ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

V4RL VISION FOR ROBOTICS LAB
Vision for Robotics Lab
Prof. Margarita Chli

Semester Thesis

Semantic Segmentation-Aided Bundle Adjustment

Autumn Term 2023

Supervised by:
Lucas Teixeira
Ruben Mascaro

Author:
Alain Schöbi

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Semantic Segmentation-Aided Bundle Adjustment

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

Alain	Schöbi

Student supervisor(s)

Lucas	Teixeira
Ruben	Mascaro

Supervising lecturer

Margarita	Chli

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/plagiarism-citationetiquette.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Munich, 2024-02-07
Place and date


Signature

[1] Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Contents

Chapter 1

Motivation

The use of robotics in agriculture and in nature in general has attracted considerable attention in recent years. Diverse methods have manifested their interested with various applications and benefits. In particular, employing Unmanned Aerial Vehicles (UAVs), commonly referred to as drones, to monitor trees or crops, is slowly revolutionizing traditional farming into what is now called *smart farming*. Methods have also been devised to build 3D digital twins of vegetation enabling a deeper understanding of nature. At the heart of all these applications is computer vision, which enables to capture a 3D environment by the use cameras.
While numerous studies have tackled the problem of capturing a 3D environment using cameras, there has been limited research into adapting and specializing these methods for natural environments.
Techniques such as Structure from Motion (SfM) exhibit notable limitations when applied to natural environments. Indeed, natural environments pose significant challenges for the following, non-exhaustive reasons. First, even a slight breeze can make leaves and branches move introducing dynamic objects into the scene. Second, the similar visual appearance of leaves or trees results in images with highly repetitive textures. Third, sunlight and reflections on leaves create areas with high contrast. Lastly, due the inherent structure of vegetation, occlusions between leaves, branches, and tree trunks are inevitable. All these factors make it challenging for conventional SfM methods to match features across multiple images.

Figure 1.1: **Left:** Smart farming. **Right:** 3D digital twin of a tree.

Chapter 2

Goal

Our work aims at extending and improving existing SfM methods to apply them to natural environments. To achieves this, we leverage semantic information and investigate how to use it.
Semantic information refers to the meaningful interpretation of an image. In particular, the semantic segmentation of an image implies segmenting it into different parts which all have a similar meaning. For instance, for an image containing a tree, we can identify parts belonging to the tree, to the ground or to the sky, thereby segmenting the image into three semantic classes, as illustrated in Figure 2.1.

Figure 2.1: **Left:** An example image of a tree. **Right:** The semantic segmentation.

The motivation to employ semantically segmented images alongside the usual color images is to address or mitigate some of the previously mentioned challenges in natural environments. For instance, in scenarios where images lack texture or contain repetitive texture, traditional SfM methods relying solely on color images would struggle a lot. In contrast, the semantic information would remain unaffected, thereby offering valuable information.
Our work specifically focuses on the problem of camera pose estimation. Camera pose estimation is the problem of estimating the position and orientation where each image has been taken in the environment. More formally, the objective is to obtain the 6 degrees-of-freedom camera pose for each image.
In summary, our work investigates how to leverage semantic information to improve camera pose estimation in natural environments, as depicted in Figure 2.2.

Figure 2.2: Using color and semantic images for camera pose estimation.

Chapter 3

Related Work

3.1 Semantic 3D Reconstruction

Several works have tackled the problem of semantic 3D reconstruction. Semantic 3D reconstruction refers to the process of reconstructing the three dimensional geometry of a scene while also assigning semantic labels to the reconstructed geometry. In these cases, the camera poses are most often assumed to be known perfectly or accurately enough to avoid further refinement. Thus, the problem of semantic 3D reconstruction is fundamentally different from the problem of camera pose estimation. Nevertheless, these works still offer valuable information on how to deal with semantics, which is why they are relevant for our work.
Häne et al. [1, 2] propose to jointly solve the problem of semantic segmentation and 3D reconstruction using an energy minimization formulation. To achieve this, they assume that the camera poses and intrinsics are known, and for each view, they use a color image, a semantic map and depth map. They represent the 3D scene by a volumetric voxel grid and employ an energy comprising a unary and pairwise potential. The unary potential penalizes voxels inconsistent with the semantic maps and depth maps, while the pairwise potential penalizes unusual transitions between semantic classes in the voxel grid. This approach enables them to accurately infer weakly observed surfaces of the scene.
An extension of this work by Bláha, Vogel et al. [3] introduces an adaptive multi-scale volumetric voxel grid enabling the method to be used for large-scale 3D semantic reconstruction.
To address the limitations of manually handcrafted semantic class transition priors, Cherabier, Schönberger et al. [4] leverage deep learning. In their work, they use an end-to-end trainable network to learn the shape priors for each semantic class.

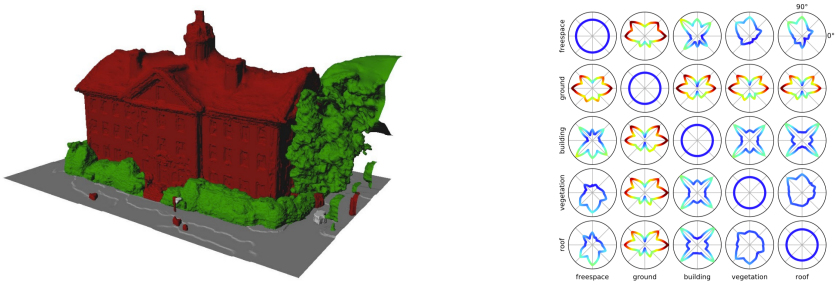

Figure 3.1: **Left:** Example of semantic 3D reconstruction using [1]. **Right:** Learned shape priors between various semantic classes using [4].

