Efficient 3D Tracking in Urban Environments with Semantic Segmentation

Martin Hirzer[1]
hirzer@icg.tugraz.at
Clemens Arth[1]
arth@icg.tugraz.at
Peter M. Roth[1]
pmroth@icg.tugraz.at
Vincent Lepetit[1,2]
lepetit@icg.tugraz.at

[1] Institute of Computer Graphics and Vision
Graz University of Technology
Austria
[2] Laboratoire Bordelais de Recherche en Informatique
University of Bordeaux
France

Abstract

In this paper, we present a new 3D tracking approach for self-localization in urban environments. In particular, we build on existing tracking approaches (*i.e.*, visual odometry tracking and SLAM), additionally using the information provided by 2.5D maps of the environment. Since this combination is not straightforward, we adopt ideas from semantic segmentation to find a better alignment between the pose estimated by the tracker and the 2.5D model. Specifically, we show that introducing edges as semantic classes is highly beneficial for our task. In this way, we can reduce tracker inaccuracies and prevent drifting, thus increasing the tracker's stability. We evaluate our approach for two different challenging scenarios, also showing that it is generally applicable in different application domains and that we are not limited to a specific tracking method.

1 Introduction

Accurate geo-localization of images is crucial for applications such as outdoor Augmented Reality (AR), autonomous driving, mobile robotics and navigation. Since GPS and compass information are often not precise enough, especially in urban environments, there has been a considerable scientific interest in computer vision methods that register and track mobile devices within a global reference frame. One common way is to rely on pre-registered images from the surroundings to obtain an accurate 3D location and orientation [23, 24].

However, such methods quickly become impractical, as many images need to be captured and registered in advance. Moreover, such pre-registered image collections typically capture only one specific appearance of the recorded scene, and matching them under different illumination or season conditions is still difficult. To overcome these drawbacks, much simpler 2.5D maps can be used (*e.g.*, [2, 21, 26]). These maps are generated from the buildings' outlines and their approximate heights and are broadly available, for instance via OpenStreetMap[1]. However, as these models are not textured, it is difficult to use them directly.

[1] www.openstreetmap.org

Thus, the first contribution of this paper is to exploit the information given by 2.5D maps overcoming these drawbacks. In particular, as illustrated in Fig. 1, we align a 3D rendering created from these 2.5D maps with a semantic segmentation of the input images by maximizing the pose likelihood. A key aspect here is that we focus on the information relevant for our task and additionally introduce rather abstract classes such as the buildings' edges, which prove to be very valuable. In fact, we use the angles about the vertical axis provided by accelerometers to rectify the input image such that the columns of the image correspond to vertical lines in 3D. Contrary to other measures, these angles are very reliable. Thus, we can simply finely sample the pose space around an initial prediction and present, as second contribution, an efficient method to compute the pose likelihood to finally obtain a good pose estimate.

As third and main contribution, these ideas are embedded into a 3D tracking pipeline, as illustrated in Fig. 2. In this way, typical tracking inaccuracies such as drift can be avoided and we are able to provide a stable and robust estimate of the exact camera pose, which is illustrated for two different scenarios.

In the remainder of this paper, we first discuss related work in Sec. 2. We then present our 3D tracking approach exploiting 2.5D maps and semantic segmentation in Sec. 3. Next, the thus obtained results on two challenging scenarios, compared to valid baselines, are shown in Sec. 4. Finally, we summarize and conclude the work in Sec. 5.

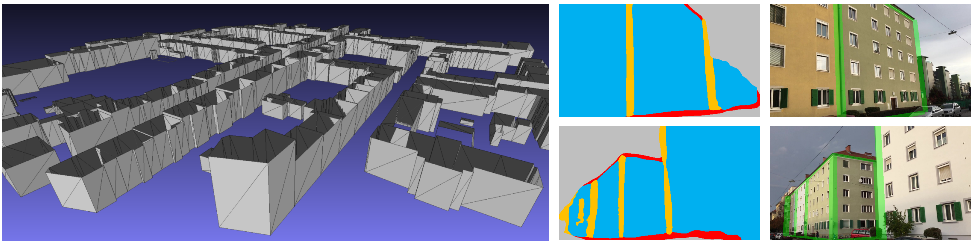


Figure 1: The proposed 3D tracking method registers the camera in video sequences using a simple 2.5D map (left) and a semantic segmentation of the frames (middle). The thus obtained aligned model overlaid over the frames (right).

