Real-Time Self-Localization from Panoramic Images on Mobile Devices

Clemens Arth* Manfred Klopschitz† Gerhard Reitmayr‡ Dieter Schmalstieg§

Graz University of Technology, Austria

ABSTRACT

Self-localization in large environments is a vital task for accurately registered information visualization in outdoor Augmented Reality (AR) applications. In this work, we present a system for self-localization on mobile phones using a GPS prior and an online-generated panoramic view of the user's environment. The approach is suitable for executing entirely on current generation mobile devices, such as smartphones. Parallel execution of online incremental panorama generation and accurate 6DOF pose estimation using 3D point reconstructions allows for real-time self-localization and registration in large-scale environments. The power of our approach is demonstrated in several experimental evaluations.

Index Terms: I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—3D/stereo scene analysis I.4.8 [Image Processing And Computer Vision]: Scene Analysis—Tracking; I.5.4 [Pattern Recognition]: Applications—Computer Vision C.5.3 [Computer System Implementation]: Microcomputers—Portable devices (e.g., laptops, personal digital assistants)

1 INTRODUCTION

For visually pleasing results in Augmented Reality (AR) applications, highly accurate registration of the camera with respect to the augmented environment is necessary. Current "AR browsers" on mobile phones rely on the built-in sensors to provide localization. Depending on the type and scale of the scenario, the availability of sensory information can vary significantly. For example, GPS data is only accessible in outdoor scenarios, while Bluetooth or WiFi triangulation is only a reasonable choice for indoor environments. Besides the lack of availability, a major problem is also the lack of accuracy, which is insufficient for high quality AR applications. In outdoor scenarios the GPS positioning error might be in the range of 10 to 20 meters: consumer-grade GPS only allows coarse position estimation in 3D plus bearing if a compass is available.

For a high-quality AR the current camera pose with respect to the environment must be estimated with full 6 degrees-of-freedom (6DOF) at real-time update rates. Previous research primarily uses computer vision techniques to compute the localization. Such 6DOF localization works on mobile devices for small workspaces [23, 12], but scaling such techniques to larger environments, such as an entire city, is challenging.

1.1 Related Work

Some recent work formulates the city-scale localization as an image retrieval problem: assuming a database created from GPS-tagged images, find the closest matching images to the current view and return the GPS information or a quantity derived from it; for example, consider the work by Zhang and Kosecka [24]. A similar approach operating in near-real time is described by Schindler *et al*. [18]. The paper by Zhu *et al*. [25] computes more accurate 2D coordinates in real time, but relies on a dual stereo camera setup, while the other systems use a conventional single camera.

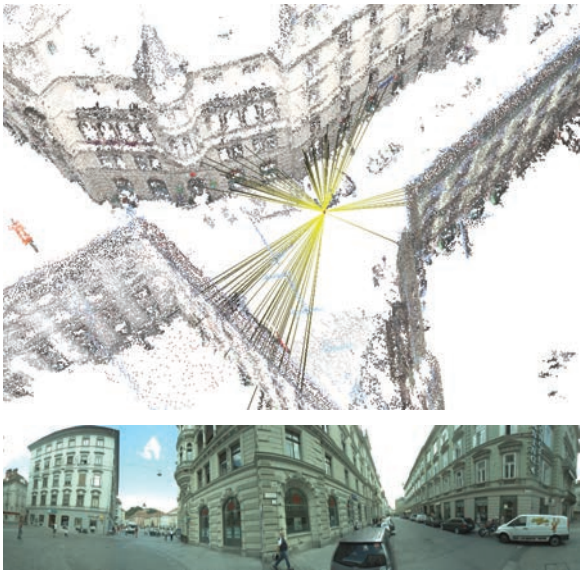


Figure 1: Localization result given the panoramic image shown at the bottom.

In contrast to these approaches, which perform essentially 2D localization, AR requires full 6DOF. Outdoor self-localization with 6DOF was shown by Reitmayr and Drummond [16], but this work relies on a textured polygonal model of the environment and does not necessarily scale to large environments. Likewise, the seminal mapping and tracking work by Klein and Murray [11] is intended for small workspaces. In contrast, Irschara *et al*. presented a 6DOF localization method for large environments using vocabulary trees and a sparse point-cloud reconstruction [9]. By inserting synthetic views during database creation, visibility of features from given viewpoints is inferred, which is later used to compress the database size and to improve the localization results. Real-time performance is achieved through the use of a GPU in order to deal with the high computational cost of the method. Li *et al*. [14] improved accuracy over this work, but they do not report real-time frame rates.

Existing 6DOF localization systems have in common that they are computationally intensive and not directly suitable for mobile devices. Recent work has therefore examined how localization can be enabled using limited computational and storage resources. Takacs *et al*. [21] perform keypoint detection and matching directly on the mobile phone. Features are clustered in a regular 2D grid and pre-fetched by proximity. Each grid element contains a set of clustered meta-features that represent the most repeatable and stable

*e-mail: arth@icg.tugraz.at
†e-mail: klopschitz@icg.tugraz.at
‡e-mail: reitmayr@icg.tugraz.at
§e-mail: schmalstieg@icg.tugraz.at

IEEE International Symposium on Mixed and Augmented Reality 2011
Science and Technolgy Proceedings
26 -29 October, Basel, Switzerland
978-1-4577-2184-7/10/$26.00 ©2011 IEEE

features of the scene. While this technique operates in real time, it does not provide full 6DOF. Klein's and Murray's [12] study showed how parallel mapping and tracking with 6DOF can be computed on a mobile device, they accepted a limitation to rather small workspaces as a trade-off, however.

This work builds on the system by Arth *et al*. [2], which uses an approach based on geo-registered 3D sparse point-cloud reconstructions of urban environments. The large amounts of sparse points are partitioned into smaller sets using the idea of *Potentially Visible Sets* [1], while feature matching is performed using vocabulary trees. An evaluation of this approach was conducted in a small-area office environment only, leaving an outdoor application of the system an open issue, which will be addressed in this paper. However, a major outcome of their evaluation was that the success rate and the accuracy of self-localization significantly grows with the widening of the cameras field of view (FOV). Dealing with this issue is one of the main goals of this paper.

1.2 Contribution

In this paper we are presenting a system that solves the self-localization task in large-scale outdoor urban scenarios giving robust and accurate results (see Figure 1). Our method computes location using a 3D point cloud reconstruction generated offline, as originally proposed by Arth *et al*. [2].

A novel technique for capturing images of the environment is used, which overcomes the problem of the narrow FOV of mobile device cameras. A high-quality and visually pleasing looking panorama of the environment is generated in an online and real-time manner. Combining the calculation of natural features with a robust pose estimation algorithm, the user's position can be estimated almost instantly.

The high degree of accuracy and robustness of our method allows for a level of augmentation quality that is considerably higher than possible with previous approaches.

