Efficient and Robust 3D Object Reconstruction based on Monocular SLAM and CNN Semantic Segmentation*

Sergey Triputen*[1], Thomas Weber*[1], Atmaraaj Gopal[1], Steffen Eißele[1],
Christian Höfert[1], Kristiaan Schreve[2], and Matthias Rätsch[1]

[1] Reutlingen University, 72762 Reutlingen, Germany
{sergey.triputen,thomas.weber,matthias.raetsch}@reutlingen-university.de
https://www.visir.org
[2] University of Stellenbosch, Stellenbosch, South Africa
kschreve@sun.ac.za
* these authors contributed equally to this work

Abstract. Various applications implement SLAM technology, especially in the field of robot navigation. We show the advantage of SLAM technology for independent 3D object reconstruction. To receive a point cloud of every object of interest void of its environment, we leverage deep learning. We utilize recent CNN deep learning research for accurate semantic segmentation of objects. In this work, we propose two fusion methods for CNN-based semantic segmentation and SLAM for the 3D reconstruction of objects of interest in order to obtain a more robustness and efficiency. As a major novelty, we introduce a CNN-based masking to focus SLAM only on feature points belonging to every single object. Noisy, complex or even non-rigid features in the background are filtered out, improving the estimation of the camera pose and the 3D point cloud of each object. Our experiments are constrained to the reconstruction of industrial objects. We present an analysis of the accuracy and performance of each method and compare the two methods describing their pros and cons.

Keywords: 3D reconstruction · SLAM· LSD-SLAM· monocular camera · CNN· semantic segmentation · bin-picking · collaborative robot · depth estimation

1 Motivation

Our research is mainly motivated by tasks given by our industrial partners in facial recognition, industrial automation, and robot navigation.

While they have distinct specifications and requirements for their preferred task solutions, there is a common overlap of interest in 3D surface reconstruction, as seen in fig. 1.

* This work is partially supported by a grant of the BMBF FHprofUnt program, no. 13FH049PX5

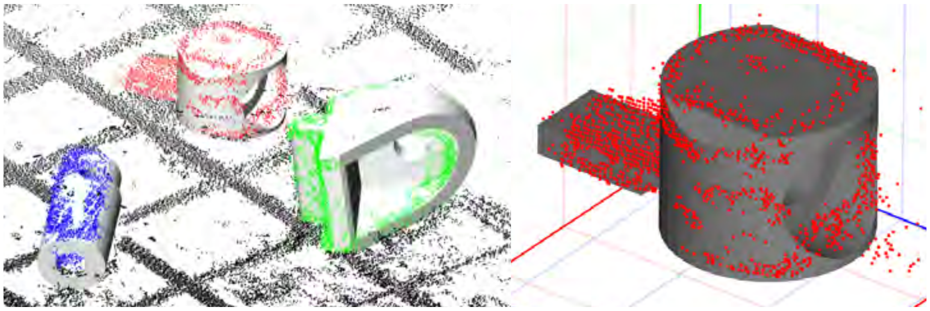

Fig. 1: (left) Point cloud produced from LSD-SLAM with the respective 3D model of the object aligned to its point cloud. (right) Point cloud of red labeled object of interest.

Therefore, our group has been looking for possible solutions or a combination of solutions to fulfill the 3D reconstruction challenge and to possibly improve it. In particular, we analyze our proposed solution and present the results concerning accuracy, performance and commercial feasibility.

In this paper, we propose two distinct methods for the 3D reconstruction of objects, characterized by semantic segmentation of the frames that are either input to an output from SLAM. To the best of our knowledge, this is the first work proposing a fusion with a focus of SLAM only to the points belonging to every single object of interest. The main contribution of this paper is a highly efficient and robust 3D object reconstruction. It is based on two different fusion methods of semantic segmentation via CNN and monocular visual SLAM. We focus the estimation of the 3D point cloud and camera trajectory obtained by SLAM to only the feature points belonging to every single object of interest. Therefore, robust and accurate 3D reconstruction, even in dynamic environments and in non-rigid world scenarios, can be obtained. Feature points not belonging to the object, within the background, or with low accuracy and noise are filtered out. We analyze the accuracy and performance of the fusion methods of CNN-based semantic segmentation with LSD-SLAM for 3D reconstruction.

