Online Creation of Panoramic Augmented Reality Annotations on Mobile Phones

Tobias Langlotz, Daniel Wagner, Alessandro Mulloni and Dieter Schmalstieg
Graz University of Technology, Graz, Austria
{langlotz, wagner, mulloni, schmalstieg} @ icg.tugraz.at

Abstract

We present a novel approach for creating and exploring annotations in place using mobile phones. The system can be used in large-scale indoor and outdoor scenarios and offers an accurate mapping of the annotations to physical objects. The system uses a drift-free orientation tracking based on panoramic images, which can be initialized using data from a GPS sensor. Given the current position and view direction, we show how annotations can be accurately mapped to the correct objects, even in the case of varying user positions. Possible applications range from Augmented Reality browsers to pedestrian navigation.

Keywords

[4.II.XIV.I] User Interfaces, [8.V.I.II] Artificial, augmented, and virtual realities

Figure 1: Our vision-based system presents an improvement over regular compass-based annotation systems: By creating and storing panoramas, we are able to locate and visualize annotations with pixel-accuracy.

1. INTRODUCTION

The idea of superimposing geo-referenced information in place using Augmented Reality (AR) was envisioned by Spohrer in his essay on the WorldBoard [1]. This idea became recently

popular in applications like Layar[1] because mobile phones equipped with camera, compass and GPS provide an inexpensive, albeit crude platform for AR. One main concern is that GPS sensors and compasses have limited accuracy and cannot provide accurate pose information. Furthermore the update rates of these sensors are in within 1Hz. The overlays onto the live video image in a mobile phone's viewfinder are therefore roughly placed, sometimes resembling a directional hint rather than an overlay to a particular location.

In this paper, we present a novel system that tackles some limitations of current AR systems on phones. We address the limited accuracy of the compass by using vision-based orientation tracking, enabling accurate object registration. However, vision tracking can only work in relation to an image database or three-dimensional reconstruction, which must either be predetermined or constructed on the fly. We therefore employ a natural-feature mapping and tracking approach for mobile phones, which is efficient and robust enough to allow tracking with three degrees of freedom. By assuming pure rotational movements, the system creates a panoramic map from the live video on the fly (Figure 1) and simultaneously tracks from it.

This approach cannot be compared to a full 3D reconstruction of the environment. However, it has a number of distinct advantages: The application can be operated from any assumed standpoint without having to walk to special designated hotspots. It corresponds well to the way in which people explore an environment, i. e., by finding an interesting location and then looking around. The system runs at real-time rates of up to 30Hz on mobile phones and can be deployed spontaneously as it does not require any preparations.

We investigate the potential of in-situ creation of AR annotation of the environment directly on the mobile phone. Previous authoring tools were mostly bound to desktop computers, or could operate only at the accuracy of the employed mobile sensors. Our approach allows creating annotations in place and storing them in a self-descriptive way on a server, in order to allow a later re-identification. We use GPS information for efficient indexing, but identify the label positions using template matching against the panoramic map. This approach yields accurate and robust registration of annotations with the environment, even if seen from a slightly different position compared to where the annotation was created.

Figure 2 shows the workflow of our system: Peter starts by creating a panoramic map and labels objects of interest. The annotations, Peter's GPS location and a description of the visual appearance of the annotated area are transmitted to a server. Later, Mary wants to retrieve annotations authored by Peter. Her phone notifies her when she is close to locations annotated by Peter, using GPS information. A map view allows her to reach a spot close to where Peter was when he created the annotations. After pointing up, the phone uses a newly created panorama for efficiently matching Peter's annotations to the environment. As soon as the area supporting a particular annotation is re-detected, Mary's phone displays the corresponding annotation. Mary is now able to create additional annotations herself.