Moreover, placing new augmentations online into a live view can now be combined with highly exact geo-referenced positioning, which facilitates accurate content creation and contextual assignment.

2 System Overview

In our approach we describe a system that delivers high quality self-tracking across a wide area (such as a whole city) with six degrees of freedom (6DOF) for an outdoor user operating a current generation smartphone or a similar mobile device. The system was designed for the characteristics – both advantageous and disadvantageous – of mobile devices. In particular, we were assuming the following design constraints:

- The algorithms must run directly on the mobile device, since continuous communication with a server incurs high latency for smooth operation. This implies that only moderate computational and storage resources can be used.

- Mobile device cameras typically have a rather limited FOV of 40 degrees or less. This fact makes it problematic to compute accurate localization, since often too few high-quality interest points are contained in a single image. A major goal of this work is to overcome this restriction.

- In addition to the camera, mobile devices provide additional sensors, such as GPS, compass or linear accelerometer. The quality of these sensors is only mediocre. Nonetheless, they provide additional input that we were using to solve the tracking problem.

- We assume that a 3D reconstruction of the environment is ready to be used for model-based tracking. With the rapid proliferation of 3D maps services on the Internet, availability of 3D reconstruction data is a reasonable assumption. However, it must be noted that street-side image data from map services does not cover all locations equally well. Consequently, existing solutions for wide-area localization suffer in less well covered areas, in particular, if repetitive structures are visible. As a remedy, we reduce the search space with a location prior.

- Given a location prior and a pedestrian moving at rather limited speed, current wireless wide area networks allow incremental pre-fetching of a reasonable amount of data for model-based tracking (e.g., a few tens of megabytes per hour). The resulting bandwidth requirement is equivalent to online map services on a mobile device. Our system uses this approach to download relevant portions of a pre-partitioned database on demand.

- Users typically consult a navigation service on a mobile device at decision points [6]. It is therefore realistic to expect that the user is standing still for a moment while orienting the device. We require this behavior for initializing the tracking, and assume the user's cooperation.

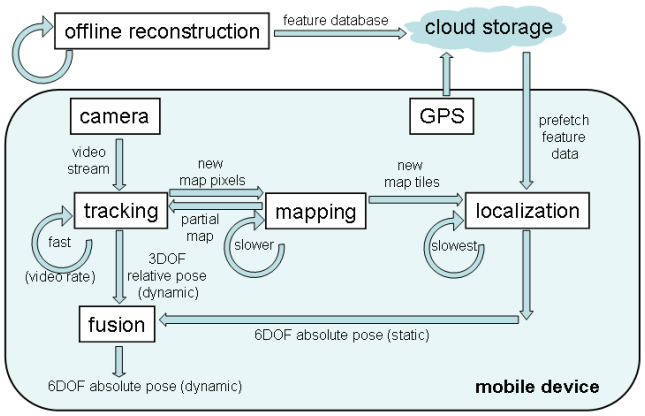


Figure 2: Overview of the localization system – three activities are performed simultaneously: orientation tracking, mapping of a panoramic image, and localization of the panoramic image relative to a large-scale 3D reconstruction. The results are combined into a 6DOF absolute pose measurement.

In the following, we describe a system that fits the above constraints (cf. Figure 2). Overall, the system is composed of an incremental orientation tracking part operating with 3DOF [22], and a model-based localization part operating with absolute 6DOF, but at a slower pace. All parts of the system execute on a mobile device simultaneously, but at different update rates, in the spirit of [11].

At startup the user is required to explore the environment through the camera's viewfinder. The video stream from the camera is fed to a feature-based orientation tracking thread, which runs at video frame rate. At the same time, the mapping thread builds a panoramic image whenever it receives previously unmapped pixels from the tracking thread.

The panoramic image is subdivided into tiles. Whenever a tile is completely covered by the mapping thread, it is forwarded to the localization thread, which compares the features found in the new tile to its database of sparse features from the 3D reconstructed model. If the localization thread successfully recovers an absolute pose, it forwards this information to the fusion step.

The fusion step combines the device's current incremental orientation with the absolute pose recovered from the panoramic map. Therefore, the fusion yields dynamic pose updates with 6DOF, albeit from a semi-static position.

Computing the localization from the partial panoramic image decouples tracking from localization effectively. This allows sustaining real-time update rates for the tracking and a smooth AR experience. At the same time the use of the partial panoramic image overcomes the disadvantages of the narrow field of view of the mobile device's camera – a user can improve the panorama until a successful localization can be performed, without having to restart the tracking.

In the following sections, we provide more details on the individual system components. Section 3 gives background on the online panorama generation. Section 4 discusses the offline reconstruction, which provides the necessary model data for the global localization. Section 5 describes the core of the new system, the localization process itself. Section 6 presents experimental results that support our claim concerning improved localization accuracy and Section 7 finally draws some conclusions.

3 Panorama Generation

High quality panorama generation is a well-known image stitching task. For a detailed overview of image stitching, the interested reader is referred to a comprehensive tutorial on this topic by Szeliski [20]. In most cases, the task of finding the geometrical relationship between individual images can be solved sufficiently well by determining image point correspondences, *e.g.*, by using the well-known SIFT algorithm, as also used in the popular Autostitch software [4, 5, 15]. The majority of panorama creation methods are working on high-resolution still images and rely on significant amounts of computational and memory resources. Since the actual process of framing individual images is still prone to camera artifact errors, these methods incorporate complex algorithms to remove seams and other visual artifacts.

Our panorama creation method works on the continuous image feed from a mobile phone camera and has to cope with the resources available on the device. Since our approach relies on an online mapping approach, only incremental techniques can be used. Many existing panorama stitching techniques, which rely on a complete set of source images for the creation of the panorama, are not applicable.

The panorama generator tracks relative orientation with 3DOF and simultaneously builds a cylindrical environment map. We are assuming that the user does not change position during panorama creation, *i.e.*, only a rotational movement is considered, while the camera stays in the center of the cylinder during the entire process of panoramic mapping (*c.f.* Figure 3). The first frame is mapped onto the cylinder surface to build the initial portion of the emerging panorama.

While the user is rotating the device, consecutive frames from the camera are processed. The algorithm computes the rotation between the current camera image and the already mapped panorama by extracting FAST corners in the live image. These features are matched with normalized cross-correlation against the tracking dataset taken from the partial panorama.

For each new frame the area in the panoramic view, which has not yet been mapped, is determined automatically. If this area is not empty, the panorama is extended with the new pixels. The map is subdivided into tiles of 64×64 pixels. Whenever a tile is entirely filled, the contained features are added to the tracking dataset. A closer description of the approach can be found in the work of Wagner *et al.* [22].

4 Reconstruction and Global Registration

The reconstruction of urban environments is a large field of research. Powerful tools are available for public use that help in fulfilling this task automatically. For example, the *bundler* software by [19] can be used to reconstruct large image collections. The task of accurately aligning the reconstructions with respect to the real world can be done semi-automatically using GPS priors from the reconstruction images.