We contribute (1) two fusion methods of CNN-based segmentation with SLAM for the 3D reconstruction; (2) an analysis of the accuracy and performance of the proposed methods; and as a significant novelty, (3) a pipeline to reconstruct the 3D point cloud of every single object of interest void of the background.

2 Problem Definition and Related Work

Our research targets to reconstruct the point cloud of a single object of interest with a monocular SLAM technology. There are various approaches available for 3D reconstruction. Considering the advantages that a relatively low-cost and smaller monocular camera could provide in an industrial environment, we choose to employ and improve the monocular SLAM technology in our experiments. The monocular camera is especially optimal for applications on embedded platforms, where CMOS image sensors are preferred. A further description of feasible depth sensing options is presented in [23], where the properties of different direct depth measurement sensors (i.e. stereo and RGB-D cameras, lasers, and optical scanning) and the depth estimation algorithm with SLAM are outlined.

There is a state-of-the-art visual odometry solution, the Semi-direct Visual Odometry (SVO) [7, 8, 19], which allows robust feature point tracking in dynamic environments and produces accurate point clouds. Svo is exceptionally stable and reliable when used with event-based cameras. These cameras are however significantly more expensive in comparison to monocular cameras and do not offer functionality to extract the point cloud of an object of interest. Nevertheless, the methods that we propose in this paper could be implemented with SVO to semantically label and reconstruct an object in the scene.

Since monocular SLAM produces semi-dense reconstruction [5,6,11,12,16,17], it is necessary that its accuracy is analyzed before approaching methods to improve the reconstruction quality. However, whenever monocular SLAM is employed there is the point cloud scaling and space alignment problem, which is a well-known problem among SLAM researchers and has been solved in various forms, i.e. through inertial measurement unit (IMU) sensor integration [9] and forward kinematics [23], to name a few.

The work in [18, 22] propose solutions based on the combination of semantic segmentation with monocular SLAM, wherein they semantically label the point cloud from SLAM with bounding boxes [18] or pixel-wise [22]. Our goal is to reconstruct a point cloud for a single predetermined object instead of the whole scene. An example of an industrial bin-picking use-case for the point cloud would be to register it to the 3D model of the object, which scales the point cloud and then determines the gripping point on the object's surface. We propose a reconstruction solution with optimal accuracy based on CNN semantic segmentation [1, 14, 21] in combination with SLAM technology. Given that the CNN is trained to recognize a specific set of industrial objects, the real dimensions of the recognized object are known and could be used to scale the object's point cloud. Contrary to [18], we also measure and evaluate the accuracy of the reconstruction.

3 System Components Overview

In order to implement the proposed methods, we first outline our experimental constraints and the subsystems of choice. We present a list of objects of interest, the SLAM technology used, the CNN-based masking of the input data and framework of our proposed solution.

3.1 Objects of Interest

We have determined the requirements for the objects of interest to improve our research result relevancy towards industrial applications. The chosen objects have to be present in industrial environments and their 3D CAD models have to be available to analyze the accuracy of reconstruction.

The objects we have selected, as shown in fig. 2, are parts of the industrial building kit connectors from *item*®. The 3D CAD models of each of these objects are available for download in several formats from the online product catalog.

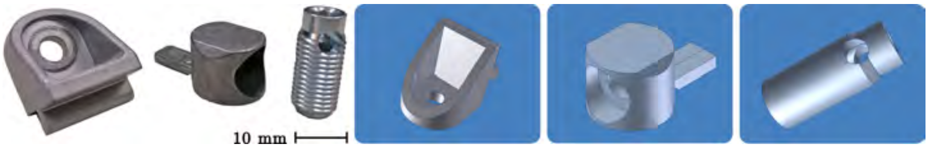

Fig. 2: The three industrial objects of interest and their 3D models: (f.l.t.r.) angle clamp bracket, universal-butt-fastener and automatic-butt-fastener.