2 Related Work

A core requirement for outdoor AR systems, autonomous driving or mobile robotics is an accurate, robust and large-scale camera registration in a global reference frame. Early works tried to tackle the problem by using GPS and compass sensor information [9]. Since this is often not accurate enough, image-based localization techniques have been developed in order to improve the computed camera pose estimate.

Such approaches usually take as input one or more camera images and optionally prior information from device sensors. Using this information, the 3D position or full 6 degrees of freedom (DoF) pose of the input image can be computed by matching 2D image points to pre-registered 3D scene points stored in a database obtained from pre-captured scene images. For example, [24] performs image-based localization based on a database that contains 20km of urban street-side imagery. [23] tackles the problem by an active correspondence search, combining both search directions when matching points, *i.e.*, 2D-to-3D and 3D-to-2D, which results in improved accuracy and performance.

Despite using advanced matching strategies, though, a core problem of these approaches remains their limited scalability, since many images need to be captured in advance for each new scene in order to create the databases. Furthermore, these databases present just snapshots of the corresponding locations. Thus, feature point matching under changing conditions due to illumination, season or construction work is very challenging if not impossible at all. In order to avoid the tedious creation and maintenance of scene databases, [31] and [27] use existing image collections such as GoogleStreetView. However, these collections are rather sparsely sampled and not available for many regions and countries, which severely limits their applicability to image-based localization for outdoor scenarios.

Another line of research avoids creating scene databases altogether by leveraging widely available, untextured 2D cadastral maps annotated with per-building height information, which we refer to as 2.5D maps or elevation models, generated for example from LIDAR data. In particular, [18] establishes line correspondences between the input image and a 2.5D map of the scene. However, due to insufficient accuracy regarding the image orientation, additional user interaction is required. Similarly, [21] searches for 3D-2D line and point correspondences between an image and a 2.5D map. Therefore, an already registered, second image is required as starting point for the method, which means that the first image of a sequence has to be manually annotated.

Other works ease the problem by using panoramic images, since the additional information contained in wide field of view images facilitates localization [1]. For instance, [8] proposes a building façade orientation descriptor to register panoramic images with 2D maps. However, since mobile devices typically have a rather narrow field of view, a descriptor such as the one proposed by [8] is not discriminative enough in such situations. [5] first detects vertical building outlines and façade normals in panoramic images, yielding 2D fragments, and then matches the obtained fragments with a 2D map. [6] matches a descriptor computed from vertical building outlines in perspective images with a 2D map. As a drawback, it requires some manual input to make detection of vertical edges and vanishing points easier.

More recently, SLAM-based systems have been proposed for use in outdoor localization tasks. For example, [29] and [19] globally align the local SLAM map from a stereo image pair, which, however, requires the user to walk several meters in practice in order to span the necessary baseline. In contrast, [2] leverages untextured 2.5D models and shows that it is possible to instantly initialize and globally register a local SLAM map without having the user perform any specific motions for initialization. However, the method relies on finding the corners of buildings in the input image by extracting vertical line segments, which is a rather vulnerable approach. In contrast, we build upon recent advances in semantic segmentation in order to identify the buildings' edges much more reliably.

In general, using segmentation in tracking is not a new idea (*e.g.*, [10, 20]), however, only specific objects are typically considered. In contrast, our method, which is trained on independent data, can handle even unseen buildings. Moreover, we provide a very efficient method to evaluate the cost function, allowing us to finely sample the pose space and avoid local minima, which are another pitfall for 3D tracking. [26] exploits segmentation in order to optimize the camera pose over a continuous 6D cost space. However, the method relies on relatively detailed models and panoramas and considers only the façades as a segment class, while we show that edges are important, especially when the field of view is narrow. [3] also uses semantic labeling, but considers landscape images. Moreover, the method only estimates the orientation of an image with respect to the model, but not its 3D position.

Building upon the success of deep learning for classification, [13] trains a CNN to directly regress the 6 DoF camera pose from a single input image. In particular, transfer learning from large scale classification to the re-localization task is leveraged. However, for each new scene, the network has to be re-trained, thus, severely limiting the practical applicability of this approach.