[1] http://layar.com/

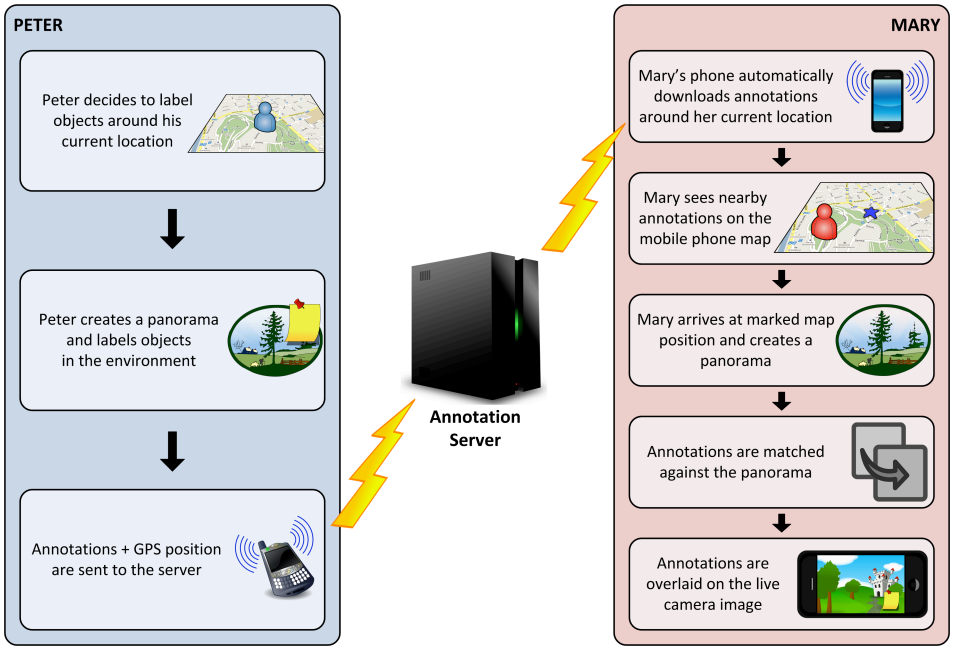

Figure 2: The workflow of the panoramic AR annotation system involves two users – Peter creates annotations and at a later time Mary browses through these annotations.

2. RELATED WORK

Previous work can be divided into two areas: work on annotation authoring and work describing approaches for real-time tracking of mobile devices in large-scale environments.

Current AR applications on mobile phones augment the world with annotations bound to physicals objects by using the current GPS position and orientation information from an accelerometer and a digital compass. This kind of applications resembles Spohrer's WorldBoard [1], a geo-referenced information display using AR on a handheld device. The Touring Machine [2] was the first prototypical mobile AR system to demonstrate the advantage of augmenting information over physical objects by using a backpack AR system and a head-mounted display. Later work by Kooper and MacIntyre [3] showed how the display of geo-registered information with AR could be linked to online information sources such as the web. However, these prototypes achieved acceptable registration performance using bulky equipment or stationary infrastructure and were not intended for daily use.

Montiel and Davison created a visual compass [4] based on single-camera simultaneous localization and mapping (SLAM). They used an extended Kalman filter formulation of the tracking problem to compute orientation from dynamically acquired landmark features. Since their approach creates a sparse 3D reconstruction of the environment, the system is not restricted to rotations only. Reitmayr et al. [5] describe a SLAM system for sharing dynamically generated annotations with a remote observer. Klein and Murray recently introduced a variant of SLAM-

based tracking, which can run on mobile phones [6]. However, all these SLAM systems work only in small areas and the maps are not designed to store annotations permanently.

Only a few related works focus on creating annotations directly in an AR view. Early work on in-situ authoring targeted placing virtual objects in the real scene, supporting users through triangulation from different views [7]. Rekimoto et al. [8] presented Augmentable Reality, which allowed annotating an environment prepared with barcode markers referring to contextual information. More recently, Wither et al. [9] showed how to solve the problem of missing depth information of annotations by using aerial maps. Later Wither et al. [10] used a laser range finder to automatically calculate the depth information from a given position and orientation, allowing better label placement.