3.2 SLAM Technology Overview

Our proposed solutions could be incorporated into almost all available monocular SLAM systems. The SLAM technology that we have chosen to use for our experiments is the LSD-SLAM developed by the Computer Vision team from the Technical University of Munich [6], due to the prior experience and in-depth understanding of their open-source code. For most of the experiments done with LSD-SLAM, it is sufficient to consider the technology as a complete black box solution. However, our implementation requires that the processing pipeline of LSD-SLAM is understood. It is made up of several software nodes, namely the `slam_core` and the `slam_viewer`. These nodes have distinct functions.

`slam_core`: The task of the `slam_core` is to receive a stream of input frames and produce keyframes. Our research requires distinguishing the pre-processing and tracking components in `slam_core`. We prioritize a thorough understanding of the pre-processing stage and its functions. This section contains algorithms to determine feature points based on gradient vectors. These feature points would then be used by the tracking algorithm to estimate the camera pose. They are also the criteria to determine the subsequent keyframe. It is significant to note that the feature points are determined in the pre-processing section, based on the pixel gradients. Being aware of this assists in understanding the logic behind our proposed solution description. Additionally, the quality of the depth estimations in the keyframes is the basis for our accuracy analysis.

`slam_viewer`: The task of the `slam_viewer` is to receive all keyframes from the `slam_core`, compute the estimated depths based on the estimated camera pose and feature points, and represent these depths as a point cloud. Nonetheless, the node only provides the calculated final point cloud of the scene. This is insufficient for our research. We require further information to represent the estimated depths of a keyframe, and various mathematical and statistical tools for our keyframe-by-keyframe analysis. To comply with these requirements, we implement the `slam_viewer` node source code in a MATLAB environment, where a representation of the point cloud from each keyframe is visualized.

3.3 Semantic Segmentation Overview

The semantic segmentation of an image means assigning a class label pixel-wise. Multiple instances of the same objects in an image are assigned the same class labels. Thus they are not necessarily distinguishable or separated, so-called instanced. However mere segmentation of objects and "background noise" is sufficient for our task. The development of CNN architecture is not a part of our research. We opt for an Mask R-CNN with pre-trained weights for the MS COCO data-set [15] that we fine-tune in training based on our requirements. The semantic segmentation solution that we employ for our experiments is based on CNN because it boasts robust and accurate results [10]. Our requirements for a CNN based semantic segmentation solution are the ease of use, well documented with examples, compatible with well-known CNN frameworks (i.e. *Tensorflow, Caffe*) and lastly, also provides tools for data representation.

CNNs are commonly trained with more universal data, whereas for our (bin-picking) use-case, the data is constrained to objects and backgrounds of an industrial robot work-space. These constraints allow a more robust CNN semantic segmentation as it is trained with a larger volume of the work-space pictures and relatively few classes. There is also the advantage that industrial environments are controlled to yield stable conditions resulting in optimal segmentation.

We prepared large sets of semi-synthetic ground truth data for fine-tuning the CNN. For this, we combined real-world HDRI environment maps (360° photosphere pictures) as background with physically-based renderings of the industrial objects of interest, based on techniques like [24]. The image generation is done in the free and open source 3D creation suite *Blender* with its physically-based rendering engine *Cycles*. These renderings also contain ground truth pass with segmentation masks and class assignments. Images and ground truth passes are converted into the MS COCO data format to be used to fine-tune the CNN network in training.

This procedure provides us vast amounts of suitable training data without the need for tedious manual, labor-intensive image capturing, labeling and classification, which would be unfeasible for widespread industrial usage.

4 Proposed Solutions Overview

We attempt to solve the problems by implementing solutions based on the semantic segmentation methodology, which assigns a predefined class to each pixel of an image. The work [18] proposes a solution to this approach, wherein a form of object detection (or semantic segmentation) is used in combination with SLAM. FLAIR feature encoding is used in this work to detect and build bounding boxes around the object(s) of interest. Furthermore, the problem their work seeks to solve is distinct to ours, as they employ SLAM to bolster the quality and accuracy of object recognition, whereas our goal is the inverse of it. We aim to measure and improve the accuracy of SLAM reconstruction of an object by object recognition. We expect that a CNN-based recognition would improve the accuracy of segmentation, because the latter exclusively labels each pixel of the objects of interest, while the bounding boxes from the former still include a part of the environment. We further hypothesize that a more accurate segmentation of the object(s) from the background results in a more precise reconstruction. The two methods we propose to employ CNN semantic segmentation supported monocular SLAM to reconstruct an object of interest in the scene. Although both yield a point cloud of an object isolated from its environment, there are differences in reconstruction accuracy and computing performance, we investigate.

