# Accurate Camera Registration in Urban Environments Using High-Level Feature Matching

Anil Armagan[1]
armagan@icg.tugraz.at
Martin Hirzer[1]
hirzer@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
Graz, Austria

[2] Laboratoire Bordelais de Recherche en Informatique
Université de Bordeaux
Bordeaux, France

**Abstract**

We propose a method for accurate camera pose estimation in urban environments from single images and 2D maps made of the surrounding buildings' outlines. Our approach bridges the gap between learning-based approaches and geometric approaches: We use recent semantic segmentation techniques for extracting the buildings' edges and the façades' normals in the images and minimal solvers [14] to compute the camera pose accurately and robustly. We propose two such minimal solvers: one based on three correspondences of buildings' corners from the image and the 2D map and another one based on two corners correspondences plus one façade correspondence. We show on a challenging dataset that, compared to recent state-of-the-art [1], this approach is both faster and more accurate.

## 1 Introduction

Knowing accurate 6DoF poses is an important task for many Augmented Reality and Autonomous Driving applications. As the accuracy of sensors such as GPS is limited, image-based localization methods [23, 25, 27] have been in focus of the Computer Vision community for a long time. However, they typically require prior knowledge such as a detailed 3D model [6, 26] or registered images [25, 27], which are cumbersome to acquire.

Recently, [1, 2, 24] showed that localization in urban environments can be done by aligning simple untextured 2.5D maps of the environment with the input image. 2.5D maps contain very useful information about outlines and heights of buildings. The main advantage of using 2.5D maps over a detailed 3D map or a point-cloud is that they are easy to build and thus already broadly available: In practice, such maps can be obtained from OpenStreetMap[1]. However, existing approaches using 2.5D maps have shortcomings: [24] requires an omnidirectional input image to converge to a good pose as it uses only a binary



[1] https://www.openstreetmap.org

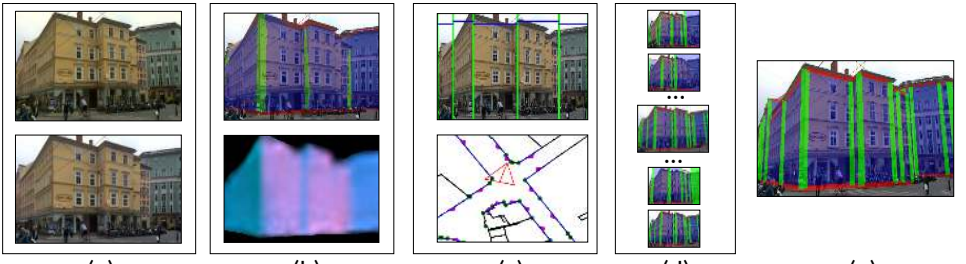



Figure 1: Overview of our approach: Input image (a, top) is vertically rectified (a, bottom). From the rectified input image, we extract façades and buildings' edges segmentation (b, top) and façades' normals (b, bottom). From the segmentation, we detect the buildings' corners and façades in the image (c, top), which are matched with corners and façades in the 2.5D map (c, bottom). From these matches, we generate possible pose hypotheses (d) using minimal solvers, and keep the pose which is most consistent with the segmentation and the normals (e).

segmentation; [2] relies on straight line segments, which are difficult to extract reliably from images; the convergence of [1] can become slow.

In this paper, we propose a method that combines the reliability of recent advanced image segmentation methods with the efficiency and accuracy of geometric pose estimation methods. More exactly, we use Deep Learning-based segmentation [4, 18, 22] to extract the buildings' edges as in [1] and in addition also their façades' normals.

We can then compute the camera pose from matches between these image features and their equivalents in the 2.5D map. To do this robustly, we consider minimal solvers [12, 13, 16] to compute camera poses from minimal sets of correspondences. Approaches based on minimal solvers were introduced to work on image features including feature points to compute geometric data such as the essential matrix between two images or a 3D pose between an object model and an image. They are typically accurate, fast, and very robust, since they can be used in a RANSAC loop.

