation2 is matched to the *place* of Navigation1. Since Navigation1 was an extension of Navigation2 it is reasonable that there are *places* that were not sufficiently visited in Navigation2.

a) Matching Results: - From the previously computed alignment of scenes we have available the rigid transformation from scene2 to scene1. By applying the registration to the clusters and matching them using smallest euclidean distance we obtain the matching shown in figure 14.

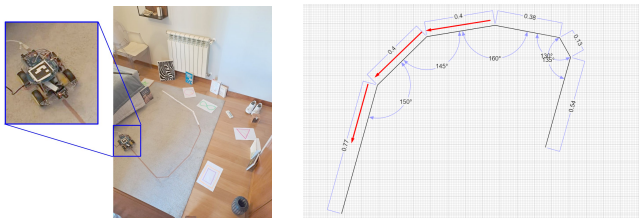
To evaluate how good the matching process was, we show in figure 13 a representative image for each *place* and the average image associated to the *place*, with the standard brightness deviation shown below.

Since images are taken in motion and a specific place is observed from several close but different perspectives it is normal for average images to present some blur. But in addition to this, the average images of each cluster seem to show that the range of images associated with clusters includes outliers. This may be due to two main reasons: there were 3D points wrongly clustered and thus their associated image should not have been included, or the 3D point itself should not have been placed there and is a bad estimate from VisualSFM.

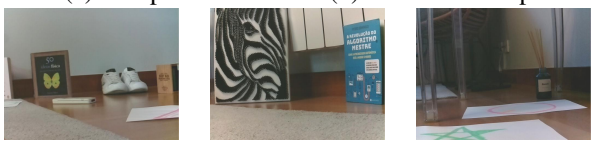
However, both the average images and the representative images of each match show that matching was successful by matching the same general location correctly from Navigation2 to Navigation1.

D. Experiment III: Place registration results and analysis

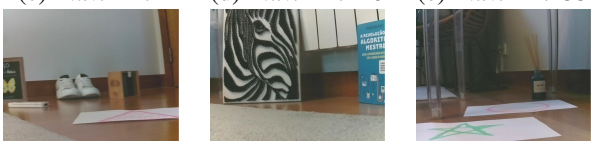
The navigations used for this experiment are a pair of navigations collected at the house of the author using approximately the trajectories depicted in red in figure 15. This experiment uses setup III, with Navigation2 having a different trajectory to Navigation1 observing the scene from a different perspective.



(a) Setup (b) Ground truth path



(c) Nav.1 fr.1 (d) Nav.1 fr.226 (e) Nav.1 fr.455



(f) Nav.2 fr.1 (g) Nav.2 fr.247 (h) Nav.2 fr.447

Fig. 11: Setup for experiment 2. The photo on the left portrays the scene the robot navigates in. The trajectory portrayed on the right is in [cm]. The red arrows indicate the sections of the trajectory taken by the robot for this navigation. We preconstructed the trajectory by using colored tape on the ground and then took measurements, using measuring tape and a protractor, for the length of each section and the angle between them respectively.

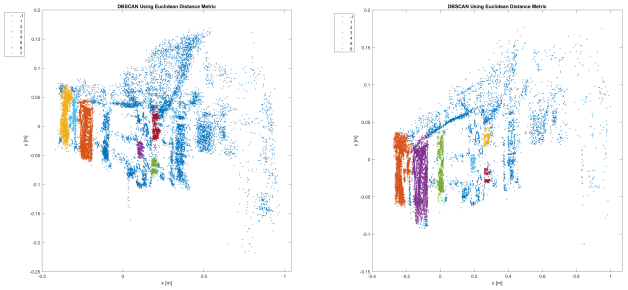


Fig. 12: Front view of pointclouds from Navigation1 (left) and Navigation2 (right) after clustering. Colours represent each cluster with dark blue representing outliers. For Navigation1 seven clusters were identified, and six clusters were identified for Navigation2. For both navigations the clustering euclidean distance threshold for DBSCAN was 0.015 and the minimum points per cluster was 200.

As in Experiment II, we utilise VisualSFM [5] to provide us with the 3D structure of the scene for this experiment. We again treat the output of this software as if it was the output of a SLAM system, and associate *places* from navigation to navigation. The localization module is, once again, turned off since there is no trajectory or pose estimation output.