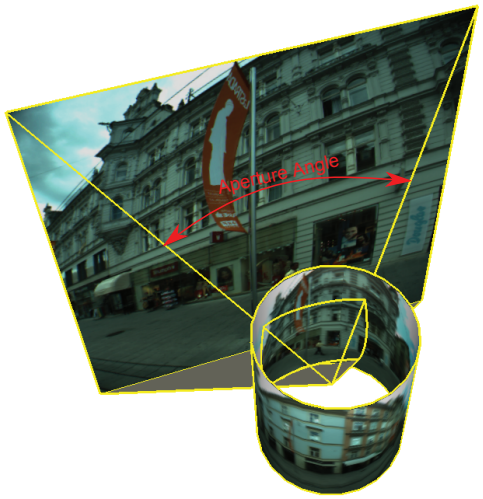


Figure 3: Panoramic Mapping of the environment onto a cylindrical surface. We use the term *angular aperture* to describe the FOV of a camera.

4.1 Structure from Motion

We are using the triplet based bottom-up reconstruction method described in [13], because we want to cover a large area with as little image data as possible, such that acquiring the source images is economical. The challenge of working with such a sparse image cover is that the incremental insertion of new views tends to be unstable. To handle this instabilities we transform the initial epipolar graph into image triplets and build the reconstruction bottom-up from the most reliable parts of the scene, as detailed in the following.

Our 3D reconstruction pipeline consists of three major steps: (i) An epipolar graph $G_{\mathscr{E}}$ is created with images as nodes and correspondences verified by epipolar geometry as edges. The feature matching process is accelerated with a bag of words approach. (ii) This graph is transformed into a graph $G_{\mathscr{T}}$ of triplet reconstructions. The nodes in this graph are all trifocal reconstructions created from $G_{\mathscr{E}}$ and are connected by overlapping views. These connections, *i.e.*, edges, of $G_{\mathscr{T}}$ are created when triplets share at least one view and pass a test for 3D point compatibility. The feature correspondences of triplets are established by using tracks from the overlapping views. (iii) These edges of $G_{\mathscr{T}}$ are then merged incrementally into reconstructions, while loop closing is handled implicitly.

4.2 Global Registration

Rather than assuming that all reconstruction data is fully available when the reconstruction process starts, our technique supports the global registration of multiple partial reconstructions that were obtained separately. This enables a more realistic workflow of acquiring a large reconstruction model and maintaining it over time.

When building a global reconstruction from several individual reconstructions, they must all be aligned in a common global coordinate system. This could be done by using a fully automatic method as presented by Kaminsky *et al.* [10], for instance. However, we chose to provide an initial rough alignment manually, and then let an algorithm refine it. Providing initial alignment can be done quickly with a suitable map tool, and prevents pathological errors resulting from too sparse image coverage and repetitive structures.

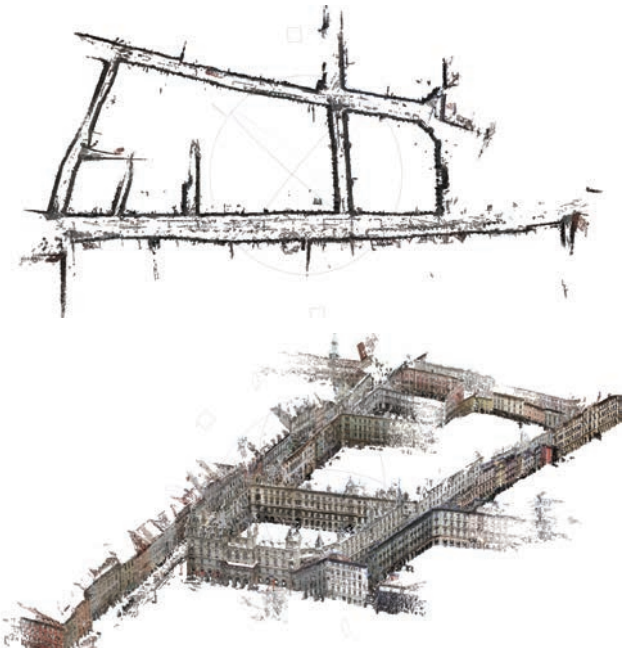


Figure 4: Sparse reconstruction data set. For better visualization a quasi-dense point cloud was created using [7].

In order to refine the alignment of two reconstructions, we calculated matches for each image in the first reconstruction to 3D features in the second reconstruction. From these matches, an initial pose estimate for the image in the first reconstruction with respect to the second reconstruction is obtained. The manual alignment is used to verify if the estimated pose is correct.

Using this approach, we can generate verified links between individual, initially not related reconstructions. We were able to improve the result of the manual alignment by using bundle-adjustment to reduce the reprojection error. A result of this approach is depicted in Figure 4.

4.3 Visibility Partitioning

Since feature database sizes grow with the covered area, it is necessary to partition the data to accommodate the storage limitations of mobile devices. We created blocks on a heuristically generated irregular grid to partition the reconstruction into smaller parts. Feature scale and estimated surface normal vectors could be added easily as additional visibility cues.

The partitioning of data blocks is on the one hand driven by visibility considerations and on the other hand by the accuracy of GPS receivers.

Most of the features in the database are generated from patches extracted from and therefore coplanar with building façades. These features can only be matched within a certain angular range[1]. Its constraint is often violated when looking down a street and viewing façades at a steep angle. In this case, which is frequent in practice, only a small area of the panorama that depicts the near streetside contains useful features, while further away features are not "visible" to the algorithm (*i.e.*, they cannot be reliably detected). We empirically determined a feature block size covering 20 meters of road direction and both sides of the road in order to yield best results.

[1] In general, this range depends on the capabilities of the feature detector. An angle smaller than $\pm 40^\circ$ seems reasonable in practice.

Figure 5: Partitioning of our feature database into individual feature blocks. Each block contains around 15000 features on average, which is around 2.2MB of memory.

An additional justification for this choice of block size is motivated by the accuracy of consumer-grade GPS receivers available in mobile devices. The accuracy of GPS estimates resides in the range of 10 to 20 meters. Given a GPS prior, the correct feature block can be determined easily. In order to avert inaccuracies, the neighboring blocks are considered as well. With this choice, the environment around an initial GPS-based position estimate is represented in a sufficiently reliable way for computing 6DOF localization.

5 Localization from Panoramic Images

For performing self-localization from panoramic images, some considerations are necessary which will be explained in the following. (Note that for the rest of the paper we will refer to the FOV also as the *angular aperture* in horizontal direction: we will use these two terms interchangeably.) In optics, the angular aperture has a slightly different meaning. For our purpose, however, we assume the use of a cylindrical model for panorama creation. In this context, the FOV of a panoramic camera directly corresponds to the arc of the cylinder circle.