Here, we show that we can use this strategy with high-level features extracted using very recent methods to compute the camera pose. We introduce two minimal solvers adapted to our application: Computing a 2D pose (2D translation+rotation) from 3 edge correspondences or from 2 edge correspondences plus a façade's normal correspondence. Note that we use a 2D projection model and as in [1], we can then compute a 3D pose from these 2D poses and information from the sensors. In practice, using 3 edges tends to be more accurate than using 2 edges and a normal, but it is also more likely that only 2 edges are visible rather than 3. We therefore use both minimal solvers in a RANSAC loop and keep the camera pose that provides the best likelihood computed as in [2]. Our experiments show that this is faster and more accurate than the very recently proposed approach [1].

In the remainder of the paper, we first discuss related work in Section 2. Then, Section 3 describes how we segment the image and estimate the façade normals to create point correspondences, and how we use minimal solvers to finally estimate the camera pose. We compare our method on a challenging dataset [2] in Section 4 and, finally, we discuss our findings in Section 5.

## 2 Related Work

Image-based geo-localization is a long standing problem in Computer Vision. Given one or more input images and possibly a prior location from other sensors, the goal is to find an accurate and robust localization of the camera.

**Image-Based Localization from Pre-Registered Images** Many previous works [23, 25, 27] focus on pre-registered images from a database to compute the pose of the input image. [23] uses a vocabulary tree to handle massive amount of data of 20 km of urban street-side imagery for image based localization. [27] and [25] use large collections of images from GoogleStreetView, which, however, is only sparsely sampled and not available at all in certain regions and countries. [15] uses a CNN to predict a 6 DoF camera pose directly from an input image. These approaches do not scale very well, as the number of images to be captured for each location is too high to represent the scene with enough numbers of densely sampled images. Varying conditions such as illumination, season, construction activity and many other sources of change make the task very hard.

**Image-Based Localization from Untextured Models** Another approach to image-based geolocalization is to use untextured 2D models as we do. [3] uses contour matching and refinement of sky silhouettes between a digital elevation model and mountain images, similar to [5], which verifies pose hypotheses by matching the image skyline with the model. These works assume that the skyline or the horizon are visible, however, this does not hold for many scenarios.

If available panoramic images are also useful, since they provide more constraints for registration. For instance, [10] registers panoramic images using a building façade orientation descriptor. Mobile devices have a narrow field of view, and a descriptor such as the one used by [10] is not discriminant enough in such situations. [8] also considers panoramic images and aims at detecting vertical building outlines and façade normals resulting in 2D fragments which are then matched with a 2D map. We find that [8] is close to our work, but in contrast we do not rely on a large field of view. We also consider high-level features that can be extracted reliably, which also influenced our choice of pose estimation using minimal solvers.

[9] proposed to compute a descriptor from vertical building outlines in perspective input images, which is then matched with a 2D map. However, the used matching strategy requires to use manual annotations of the input images. [21] combines 3D-2D line and points correspondences to make the registration. However, a reference image still needs to be manually annotated to match its SIFT features with the input image.

[1, 2, 24] rely on image segmentation for aligning the image and the 2.5D maps. [24] uses numerical optimization to converge to the correct pose using panoramic images. [2] estimates the orientation and translation independently and computes the orientation of the camera by estimating the vanishing points and the translation from buildings' corners. This last step is related to our approach, however, we rely on more sophisticated methods to extract the corners, we compute both the orientation and translation in a single step and we also show that we can compute and exploit the normal orientations of the façades. [1] trains neural networks to predict search directions to improve a pose estimate. Magnitude of the update is explored using a line search aiming at improving the log-likelihood. However, the search strategy used in [1] makes the method less efficient when the initial pose estimate is

far away from the correct pose. We compare our method with [1] on the dataset of [2] and show that our method is more accurate but also much faster.