3 Efficient 3D Tracking based on Semantic Segmentation

Since scalability and efficiency are crucial for mobile outdoor applications, our 3D tracking system does not rely on cumbersome, pre-registered image collections. Instead, we build on existing 3D tracking approaches and exploit easily obtainable 2.5D city maps and recent advances in semantic segmentation in order to correct errors induced by the tracker such as drift.

In particular, the input frame is first forwarded to the 3D tracker (see Sec. 3.1) and to a CNN-based segmentation stage (see Sec. 3.2), giving us a rough estimate of the camera pose and a semantic segmentation of the image. This information is then used in the pose refinement stage (see Sec. 3.3), where we first sample pose hypotheses around the estimated pose prior and then compute a 3D rendering of the city model for each hypothesis. The core idea now is to find the pose hypothesis that optimally aligns the rendering with the segmentation, finally yielding a more accurate estimate of the camera pose than the prior provided by the tracker. Fig. 2 shows an overview of the approach. In the following, we describe the individual steps in more detail.

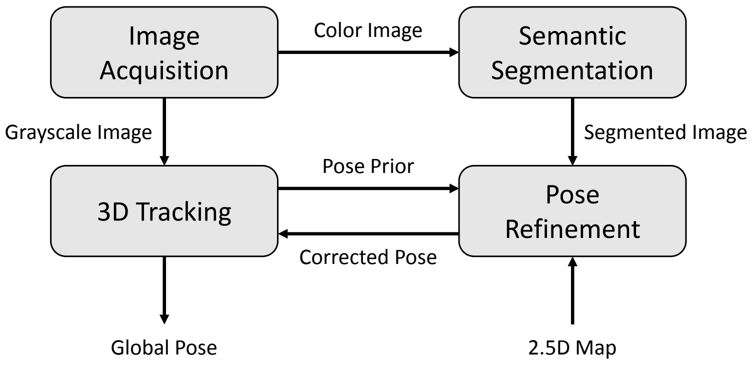


Figure 2: Correcting a 3D tracker by optimally aligning a 3D rendering of a city model estimated from a 2.5D map with a semantic segmentation of the input image.

3.1 3D Tracking

Due to the flexibility of our method, the actually used tracker is not crucial and we can use any approach that performs relative 3D tracking between consecutive frames. This, in addition, allows us also to apply different tracking methods for different application domains. In this work, we use two different trackers with different properties, a SLAM-based approach and a visual odometry algorithm. While the terms *SLAM* and *visual odometry* are sometimes used synonymously in the literature, we consider the main difference in the reconstruction of structure in the former, and the absence of explicit structure recovery in the latter.

The first tracker is a keyframe-based SLAM approach similar to PTAM [14]. Keyframe-based SLAM approaches have been widely used in the past in the domain of handheld camera motion. Although directly applicable on camera phones [15], as a drawback, they require side-ways motion and usually fail for forward motion scenarios. After covering a reasonable camera baseline between the initial keyframes, the camera pose is continuously estimated and 3D structure is recovered using fast-corners and image descriptors.

The second tracker is a lightweight visual odometry algorithm based on the work of [28]. The relative motion between consecutive keyframes is recovered by first estimating the optical flow using Lucas-Kanade [17], followed by epipolar geometry estimation through linearized Groebner solvers. These solvers have shown to give good performance in domains with restricted camera motion, such as forward vehicular motion, however, at the cost of being less accurate at turns due to the inherent numerical approximations.

After relative motion estimation, for both tracking approaches, the initial pose estimates are forwarded to the pose refinement stage, which corrects the drift of the trackers based on the 2.5D map and the semantic segmentation.

3.2 Semantic Segmentation

Recently, several deep learning based semantic segmentation methods [4, 16, 22] have been proposed, allowing for classifying large number of classes. In our case, however, we aim at segmenting only classes that are relevant to our problem, *i.e.*, classes that correspond to elements of the 2.5D map: the façades, their vertical and horizontal edges and background (roofs, ground, sky or vegetation). In particular, we use a stage-wise training procedure, where we start with a coarse network (we use FCN-32s [16]) initialized from VGG-16 [25], fine-tune it on our data, and then use the thus generated model to initialize the weights of a more fine-grained network (FCN-16s). This process is repeated in order to compute the final segmentation network having an 8 pixels prediction stride (FCN-8s).