Another method proposed by Wei, Wang et al. [5] focuses solely on semantic fusion without using semantic priors. That is, after building a 3D point cloud, they assign semantic class probabilities to each point by averaging the respective probabilities from each view containing that point. This process is illustrated in Figure 3.2. A local refinement and a global refinement are then performed to improve the semantic 3D reconstruction.

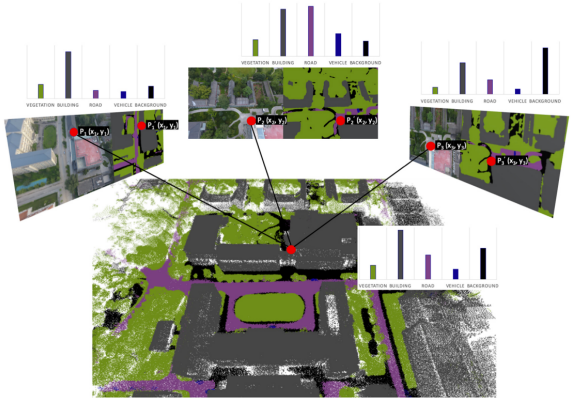

Figure 3.2: Semantic fusion illustration of [5].

Finally, Rosinol et al. [6] propose an open-source C++ implementation for semantic Simultaneous Localization and Mapping (SLAM). In addition to other features, their implementation builds a semantically annotated 3D mesh. Similarly to [5], they average the semantic class probabilities before extracting the most likely class.

3.2 Natural Environments

A review conducted by Iglhaut et al. [7] analyzed the most used and promising techniques of remote sensing methods for forests. First, one needs to distinguish the aerial approach from the terrestrial approach. In the aerial approach, the forest is sensed from above the trees, enabling to estimate the tree or canopy heights easily. In the latter approach, the forest is sensed closer to the ground making it practical to estimate individual trees or trunks.
The review also highlights the advantages of photogrammetry methods such as SfM compared to Terrestrial Laser Scanning (TLS) or Airbone Laser Scanning (ALS). While the results of photogrammetry are on a similar scale to these other methods, they exhibit much lower costs and require less planning. Indeed, photogrammetry surveys can be conducted manually with hand-held cameras or automated using UAVs.
A previous study by Surový et al. [8] also analyzed magnetic motion tracker methods in addition to laser scanning and photogrammetry methods. Despite offering high accuracy and the ability to detect concealed objects, magnetic motion trackers are too expensive for practical use in large environments.

3.3 Geometric-Aided Localization

In this final section of related work, we briefly discuss two geometry-aided localization works that significantly inspired our methods.
The first work by Lee et al. [9] focuses on localization on highways using poles. More precisely, they estimate the pose along the highway by leveraging nearby

poles. Highways are particularly challenging environments for photogrammetry techniques, with similar challenges to those in natural environments. Leveraging poles is therefore an effective approach to overcome most of these challenges.
The approach uses a prior map containing poles on highways, employed for localization. The estimated poles are detected from LiDAR data. They formulate the localization problem as an optimization problem, in which the error between the estimated poles and the corresponding ones in the prior map is minimized. This work introduces a mathematical parametrization of poles, as depicted in Figure 3.3. While our work follows a similar approach in Section 4.2, we operate in a different environment and use photogrammetry instead of LiDAR. Additionally, our objectives differ. We focus on improving camera pose estimation, whereas this work primarily addresses localization.
Our method also leverages pole-like objects, specifically using cylinders to represent tree trunks. Similar to poles on highways, tree trunks are static objects that provide valuable information.
Finally, another work by Chen et al. [10] adopts a similar approach to [9] but also leverages curbs in addition to poles. Curbs are elevated edges along roads or sidewalks that can serve as additional geometric constraints for localization. Furthermore, this work uses a Branch-and-Bound global optimization method to avoid landing in local minima.

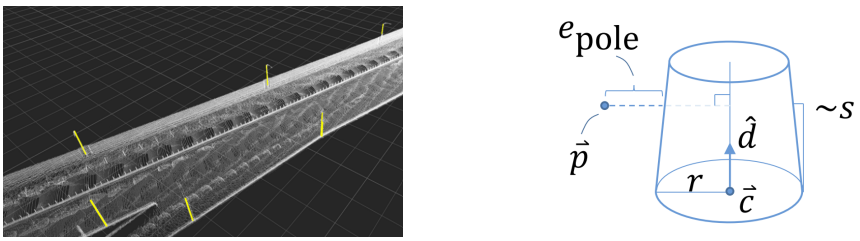

Figure 3.3: **Left:** Prior map containing poles. **Right:** Pole parametrization.

Chapter 4

Methods

With the aim of improving camera pose estimation for natural environments we propose the following method. We begin with a set of n images $I_1, \dots I_n$ taken around a single tree. First, we estimate the camera poses $p_1, \dots, p_n$ by using an existing method. Several SfM methods, such as COLMAP [11, 12], MAPLAB [13, 14] or ORB-SLAM3 [15] can be used for this initial camera pose estimation. Second, we refine these camera poses by employing the semantic maps of each image $S_1, \dots, S_n$. In summary, we begin with a rough estimation of the camera poses and then try to refine them by leveraging the semantic maps. This process is depicted in Figure 4.1.