**Minimal Solvers** Using minimal solvers has been proven to be an efficient and accurate way to deal with noisy data including outliers. They typically rely on geometrical image features for hypotheses generation to find a good fit applying the RANSAC algorithm [12]. RANSAC was actually introduced to be used with a P3P algorithm, which is a minimal solver. Since then, many minimal solvers have been introduced, for example to compute a camera pose from line correspondences [11], register a camera including internal parameters [7, 17] or the essential matrix [20] from point correspondences. They typically involve solving a polynomial system, which is also the case for one of our minimal solvers. However, since we rely on high-level features and we mainly deal with a 2D problem our minimal solvers remain simple and very fast.

# 3 Method Overview

Our input consists of a color image $I_{\text{input}}$ of an urban area, a prior on the camera pose, and a 2.5D map of the surrounding buildings. In practice, this prior is given by the sensors (such as compass and GPS) of the device used to capture the image. Our goal is then to find an accurate estimate of the camera pose.

We first describe how we extract high-level information from $I_{\text{input}}$ in order to match $I_{\text{input}}$ with the 2.5D map. We use simple methods to extract the buildings' corners and normals using semantic segmentation. More sophisticated methods could be developed, however, these methods were sufficient to show the effectiveness of our general approach. We then explain how to compute a camera pose from a minimal set of correspondences using minimal solvers and, finally, how we use these solvers in a RANSAC loop to robustly estimate the camera pose.

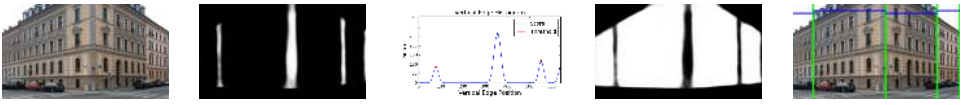
## 3.1 Extracting High-Level Features from the Input Image

Like [1, 2], we first rectify the input image $I_{\text{input}}$ using the orientation of the device with respect to the gravity vector as given by the accelerometer. These two vectors are typically accurate enough in practice. We then apply a Fully Connected Network (FCN) [18] developed for semantic segmentation to obtain probability maps $S_{\text{F}}$, $S_{\text{VE}}$, $S_{\text{HE}}$, $S_{\text{BG}}$, for the façades, vertical and horizontal edges and background (sky and ground plane), respectively.

We use these probability maps to compute the likelihood of a camera pose estimate as explained in Section 3.4 and also to extract the coordinates of the buildings' corners as explained below.

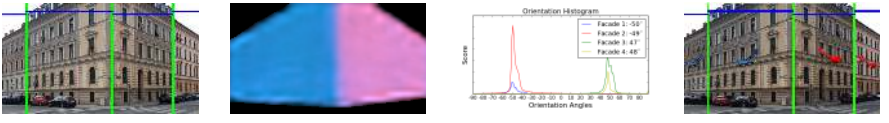
**Extracting the Buildings' corners** Our minimal solvers rely on the coordinates along the image horizontal axis of the buildings' corners. We obtain these coordinates from $S_{\text{VE}}$, the probability map for vertical edges. As shown in Fig. 2, we first compute an accumulator $A_{\text{VE}}$ for each column of $S_{\text{VE}}$:

$$A_{\text{VE}}[u] = \sum_{v=1}^{H} S_{\text{VE}}[u,v]\,, \tag{1}$$



(a) (b) (c) (d) (e)

Figure 2: Extracting building corners and façades: (a) input image after rectification; (b) probability map $S_{\text{VE}}$ for the vertical edges; (c) histogram of vertical edge probabilities and its local extrema (red), which we use as the locations for the buildings' corner; (d) probability map $S_{\text{F}}$ for the façades; (e) found corners (in green) and defined façades (in blue) shown over the rectified input image.




(a) (b) (c) (d)

Figure 3: Extracting façade normal orientations: (a) Corners and façades extracted as shown in Fig. 2; (b) surface normal estimation for the rectified input image; (c) orientation histograms for each façade; (d) final orientations assigned to each façade.

where $H$ is the number of rows of the probability map and $S_{\text{VE}}[u,v]$ is the probability of image location $[u,v]^\top$ to be on a vertical edge. We then obtain the coordinates of potential corners by extracting the local extrema of $A_{\text{VE}}$ after Gaussian smoothing. This gives us a set of column indices $U = \{u^{(i)}\}_{i=1..N_u}$ that are likely to contain buildings' corners.