Once trained, the output of the segmentation stage for a given RGB image I is a set of probability maps having the same resolution as I, one for each of our four classes, *i.e.*, façade (f), vertical edge (ve), horizontal edge (he) and background (bg):

$$S(I) = \{P_{\text{f}}, P_{\text{ve}}, P_{\text{he}}, P_{\text{bg}}\}\,. \tag{1}$$

As shown in Fig. 3, other items are typically classified as the classes of the elements they occlude. For example, pedestrians and cars will be ignored and classified as part of the façade and as part of the background. This happens because they are also ignored in our training images, and the segmentation method is powerful enough to classify them at run-time as belonging to the class of the elements they occlude. This is the desired behavior, as scene elements such as pedestrians or cars are not directly relevant in our approach.

Standard training sets for semantic segmentation usually do not contain the edges of façades, while these are crucial for our approach. Hence, in order to create ground truth data with relatively little effort, we recorded short video sequences in an urban environment. We manually aligned the first frame of each sequence with a 2.5D model and then applied a model- and keypoint-based 3D tracking system. With this approach, we were able to label the façades and their edges very efficiently. More precisely, we recorded 82 video sequences of average length of about 10 seconds. In this way, we obtained a training set of 10,846 images, which we augmented by horizontally mirroring each image, yielding a training set of 21,692 samples in total.

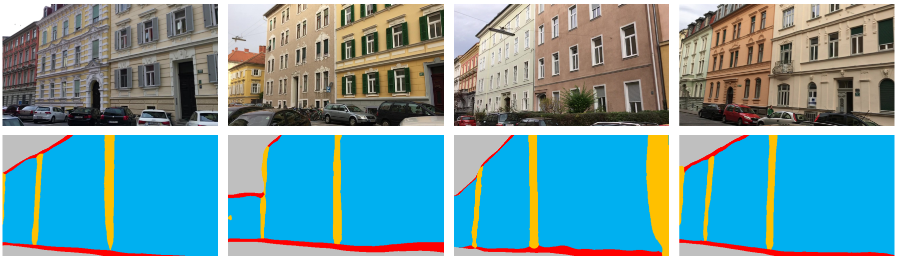


Figure 3: Input frames (top) and their semantic segmentations (bottom). The semantic segmentation is robust to occlusions by cars, pedestrians and vegetation, as can be particularly seen from the lower horizontal edges.

3.3 Pose Refinement

The input to our pose refinement stage are a coarse pose prior, a segmentation of the input frame and a 2.5D model. The core idea now is to sample pose hypotheses around the prior and to find the pose which best aligns a rendering of the model with the semantic segmentation. How well a rendering from pose **p** fits to the segmentation is evaluated via the log-likelihood

$$\mathcal{L}(\mathbf{p}) = \sum_{\mathbf{x}} \log P_{c(\mathbf{p},\mathbf{x})}(\mathbf{x}) \tag{2}$$

estimated over all image locations $\mathbf{x} = (u, v)$. Here, $c(\mathbf{p}, \mathbf{x})$ is the class at location $\mathbf{x}$ when rendering the model under pose $\mathbf{p}$; $P_c(\mathbf{x})$ is the probability for class c at location $\mathbf{x}$ given by the corresponding probability map predicted by the semantic segmentation stage in Eq. (1).

We show now that under reasonable assumptions, the sum in Eq. (2) can be computed very efficiently. We first observe that the angles between the camera and the gravity vector are usually estimated very accurately by the sensors, in contrast to other measures that can be very noisy. This allows us not only to reduce the pose search space, but also to rectify the input image such that the columns of the image correspond to vertical lines in 3D [12]. Since the same also applies for the 3D renderings, we can compute the sum over the image in Eq. (2) column by column. Let us thus rewrite $\mathcal{L}(\mathbf{p})$ as

$$\begin{aligned} \mathcal{L}(\mathbf{p}) &= \textstyle\sum_u \sum_v \log P_{c(\mathbf{p},(u,v))}(u,v) \\ &= \textstyle\sum_u \ell(u,\mathbf{p}), \end{aligned} \tag{3}$$

where u and v denote the indices of the column and the row of an image location, respectively.