4.1 Method I - CNN Semantic Segmentation applied to Output Keyframes

Description: The idea of the first method is to apply semantic segmentation to the keyframes output by SLAM, as proposed in [18]. This method would result in a complete point cloud of the scene with the objects of interest labeled, allowing the object's point cloud to be extracted from its environment. The semantic segmentation system they employ is non-CNN in combination with monocular ORB-SLAM [16, 17]. Accuracy and performance of their reconstruction are not measured nor analyzed. In contrast, we propose to use a CNN-based semantic segmentation [1, 14, 21] in combination with LSD-SLAM [6]. Their work proves the feasibility and advantages of labeling objects of interest in the SLAM reconstructed point cloud. Our aim is also to measure, analyze and improve the performance and accuracy of the 3D reconstruction. As there was no data on the accuracy of the reconstruction in their work, we re-implement this idea and measure the accuracy of the reconstruction and the computing performance of the system, with the constraints discussed in section 3.1. We hypothesize that high recognition accuracy is achievable when a CNN is used for semantic segmentation. This CNN solution could be more processing power demanding, causing a drop in performance, depending on the optimization of the CNN architecture.

Implementation: We present a solution, where the semantic segmentation is applied to the keyframes output from the LSD-SLAM core, as shown in fig. 3. A keyframe is made up of the source camera frame, the estimated camera pose and the feature points determined. The camera frames from the tuple of keyframes are input to the semantic segmentation CNN block to recognize, segment and label the pixels in the frames. This block outputs frames, where the background pixels are labeled a monotone grey, while the pixels of interest are segmented and the masks are colored, as shown in fig. 5-B. The labeled frames are used as a mask to recognize the feature points of interest, from which the feature points of the objects of interest could be determined and isolated. As the point cloud is constructed based on the feature points and the estimated camera pose, the point cloud of the object of interest could be recognized and isolated. Examples resulting from this solution are shown in fig. 5-E-F-G.

4.2 Method II - CNN Semantic Segmentation applied to Input Frames

Description: We propose another method of semantic segmentation integration with SLAM, where we recognize the objects in the stream of frames from the monocular camera, instead of the keyframes. As the keyframes themselves are determined based on the feature points on the stream of frames of the camera, we expect a difference in the accuracy and performance with this method. We expect to reduce a significant amount of background in the frame, which means LSD-SLAM would track, estimate the depth and produce a point cloud of only the object of interest. This could result in a better SLAM performance, as there are significantly fewer feature points in the frames. As a result of a focused area of interest on the frame and the background being filtered out, the tracking process in SLAM is speculated to be more efficient and accurate. This is due to minimal variance in the distance between the tracked feature points, given that the object of interest is generally much closer to the camera than the background is. As the camera moves, there will be a change in perspective. The displacement of the tracked feature points is proportional to the distance of the elements in the frame to the camera. Nevertheless, there would be a loss in performance as the semantic segmentation is executed on every camera frame. This could be limiting for embedded systems, but as deep learning frameworks have been optimized to be GPU-accelerated [13, 20], this would still be a valid solution for use cases with GPU supported architectures.

Implementation: To solve this task, we extend the *Frames Source* section of LSD-SLAM, as shown in fig. 4, by processing the frames before they are input to `slam_core`. While a tuple of RGB frames captured by the monocular camera is gray-scaled, the same tuple is input to the semantic segmentation CNN block to recognize, segment and label the pixels in the frames. Similar to the previous method, the background is labeled a monotone grey, and the objects of interest are respectively colored, as seen in fig. 6-B. From the tuple of labeled frames, the label of the object of choice is determined (green in this example). We then build binary masks out of these frames by setting the determined pixels of interest generic white, and the rest generic black, as exemplified in fig. 6-C. These masks are further blurred in gray-scale with a suitable blur factor to reduce the gradients at the edges of the pixel region of interest. This is a necessary step as the gradients are the criteria for determining feature points, and the shading done while building the mask would distort the gradients at the edges of the object. The tuple of masks with their blurred edges are used to filter the grey-scale frames tuple. The resulting frames present only the grey-scaled pixels of the object of choice, while the rest are colored black, as shown in fig. 6-E. These frames with the background filtered out subsequently are input to `slam_core`.