Envisor [11] shows a vision-based approach for orientation tracking. It tracks the camera orientation in real-time and simultaneously creates an environment map by calculating the optical flow between successive frames. These measurements are refined with more computationally expensive landmark tracking to avoid the drift introduced by frame-to-frame feature matching. While the results of this approach are similar to our approach, Envisor cannot run on phones due to the high computational cost of the method requiring extensive GPU processing to run in real-time.

3. PANORAMIC MAPPING AND TRACKING

Our system is based on a simultaneous mapping and tracking approach, operating on cylindrical panoramic images. The algorithm is conceptually comparable to SLAM, however we do not create a 3D map of the environment, but limit the map to a 2D panorama. This simplification works well for users who are standing still while turning the phone. Our method creates a cylindrical map of the environment on the fly and simultaneously uses this map for tracking the camera orientation. Our approach requires ~15ms per frame on a smartphone and allows for applications running at interactive frame rates (30Hz). In this section we briefly introduce our method, which is described in full detail elsewhere [12].

3.1. Panoramic Mapping

Our panoramic mapping method assumes that the camera undergoes only rotational motion. Under this constraint, there are no parallax effects and the environment can be mapped onto a closed 2D surface. Although a perfect rotation-only motion is unlikely for a handheld camera, our method can tolerate enough error for casual operation. Especially outdoors, where distances are usually large compared to the translational movements of the mobile phone, mapping errors tend to be negligible. A detailed analysis on the effect of violating the pure rotation requirement with respect to the distance of the mapped objects is given in [11].

We use a cylindrical mapping model, which does not suffer from discontinuities as cubic environment maps do. When the mapping process starts, the first camera frame is completely projected into the map and serves as a starting point for tracking. We assume that the phone is held with zero pitch and roll during the first frame. For mobile phones with a linear

accelerometer, roll and pitch can be automatically inferred to initialize the application. For subsequent camera frames, projecting only those parts of the image that have not yet been mapped, preserves compute cycles. The map is organized in tiles and a tile is only considered for tracking after it is completely filled with pixels. Pixel-accurate book-keeping for the mapping is done using a run-length-encoded coverage mask. It allows us to quickly sort out map pixels that do not require updating. As a result, every pixel of the map is written only once and only few (usually <1000) pixels are mapped per frame, which guarantees high frame rates.

3.2. Panoramic Tracking

The camera orientation, needed for the mapping process, is tracked with an efficient and accurate method using the map as it is built.

We apply an active search procedure based on a constant velocity motion model to track keypoints from one frame to the next. Keypoints in the map are compared against their counterparts in the camera image. We subdivide the map into 32x8 cells. Once a cell has been completely mapped, keypoints for that cell are extracted using a corner detector. The tracking approach is generally drift-free, but errors in the mapping process still accumulate so that the map is not 100% accurate: As a result, our method allows loop closing, which can minimize errors that accumulate over a full 360° horizontal rotation.

The motion model provides a rough estimate for the camera orientation in the next camera frame, which is then refined based on normalized cross correlation (NCC) template matching: Based on the estimated orientation, keypoints from the map are projected into the camera image and an 8x8 pixel wide patch around the projected keypoints is matched using NCC.

As long as tracking succeeds, we store the camera frames at quarter resolution together with their estimated pose. When tracking fails, we compare the current camera image against all stored keyframes and take the pose of the best match as the coarse guess to re-initialize the tracking process. In practice, tracking quickly restarts within 45ms (on ASUS P565) as soon as the user points the camera into a previously observed direction.

4. ANNOTATION DETECTION AND TRACKING

In our work on panoramic mapping and tracking [12], the created map was saved together with 2D map locations of annotations, so that another user could reload the map and explore the annotations. Since we did not store the keyframes together with the map, we used PhonySIFT [13] to register the loaded map with the camera images. This approach requires the user to be very close - ranging from 20cm to 100cm, depending on the distance of the object in the camera frame - to where the map was originally created. If the standpoint deviates too much, the map may not be registered, or the annotations may not be correctly aligned to the physical objects. The sensitivity to the standpoint, together with the elevated memory requirements for storing and transmitting a complete map, is a major limitation of this previous approach.