The experiments in [2] indicated that increasing the field of view also increases the robustness of localization. One possible way to increase the field of view without modifying the device itself is by letting the user capture panoramic images. Figure 6 illustrates the basic idea of how we use panoramic images to increase the field of view for better image-based localization. A partial or complete panorama can be used for querying the feature database. Features extracted from the panoramic image are converted into rays and used directly as input for standard 3-point pose estimation. An alternative approach would be to use the unmodified source images that were utilized to create the panorama, and feature point tracks. These tracks can be converted to rays in space using the relative orientation of the images. However, we chose to work directly with the panoramic image for reasons of simplicity and lower storage requirements.

5.1 Three Point Pose Estimation

The classical three point perspective pose estimation ($P3P$) problem [8] is also applicable for localizing panoramic images. Figure 7 shows the geometry of the problem.

For pinhole camera models a known camera calibration means that the image measurements m_i can be converted to rays v_i and their pairwise angle $\angle(v_i, v_j)$ can be measured. In this case three

(a) (b) (c)

Figure 6: Extending the field of view illustration. (a) Relative orientation of images with the same projection center. (b) Feature points are extracted in the blended cylinder projection of the images. (c) Inlier feature matches for the panoramic image after robust pose estimation against a small office test database, the lines connect the center of projection with the matched database points.

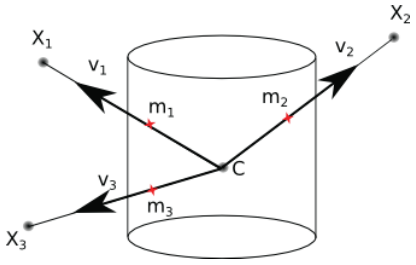


Figure 7: The geometry of the *P3P* problem for panoramic camera models is the same as for pinhole models. The law of cosines relates the unknown distances of 3D points and the camera center $x_i = \|C - X_i\|$ with the pairwise angles $\angle(v_i, v_j)$ of the image measurement rays.

known 3D points X_i and their corresponding image measurements m_i give rise to 3 pairwise angle measurements. These are sufficient to compute a finite number of solutions for the camera location and orientation. Converting our panoramic image measurements m_i to rays v_i and thus to pairwise angle measurements $\angle(v_i, v_j)$ leads to the same equation system as in the pinhole case.

For three observed 3D points X_i, the pairwise 3D point distances l_{ij} can be computed. Furthermore the angles between pairs of image measurements θ_{ij} are known from the corresponding image measurements m_i. The unknowns are the distances x_i between the center of projection C and the 3D point X_i:

$$\begin{aligned} l_{ij} &= \|X_i - X_j\| \\ \theta_{ij} &= \angle(v_i, v_j) \\ x_i &= \|C - X_i\|. \end{aligned}$$

Using the law of cosines each of the three point pairs gives one equation:

$$l_{ij}^2 = x_i^2 + x_j^2 - 2x_i x_j cos\theta_{ij}$$

This is the same polynomial system as in the case of the more commonly used pinhole camera model and can be solved with the same techniques, see for example [8]. The main difference is that in the pinhole case the camera calibration matrix K is used to convert image measurements to vectors and therefore pairwise Euclidean angle measurements, while in our case, the rays are defined by the geometry of the cylindrical projection.

5.2 Optimization

The three point pose estimation is used in a RANSAC scheme to generate hypotheses for the pose of the camera. After selecting inlier measurements and obtaining a maximal inlier set, a non-linear optimization for the pose is applied to minimize the reprojection error between all inlier measurements m_i and their corresponding 3D points X_i.

To avoid increasing error distortions towards the top and bottom of the panoramic image we defined a meaningful reprojection error that is independent of the location of the measurement m_i on the cylinder. We approximated the projection locally around the measurement direction with a pinhole model and applied a constant rotation R_i to both the measurement ray v_i and the camera pose to move the measurement ray into the optical axis. The rotation R_i is defined such as

$$R_i v_i = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T.$$

The remaining degree of freedom can be chosen arbitrarily. This rotation R_i is constant for each measurement ray v_i and therefore not subject to the optimization.

The imaging model for the corresponding 3D point X_i is then given by

$$\begin{aligned} \begin{pmatrix} u \\ v \end{pmatrix} &= \text{proj}(R_i T X_i), \quad \text{where} \\ \text{proj}(\begin{pmatrix} x & y & z \end{pmatrix}^T) &= \begin{pmatrix} \frac{x}{z} \\ \frac{y}{z} \end{pmatrix}, \end{aligned}$$

and T is the camera pose matrix representing a rigid transformation.

The optimization minimizes the sum of all squared reprojection errors as a function of the camera pose matrix T:

$$E(T) = \sum_i \| \text{proj}(R_i v_i) - \text{proj}(R_i T X_i) \|^2 = \sum_i \| \text{proj}(R_i T X_i) \|^2.$$

Note that the rotation R_i rotates the measurement into the optical axis of the local pinhole camera and therefore the projection $\text{proj}(R_i v_i) = (0\ 0)^T$. The camera pose matrix T is parameterized as an element of $SE(3)$, the group of rigid body transformations in 3D. The solution T_{min}

$$T_{min} = \arg\min_T E(T)$$

is found through iterative non-linear Gauss-Newton optimization.

6 Experiments

In the following, we present evaluation results illustrating several aspects of our work. One goal of our investigation is to gain insight into the relationship between the cameras FOV and the resulting localization accuracy. Another goal is to demonstrate the accuracy of the method and its applicability for high-quality AR.

Offline Reconstruction	
Camera	Canon EOS 5D + 20*mm* wide angle lens
# of images	4303
image resolution	5616x3744 pixels
# of sparse 3D points	816,948
total size of database	122,769 kB
# of feature blocks	55
# of sections	29
Reference Images	
Camera	Point Grey Ladybug 3
image resolution	2048x512 pixels (resized for compatibility)
# of images	204
Test Images	
Camera Phone	Nokia N900
image resolution	2048x512 pixels
average aperture angle	$\sim 225^\circ$
# of images	80

Table 1: Details from our reconstruction of the Graz city center and the panoramic test images captured.

6.1 Localization Database and Panoramic Images

As the raw material for the localization database, we collected a large set of images from the city center of Graz, Austria. A Canon EOS 5D SLR camera with a 20*mm* wide-angle lens was used, and 4303 images were captured at a resolution of 15M pixels. By using the reconstruction pipeline described in Section 4, a sparse reconstruction containing 800K feature points of the façades of several adjacent streets was created. As natural features we used a scale-space based approach similar to the work of Bay *et al*. [3]. The entire reconstruction was registered manually with respect to a global geographic coordinate system, and partitioned into 55 separate feature blocks. These blocks were combined again into 29 larger sections according to visibility considerations.