**Extracting the Façades' Normals** Our second minimal solver also relies on the façades' normals. We trained a second FCN to predict the façades' normals $N(I_{\text{input}})$ using registered training images. At run-time, this gives us a normal estimate in the form of an angle in the range $[-90°;+90°]$ for each pixel of the input image. As shown in Fig. 3, we also identify the façades in the input image as intervals along the image horizontal axis between two consecutive possible $u^{(i)}$ and $u^{(i+1)}$, extracted as explained in the previous paragraph. We also consider the interval between the first column on the left of the image and the first detected corner $u^{(1)}$ and the interval between the last detected corner $u^{(N_u)}$ and the last column $W$ on the right of the image. To summarize, the extracted façades are therefore denoted $f^{(0)} = [1;u^{(1)}]$, $f^{(i)} = [u^{(i)};u^{(i+1)}]$ for $i \in [1..(N_u-1)]$, and $f^{(N_u)} = [u^{(N_u)};W]$. We denote by $F$ the set $\{f^{(i)}\}_{i=0..N_u}$.

For each façade $f^{(i)}$ $i \in [0;N_u]$, we estimate its normal using a method inspired by the SIFT descriptor to compute a dominant gradient orientation [19]: We quantize the normal angles into bins, and each pixel between columns $u^{(i)}$ and $u^{(i+1)}$ votes for the bin corresponding to its predicted normal. The votes are weighted by the probability of the pixel to lie on a façade, as predicted in $S_{\text{F}}$. We finally take the normal orientation $n^{(i)}$ for façade $f^{(i)}$ as the orientation corresponding to the bin with the largest score.

## 3.2 Minimal Solvers

We consider two minimal solvers to compute the camera pose and evaluate the best one for final pose estimate. Since we consider high-level features that can be extracted and matched from the image and the 2.5D map, these minimal solvers are relatively simple.

### 3.2.1 Using Three Corner Correspondences

Let us consider three correspondences between buildings' corners extracted from the image and buildings' corners extracted from the 2.5D map:

$$u_1 \leftrightarrow [x_1, y_1]^\top, \quad u_2 \leftrightarrow [x_2, y_2]^\top, \quad u_3 \leftrightarrow [x_3, y_3]^\top,$$

where $u_i \in U$ and $[x_i, y_i]^\top$ lie on the ground plane. We want to estimate the 2D location $[t_x, t_y]^\top$ of the camera on the ground plane and its orientation $\theta$. Let us denote by $\mathrm{P}(\mathbf{t}, \theta; \mathbf{m})$ the projection of a 2D point $\mathbf{m}$ on a column of the rectified image:

$$\mathrm{P}(\mathbf{t}, \theta; \mathbf{m}) = \frac{(\mathbf{K}(\mathbf{R}_\theta \mathbf{m} + \mathbf{t}))_0}{(\mathbf{K}(\mathbf{R}_\theta \mathbf{m} + \mathbf{t}))_1},$$

where $\mathbf{K} = \left[\begin{smallmatrix} k_u & u_0 \\ 0 & 1 \end{smallmatrix}\right]$ is the intrinsic parameters for the rectified image for the horizontal axis, $\mathbf{R}_\theta = \left[\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right]$ is a 2D rotation matrix, and $(.)_0$ and $(.)_1$ denote the first and second coordinates of a vector, respectively. From the 3 correspondences, we get three equations of form $u_i = \mathrm{P}([t_x, t_y]^\top, \theta; [x_i, y_i]^\top)$ with $i = 1, 2, 3$. Introducing $c = \cos\theta$ and $s = \sin\theta$, we can transform these 3 equations into 3 linear equations and one quadratic equation since $c^2 + s^2 = 1$. After applying Gauss-Jordan elimination to the 3 linear equations, we get

$$\begin{bmatrix} . & . & . & . \\ 0 & . & . & . \\ 0 & 0 & . & . \end{bmatrix} \begin{bmatrix} t_x \\ t_y \\ c \\ s \end{bmatrix} = \mathbf{b}. \tag{2}$$