In this paper, we present a novel method for storing, detecting and tracking annotations that targets these problems. This method does not rely on previously created maps for tracking, as

one can always build a new map on the fly. Rather than describing the annotations by a position in a previously created map, we store them in a self-descriptive way suitable for robust re-detection in a new map.

Using SIFT (Scale-Invariant Feature Transform) or similar descriptors to store keypoints surrounding the annotation in the camera image is not a suitable solution. While we have an efficient SIFT-based solution [13] for tracking on mobile phones, building a support search structure on the whole 2048x512 pixel panorama and maintaining it every time a cell gets updated, is currently too slow to run in real-time on a phone. Furthermore, due to the size of the support area SIFT is known to be problematic when matching small objects (<50x50 pixels).

In our approach, points of interest are not matched against the camera image, but against the panoramic map. This allows us to search in regions that have been seen, but are not in the camera view anymore. Hence, we can decouple object detection from the current camera view and run it in the background.

These special restrictions also make several of SIFT's features unnecessary: Since the map is always expected to be more or less upright, rotation invariance is not required. Maps are recreated at similar locations so that scale invariance is also not required. Instead, we propose identifying label positions using normalized cross correlation, which shares only the brightness and contrast invariance with the SIFT descriptor.

We describe a single annotation by using 9 templates in a 3x3 configuration (see Figure 3). Each template has a size of 16x16 pixels, as we empirically determined that this configuration is giving the best detection rate and using small templates, which are independently located in the map, makes them more robust to small scale and rotation changes rather than using one large template: The 3x3 templates are not required to perfectly reproduce the arrangement in the original map. Instead they must only roughly form the original arrangement with a tolerance of 5 pixels in any direction (see Figure 3). This makes the annotation detection robust to small non-uniform scaling such as when an object is seen from a slightly different angle.

Compared to a complete map, which requires about 1 megabyte of storage, each annotation requires only ~2 kilobytes. Furthermore, we can easily combine annotations from different users by loading all annotations created in a close proximity. Finally, detecting independent annotations is generally more robust to slight offsets in the user position than matching a complete map from a different location.

4.1. Walsh transforms for faster template matching

In a typical scenario we must match dozens of annotations, described by image templates, against a map with a size of 2048x512 pixels. Matching a large number of templates against an image of this size is slow. We therefore use Walsh transforms [14] as a pre-check, since they have several appealing properties:

- Walsh transforms are fast to execute, and using integral images [15] makes the execution speed independent of the templates' size.
- Matching multiple templates against the same image scales well, because the same Walsh transform of the image can be matched against an arbitrary number of transformed templates.

Integral images [15] are memory intensive, but an even more severe issue is that integral images are difficult to create for incomplete images such as the panoramic map, which is subject to change by successively adding new pixels to the image: whenever the map is updated, most of the integral image would have to be updated too.

To solve this we subdivide the map into tiles: A tile is only considered for matching against annotation templates once it is completely filled. We can then build an integral image for each tile with enough overlap to the right and bottom, so that we can place the templates at every possible pixel location inside that tile, performing a dense search. For each pixel location, we create eight Walsh transforms, which are then compared against the Walsh transforms of the annotations' templates.

Walsh transforms are fast to compute, but only give a lower bound of the matching error. Hence, for good matches we also apply NCC as an additional check. For each template, we keep the 10 best matching locations together with their NCC score. If at least four of the nine templates have been matched, we check if they form a 3x3 arrangement in the map (Figure 3). Our tests show that 4 out of 9 provide a good balance between false positives and failed detections. Once this check succeeds, we mark the annotation as detected.

Figure 3: (Left) The support area of an annotation is described using a 3x3 grid of templates encoded using a Walsh transform. (Right) Matching the templates from a slightly different camera perspective.

4.2. Real-time scheduling of annotation detection

As mentioned before, the annotation templates are matched against the map rather than the camera image. We can therefore schedule the matching to guarantee a desired frame rate: Each finished tile is not checked immediately, but put into a queue instead. During each frame, the system schedules only as much work from the queue as allowed by the given time budget. Since

the operations are simple and their timings are predictable, we can easily limit the workload so that the time budget is not exceeded.