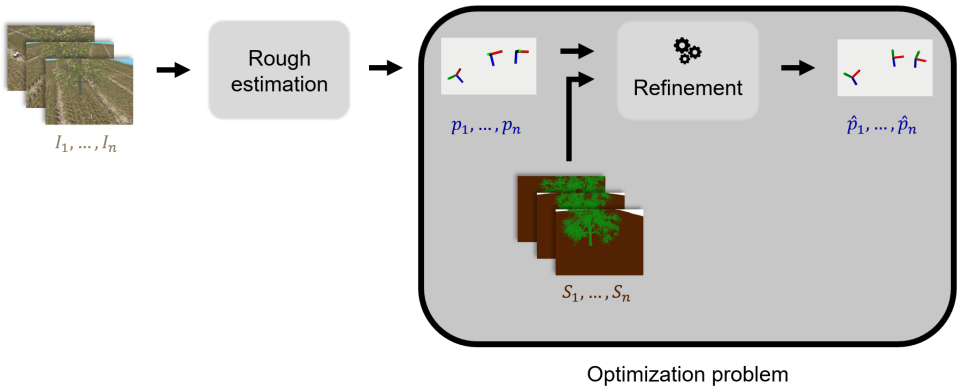

Figure 4.1: Illustration of the proposed method.

Inspired by prior work, we formulate the refinement step as an optimization problem where the camera poses act as optimization variables. Such an optimization problem, where the camera poses are being refined, is often referred to as a Bundle Adjustment (BA).

Alongside with the optimization variables, we also need to employ an error function, which will be minimized during the optimization. In line with our motivation, this error function is based on semantics and evaluates the consistency of the semantic classes across every image. That is, if the semantics are consistent between the images, the error should be low; conversely, if the semantics are inconsistent the error should be large.

In this work, we propose two different approaches that employ distinct error functions. These two approaches, namely a pixelwise and a geometric approach, are described in section 4.1 and 4.2, respectively.

4.1 Pixelwise Approach

In this first approach, we evaluate the semantic consistency on a per-pixel basis. Figure 4.2 illustrates the process for a single pixel in yellow. Note that for this approach we additionally assume that the depth maps $D_1, \ldots, D_n$ for each view are available.

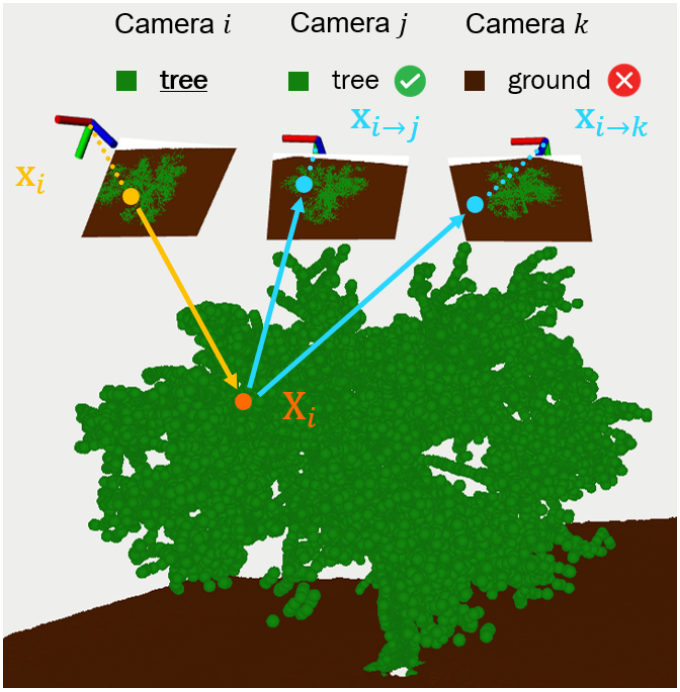

Figure 4.2: Semantic error computation example for one single pixel.

In short, a pixel in a image is considered. It is then back projected to 3D using its depth, and reprojected to the other cameras. Next, the semantic classes at the reprojected locations and the initial pixel are compared using a zero-one loss. This procedure is repeated for every pixel in every image and the errors are summed together.
Here is a more detailed description for one example pixel $\mathbf{x}_i = (x_i, y_i)$ in image i:

1. The pixel $\mathbf{x}_i$ is first back-projected to 3D by using the provided depth $D_i(\mathbf{x}_i)$. This results in the 3D point $\mathbf{X}_i = (X_i, Y_i, Z_i)$.
2. The 3D point $\mathbf{X}_i$ is then reprojected to the other cameras $j = 1, \ldots, n$ with $j \neq i$. This results in $n - 1$ pixel coordinates $\mathbf{x}_{i\to j}$.
3. The semantic classes at the reprojected points in all the images are accessed, that is $S_j(\mathbf{x}_{i\to j})$.
4. The original semantic class $S_i(\mathbf{x}_i)$ is compared with all other classes $S_j(\mathbf{x}_{i\to j})$ for $j = 1, \ldots, n$ with $j \neq i$.
5. A zero-one error function is employed

$$\sum_{\substack{j=1 \\ j\neq i}}^{n} \mathbb{1}[S_i(\mathbf{x}_i) \neq S_j(\mathbf{x}_{i\to j})]$$

where $\mathbb{1}[\cdot]$ represents the indicator function.