For studying our approach, we also created a set of reference panoramic images. We captured a set of 204 panoramas using a *Point Grey Ladybug 3* spherical camera. The images were captured along a walking path through the reconstructed area, and were resized to 2048x512 pixels to be compatible with our localization system. Note that we did not enforce any particular circumstances for the capturing of the reference panoramic images, they were rather resembling casual snapshots. The reference images and the images used for reconstruction were taken within a time period of about 6 weeks, while the imaging conditions were allowed to change slightly.

Since the spherical camera delivers ideal panoramic images, the results might not resemble realistic conditions for a user to capture a panorama. For this reason we additionally captured a set of 80 images using our panorama mapping application. These images were taken about one year after the acquisition of both other datasets, while a significant amount of time had passed. The capturing conditions were almost the same, *i.e*. high noon and partially cloudy sky. The images expose a high amount of clutter and noise due to exposure changes of the camera[2]. An important fact is that in almost all images only one side of the street could be mapped accurately. This results from the violated condition of pure rotation around the camera center during mapping.

Some details about the reconstruction and test images are summarized in Table 1.

6.2 Aperture Dependent Localization Performance

By using our panorama generation method, the handicap of the narrow FOV of mobile phone cameras can be managed. However, one remaining question is how the success of the localization procedure and the localization accuracy relates to the FOV of a camera in general.

We ran an exhaustive number of pose estimation tests given our set of panoramic images to measure the dependence of the localization success rate on the angular aperture. We modeled a varying FOV by choosing an arbitrary starting point along the horizontal axis in the panoramic image, and by limiting the actually visible area to a given slice on the panoramic cylinder around this starting point. In other words, only a small fraction of the panoramic image relating to a given FOV around the actual starting point is considered for pose estimation. The angular aperture was incrementally increased in steps of 5° from 30° to 360°.

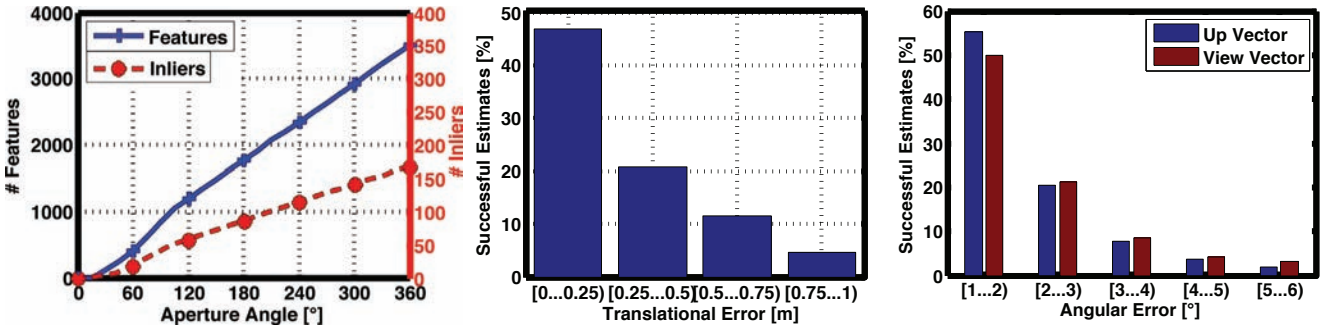
We hypothesized that in urban scenarios, the localization procedure is likely to fail if the camera with a small FOV is pointing down a street at a steep angle. The same procedure is more likely to be successful if the camera is pointing towards a façade. Consequently, the choice of the starting point is crucial for the success or failure of the pose estimation procedure, especially for small angular apertures. To verify this assumption, we repeated the random starting point selection five times, leading to a total of 68,340 tests.

In Figure 8(a), the total number of inliers and features is shown. The number of inliers is approximately 5% of the number of features detected in the entire image. In Figure 8(b) and (c), the translational and rotational errors of all successful pose estimates are depicted. Due to the robustness of the approach it is unlikely that a wrong pose estimate is computed; in ill-conditioned cases the pose estimation cannot establish successful matches and fails entirely. As ground truth, we consider the pose estimate with the most inliers calculated from a full 360° panoramic image. The translational error lies in the range of several centimeters, while the rotational error is below 5°. This indicates that the pose estimate were highly accurate if it is successful.

The success rate of our localization procedure with respect to the angular aperture is depicted in Figure 9. We measured the localization performance considering different thresholds for the translation error to accept or reject a pose as being valid. In order to measure the difference between brute-force based feature matching and our tree-based matching approach, in Figure 9 (a) and (b) the results for both approaches are depicted. The tree-based approach has an approximately 5-10% lower success rate. Since building façades expose a high amount of redundancy, the tree-based matching is more likely to establish wrong correspondences. Thus, a lower success rate is reasonable. For a small threshold the performance is almost linearly dependent on the angular aperture. This is an important result since it proves that for solving the localization task the FOV should be as wide as possible, *i.e*. a full 360° panoramic image in the ideal case. An additional result is that for a small FOV and an arbitrarily chosen starting point (corresponding to an arbitrarily camera snapshot), the localization procedure is only successful in a small number of cases. Even if the snapshot is chosen to contain parts of building façades, the localization approach is still likely to fail due to the relatively small number of matches and even smaller number of supporting inliers. With increasing aperture values all curves converge which is an indication that the pose estimates get increasingly accurate.

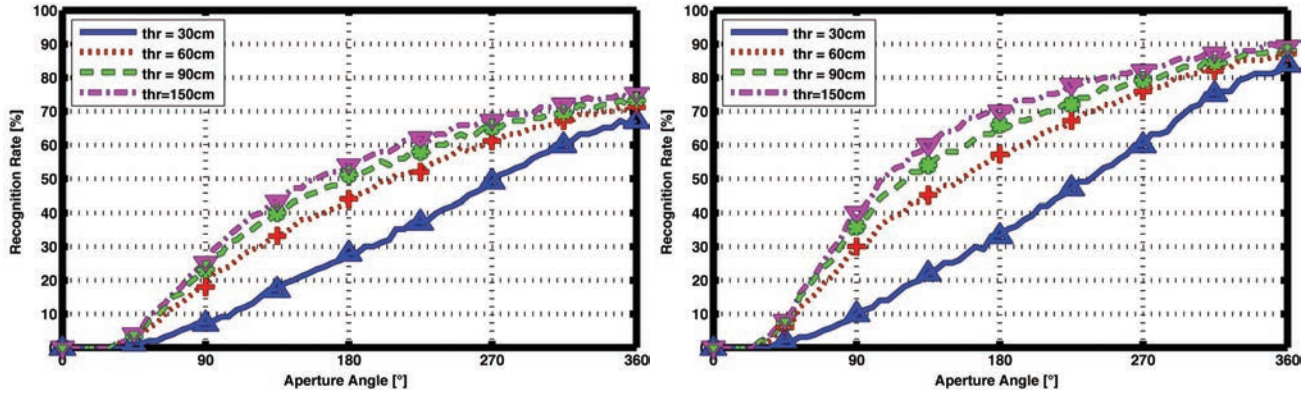
By now we only considered random starting points for capturing the panorama. However, a reasonable assumption is that the user starts capturing a panoramic snapshot while pointing the camera towards building façades intentionally, rather than somewhere else. Thus, we defined a set of starting points manually for all our reference images and conducted the previous experiment again. In Figure 10, the success rate for both matching approaches is depicted given different thresholds and our manually chosen starting points. For small aperture values the success rate is between 5 and 15% higher than for randomly chosen starting points, if the threshold on pose accuracy is relaxed (compared to Figure 9 (a) and (b) respec-

[2]Note that we simply used auto-exposure setting for acquisition.