The last equation has the form $ac + bs = d$. We can then replace $c$ in $c^2 + s^2 = 1$ by $c = (-bs + d)/a$, and solve for $s$. Once we know $s$, we can easily compute $c$, then $t_x$ and $t_y$. Since there are two possible solutions for $s$ from the quadratic equation, this gives two possible camera poses. However, the one that is in the opposite direction of $\tilde{\mathbf{p}}$ can be discarded.

### 3.2.2 Using Two Corner Correspondences and One Façade Correspondence

Let us now consider two correspondences between buildings' corners extracted from the image and buildings' corners extracted from the 2.5D map and one correspondence between a façade extracted from the image and a façade extracted from the 2.5D map:

$$u_1 \leftrightarrow [x_1, y_1]^\top, \quad u_2 \leftrightarrow [x_2, y_2]^\top, \quad f \leftrightarrow \mathcal{F},$$

where $f \in F$ and $\mathcal{F}$ is a façade in the 2.5D map. For the following derivations none of the extremities of $\mathcal{F}$ has to be visible in the image, which makes this solver interesting when only two edges are visible. Unfortunately, this solver tends to be less accurate than the previous one because of noise in the normal estimation.

$\theta$ is the angle between $n(f)$, the normal of the façade observed in the image as explained in Section 3.1, and $n(\mathcal{F})$, the normal of the façade in the 2.5D map model. It can thus be computed as

$$\theta = \arccos(n(f) \cdot n(\mathcal{F})) \,. \tag{3}$$

Once $\theta$ is known, it is easy to compute $t_x$ and $t_y$ by solving a system of two linear equations.

## 3.3 Creating Pose Hypotheses

At test time, we do not know the correct correspondences between the input image and the 2.5D map. We therefore need to consider all possible correspondence hypotheses between the features extracted from the image and the buildings in the surrounding of the pose provided by the sensors. For example, the exhaustive set of correspondences for the three corner solver has the size $N_u N_\mathcal{V}(N_u-1)(N_\mathcal{V}-1)(N_u-2)(N_\mathcal{V}-2)/3!$, where $N_u$ is the number of corners extracted from the image and $N_\mathcal{V}$ is the number of corners in the 2.5D map; typical values for $N_u$ and $N_\mathcal{V}$ are 4 and 100, respectively.

We, however, need not to consider all hypotheses from the exhaustive set, since the corners, and the façade in the case of the second solver, need to be visible to estimate a possible solution. We therefore pre-process the 2.5D map and create for each corner $\mathcal{V}$ a list $\mathcal{L}_\mathcal{V}$ containing all corners that can be visible simultaneously. At run-time, we create a list $\mathcal{W}$ of potentially visible corners given the pose prior provided by the sensors, made of the corners that are in or close to the field of view of this pose prior.

The possible hypotheses we need to consider can therefore be written as $u_1 \leftrightarrow \mathcal{V}_1$, $u_2 \leftrightarrow \mathcal{V}_2$, $u_3 \leftrightarrow \mathcal{V}_3$, with $(u_1,u_2,u_3) \in U^3$ and $u_1 \neq u_2$, $u_1 \neq u_3$, $u_2 \neq u_3$, and

$$\mathcal{V}_1 \in \mathcal{W}, \quad \mathcal{V}_2 \in \mathcal{W} \cap \mathcal{L}_{\mathcal{V}_1}, \quad \mathcal{V}_3 \in \mathcal{W} \cap \mathcal{L}_{\mathcal{V}_1} \cap \mathcal{L}_{\mathcal{V}_2} \,.$$

These constraints on $\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3$ allow us, on average, to consider only 1.5% of the hypotheses in the exhaustive set in our experiments. The same approach can of course also be used for the second minimal solver by considering a list, for each corner, of the façades that can be visible simultaneously.