To efficiently compute the sum $\ell(u, \mathbf{p})$, analogously to integral images [7, 30], we apply the concept of 'integral columns'. Let us define such integral columns for the probability map of class c as

$$\mathbb{P}_c(u,v) = \sum_{j=0}^{v-1} \log P_c(u,j), \tag{4}$$

which can be computed efficiently similarly to integral images:

$$\begin{cases} \mathbb{P}_c(u,0) = 0 \\ \mathbb{P}_c(u,v) = \mathbb{P}_c(u,v-1) + \log P_c(u,v). \end{cases} \tag{5}$$

Note that $\boldsymbol{P}_c$ only depends on the segmentation and needs only to be computed once and independently of the number of evaluated pose samples. For instance, considering a typical image column u, such as illustrated in Fig. 3 consisting of the labels background, horizontal edge, façade, horizontal edge and background, $\ell(u,\mathbf{p})$ can easily be computed by

$$\begin{aligned}\ell(u,\mathbf{p}) \quad = \quad & \boldsymbol{P}_{\text{bg}}(u, v_{bh}+1) - \boldsymbol{P}_{\text{bg}}(u,0) + \\ & \boldsymbol{P}_{\text{he}}(u, v_{hf}+1) - \boldsymbol{P}_{\text{he}}(u, v_{bh}) + \\ & \boldsymbol{P}_{\text{f}}(u, v_{fh}+1) - \boldsymbol{P}_{\text{f}}(u, v_{hf}) + \\ & \boldsymbol{P}_{\text{he}}(u, v_{hb}+1) - \boldsymbol{P}_{\text{he}}(u, v_{fh}) + \\ & \boldsymbol{P}_{\text{bg}}(u, V) - \boldsymbol{P}_{\text{bg}}(u, v_{hb}),\end{aligned} \tag{6}$$

where V is the number of rows of the segmentation, and the pose $\mathbf{p}$ is represented by the transition rows v_* between individual classes:

- v_{bh}: the row of transition between the background (sky) and the top horizontal edge
- v_{hf}: the row of transition between the top horizontal edge and the façade
- v_{fh}: the row of transition between the façade and the bottom horizontal edge
- v_{hb}: the row of transition between the bottom horizontal edge and the background (ground plane)

Note that for some columns, not all v_* are present, and that there can be another type of transition including the vertical edge class. In these cases, the sum above has to be adapted, which, however, is straightforward.

4 Experiments

In this section, we first illustrate the advantages of considering the edges of the buildings in the context of our tracking problem. Then, we demonstrate our approach for two different scenarios also building on two different tracking approaches. In this way, we cannot only show the benefits of the proposed approach, but also its generality.

4.1 Importance of Edges

Vertical and horizontal edges are usually not considered in standard semantic segmentation problems, however, they are very useful for our 3D tracking problem. As Fig. 4 shows, if the buildings are in a configuration that forms a discriminative shape in the image, segmenting only the façades can be sufficient. Even with these discriminative configurations, though, tracking can still fail. On the other hand, if the buildings are aligned in a row for example, the registration is not constrained enough without the edges, and tracking is not possible.

4.2 Motion Estimation Using SLAM for Handheld Cameras

First, we demonstrate our method on several challenging sequences from a handheld camera using SLAM. These sequences were not used for training the semantic segmentation, and, furthermore, have also been captured during a different season than the training sequences. For each frame, we use 245 pose samples around the prior provided by the SLAM tracker, taken within a square of $\pm 3m$ on the ground plane, and in a range of $[-6°; +6°]$ for the orientation around the model's up-vector. Since the movement along the z-axis and the orientation

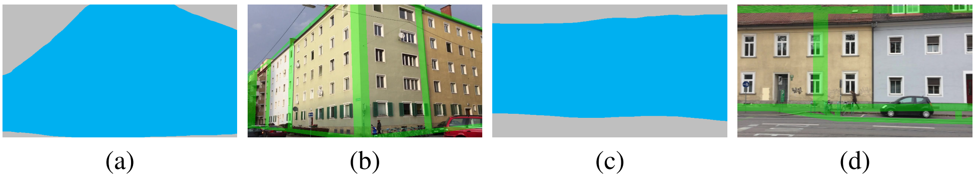


Figure 4: Façade only segmentations (a, c) and registration with our method overlaid over the input frame (b, d). The pair (a, b) shows a configuration where segmenting the façades alone is sufficient for successful tracking, while for the pair (c, d), additional constraints from the edges would be required.

around the other two rotation axes seem less affected by tracking errors and drift, we do not generate samples for these parameters. We compared our method against the unmodified camera trajectory of the SLAM system described in Sec. 3.1. As Fig. 5 shows, the SLAM-based tracking system is prone to drift, while we can register the sequences successfully, thanks to the "anchors" provided by the semantic segmentation.