For registration using plane-to-plane ICP we utilized an

[5]VisualSFM is a GUI application for 3D reconstruction using structure from motion (SFM).
http://ccwu.me/vsfm/index.html

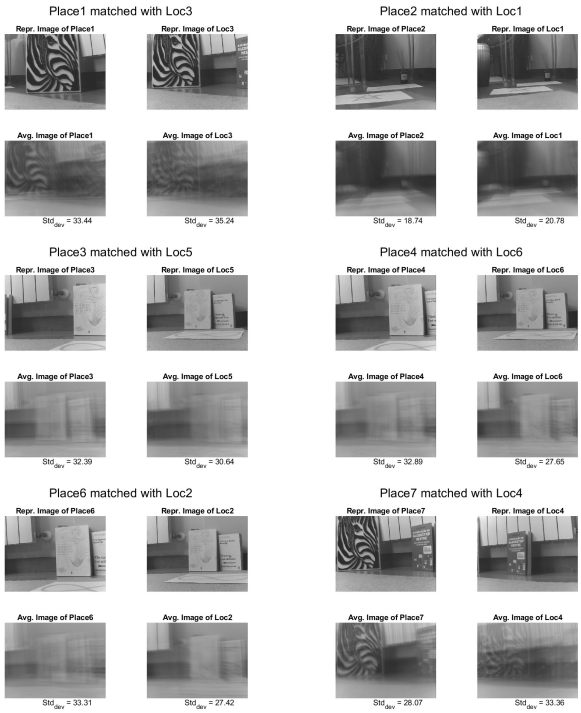


Fig. 13: Reference images for each *place* match for Experiment II. Each match presents two sets of images, images related to the *place* of Navigation1 that was matched (left) and images related to the matched cluster of Navigation2 (right). For each *place* and cluster, two images are shown. A representative image of the cluster, and the average image taken by using all images related to said cluster. The representative image is obtained by finding the closest image in the cluster set to the average image. Below each average image we show the brightness standard deviation found for the cluster.

inlier-ratio of 0.45 for this pair of navigations. The root average squared error of the registration was 0.0057 in normalized distance units.

In figure 16 we show the pointclouds from a front view with the coloured clusters obtained from DBSCAN, note the axis' relative size is not kept.

Since the scene was observed from a different perspective, we can notice a change in the structure of the resulting pointclouds and therefore, the most dense regions. Regions that were very well represented in Navigation1, such as the zebra on the left side of the scene, were not as well represented in Navigation2. In both navigations the right side of the scene has lower density of points.

However, as we will present in the matching section, for this experiment the results were still comparable to previous experiments.

a) Matching Results: -

From the previously computed alignment of scenes we have available the rigid transformation from scene2 to scene1. By applying the registration to the clusters and matching them using smallest euclidean distance we obtain the matching shown in figure 18.

To evaluate how good the matching process was, we show in

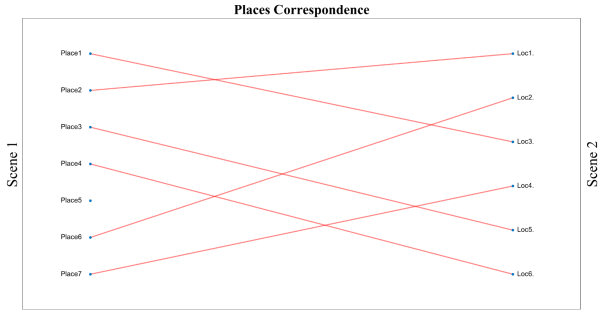
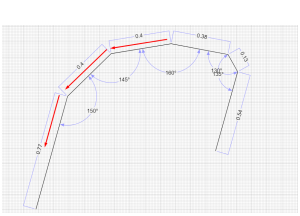
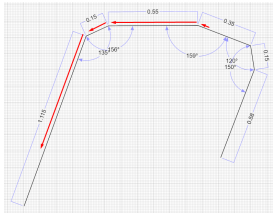


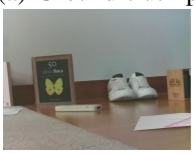
Fig. 14: Visual representation of *place* matching between navigations, according to closest *place* centroids for Experiment II. For this experiment all clusters found in Navigation2 found a *place* match in Navigation1. However, not all *places* from Navigation1 had a correspondence. This might happen if a different amount of clusters is found or if the closest *place* from Navigation1 is the same for two clusters found in Navigation2, in which case only the closest cluster of Navigation2 is matched to the *place* of Navigation1.