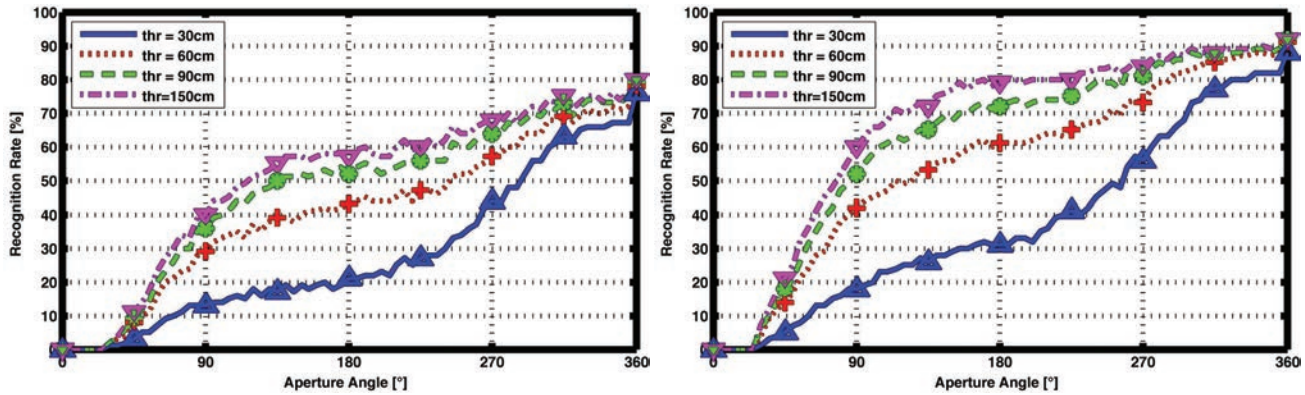
(a) (b) (c)

Figure 8: Localization Performance for a varying FOV. (a) The average number of inliers is approximately 5% of the features and increases linearly with the number of features detected in the entire image. (b) For around 84% of all successfully determined poses, the positional error is below $1m$. (c) For around 88% of all successfully determined poses, the rotational error for both the view vector and the upvector is below 5 degrees.



(a) (b)

Figure 9: Success Rate for different thresholds applied on the translation error and random starting points. In (a) the tree-based matching is depicted, while in (b) the exhaustive matching based results are shown. The results of the tree-based matching are 5-10% worse than the results of the exaustive matching approach.



(a) (b)

Figure 10: Success Rate for different thresholds applied on the translation error and manually selected starting points. The tree-based matching results are depicted in (a), the exhaustive matching based results are shown in (b). Compared to Figure 9, for small FOV higher success rates are achieved under a relaxed accuracy constraint.

tively). This result implies that successful pose estimates can be established more easily, but at the expense of a loss of accuracy. Since the features are not equally distributed in the panoramic image, the curves become saturated in the mid range of aperture values, mostly due to insufficient new information being added at these angles. For full 360° panoramic images, the results are identical to the ones achieved in the experiments before.

The entire path of panoramic images is shown in Figure 12, starting from the lower left area (red cylinders) and ending in the upper left area (cyan cylinders). In the upper right part the localization fails due to missing parts in the reconstruction. Localization again succeeds in areas where enough texture is visible (violet cylinders).

6.3 Pose Accuracy

For measuring the pose accuracy depending on the angular aperture, we ran a Monte-Carlo simulation on one sample panoramic image. Again, we simulated different angular apertures from 40° to 360° in steps of 5°. For each setting, we conducted 100,000 runs with random starting points, perturbing the set of image measurements with Gaussian noise of 2σ. This corresponds to a measurement error for features in horizontal and vertical direction of at most ± 5 pixels.

In Table 3, the resulting pose estimates are shown for different settings of the aperture angle. All poses are considered with a translational error of at most $1m$. The camera pose uncertainty follows an unimodal distribution for uniformly distributed 3D points. In our real test environment 3D points are distributed more systematically. This is well reflected in our results, the centers follow a multimodal distribution. The pose estimates cluster visibly in multiple centers, increasing values of the aperture angle decreases the variances around the centers. For a full panoramic image, all pose estimates converge into a single pose with minimal variance and the number of mixture components is reduced to one, i.e. an increasing FOV decreases the variance and the complexity of the pose uncertainty.

There are multiple reasons for this behavior. First, for a small FOV, only a small part of the environment is visible and can be used for pose estimation. A small field of view mainly affects the estimation of object distance, which, in turn, reduces the accuracy of the pose estimate in the depth dimension. A second reason for inaccurate results is that the actual view direction influences the quality of features used for estimating the pose, especially for a small FOV. Since the features are non-uniformly distributed for viewing directions towards façades, the estimation problem can be constrained better due to a higher number of matches. In contrast, for a camera pointing down a street at a steep angle, the number of features for pose estimation is considerably lower, and the pose estimation problem gets harder. Finally, due to the least squares formulation of the pose estimation algorithm, random noise present in the feature measurements gets less influential for increasing aperture angles. As a consequence, the pose estimates converge to multiple isolated positions. These images already cover large parts of the panoramic view (50-75% of the panorama). A single common estimate is maintained for full 360° panoramic images.

6.4 Runtime Estimation

To prove the usability of our approach on mobile devices, we took runtime measurements of the most important parts of our algorithm on a *Nokia N900* smartphone featuring an ARM Cortex A8 CPU with 600MHz and 1GB of RAM. The results were averaged over a localization run involving 10 different panoramic images. The results of this evaluation are given in Table 2.

The feature extraction process consumes the largest fraction of the overall runtime. Since the panoramic image is filled incrementally in an online run, the feature extraction process can be split up to run on small image patches (*i.e.*, the newly finished tile in the panorama caption process). Given a tile size of 64x64 pixels, the average time for feature extraction per tile is around 11.75 ms. As features are calculated incrementally, the time for feature matching is split up accordingly to around 0.92 ms per cell. To improve the accuracy of the pose estimate, the estimation procedure can be run multiple times as new matches are accumulated over time.

Test Results		Algorithm	Time [ms]
# of images	10	Feature extraction	3201.1 (11.75 / tile)
avg. # of features	3008	Matching	235.9 (0.92 / tile)
avg. # of matches	160	Robust pose estimation	39.0
avg. # of inliers	76	First frame (15 tiles)	< 230

Table 2: Results of our runtime estimation for different parts of our algorithm on the *Nokia N900* smartphone.