This error computation is then repeated for every pixel $\mathbf{x}_i$ in the image i with a predefined pixel step size, allowing a more efficient yet less precise semantic evaluation. Finally, this error computation is also repeated for every image $i = 1, \ldots, n$. In summary, the pixelwise semantic error for n images can be expressed as:

$$\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{\substack{\mathbf{x}_i \in \text{image}_i \\ \text{with step size}}} \mathbb{1}[S_i(\mathbf{x}_i) \neq S_j(\mathbf{x}_{i \to j})]$$

4.1.1 Important Considerations

During the process of backprojection and reprojection, we need to address three important issues carefully.
First, if the provided depth value $D_i(\mathbf{x}_i)$ is not valid, we simply skip pixel $\mathbf{x}_i$ in the semantic error computation. Second, if the reprojected pixel $\mathbf{x}_{i \to j}$ falls outside of image j after the reprojection, we set the semantic error to zero. Third, we must handle occlusions properly. For instance, consider an object being occluded by another object, which have two different semantic classes. Then, even with perfect camera poses, a semantic mismatch will occur. This situation is depicted in Figure 4.3, where a blue box occludes the ground when observed from camera j.
To handle this last issue, we introduce a depth error threshold, denoted as d_{error}. When reprojecting a 3D point $\mathbf{X}_i$ to image j, we compare the measured depth between $\mathbf{X}_i$ and camera j with the depth retrieved from the depth map at the reprojected pixel location $D_j(\mathbf{x}_{i \to j})$. If the difference in depth exceeds the depth error threshold d_{error}, the semantic error is set to zero.

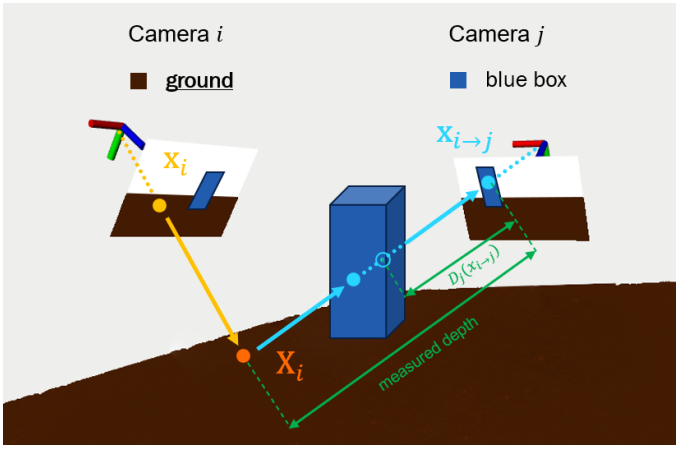

Figure 4.3: Measured depth and depth retrieved from the depth map D_j.

4.2 Geometric Approach

In this second approach, we focus on the scenario of a single tree and images captured around it. We also assume that the semantic maps contain three distinct classes: trunk, canopy, and ground. An illustration of these semantic classes is presented in Figure 4.4, with the trunk class in green, the canopy in blue and the ground in brown.
Inspired by [9, 10], we parametrize the scene using geometric objects. Specifically, we employ a cylinder to model the tree trunk. Note that this cylinder does not physically exist in the scene, it is only a man-made representation of tree trunk.

Figure 4.4: **Left:** Semantic classes. **Right:** Scene parametrization with a cylinder.

4.2.1 Error Function

The proposed error function in this approach uses the cylinder to evaluate the semantic consistency. To achieve this, the cylinder is first reprojected onto every camera view. For each view, the semantic consistency is then assessed by comparing the reprojected cylinder with the tree trunk in the semantic image. To achieve this, the pixels of the semantic image are classified into four categories depending whether they belong to the tree trunk or not, and to the reprojected cylinder or not. An example of this procedure is depicted in Figure 4.5. Finally, an Intersection over Union (IoU) metric, based on these four categories, is used to directly assess the alignment of the cylinder with the tree trunk. In Figure 4.5, the IoU corresponds to the area in green divided by the sum of the areas in green, orange and red.

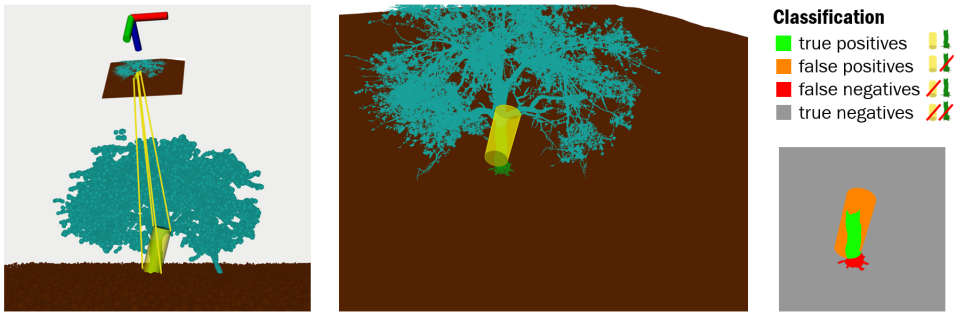

Figure 4.5: **Left:** Cylinder reprojection onto a camera. **Middle:** Visualization of the reprojected cylinder in the semantic image. **Right:** IoU computation.

As a higher IoU indicates better alignment, the goal is to maximize the IoU for each view. To capture this idea, we use the following error function described in equation

4.1. Note the square term, which penalizes strong misalignments even more.

$$\sum_{i=1}^{n}(1 - \text{IoU}_i)^2 \tag{4.1}$$

4.2.2 Optimization Problem

In the geometric approach, we refine both the camera poses and the cylinder. An initial estimation of the cylinder can be obtained manually. For instance, one can triangulate 3D points belonging to the trunk and then estimate a cylinder that contains them. The error function used in the optimization problem is described above in equation 4.1.