Our system can therefore run at constant speed on any phone that is able to perform real-time panoramic mapping and tracking. On fast phones, annotations are detected quickly, whereas on slower phones it takes longer. We benchmarked the detection on a current smartphone (ASUS P565). Matching one cell against 12 annotations takes ~54ms. Targeting a frame rate of 20Hz (50ms per frame) allows scheduling ~10ms for detection of every frame.

Once all available map cells have been searched for annotations, surplus compute time can be used to search at different scales for increased scale invariance, until new map tiles are completed.

5. APPLICATIONS

5.1. Browsing annotations

We apply the described technique in an AR browser application. Initially the user sees the environment in an aerial map (Figure 4, left). This 2D map view shows the current GPS position of the user and highlights nearby annotations. The user can employ the map to navigate the environment and to find annotated spots. Once the user walks closer to an annotated spot, the application downloads the annotation data from a server. All annotations in immediate proximity, as indicated by their GPS tags, are considered, so that inaccuracies in the GPS data do not affect the experience.

Figure 4: (Left) 2D map overview showing nearby annotations. (Right) First-person view of the annotated panorama.

If the user decides to browse the annotations, they can switch to a first-person view (see Figure 4, right), where they see the current camera image. This automatically triggers the system to start the panoramic mapping and tracking. As the user rotates the phone in order to explore the environment, the application finds the correct position of the surrounding annotations as the best

match of the stored template in the newly created panoramic map. Once a template is successfully matched, the view is updated by displaying the annotation at the correct position. Furthermore, the preview map is updated by displaying the position of the annotation in the miniaturized version of the panoramic image. This assists the user in finding the annotations from their current position.

5.2. Creating annotations

The process for creating new annotations is similar to exploring annotations. The user moves to a position where they want to create an annotation. Switching to the first-person view prompts the application to start tracking the orientation and creates a panoramic image of the current environment. The user can now create annotations by simply touching the display at the desired position and entering a textual description or a voice annotation. A self-contained annotation is stored as a 48x48 pixel sub-image centered on the chosen point in the panorama image. This sub-image is later used for template matching. Besides the annotation itself, the sub-image is the only information required for finding the annotation anchor point again.

Sharing of annotations is supported by a server-side database to which annotations are uploaded. The server application uses standard Web software and protocols (Apache/Tomcat, MySQL). For better indexing, we tag each annotation with the current GPS coordinate and user information before uploading it to the server. GPS is used to file the submitted annotations according to a spatial index, so that queries for information in the vicinity of a particular standpoint can be executed efficiently. Information on the user's identity and optionally provided tags allow efficient filtering of large amounts of annotations.

6. RESULTS

6.1. Preliminary feedback on usability

We used our application prototype for a first exploratory field trial in order to receive user feedback. We recruited eight users (3female/5male, age 22-34yr) with no previous knowledge of AR. Two sets of six annotations were prepared for each user in an urban outdoor environment, with the labeled objects being 10-200m away from the user. In each set of six annotations, two annotations were created from a slightly different position (5m away). The first set was made one day before the user trials, while the second set was always created anew at a maximum of 30 minutes before the user tests. This meant that all the users had to browse one set, which did not match the current environment conditions (e.g. lighting, shadows) and a second set, which matched the current environment conditions, but was different for each user. During the test, we asked the users to identify the labeled objects and to label some new objects. After the trial we used a semi-structured interview to collect user feedback.

The test showed that 43 out of 48 annotations could be detected (89.53%) in the case where annotations were recorded under similar environment conditions. The detection rate was lower

for annotations created under a different environment condition (27 out of 48, 56.25%). There were no false positive detections during any of the tests.

All users were able to annotate the given objects. While users found the display to be very small for clearly identifying objects, nobody perceived this as a real problem: Most of the users' gaze often switched from the display to the real environment for verification. To cope with the small-screen constraint, users proposed adjusting the size of annotation points and adding a video zoom function for annotating very small objects.