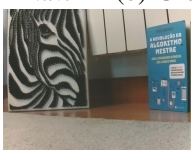
(a) Ground truth path Nav.1



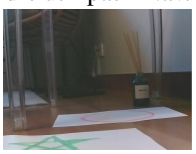
(b) Ground truth path Nav.2



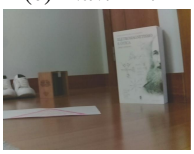
(c) Nav.1 fr.1



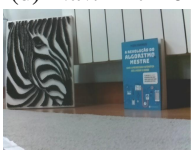
(d) Nav.1 fr.226



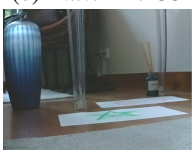
(e) Nav.1 fr.453



(f) Nav.2 fr.1



(g) Nav.2 fr.286



(h) Nav.2 fr.466

Fig. 15: Setup for experiment 3. The trajectories portrayed are in [cm]. The red arrows indicate the sections of the trajectory taken by the robot for each navigation. We preconstructed the trajectories by using colored tape on the ground and then took measurements, using measuring tape and a protractor, for the length of each section and the angle between them respectively.

figure 17 a representative image for each *place* and the average image associated to the *place*, with the standard brightness deviation shown below.

Results for this experiment seem to match the previous results. Blurring of average cluster images can still be observed. As mentioned before, this is likely due to two reasons. The use of images during motion, which displace the view slightly.

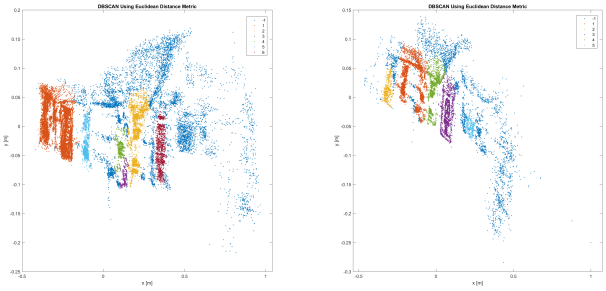


Fig. 16: Front view of pointclouds from Navigation1 (left) and Navigation2 (right) after clustering. Colours represent each cluster with dark blue representing outliers. For Navigation1 six clusters were identified, and five clusters were identified for Navigation2. For both navigations the clustering euclidean distance threshold for DBSCAN was 0.025 and the minimum points per cluster was 250.

And the inclusion of 3D points in the cluster that are farther off from the average image. This can be tuned by considering lower values for clustering distance but might lead to inability to model some *places*.

We can still observe some overlap of *places* happening, however, both the average images and the representative images of each match show that matching was successful by matching the same general location correctly from Navigation2 to Navigation1.

V. Conclusion and Future Work

This work aimed to propose a topological mapping strategy using a low-cost mobile robot that could enable the robot to recognize previously mapped areas. Our approach relies on an estimation of the 3D structure of the environment and the notion that clusters in this structure can be considered sign posts for navigation.

For evaluating our proposal we utilized and in-house built mobile robot equipped with a monocular camera sensor and an low-cost IMU. We have also utilized a public dataset in one of our experiments.

Results seem to suggest the methodology proposed can provide topological localization for the scenarios reviewed.

However, in more challenging scenarios there might be a need to either add another type of sensor that directly observes distances, such as a depth camera.

Real world locations appeared repeated as different nodes of the map even if they were correctly matched in the second navigation. Evaluation of the clustering via images revealed that there were outlier locations included in the cluster.

Both of the mentioned issues are influenced by the clustering distance and minimum number of points per cluster parameters. In the future, pursuing a mathematical way to model how to choose these parameters automatically given a certain 3D pointcloud seems promising.

Another possible avenue for future research is the labelling step of the methodology. Object-detection can be used to give semantic meaning to the found clusters, enabling the robot to have a more human-like understanding of the environment.