Moreover, some poses computed by the solvers are clearly erroneous because they are very far for the prior pose. We therefore discard them without evaluating them. This step allows us, on average, to finally consider only 0.03% of the hypotheses in the exhaustive set in our experiments. To select the best hypothesis among the remaining ones, we rely on their log-posterior, as explained below.

## 3.4 Evaluating Hypotheses

[1] uses the log-likelihood of the image segmentation $S(I_{\text{input}})$ given the pose to evaluate the quality of the pose, considering the four classes defined in Section 3.1. Here, we also want to consider the façades' normals $N(I_{\text{input}})$ to obtain a better evaluation. We also take into account the pose prior provided by the sensors: Some images are potentially ambiguous, such as in the last row in Fig. 5, where several buildings of similar lengths are aligned with each other. Thus, only from the image segmentation, it is not possible to decide which building the camera is facing. In such a case, we would like to keep among the possible camera poses the one that is closest to the pose prior. We therefore look for the pose $\mathbf{p}$ that maximizes

$$P(\mathbf{p} \mid I_{\text{input}}) = P(\mathbf{p} \mid S(I_{\text{input}}), N(I_{\text{input}})) \propto P(S(I_{\text{input}}), N(I_{\text{input}}) \mid \mathbf{p}) P(\mathbf{p}) \,. \tag{4}$$

Let the pixels be independent given a pose, we have

$$P(S(I_{\text{input}}), N(I_{\text{input}}) \mid \mathbf{p}) = \prod_{\mathbf{x}} P(S(I_{\text{input}})[\mathbf{x}] \mid \mathbf{p}) P(N(I_{\text{input}})[\mathbf{x}] \mid \mathbf{p}),$$

where $\mathbf{x}$ takes all the possible pixel location values in the image. Let's consider $R_{\text{F}}$, $R_{\text{HE}}$, $R_{\text{VE}}$, $R_{\text{BG}}$, the binary maps for the classes façade, horizontal edge, vertical edge and background as in [1] created by rendering in 3D the 2.5D map from pose $\mathbf{p}$, and $R_N$ the rendering of the façade normals. We assume that $P(N(I_{\text{input}})[\mathbf{x}] \mid \mathbf{p})$ follows a Gaussian distribution centered on $R_N[\mathbf{x}]$ if $\mathbf{x}$ lies on a façade according to the rendering (i.e., $\mathbf{x} \in R_{\text{F}}$). If $\mathbf{x} \notin R_{\text{F}}$, the normal estimation does not provide a reliable estimate as it was trained only for façades and we use a constant value for $P(N(I_{\text{input}})[\mathbf{x}] \mid \mathbf{p})$. We also consider that $P(\mathbf{p})$ follows an isotropic Gaussian distribution centered on $\tilde{\mathbf{p}}$. Taking the logarithm of Eq. (4), we obtain the log-posterior

$$s_{\mathbf{p}} = \sum_{c \in \{\text{F,HE,VE,BG}\}} \sum_{\mathbf{x} \in R_c} \log S_c[\mathbf{x}] - \lambda_N \sum_{\mathbf{x} \in R_{\text{F}}} (N(I_{\text{input}})[\mathbf{x}] - R_{\text{F}}[\mathbf{x}])^2 - \lambda_p \|\mathbf{p} - \tilde{\mathbf{p}}\|_2^2 \qquad (5)$$

plus terms that do not depend on $\mathbf{p}$, where the parameters $\lambda_N$ and $\lambda_p$ are constant and empirically estimated from the data to match the noise in the normal prediction and the pose prediction from the sensors. We finally keep the pose estimate provided by the minimal sensors described in Section 3.2 on the hypotheses generated as explained in Section 3.3 that maximizes this log-posterior. From this 2D pose estimate, we obtain a 3D pose by using the angles w.r.t. gravity from the sensors and fixing the altitude of the camera to $1.6m$ as in [1].