All users agreed that the tracking was stable and fast. They experienced occasional loss of tracking, which was signified by a question mark on the screen, but were consistently able to recover quickly by pointing the camera towards a previously a visited region (see 3.2). Six out of the eight users stated that as they became more familiar with the applications, they could avoid tracking problems. This was also noticeable as users progressed from a very stiff to a more relaxed posture over time. Users reported that they mostly broke the panorama-based orientation tracking by moving too fast or by pointing the phone to the sky.

For the detection of the annotations, six out of the eight users had the feeling that they could improve detection by exploring the neighborhood of an annotation, while the remaining users said that the label was at the correct position right after having the position in the camera frame. None of them noticed any drifting or jumping in the labels once detected.

Finally the user interface generally received positive comments, in particular the panorama preview function, which was employed by all but one user for orientation and to identify unexplored regions. Five of the eight users also took advantage of the preview to locate known positions for reinitializing the tracking. All users agreed that the browsing operation is easy, enjoyable and the tracking works robustly.

6.2. Matching Results

The results of the preliminary user test showed significant differences in matching quality. A further analysis showed that changing light conditions through the day caused most of the failed matches.

New or missing shadows can largely change the appearance of objects. Figure 5a shows an example of a failed match: In the morning the wall behind the sign is half dark and half bright, whereas in the afternoon the whole background has similar brightness and is shadowed by a tree. Figure 5c shows how object structures (visible as self-shadows) vanish once the object itself is in the shadow. Figure 5d shows artifacts due to parallax caused by a different user location. In contrast, the house in Figure 5b matched throughout the whole day, because the major structures where always well visible.

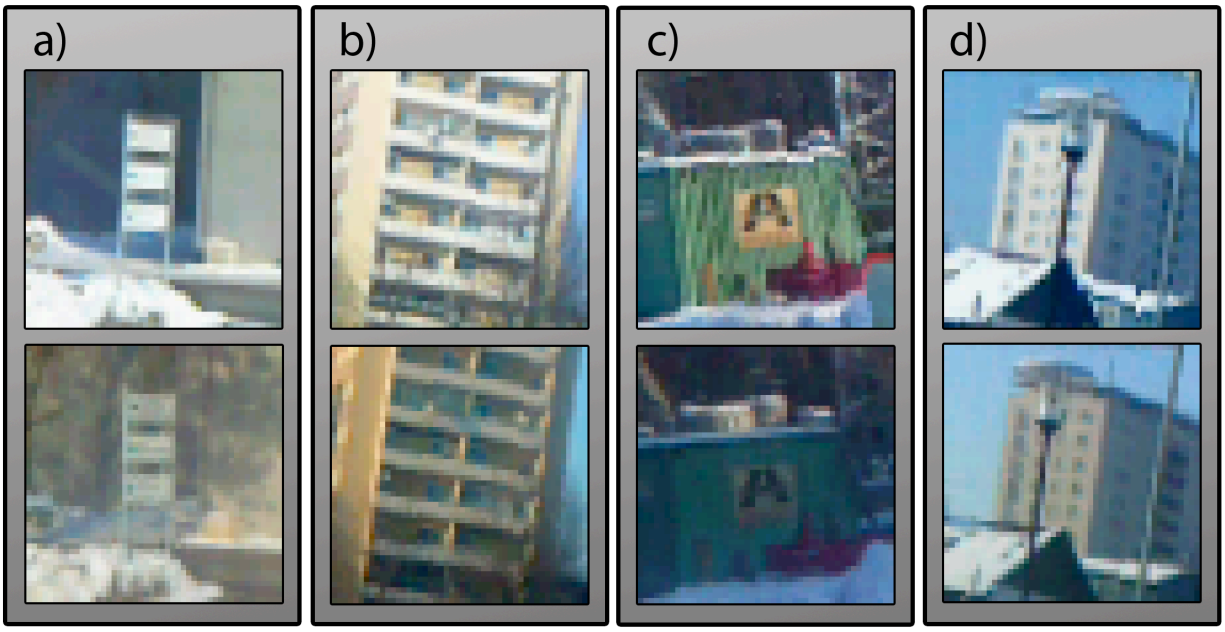

Figure 5: Appearances of points of interest in maps taken at different time of day or different location.