In contrast to previous methods, we do not rely on pre-registered images. This makes our method not only more convenient, but also more robust: We are much less affected by illumination variations, occlusions or other changes in the scene. In particular, Sequence #5 was recorded from the same scene under different illumination conditions, once on a cloudy day, and once with bright sunlight casting shadows on the façades. Since the segmentation is robust enough to cope with such variations, our method is not affected and still able to successfully track the sequences. Table 1 shows some quantitative results in terms of rotation error around the model's up-vector and position error on the ground plane.

	Sequence #1					Sequence #3					Sequence #5				
Frame	250	350	450	550	650	200	400	600	800	1000	210	310	410	510	610
Seg rot. error [°]	0.9	0.6	4.3	1.4	0.6	0.9	1.4	1.6	0.6	1.3	0.3	3.7	2.0	2.9	3.1
SLAM rot. error [°]	3.9	3.6	4.3	4.4	5.4	2.1	4.6	4.4	3.6	1.7	3.3	6.7	8.0	8.9	9.1
Seg pos. error [m]	1.0	0.8	2.4	1.2	1.5	1.0	1.0	2.2	1.4	1.3	0.2	2.7	1.9	3.4	2.3
SLAM pos. error [m]	1.4	1.5	1.3	1.2	1.3	0.5	2.6	1.1	1.5	2.2	2.3	1.3	1.9	0.9	2.2

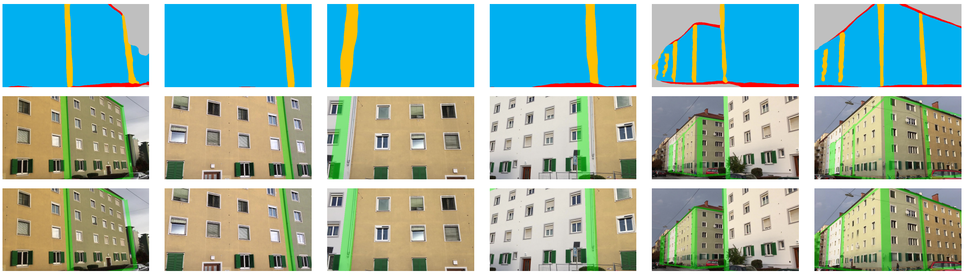
Table 1: Rotation and position errors for our method (Seg) and SLAM-based tracking (SLAM) on three sequences.

4.3 Vehicle Trajectory Estimation Using Visual Odometry

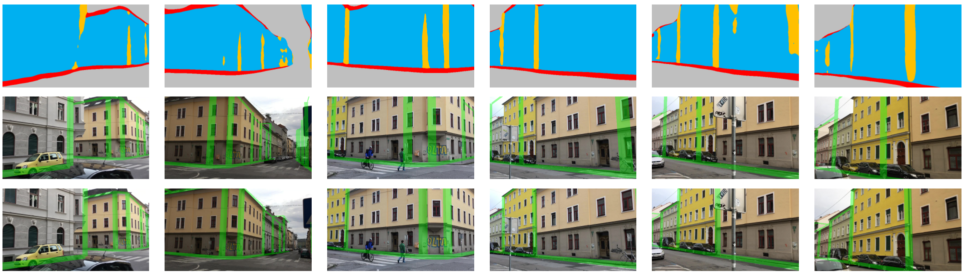
To show the generality of our approach, we run a second experiment, where we combine our pose estimation method with the visual odometry tracker described in Sec. 3.1 for a camera rig mounted on the roof of a car. The poses are sampled along the trajectory estimated by the tracker using the same settings as described above, and corrected from the best sampling result.