4.2.3 Cylinder Parametrization

To optimize over a cylinder, it needs to be represented by some mathematical variables. In line with this, we propose two simple parametrizations, illustrated in Figure 4.6. The first parametrization uses a 3D point representing the bottom center of the cylinder, an orientation described by a unit quaternion, and two parameters for the height and the radius. The second parametrization uses two 3D points representing the bottom and top centers of the cylinder, along with a radius parameter.

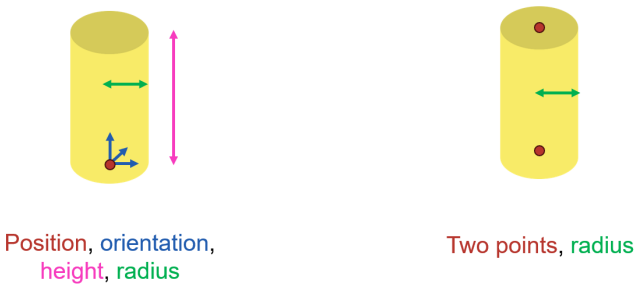

Figure 4.6: The proposed parametrizations of a cylinder.

4.2.4 Cylinder Projection

The projection of a cylinder onto a camera can be decomposed into two distinct parts. Essentially, the projection of a cylinder consists of two ellipses and a parallelogram.
The parallelogram is defined by the projection of the four 3D points situated at the corners of the cylinder. These corners represent the points located at the boundaries of the cylinder when observed from a given camera, as illustrated in red in Figure 4.7. To compute these corner points, a geometric approach is employed, as depicted in Figure 4.7. Essentially, the idea consists in casting two rays from the camera that are tangent to the cylinder.
The second part of the cylinder projection involves computing the projection of the two circles. The projection of a circle onto a camera results in an ellipse. By leveraging homogeneous coordinates to represent ellipses and using a geometric analysis, we can derive a closed-form solution for the projection of a circle. The following method is illustrated in Figure 4.8.
Let R_π, t_π represent the transformation from the coordinate frame π containing the circle to the camera frame. Let $\mathbf{x}_\pi^T C_\pi \mathbf{x}_\pi = 0$ implicitly represent the circle in the

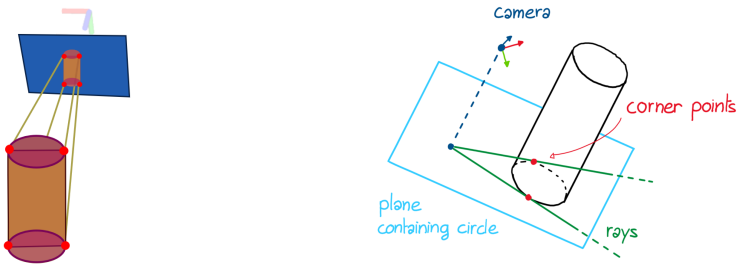

Figure 4.7: **Left:** Cylinder projection. **Right:** Corner points computation.

frame π, with $\mathbf{x}_\pi = (x_\pi, y_\pi, 1)^T$, and let $\mathbf{x}^T C \mathbf{x} = 0$ represent the resulting ellipse in image coordinates with $\mathbf{x} = (x, y, 1)^T$. The matrices C_π and C are symmetric 3×3 matrices. Then, the following relation holds:

$$C = H^{-T} C_\pi H^{-1}, \quad \text{with } H = K \cdot \begin{bmatrix} | & | & | \\ R_{\pi,x} & R_{\pi,y} & t_\pi \\ | & | & | \end{bmatrix},$$

where $R_{\pi,x}$ and $R_{\pi,y}$ respectively denote the first and second column of the rotation matrix R_π, and K is the camera intrinsic matrix.

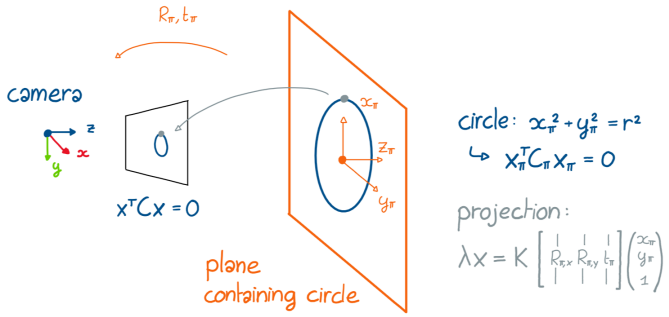

Figure 4.8: Illustration of the cricle projection method.

Chapter 5

Results

In the following analysis, we consider synthetic datasets with images captured around a single tree. The synthetic dataset were generated using Vulkan Glasses from the Vision for Robotics Lab (V4RL).

5.1 Pixelwise Approach

In the pixelwise approach, we assess whether or not the proposed pixelwise semantic bundle adjustment is able to reduce the camera pose error. The camera pose error refers to the translation and angular error between some estimated camera poses and the ground truth camera poses. Since we are using synthetic datasets for the evaluation, the ground truth camera poses $\bar{p}_1, \ldots, \bar{p}_n$ are known.
To achieve this, we first employ COLMAP as SfM algorithm to estimate the initial camera poses $p_1^{\text{init}}, \ldots, p_n^{\text{init}}$. These initial camera poses are then refined using the proposed pixelwise approach leading to new refined camera poses $\hat{p}_1, \ldots, \hat{p}_n$. Finally, we compute the camera pose error of the initial camera poses $p_1^{\text{init}}, \ldots, p_n^{\text{init}}$ before the refinement step), and of the refined camera poses $\hat{p}_1, \ldots, \hat{p}_n$ (after the refinement step). These two errors can then be compared to assess our method. This process is depicted in Figure 5.1.