7. CONCLUSION AND FUTURE WORK

We showed how panoramic images can be built in real-time on a mobile phone and used to annotate the physical environment with an AR interface. Annotations can be tagged with the user's GPS position, stored on a server and retrieved by other users. A first study showed that this operation is easy and that the tracking works robustly.

Our current detection approach is optimized for no false positives and speed. In the future we plan to improve the re-detection rate under environmental changes: We will look into describing annotations as sets of data collected from multiple panoramas with various lighting conditions. It will be interesting to evaluate how many panoramic sources are required for robust results. Finally, the availability of compass and accelerometer will allow us to position annotations – albeit with reduced accuracy – even in case of failure of our vision-based matching. Finally, sensor based tracking together with geo-referenced annotations would allow us to improve the template matching process as we can narrow down the search area within the panorama.

8. ACKNOWLEDGMENTS

This work was sponsored by the Christian Doppler Laboratory for Handheld Augmented Reality, the Austrian Science Fund FWF under contract W1209-N15 and the EU Integrated Project FP6-IST-27571.

9. REFERENCES

[1] J.C. Spohrer, "Information in places", *IBM SYSTEMS JOURNAL*, vol. 38, 1999, pp. 602-628.

[2] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster, "A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment", *Personal Technologies*, vol. 1, 1997, pp. 208-217.

[3] R. Kooper and B. Macintyre, "Browsing the Real-World Wide Web: Maintaining Awareness of Virtual Information in an AR Information Space", *World Wide Web Internet And Web Information Systems*, 2001.

[4] J. Montiel and A. Davison, "A visual compass based on SLAM", *Proceedings of the 2006 IEEE International Conference on Robotics and Automation, ICRA 2006*, IEEE, 2006, pp. 1917-1922.

[5] G. Reitmayr, E. Eade and T. Drummond, "Semi-automatic Annotations in Unknown Environments", *Proceedings IEEE Symposium on Mixed and Augmented Reality*, 2007, pp. 67-70.

[6] G. Klein and D. Murray, "Parallel Tracking and Mapping on a Camera Phone", *Proceedings of IEEE Symposium on Mixed and Augmented Reality*, 2009.

[7] Y. Baillot, D. Brown D. and S. Julier, "Authoring of physical models using mobile computers", *Proceedings of ISWC 2001*, pages 39–46, 2001.

[8] J. Rekimoto, Y. Ayatsuka, and K. Hayashi, "Augment-able Reality: Situated Communication through Physical and Digital Spaces", *Proceedings of the 2nd IEEE International Symposium on Wearable Computer*, 1998, p. 68.

[9] J. Wither, S. DiVerdi, and T. Höllerer, "Using aerial photographs for improved mobile AR annotation", *IEEE/ACM International Symposium on Mixed and Augmented Reality, 2006. ISMAR 2006*, 2006, pp. 159–162.

[10] J. Wither, C. Coffin, J. Ventura, and T. Höllerer, "Fast annotation and modeling with a single-point laser range finder", *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, IEEE, 2008, pp. 65-68.

[11] S. DiVerdi, J. Wither, and T. Höllerer, "Envisor: Online Environment Map Construction for Mixed Reality", *Proceedings of the IEEE Virtual Reality Conference 2008*, IEEE, 2008, pp. 19-26.

[12] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg, "Realtime Panoramic Mapping and Tracking on Mobile Phones", *Accepted for the IEEE Virtual Reality Conference 2010 (VR 2010)*.

[13] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg, "Pose Tracking from Natural Features on Mobile Phones", *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2008, pp. 125-134.

[14] P. Nillius, J.O. Eklundh, "Fast block matching with normalized cross-correlation using Walsh transforms", report, ISRN KTH/NA/P--02/11--SE, Sept. 2002.

[15] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features", *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001, pp. 511-518.