The results are depicted in Fig. 6, where we show the trajectory obtained by our approach (solid red line) along with the trajectory of the visual odometry tracker (dotted blue line) on a part of Sequence #18 from the KITTI dataset [□][2]. While the trajectory of the visual

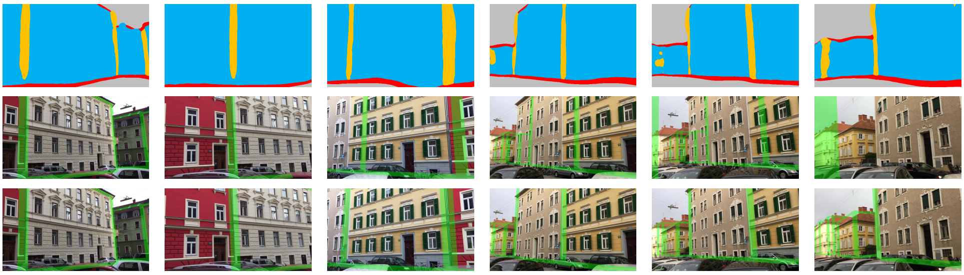
[2] Although the KITTI sequences were recorded with a stereo rig, we used the images from the left camera only.



Sequence #1, frames #43, #103, #163, #203, #343 and #723.



Sequence #3, frames #223, #483, #663, #823, #923 and #1123.



Sequence #5, frames #43, #83, #183, #243, #363 and #623.

Figure 5: 3D tracking on challenging sequences. Semantic segmentation of the input frame (top row), registration with a SLAM-based tracker overlaid over the input frame (middle row), and registration with our method overlaid over the input frame (bottom row).

odometry tracker deviates from the real motion path over time, our method removes most of the errors and aligns the trajectory very well.

We want to note explicitly that the segmentation was not tuned to this vehicular motion domain and that the combination of 2.5D model data from OpenStreetMap and the images from KITTI is very challenging for several reasons. First, the images are highly over- or underexposed, especially at the image areas of the façades. Second, the model is highly inaccurate at corners and in narrow streets, which becomes clearly observable during rendering. Third, the height of the buildings is unknown, so is the actual building altitude. In particular, following the data source, all buildings are located on a single ground plane, forcing the tracker to stay on the ground plane even if the real motion trajectory is slightly uphill or downhill (*i.e.*, $\pm 2m$ on the trajectory shown in Fig. 6).

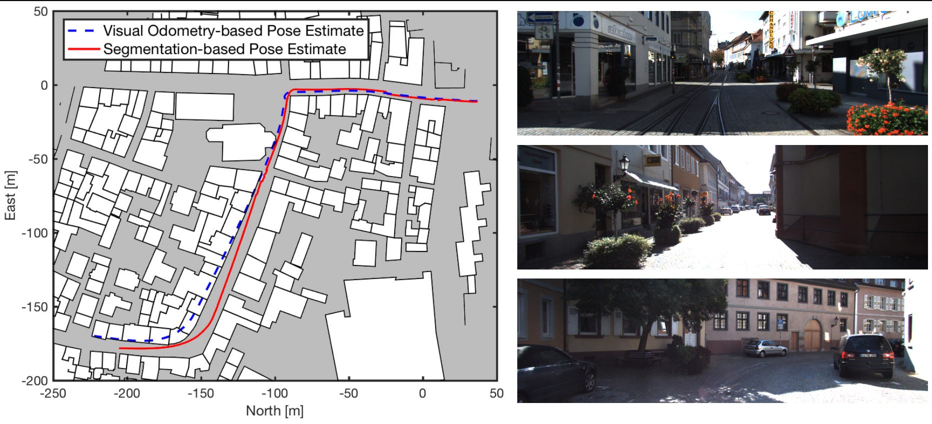

Visual Odometry-based Pose Estimate
Segmentation-based Pose Estimate
East [m]
North [m]
50
0
-50
-100
-150
-200
-250
-200
-150
-100
-50
0
50

Figure 6: Trajectory estimated via visual odometry (dotted blue line) and the corrected trajectory using our segmentation-based tracking (solid red line) (left). Illustrative examples from KITTI Sequence #18 showing highly over- or underexposed façade areas (right).

5 Conclusion

In this paper, we presented a 3D tracking method which additionally takes into account environmental information in terms of 2.5D maps. The key idea is to estimate a coarse pose using the tracker and to refine this pose later on using the environmental information. For that purpose, a 3D model is rendered from the 2.5D maps, which is then optimally aligned using a semantic segmentation of the corresponding input image. In this way, we get a robust and accurate estimation for the pose and do not need pre-registered reference images, which are cumbersome to obtain. We demonstrated that our approach is very general, as we applied it for two different scenarios also using two different tracking approaches. In particular, introducing additional semantic classes (*i.e.*, vertical and horizontal edges) has proven to be very beneficial, where the currently fast progress in semantic segmentation would allow for further improvements in the future.