5 Accuracy Analysis

When it comes to 3D reconstruction, one of the primary metrics to determine the quality of reconstruction is the accuracy. A lot of the work in SLAM gives

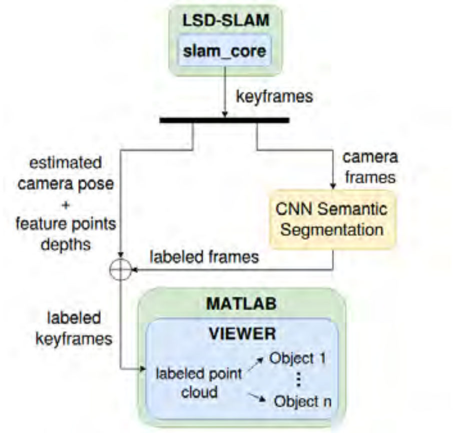

Fig. 3: Method I overview of the data flow and subsystems for the keyframes labeling solution.

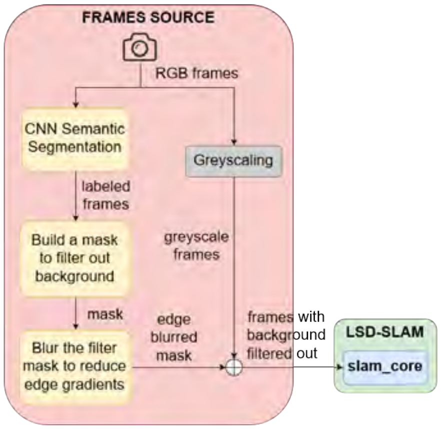
FRAMES SOURCE
RGB frames
CNN Semantic Segmentation
Greyscaling
labeled frames
Build a mask to filter out background
greyscale frames
mask
edge blurred mask
Blur the filter mask to reduce edge gradients
frames with background filtered out
LSD-SLAM
slam_core

Fig. 4: Method II overview of the data flow and subsystems for the all input frames labeling solution.

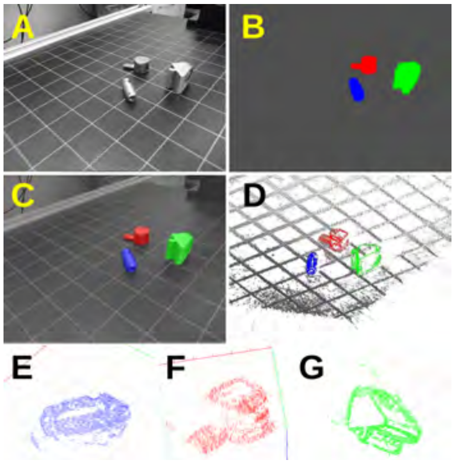

Fig. 5: The fusion of the CNN semantic segmentation with keyframes and the resulting isolated point clouds of each object in the frame. (a) The input camera frame from a selected keyframe. (b) The semantically labeled frame. (c) Input frame with labels overlay. (d) Semantically labeled point cloud of the selected keyframe. Point clouds of the (e) blue, (f) red and (g) green labeled objects of interest.

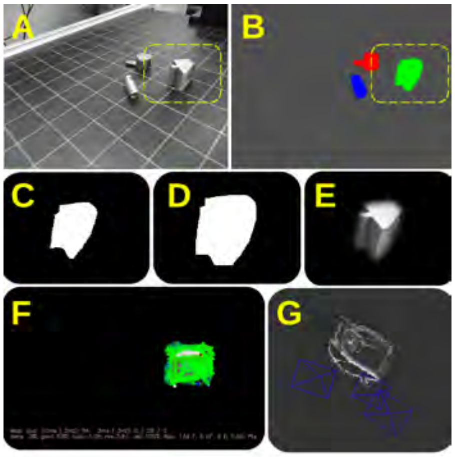

Fig. 6: The fusion of the CNN semantic segmentation with the input frames. (a) An input camera frame. (b) The labeled frame. The yellow dashed line is the region of interest (ROI) with the selected object of interest. ROI of (c) the resulted mask from fusion (d) the blurred mask, and (e) the final filtered frame. (f) The LSD-SLAM debug frame with feature points highlighted green and (g) the reconstructed point cloud of the chosen object as shown in the slam_viewer.