# 4 Experiments

We briefly describe the segmentation and the normal estimation networks and training data used to train these networks. Then, we explain the evaluation data and discuss the results of our minimal solvers. Finally, we compare our results with those obtained by a very recent work [1].

## 4.1 Training the Networks

We use the same architecture as in [18] for training the segmentation and the normal estimation networks. These networks were trained on about 25000 images taken from 95 short video sequences with known ground truth poses for each frame. Each image was vertically rectified and fed to the networks as an input. Ground truth data for the semantic segmentation and the normal estimation for each image is computed by rendering the 2.5D model under the ground truth pose for that frame. Both networks take $I_{\text{input}}$ as an input. The segmentation network outputs the probability of belonging to a class for each pixel in the image. The normal estimation network outputs the normal for each pixel.

## 4.2 Evaluation Data

We tested our approach on the same dataset as [1]. This dataset is made of 40 images captured with an *Apple iPad Air* providing an orientation and location estimate that we use as pose prior $\tilde{\mathbf{p}}$. The images were registered from manual correspondences to get ground-truth pose to calculate relative errors for the orientation and the position. The orientation error

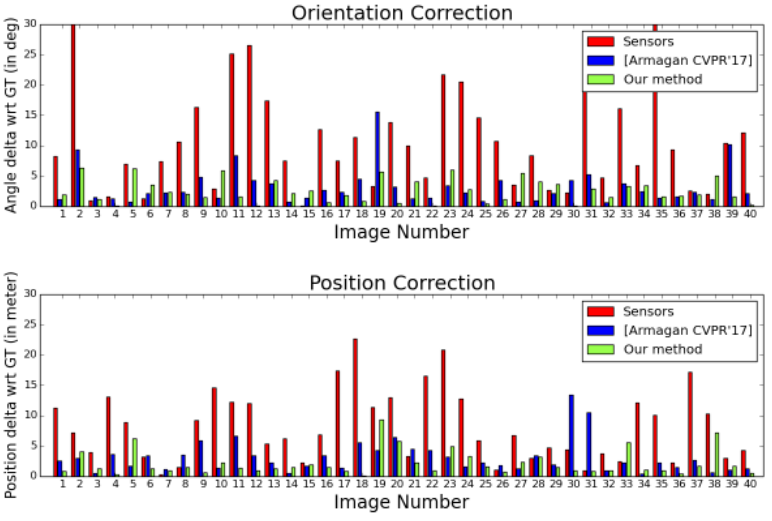



Figure 4: Orientation and location errors for the poses from the sensors, the poses obtained by [1], and the poses obtained by our method.

between the estimated orientation and the orientation of the ground-truth pose is calculated as sum of squared angular differences for all axes. The position error is calculated similarly to the orientation error. The mean orientation error of the sensors over all images varies from $0.25^\circ$ to $49^\circ$ and mean location error from $0.25m$ to $23m$ with standard deviations $9.97^\circ$ and $5.73m$, respectively.

## 4.3 Evaluation

**Accuracy.** Fig. 4 compares the poses provided by the sensors, the poses estimated by [1], and the poses recovered by our method. We apply our method on 40 test images. Our method decreases the mean location error from $13.4m$ to $2.1m \pm 2.05$ and the mean orientation error from $11.3^\circ$ to $2.51^\circ \pm 1.8$, which is significantly better than the one obtained by [1]: $3.1m \pm 2.62$ for the location error and $3.2^\circ \pm 3$, which represents a 30% improvement. Fig. 5 shows some qualitative results.

**Computation times.** The exhaustive number of possible matches in our approach can potentially become very large: For example, it is equal to 4 million if $N_u$, the number of corners in the image, is 4 and $N_\mathcal{V}$, the number of corners in the 2.5D map, is 100 in the case of the first solver. However, our heuristics described Section 3.3 keeps 59,722 hypotheses which we run the solvers for. The solvers are very fast as they require only $10\mu s$ for each hypothesis: Running the solvers therefore takes about $0.6s$. Out of these 59,722 poses, we obtain on average 1255 poses that are close enough to the pose prior to be evaluated using the log-posterior of Eq. (5). Compared to 6000 required pose evaluations of [1], this gives a speedup factor of about 4 for our approach. In total, our implementation therefore needs about 4 seconds in average for all images to provide the final pose.