Acknowledgment

This work was funded by the Christian Doppler Laboratory for Semantic 3D Computer Vision.

References

[1] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg. Wide Area Localization on Mobile Phones. In *International Symposium on Mixed and Augmented Reality*, 2009.

[2] C. Arth, C. Pirchheim, J. Ventura, D. Schmalstieg, and V. Lepetit. Instant Outdoor Localization and SLAM Initialization from 2.5D Maps. In *International Symposium on Mixed and Augmented Reality*, 2015.

[3] G. Baatz, O. Saurer, K. Köser, and M. Pollefeys. Leveraging Topographic Maps for Image to Terrain Alignment. In *3DimPVT*, 2012.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *arXiv:1511.00561*, 2015.

[5] T. Cham, A. Ciptadi, W. Tan, M. Pham, and L. Chia. Estimating Camera Pose from a Single Urban Ground-View Omnidirectional Image and a 2D Building Outline Map. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[6] H. Chu, A. Gallagher, and T. Chen. GPS Refinement and Camera Orientation Estimation from a Single Image and a 2D Map. In *IEEE Workshop on Mobile Vision*, 2014.

[7] F. Crow. Summed-Area Tables for Texture Mapping. In *ACM SIGGRAPH*, 1984.

[8] P. David and S. Ho. Orientation Descriptors for Localization in Urban Environments. In *International Conference on Intelligent Robots and Systems*, 2011.

[9] S. Feiner, B. Macintyre, T. Höllerer, and A. Webster. A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. *Personal and Ubiquitous Computing*, 1(4):208–217, 1997.

[10] J. Gall, B. Rosenhahn, and H.-P. Seidel. Drift-Free Tracking of Rigid and Articulated Objects. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[12] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[13] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *International Conference on Computer Vision*, 2015.

[14] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *International Symposium on Mixed and Augmented Reality*, 2007.

[15] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *International Symposium on Mixed and Augmented Reality*, 2009.

[16] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[17] B.D. Lucas and T Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, 1981.

[18] N. Meierhold and A. Schmich. Referencing of Images to Laser Scanner Data Using Linear Features Extracted from Digital Images and Range Images. *International Society for Photogrammetry and Remote Sensing*, XXXVIII(3/W8):164–170, 2009.

[19] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt. Scalable 6-DOF Localization on Mobile Devices. In *European Conference on Computer Vision*, 2014.

[20] A. Milan, L. Leal-Taixe, K. Schindler, and I. Reid. Joint Tracking and Segmentation of Multiple Targets. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[21] S. Ramalingam, S. Bouaziz, and P.F. Sturm. Pose Estimation Using Both Points and Lines for Geo-Localization. In *International Conference on Robotics and Automation*, 2011.

[22] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, 2015.

[23] T. Sattler, B. Leibe, and L. Kobbelt. Improving Image-Based Localization by Active Correspondence Search. In *European Conference on Computer Vision*, 2012.

[24] G. Schindler, M.A. Brown, and R. Szeliski. City-Scale Location Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2007.

[25] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014.

[26] A. Taneja, L. Ballan, and M. Pollefeys. Registration of Spherical Panoramic Images with Cadastral 3D Models. In *3DimPVT*, 2012.

[27] G. Vaca-Castano, A.R. Zamir, and M. Shah. City Scale Geo-Spatial Trajectory Estimation of a Moving Camera. In *Conference on Computer Vision and Pattern Recognition*, 2012.

[28] J. Ventura, C. Arth, and V. Lepetit. Approximated relative pose solvers for efficient camera motion estimation. In *Workshop on Computer Vision in Vehicle Technology, held in conjunction with ECCV*, 2014.

[29] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. Global Localization from Monocular SLAM on a Mobile Phone. In *IEEE Virtual Reality Conference*, 2014.

[30] P. Viola and M. Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[31] A.R. Zamir and M. Shah. Accurate Image Localization Based on Google Maps Street View. In *European Conference on Computer Vision*, 2010.