priority to and describes the accuracy of the estimated camera pose and path; however, we aim to analyze the accuracy of the reconstruction of a single object by a SLAM system. Hence, we describe here the methodology used for our result measurement and analysis. We define the accuracy of the reconstruction of an object as the normalized average error between the reconstructed point cloud and the 3D model of the object. The point cloud is first registered to the 3D model before the cloud-to-mesh distance is calculated. The registration is done to scale and transform the reconstructed point cloud to "fit" either the mesh or the dense point cloud from the 3D model. This is generally done via a minimization task that reduces the error between the two entities. There are many algorithms for the registration and cloud-to-mesh error calculation, and we choose to use the algorithms implemented in the open-source software for point cloud and 3D model processing, *CloudCompare* [4]. It provides several tools for manual editing and rendering of 3D point clouds and meshes, from which the most significant for our goal are tools for registration [2] and cloud-to-mesh distance computation [3].

To measure the accuracy of our reconstruction, we first import one instance of the point cloud, cf. fig. 5-E-F-G, and one of the object's 3D model as a mesh, shown in fig. 2. We then register the point cloud to the mesh, constraining the registration calculation with a parameter for the maximum error allowed. This results in the point cloud being optimally transformed and fitted to the mesh. From there on, we compute the cloud-to-mesh distance of every point in the point cloud. We obtain mean and standard deviation of the reconstruction error.

These steps are done for every keyframe with the parameters controlled, for both of the methods we propose.

6 Results

In this section, we present the results of our experiments. They are in the form of measurements and statistics of a set of metrics for the implementations of both proposed methods. The performance of both the proposed methods and accuracy of the 3D reconstruction are analyzed and compared to the original LSD-SLAM of the Technical University of Munich.

6.1 Performance Analysis Results

The processing time demanding block in the pipeline for each experiment is the tracking of feature points by `slam_core` and the semantic segmentation with CNN. Both methods increase processing time compared to the original LSD-SLAM, as there is the additional semantic segmentation of the frames. The empirical metrics defined before measuring the performance are listed in table 1.

The keyframe per frame factor is calculated by the ratio of the number of keyframes (KFs) to the number of input frames from the camera. We understand that the value from one experiment is non-reproducible, as it depends heavily on various variables, i.e., lighting, speed and trajectory of camera motion, and

Table 1: Metrics for analyzing the performance obtained by averaging data from LSD-SLAM and CNN semantic segmentation.

Metric	Mean Value
KF per frame factor	0.0765
Track time per KF [ms]	4311
CNN segmentation time per frame [ms]	128.14
Number of feature points per KF	87 243

Table 2: Performance metrics for analysis of method II.

Metric	Method I	Method II
Mean feature point per KF factor	1	0.074
Mean KF per frame factor	0.0765	0.1832

the camera frame rate, and so we obtain a mean value from several experiments. Similarly, the mean tracking time for every keyframe, mean time for CNN inference of a camera frame with a 640×480 px resolution and the mean number of feature points per keyframe.

Determining these parameters enables approximation of the time for tracking and segmentation in our methods. For method I, it is relevant to measure the factor between input and output keyframes, since we expect a large number of feature points. The run-time for method I is the sum of the processing time of the original LSD-SLAM to reconstruct the scene and keyframes CNN inference.

To evaluate the performance of method II, we calculate additional metrics as shown in table 2. Specifically for method II, the performance is determined by the amount of background filtered before tracking. Therefore, we determine the ratio of feature points generated by this method to the mean number of feature points per keyframe, as given in table 1. The parameter is dependent on the accuracy of the segmentation and the density of feature points in the keyframes. The run-time for method II also varies due to significantly fewer feature points being generated per keyframe. Given a tuple of camera frames with our system specifications, our results prove that method II is twice as fast as method I, based on the values of our performance metrics.