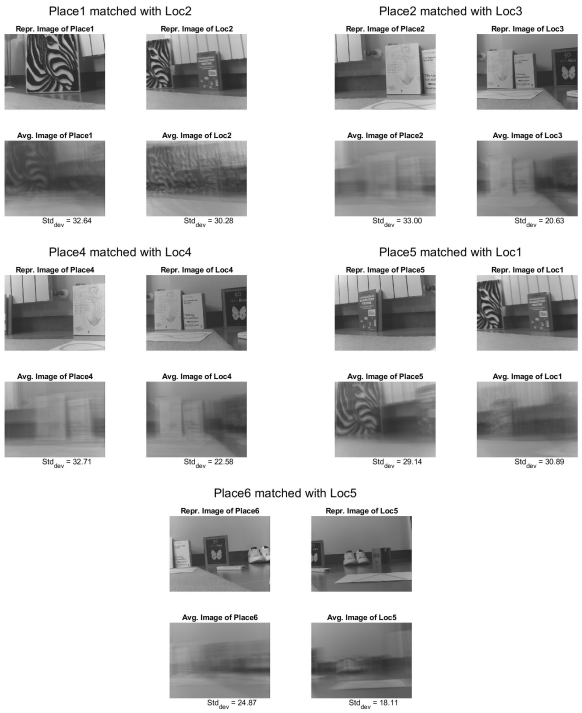


Fig. 17: Reference images for each *place* match for Experiment III. Each match presents two sets of images, images related to the *place* of Navigation1 that was matched (left) and images related to the matched cluster of Navigation2 (right). For each *place* and cluster, two images are shown. A representative image of the cluster, and the average image taken by using all images related to said cluster. The representative image is obtained by finding the closest image in the cluster set to the average image. Below each average image we show the brightness standard deviation found for the cluster.

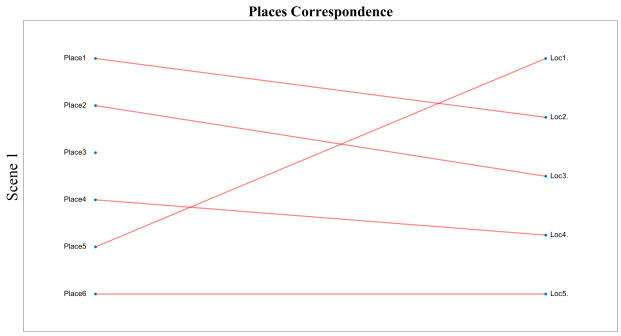


Fig. 18: Visual representation of *place* matching between navigations, according to closest *place* centroids for Experiment III. For this experiment all clusters found in Navigation2 found a *place* match in Navigation1. However, not all *places* from Navigation1 had a correspondence. This might happen if a different amount of clusters is found or if the closest *place* from Navigation1 is the same for two clusters found in Navigation2, in which case only the closest cluster of Navigation2 is matched to the *place* of Navigation1.

REFERENCES

[1] I. Armeni, Z.Y. He, A. Zamir, J. Gwak, J. Malik, M. Fischer, and S. Savarese. 3D scene graph: A structure for unified semantics, 3D space, and camera. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5663–5672, 2019.

[2] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart. Topomap: Topological mapping and navigation based on visual SLAM maps. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3818–3825, 2018.

[3] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[4] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. M. Montiel. Towards semantic SLAM using a monocular camera. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1277–1284, 2011.

[6] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[7] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.

[8] Emilio Garcia-Fidalgo and Alberto Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, 64:1–20, 2015.

[9] E. Giacoumidis, Y. Lin, M. Jarajreh, S. O'Duill, K. McGuinness, P. F. Whelan, and L. P. Barry. A blind nonlinearity compensator using DBSCAN clustering for coherent optical transmission systems. *Applied Sciences*, 9(20), 2019.

[10] Guoquan Huang. Visual-inertial navigation: A concise review. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9572–9582, 2019.

[11] T. Lupton and S. Sukkarieh. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, 28(1):61–76, 2012.

[12] S. Bonnabel M. Brossard and A. Barrau. Unscented Kalman filtering on Lie groups for fusion of IMU and monocular vision. *International Conference on Robotics and Automation (ICRA)*, 2018.

[13] Z. Murez, T. As, J. Bartolozzi, A. Sinha, V. Badrinarayanan, and A. Rabinovich. Atlas: End-to-end 3D scene reconstruction from posed images. In *ECCV*, 2020.

[14] P. Nogueira. Visual inertial odometry for mobile home robots, Master's Dissertation. In *Instituto Superior Técnico / UTL, Lisbon, Portugal*, 2021.

[15] T. Qin, P. Li, and S. Shen. VINS-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[16] T. Qin, J. Pan, S. Cao, and S. Shen. A general optimization-based framework for local odometry estimation with multiple sensors, 2019.

[17] J. Redmon and Rand Farhadi A. Divvala, Sand Girshick. You Only Look Once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[18] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart. Maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, 2018.

[19] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[20] G. J. Stein, C. Bradley, V. Preston, and N. Roy. Enabling topological planning with monocular vision. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1667–1673, 2020.

[21] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis. VINS on wheels. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5155–5162, 2017.

[22] Z. Yu, L. Zhu, and G. Lu. VINS-motion: Tightly-coupled fusion of VINS and motion constraint. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7672–7678, 2021.