Figure 11: Panoramic images captured with our mapping approach. No exposure adjustment was used to increase the contrast or the visual appeal of the images.

Given an input image size of 320x240 pixels and a tile size of 64x64 pixels, the estimated time for the first frame being mapped is around 230 ms. This results from the maximum number of tiles finished at once (15), plus the time for matching and pose estimation. The average time spent for localization throughout all following frames can be estimated similarly by considering the number of newly finished tiles. However, this amount of time remains in the range of a few milliseconds.

6.5 Panoramas captured under Realistic Conditions

To test the performance of our algorithm on images captured under realistic conditions our localization approach was run on the second test set of 80 panoramas captured by our mapping application. Although a significant amount of time had passed between the initial reconstruction and the acquisition of the test dataset, using exhaustive feature matching our approach was successful in 51 out of 80 cases (63.75%). The tree-based matching approach was only successful in 22 of 80 cases (27.5%), however. A pose estimate was considered successful if the translational error was below $1m$ and the angular error was below 5°. These results mainly align with the results discussed in Section 6.2. The tree-based matching approach is more sensitive to changes of the environment and the increasing amount of noise respectively, which directly results in inferior performance.

We consider these results to be exceptionally good. We did not compensate for exposure changing artefacts in our panoramic images (see Figure 11). Given an aperture angle of $180 - 270°$ the results correspond to our previous observations. The fact that the localization procedure still delivers reasonable results even a year after reconstruction is remarkable, however.

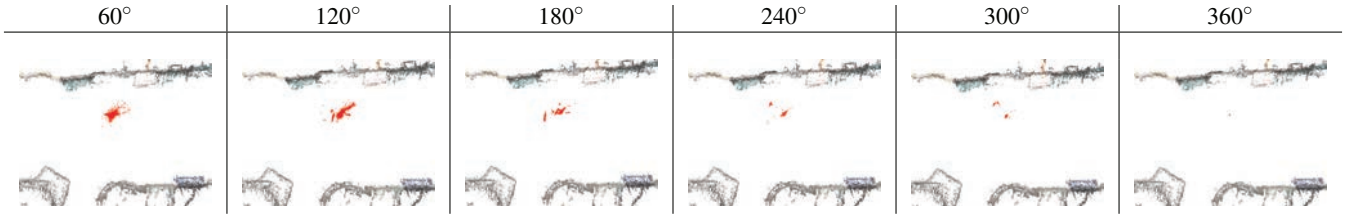


Table 3: Monte Carlo Simulation of Localization Performance for a varying FOV. With an increasing aperture angle, the pose estimates converge into smaller clusters. Finally, for a full panoramic image all pose estimates converge to a single position.

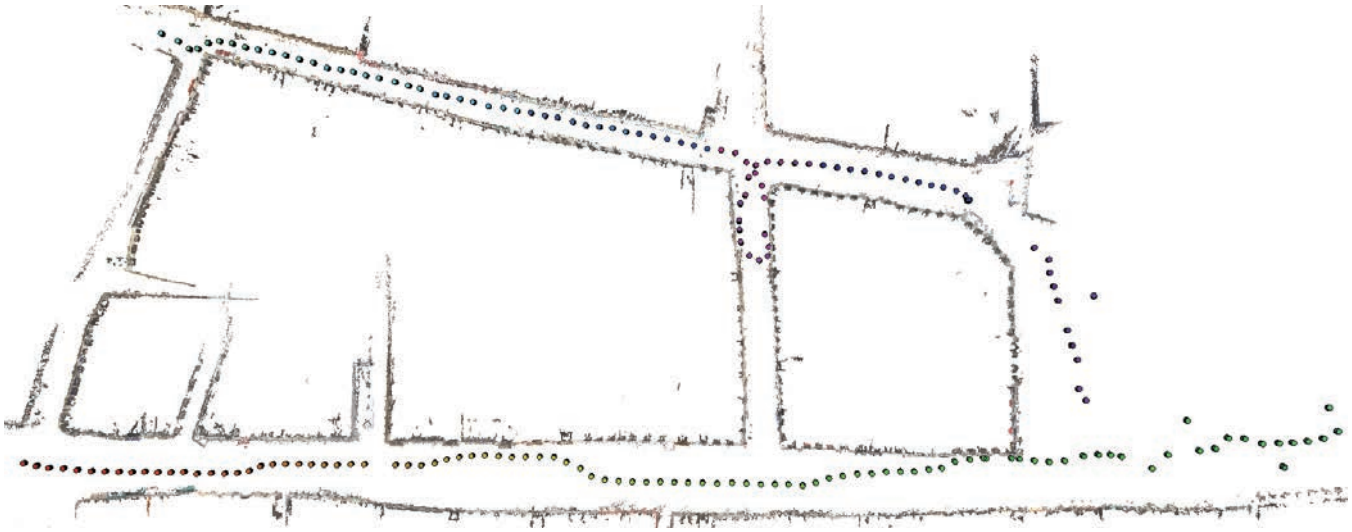


Figure 12: The path of panoramic images through the reconstruction. The camera positions are drawn as color coded cylinders, starting from the bottom left area and ending in the upper left area. Some localization results fail in the upper right area due to missing parts in our reconstruction.

6.6 Live Augmentation

To demonstrate the capability of our system to field real-time AR experiences on current mobile phone hardware, we are presenting an application scenario involving two dwarfs running through the city of Graz. Snapshots from the video sequence accompanying this paper are depicted in Figure 13. The entire video is available as supplementary material.

The screenshots show that 3D models are accurately registered with the real world environment. Some minor errors are still visible, which are mainly caused by parallax effects. These effects result from the incorrect assumption of pure rotational movement around the center of projection. This assumption cannot be hold permanently all the time, so small errors become apparent especially for close-by objects.

6.7 Discussion

From our evaluation we concluded that the pose estimation algorithm gives highly accurate results in the case of successful completion. Giving details about the reprojection error, the traditional indicator for accuracy was omitted, since we considered it as an unsuitable measure in our case. The panoramic mapping procedure introduces a certain degree of discretization error, while the assumption of pure rotational movement during mapping creates another source of inaccuracy. The pose estimation procedure internally minimizes the distance between the projection (*i.e.*, mapping) of 3D points and the corresponding measurements. This has more algorithmic relevance than direct implications on the perceived quality of the augmentation.

The number of inliers for pose estimation lies in the range of 5-10% of the features matched, and in the range of about 1-2% of the entire number of features detected. On the one hand, this means that our pose estimation algorithm is very robust against outliers. On the other hand, the use of alternative, additional feature types should be considered to make the entire localization procedure more reliable.