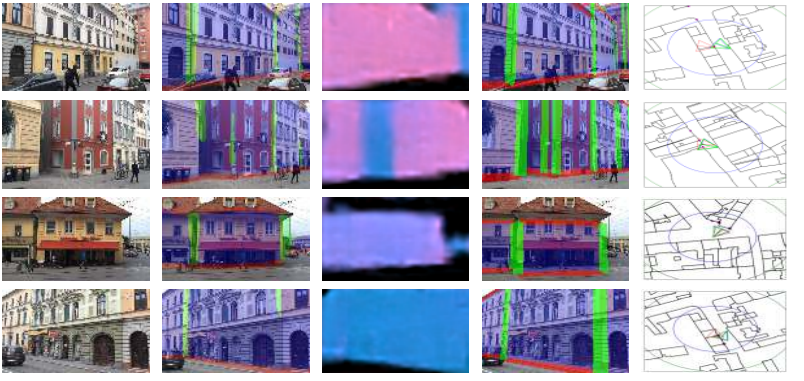



Figure 5: Some poses obtained using our method. (a) Input image, (b) segmentation, (c) normals, (d) rendering of the map from the pose estimated with our method, and (e) 2.5D maps with blue: ground truth pose, red: sensor pose, and green: the pose obtained with our method. All corners and normals within green circle, with radius $75m$, in the 2.5D maps are used in the hypothesis space and only the solutions that lies within blue circle, with radius $40m$, are kept for evaluation in the next step. The poses for the two first examples were found by the first minimal solver, the poses for the two last examples by the second minimal solver.

# 5 Conclusion

We presented an approach to camera registration where we combine the reliability of high-level features extraction using very recent techniques and the accuracy and efficiency of well established methods based on minimal solvers. We believe that this is a very general and promising direction for future research, where other high-level features are simultaneously extracted from images, matched and used to compute a pose with novel minimal solvers.

# Acknowledgment

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

# References

[1] A. Armagan, M. Hirzer, P. M. Roth, and V. Lepetit. Learning to Align Semantic Segmentation and 2.5D Maps for Geolocalization. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[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. Large Scale Visual Geo-Localization of Images in Mountainous Terrain. In *European Conference on Computer Vision*, 2012.

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

[5] M. Bansal and K. Daniilidis. Geometric Urban Geo-Localization. In *Conference on Computer Vision and Pattern Recognition*, 2014.

[6] 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 *Conference on Computer Vision and Pattern Recognition*, 2016.

[7] M. Bujnak, Z. Kukelova, and T. Pajdla. A General Solution to the P4P Problem for Camera with Unknown Focal Length. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[8] 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.

[9] 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.

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

[11] M. Dhome, M. Richetin, J. Lapreste, and G. Rives. Determination of the Attitude of 3D Objects from a Single Perspective View. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, 1989.

[12] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[13] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete Solution Classification for the Perspective-Three-Point Problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003.

[14] B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. *International Journal of Computer Vision*, 13(3):331–356, 1994.

[15] 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.

[16] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic Generator of Minimal Problem Solvers. In *European Conference on Computer Vision*, 2008.

[17] Z. Kukelova, M. Bujnak, and T. Pajdla. Real-Time Solution to the Absolute Pose Problem with Unknown Radial Distortion and Focal Length. In *International Conference on Computer Vision*, 2013.

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

[19] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[20] D. Nister. An Efficient Solution to the Five-Point Relative Pose Problem. In *Conference on Computer Vision and Pattern Recognition*, 2003.

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

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

[25] 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.

[26] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Perez, and P. H. S. Torr. Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction. In *International Conference on Robotics and Automation*, 2015.

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