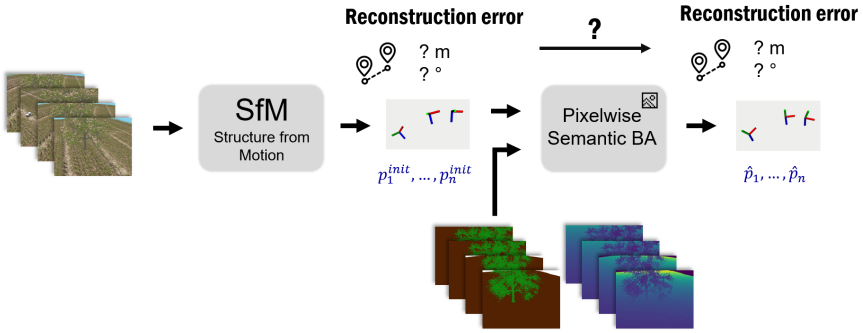

Figure 5.1: Evaluation process for the pixelwise approach.

Table 5.1 presents the mean camera pose error before and after the refinement step for various datasets. Each cell contains the mean translation error in centimeters and the mean angular error in degrees. The last column displays the relative change of error in percentage for the translation and angular errors. A negative error change indicates an improvement in camera poses, while a positive change indicates a deterioration.

Dataset	Error Before	Error After	Error Change
Dataset 1	2.90cm	2.52cm	-13%
	0.580°	0.639°	+10%
Dataset 2	**2.43cm**	**1.66cm**	**-31%**
	0.275°	**0.216°**	**-21%**
Dataset 3	2.99cm	2.77cm	-7.4%
	0.177°	0.170°	-4.0%
Dataset 4	1.03cm	1.47cm	+43%
	0.305°	0.285°	-6.6%
Dataset 5	1.70cm	1.89cm	+11%
	0.664°	0.703°	+5.9%
Dataset 6	1.13cm	1.02cm	-9.7%
	0.218°	0.302°	+39%

Table 5.1: Translation and angular error comparison before and after using the pixelwise semantic BA for various datasets. The change of error for the translation and angular error is displayed in the last column.

From the table, we observe that for datasets 1-3, the error after the refinement is lower. This indicates that the proposed method was able to improve the camera poses. However, for datasets 4-6, the error remains the same or even increases. The reason is that datasets 1-3 appeared more challenging for COLMAP, resulting in a larger initial error. Thus, our method still had room to improve the camera poses. On the other hand, for datasets 4-6, COLMAP obtained very accurate camera poses and our method struggled to refine them further. This can be attributed to the complexity of the scene, which leads to local minima in the error function.
While the camera poses error significantly decreased for datasets 1-3, it's important to note that these results employed ground truth semantic and depth maps. In practice, such accurate maps are usually unavailable, posing more difficulties for our method. However, in real-word scenes, the initial camera pose estimation from a SfM algorithm such as COLMAP would likely be less accurate, thereby leaving more room for our method to refine the camera poses.
Lastly, we observed that the choice of the depth error threshold d_{error} significantly impacts the method. Based on experiments, relatively large values around 1-2m show the most promising results.

5.2 Geometric Approach

5.2.1 Tree Trunk Localization (fixed camera poses)

Regarding the geometric approach, we first evaluate how well the estimated cylinders match a given tree trunk. For this, we only optimize over the parameters of the cylinder while keeping the camera poses fixed.
Figure 5.2 presents the mean IoU throughout the optimization process, depicting values at each optimization step. The various curves represent experiments with different initial cylinders. From the plot we observe that the IoU first shows a rapid increase followed by a convergence to a specific value. This further validates the choice of the error function specified in Equation 4.1 in order to maximize the IoU of each camera.

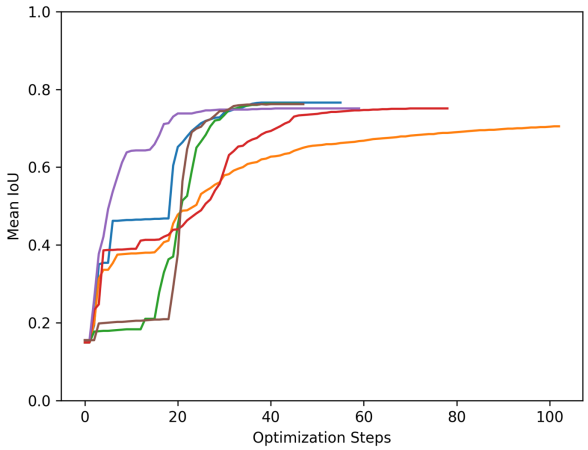

Figure 5.2: Progression of the mean IoU for various initial cylinders.

For the quantitative evaluation of the estimated cylinder, we manually define a ground truth cylinder representing the trunk, as illustrated in Figure 5.3. However, note that the definition of this ground truth cylinder is subject to interpretation, considering that tree trunks are not perfect cylinders.
Table 5.2 presents the error between the estimated cylinders and the ground truth cylinder, including radius, height, position and orientation errors. The relative errors for the radius and height are indicated within the parentheses. The experiments were conducted multiple times with different initial cylinders to assess the robustness of the method. Additionally, various numerical differentiation step sizes and the two parametrizations of the cylinder were employed in this experiment.