The real-time applicability of our approach makes it highly suitable for the use on mobile phones. Although the size of the feature database as a whole is still prohibitive, holding the entire database on external storage on the device or downloading feature blocks occasionally, is a reasonable option. The smartphone used in our evaluation is a mid-range device in terms of computational power. More recent devices with fast multi-core CPUs are more likely to be capable of dealing with the computational demands of the method presented in this paper. Note that we did not investigate the use of exhaustive matching on the mobile device. Given a reasonable amount of a few thousand features, realistically brute-force matching cannot be applied under real-time constraints at present.

In this work, we do not include any information from non-visual sensors like compasses or gyroscopes. Obviously, the use of such sensors can help with multiple steps of our approach. For example, through compass information a guided matching scheme could be employed, pre-filtering features based on the current view direction and visibility constraints. An example of combining multiple heterogeneous sensors and visual tracking has for example been described in the work of Schall *et al*. [17], for instance.

7 Conclusion

We presented a highly accurate outdoor localization system using panoramic images which can be used in real-time on current mobile devices. The most important characteristics of our approach are its

Figure 13: Five sample snapshots from the augmented video sequence. After initialization, the panoramic preview on the bottom is removed to improve the visibility of the augmentations.

high degree of accuracy and its low computational demands that make it suitable to run on off-the-shelf mobile phones. We managed to overcome the problem of narrow FOV of current cameras on mobile devices by employing a novel technique for image capturing. Our approach can be used intuitively and in a very straight forward way, as the user is simply asked to capture the environment as a visually pleasing panoramic snapshot, and our approach allows for augmentations of considerably higher quality than possible with previous outdoor approaches.

Building a localization system with low computational demands and a high degree of robustness and accuracy is a challenging task. A significant unresolved issue is the acquisition and maintenance of the vast amounts of data needed for highly accurate reconstruction and subsequent localization. Databases have to contain information covering the visual variations for different times of the day and have to capture the appearance of the environment throughout the year. Building suitable and maintainable representations is an open issue for all localization tasks employing visual sensors.

Large companies like Microsoft or Google tend to off-load the localization task to the *cloud*, mainly due to computational and maintenance reasons. However, we do not consider this as a reasonable option given the constraints of limited bandwidth and real-time operation.

For future work, we mainly consider the fusion of multiple heterogeneous sensors, such as recent micro-gyroscopes, to further enhance our localization approach. Moreover, we want to explore the use of more powerful mobile device hardware to make our approach more robust. Finally, upcoming stereo or depth cameras in mobile phones could be used to create a tight feedback loop between self-localization, image capturing, updating and maintaining the database representations used for the localization task. However, such a solution will be highly dependent on the characteristics of the actual sensor hardware.

Acknowledgements

This work was partially sponsored by the Christian Doppler Laboratory for Handheld Augmented Reality and the Austrian Science Foundation *FWF* under contract W1209-N15. Many thanks go to Lukas Gruber and to Christian Pirchheim for the interesting discussions and valuable help.

References

[1] J. M. Airey, J. H. Rohlf, and F. P. Brooks, Jr. Towards Image Realism with Interactive Update Rates in Complex Virtual Building Environments. In *Proc. Symposium on Interactive 3D Graphics*, pages 41–50, New York, NY, USA, 1990. ACM.

[2] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg. Wide Area Localization on Mobile Phones. In *ISMAR*, pages 73–82, 2009.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-Up Robust Features (SURF). *CVIU*, 110(3):346–359, June 2008.

[4] M. Brown and D. G. Lowe. Recognising Panoramas. In *ICCV*, page 1218, 2003.

[5] M. Brown and D. G. Lowe. Automatic Panoramic Image Stitching using Invariant Features. *IJCV*, 74(1):59–73, 2007.

[6] L. Chittaro and S. Burigat. Augmenting Audio Messages with visual directions in Mobile Guides: an Evaluation of three Approaches. In *Proceedings of MobileHCI*, 2005.

[7] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Clustering Views for Multi-View Stereo. http://grail.cs.washington.edu/software/cmvs.

[8] R. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. *IJCV*, 13:331–356, December 1994.

[9] A. Irschara, C. Zach, J. Frahm, and H. Bischof. From Structure-from-Motion Point Clouds to Fast Location Recognition. In *CVPR*, pages 2599–2606, 2009.

[10] R. Kaminsky, N. Snavely, S. Seitz, and R. Szeliski. Alignment of 3D Point Clouds to Overhead Images. In *IEEE Workshop on Internet Vision (held in conjunction with CVPR)*, pages 63 –70, june 2009.

[11] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *ISMAR*, pages 1–10, Washington, DC, USA, 2007. IEEE Computer Society.

[12] G. Klein and D. Murray. Parallel Tracking and Mapping on a Camera Phone. In *ISMAR*, pages 83–86, Washington, DC, USA, 2009. IEEE Computer Society.

[13] M. Klopschitz, A. Irschara, G. Reitmayr, and D. Schmalstieg. Robust Incremental Structure from Motion. In *3DPVT*, 2010.

[14] Y. Li, N. Snavely, and D. P. Huttenlocher. Location Recognition using Prioritized Feature Matching. In *ECCV*, ECCV'10, pages 791–804, Berlin, Heidelberg, 2010. Springer-Verlag.

[15] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[16] G. Reitmayr and T. W. Drummond. Going Out: Robust Model-Based Tracking for Outdoor Augmented Reality. In *ISMAR*, pages 109–118, 2006.

[17] G. Schall, A. Mulloni, and G. Reitmayr. North-centred Orientation Tracking on Mobile Phones. In *ISMAR*, Seoul, South Korea,, Oktober 2010.

[18] G. Schindler, M. Brown, and R. Szeliski. City-Scale Location Recognition. In *CVPR*, 2007.

[19] N. Snavely, S. M. Seitz, and R. Szeliski. Photo Tourism: Exploring Photo Collections in 3D. *ACM Transactions on Graphics*, 25:835–846, July 2006.

[20] R. Szeliski. Image Alignment and Stitching: A Tutorial. Technical report, MSR-TR-2004-92, Microsoft Research, 2004, 2005.

[21] G. Takacs, V. Chandrasekhar, N. Gelfand, Y. Xiong, W.-C. Chen, T. Bismpigiannis, R. Grzeszczuk, K. Pulli, and B. Girod. Outdoors Augmented Reality on Mobile Phone using Loxel-based Visual Feature Organization. In *MIR*, pages 427–434, 2008.

[22] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg. Real-Time Panoramic Mapping and Tracking on Mobile Phones. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 211 –218, March 2010.

[23] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose Tracking from Natural Features on Mobile Phones. In *ISMAR*, pages 125–134, 2008.

[24] W. Zhang and J. Kosecka. Image Based Localization in Urban Environments. In *3DPVT*, 3DPVT '06, pages 33–40, Washington, DC, USA, 2006. IEEE Computer Society.

[25] Z. Zhu, T. Oskiper, S. Samarasekera, R. Kumar, and H. Sawhney. Real-time Global Localization with a Pre-built Visual Landmark Database. In *CVPR*, pages 1 –8, june 2008.