6.2 Accuracy Analysis Results

The analysis of the 3D reconstruction accuracy is carried out in the *CloudCompare* software, as described in section 5. Our metrics for accuracy are the mean normalized error of the registered point cloud from the surface of the object's mesh, and the mean normalized standard deviation of the errors. Table 3 shows that the mean error for both methods is similar. The mean standard deviation for every object from method II shown in table 4 confers that the errors are less distributed and the points closer to the surface of the 3D mesh when reconstructed with method II. We infer therefore that method II reconstructs a more accurate point cloud of an object compared to method I. This agrees with our hypothesis

Table 3: Mean normalized error of the registered point clouds from their respective 3D model surfaces.

Object	Method I	Method II
Red	0.1583	0.1501
Green	0.2151	0.2203
Blue	0.1688	0.1586

Table 4: Mean normalized standard deviation of the registered point clouds from their respective 3D model surfaces.

Object	Method I	Method II
Red	0.3071	0.1334
Green	0.6044	0.2462
Blue	0.2240	0.1189

of a more accurate 3D reconstruction, due to the reduced number of feature points when the background is filtered out.

7 Conclusion and Further Work

Both in this work proposed methods solve the initial task. We receive independent point clouds of all individual objects of interest.

In method I, we apply CNN-masking only on keyframes. The performance is slower compared to method II, albeit labeling only the keyframes. This is due to the larger number of feature points to be tracked in the keyframes, which increases process time in the LSD-SLAM block of the pipeline. Contrary to method II, we do not use a reduction of feature points for each frame. Method I is optimal for applications with minimal changes in the scene because no keyframe is generated while there is no major change. The semantic segmentation is done after the `slam_core`; thus the accuracy may be lowered due to further “background” feature points not belonging to the object. The method allows the extraction of several objects of interest from the scene’s point cloud.

Method II is more resilient to motion in the background of the object and changes in the scene than method I. The camera pose is estimated by solving a minimization task between two point clouds. Point clouds are constructed based on the projection from the camera pose. This pose calculation is sensitive to feature points furthest away from its current position in the keyframes: A slight angular shift translates feature points the more the further away they are from the camera. The minimization task is expected to compute better camera pose estimate when the further feature points are filtered out. With refined, robust CNN-masking, most of the feature points should be concentrated on the object itself; “background noise” is filtered out. The `slam_core` produces keyframes at a higher rate compared to method I. Therefore method II is optimal for 3D reconstruction in dynamic environments and even non-rigid world scenarios.

Additionally, our work can be used as an alternative solution to the scaling problem in SLAM. The CNN semantic segmentation and pose estimation can be used to compute the scaling from the 3D model of the object. The accuracy of the 3D reconstruction is dependent on the accuracy of CNN semantic segmentation. Task- and object-specific CNN architectures could yield further improvements. Adapted CNN architectures may reduce run-time and require less demanding GPUs. As a result, the application could work in real time, even on lower-end hardware.

Our further interest is investigating the feasibility, accuracy, and performance of the methods implemented on embedded low-cost systems.