Error	Using ground truth camera poses	Using estimated camera poses
Radius error [mm]	8.0 ± 1.0 (3.2%)	14 ± 14 (5.6%)
Height error [cm]	19 ± 3.1 (9.1%)	16 ± 3.3 (7.6%)
Position error [cm]	30 ± 7.1	38 ± 5.9
Orientation error [°]	2.2 ± 0.88	4.7 ± 4.9

Table 5.2: Comparison between estimated cylinders and ground truth cylinder. The relative errors for the radius and height are indicated within the parentheses.

The first column in the table shows the results obtained using the ground truth

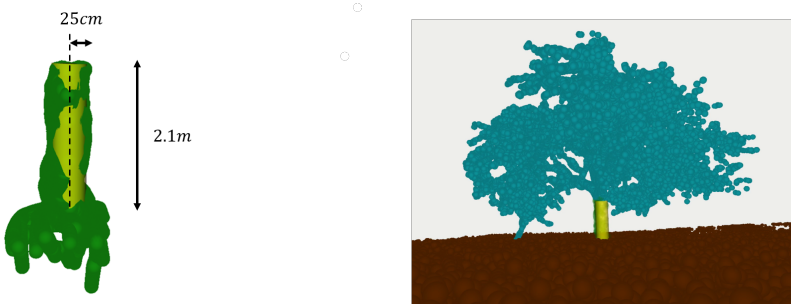

Figure 5.3: **Left:** Ground truth cylinder. **Right:** Estimated cylinder in yellow.

camera poses, while the second column shows the results obtained with the camera poses estimated by COLMAP. In both cases, our method accurately localizes the tree trunk. It is worth noting a slight increase in error when using the estimated camera poses, which is expected. Finally, we can observe a notable height and position offset. This offset can be attributed to the ground truth cylinder not accounting for the roots of the trunk, as depicted in Figure 5.3.

5.2.2 Influence of the Cylinder Parametrization

In Section 4.2.3, we explored two parametrizations to represent a cylinder. However, there does not seem to be a major difference in performance when it comes to the optimization. In some cases, we observed that the first parametrization, employing orientation, faced challenges with local minima. This can likely be attributed to the use of a quaternion manifold for orientation, adding complexity to the numerical optimization. In addition, the extra degree of freedom in the orientation due to the radial symmetry of cylinders also adds complexity. In contrast, the second parametrization, which instead employs two 3D points, appears to be a more appropriate and straightforward parametrization.

5.2.3 Geometric Semantic BA (optimizing camera poses)

The previous section discussed tree trunk localization, which means optimizing over the cylinder with fixed camera poses. Nevertheless, our objective is to improve the camera poses, implying they should also be optimization variables rather than kept fixed.

Unfortunately, optimizing over both the cylinder and the camera poses yields unsatisfactory results. This is mainly due to the simple geometry of a cylinder leading to many global minima. Indeed, when considering a single cylinder, many different camera poses exist for which the reprojection of the cylinder is the same. Such an example is depicted in Figure 5.4.

Therefore, the optimal camera poses in the optimization are not unique, which means the camera poses might end up anywhere during the optimization. This indicates that even if the optimization successfully minimizes the semantic error function and, conversely, maximizes the IoU, the camera poses might not end up closer to the ground truth ones. In conclusion, this suggests that the proposed error function is inappropriate if used on its own. Some future insights and ideas to overcome these issues are discussed in Chapter 6.

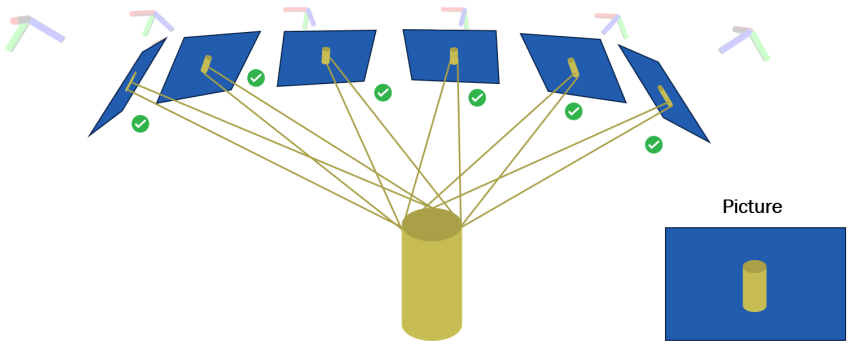

Figure 5.4: Illustration of the lack of constraints when using a cylinder.

5.3 Runtime and Implementation

The pixelwise semantic bundle adjustment and the geometric semantic bundle adjustment were both implemented in C++, building upon the standard COLMAP bundle adjustment. The optimization scheme is implemented using the Ceres Solver [16]. As mentioned previously, the synthetic datasets were generated using Vulkan Glasses from the V4RL.

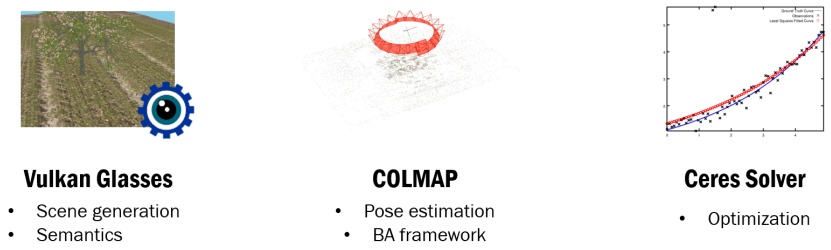

Figure 5.5: Software and tools used for the experiments and implementation.

For a dataset with 4 images of size 3072x2304, running the pixelwise semantic bundle adjustment takes approximately 5 seconds per optimization step, using a pixel step size of 5. However, with a smaller pixel step size of 2, each optimization step takes about 2 minutes, illustrating the limits of the method. Generally, the optimization converges after approximately 25 optimization steps.
In contrast, the geometric pixelwise semantic bundle adjustment only takes about 3 seconds per optimization step. When only optimizing over the cylinder with fixed camera poses, as discussed in Section 5.2.1, each optimization step lasts about 2 seconds, and convergence is reached after about 50 optimization steps. This can also be observed in Figure 5.2.

Chapter 6

Conclusion and Future Work

This work explored various ways to leverage semantic information with the aim of improving camera pose estimation in natural environments. Two distinct methods were introduced: a pixelwise approach and a geometric one. The pixelwise method demonstrated success in refining the camera poses in certain scenarios, following an initial estimation performed with COLMAP. Nevertheless, this is to be tempered with the fact that only ground truth semantic and depth maps were employed. The geometric approach, on the other hand, parametrizes the scene with a cylinder in order to evaluate the semantic consistency. The results have shown accurate localization of tree trunks by using a cylinder. Nevertheless, refining the camera poses proved impossible due to a lack of inherent constraints in this approach.

Figure 6.1: **Left:** Real image of a tree. **Right:** Synthetic image of a tree.

One of the main limitations of our work is that it has solely been tested on synthetic datasets using ground truth semantic maps. To address this, conducting experiments with real datasets becomes crucial for a more comprehensive evaluation of the proposed methods. Real datasets are expected to yield less accurate initial camera poses, giving our methods more room to refine them. Furthermore, employing estimated semantic maps, as opposed to relying on ground truth maps, is a key step in bringing the methods closer to practical application. Regarding the pixelwise semantic bundle adjustment, trying out alternative error functions to the zero-one loss could be an interesting area for future research. In addition, handling special cases of depth inconsistencies differently might enhance the performance. Finally, the primary issue in the geometric semantic bundle adjustment can be attributed to the simplicity of the cylinder shape leading to a lack of constraints. To address this, one could introduce additional constraints, such as incorporating 3D landmarks. Another approach is to alternate between the geometric semantic bundle adjustment and a traditional bundle adjustment to prevent converging to an incorrect solution.

Bibliography

[1] C. Hane, C. Zach, A. Cohen, R. Angst, and M. Pollefeys, "Joint 3d scene reconstruction and class segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 97–104.

[2] C. Häne, C. Zach, A. Cohen, and M. Pollefeys, "Dense semantic 3d reconstruction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 9, pp. 1730–1743, 2016.

[3] M. Blaha, C. Vogel, A. Richard, J. D. Wegner, T. Pock, and K. Schindler, "Large-scale semantic 3d reconstruction: an adaptive multi-resolution model for multi-class volumetric labeling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3176–3184.

[4] I. Cherabier, J. L. Schonberger, M. R. Oswald, M. Pollefeys, and A. Geiger, "Learning priors for semantic 3d reconstruction," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 314–330.

[5] Z. Wei, Y. Wang, H. Yi, Y. Chen, and G. Wang, "Semantic 3d reconstruction with learning mvs and 2d segmentation of aerial images," *Applied Sciences*, vol. 10, no. 4, p. 1275, 2020.

[6] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.

[7] J. Iglhaut, C. Cabo, S. Puliti, L. Piermattei, J. O'Connor, and J. Rosette, "Structure from motion photogrammetry in forestry: A review," *Current Forestry Reports*, vol. 5, pp. 155–168, 2019.

[8] P. Surovỳ, A. Yoshimoto, and D. Panagiotidis, "Accuracy of reconstruction of the tree stem surface using terrestrial close-range photogrammetry," *Remote Sensing*, vol. 8, no. 2, p. 123, 2016.

[9] S.-C. Lee, V. Lu, C.-C. Wang, and W.-C. Lin, "Lidar-based localization on highways using raw data and pole-like object features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 230–237.

[10] G. Chen, F. Lu, Z. Li, Y. Liu, J. Dong, J. Zhao, J. Yu, and A. Knoll, "Pole-curb fusion based robust and efficient autonomous vehicle localization system with branch-and-bound global optimization and local grid map method," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 11 283–11 294, 2021.

[11] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[13] T. Schneider, M. T. 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*, vol. 3, no. 3, pp. 1418–1425, 2018.

[14] A. Cramariuc, L. Bernreiter, F. Tschopp, M. Fehr, V. Reijgwart, J. Nieto, R. Siegwart, and C. Cadena, "maplab 2.0 – A Modular and Multi-Modal Mapping Framework," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 520–527, 2023.

[15] C. Campos, R. Elvira, J. J. Gomez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[16] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 10 2023.

Appendix A

Sources

Here are the sources of the images contained in this report:

- https://www.flaticon.com
- https://www.plantmegreen.com/pages/apple-tree-guide
- http://ceres-solver.org/nnls_tutorial.html
- https://v4rl.com/
- https://stock.adobe.com/ee/images/drone-sprayer-flies-over-apple-trees-smart-farming-and-precision-agriculture/552638357
- https://www.turbosquid.com/3d-models/realistic-tree-3d-model/1133513