As of now, the CNN semantic segmentation provides no additional information about the camera pose. However, an adapted CNN architecture could be trained to estimate information about the pose of the object. The object pose estimation could be an initial value input to the optimization task, which would facilitate faster solving and determination of the camera pose.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. CoRR **abs/1511.00561** (2015), http://arxiv.org/abs/1511.00561 3, 6
2. Besl, P.J., McKay, N.D.: A method for registration of 3-d shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (Feb 1992). https://doi.org/10.1109/34.121791, http://dx.doi.org/10.1109/34.121791 9
3. Cignoni, P., Rocchini, C., Scopigno, R.: Metro: Measuring error on simplified surfaces **17**, 167 – 174 (06 1998) 9
4. Daniel Girardeau-Montaut: CloudCompare, http://www.cloudcompare.org/ 9
5. Engel, J., Sturm, J., Cremers, D.: Semi-dense Visual Odometry for a Monocular Camera. In: 2013 IEEE International Conference on Computer Vision. pp. 1449–1456 (Dec 2013). https://doi.org/10.1109/ICCV.2013.183 3
6. Engel, J., Schoeps, T., Cremers, D.: LSD-SLAM: large-scale direct monocular SLAM. In: Eur. Conf. Comput. Vis. vol. 8690, pp. 834–849 (09 2014) 3, 4, 6
7. Forster, C., Pizzoli, M., Scaramuzza, D.: Svo: Fast semi-direct monocular visual odometry. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 15–22 (May 2014). https://doi.org/10.1109/ICRA.2014.6906584 3
8. Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems. IEEE Transactions on Robotics **33**(2), 249–265 (April 2017). https://doi.org/10.1109/TRO.2016.2623335 3
9. J. Engel and J. Sturm and D. Cremers: Camera-based navigation of a low-cost quadrocopter. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2815–2821 (Oct 2012). https://doi.org/10.1109/IROS.2012.6385458 3
10. Jafari, O.H., Groth, O., Kirillov, A., Yang, M.Y., Rother, C.: Analyzing modular CNN architectures for joint depth prediction and semantic segmentation. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 4620–4627 (May 2017). https://doi.org/10.1109/ICRA.2017.7989537 5
11. Klein, G., Murray, D.: Parallel Tracking and Mapping for Small AR Workspaces. In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. pp. 225–234 (Nov 2007). https://doi.org/10.1109/ISMAR.2007.4538852 3
12. Klein, G., Murray, D.: Parallel tracking and mapping on a camera phone. In: 2009 8th IEEE International Symposium on Mixed and Augmented Reality. pp. 83–86 (Oct 2009). https://doi.org/10.1109/ISMAR.2009.5336495 3
13. Li, C., Yang, Y., Feng, M., Chakradhar, S., Zhou, H.: Optimizing memory efficiency for deep convolutional neural networks on gpus. In: SC16: International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 633–644 (Nov 2016). https://doi.org/10.1109/SC.2016.53 7

14. Lin, G., Milan, A., Shen, C., Reid, I.D.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. CoRR **abs/1611.06612** (2016), http://arxiv.org/abs/1611.06612 3, 6
15. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. CoRR **abs/1405.0312** (2014), http://arxiv.org/abs/1405.0312 5
16. Mur-Artal, R., Tardos, J.: ORB-SLAM: Tracking and mapping recognizable features. Robotics: Science and Systems (RSS) Workshop on Multi View Geometry in Robotics (MVIGRO) (07 2014) 3, 6
17. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. CoRR **abs/1610.06475** (2016), http://arxiv.org/abs/1610.06475 3, 6
18. Pillai, S., Leonard, J.J.: Monocular SLAM supported object recognition. CoRR **abs/1506.01732** (2015), http://arxiv.org/abs/1506.01732 3, 5, 6
19. Pizzoli, M., Forster, C., Scaramuzza, D.: Remode: Probabilistic, monocular dense reconstruction in real time. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 2609–2616 (May 2014). https://doi.org/10.1109/ICRA.2014.6907233 3
20. Strigl, D., Kofler, K., Podlipnig, S.: Performance and scalability of gpu-based convolutional neural networks. In: 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing. pp. 317–324 (Feb 2010). https://doi.org/10.1109/PDP.2010.43 7
21. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.G.: Multi-view convolutional neural networks for 3d shape recognition. CoRR **abs/1505.00880** (2015), http://arxiv.org/abs/1505.00880 3, 6
22. Tateno, K., Tombari, F., Laina, I., Navab, N.: CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. ArXiv e-prints (Apr 2017) 3
23. Triputen, S., Gopal, A., Weber, T., Hofert, C., Schreve, K., Rätsch, M.: Methodology to analyze the accuracy of 3D objects reconstructed with collaborative robot based monocular LSD-SLAM. CoRR **abs/1803.02257** (2018), http://arxiv.org/abs/1803.02257 3
24. Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J., Jin, H., Funkhouser, T.A.: Physically-based rendering for indoor scene understanding using convolutional neural networks. CoRR **abs/1612.07429** (2016), http://arxiv.org/abs/1612.07429 5