Robust Relocalization and Its Evaluation for Online Environment Map Construction

Sehwan Kim, *Member*, *IEEE*, Christopher Coffin, *Member*, *IEEE*, and Tobias Höllerer, *Member*, *IEEE*

Abstract—The acquisition of surround-view panoramas using a single hand-held or head-worn camera relies on robust real-time camera orientation tracking and relocalization. This paper presents robust methodology and evaluation for camera orientation relocalization, using virtual keyframes for online environment map construction. In the case of tracking loss, incoming camera frames are matched against known-orientation keyframes to re-estimate camera orientation. Instead of solely using real keyframes from incoming video, the proposed approach employs virtual keyframes which are distributed strategically within completed portions of an environment map. To improve tracking speed, we introduce a new variant of our system which carries out relocalization only when tracking fails and uses inexpensive image-patch descriptors. We compare different system variants using three evaluation methods to show that the proposed system is useful in a practical sense. To improve relocalization robustness against lighting changes in indoor and outdoor environments, we propose a new approach based on illumination normalization and saturated area removal. We examine the performance of our solution over several indoor and outdoor video sequences, evaluating relocalization rates based on ground truth from a pan-tilt unit.

Index Terms—Environment map, virtual keyframe, vision-based tracking, camera pose relocalization, illumination changes.

1 Introduction

Augmented Reality (AR) makes the physical world a part of the user interface experience and has the potential to play a significant role in enhancing the mobile and wearable computing paradigm. Anywhere Augmentation is a conceptual extension of Mobile Augmented Reality (MAR) and its aim is to link location-specific computing services with the physical world, making them readily and directly available in any situation and location without relying on prepared environments or off-line environment models [1]. Real-time visual tracking is used to estimate the pose of a camera relative to its surroundings. However, tracking failure is inevitable, and thus, an efficient and accurate camera pose relocalization is needed to provide a user with reliable tracking results.

In this paper, we focus on the problem of camera orientation tracking with the goal of achieving real-time environment map acquisition, for which there are several motivating applications. Environment maps are useful as immersive representations of physical locations, e.g., as a backdrop in a tele-collaboration system, or in first person interfaces such as QuickTime VR or Google Street View experiences. Remote presence applications [2] can use environment maps that are updated in real-time as a simple way of representing and referring to dynamic remote environments. In AR systems, environment maps can be used to represent the light distribution around a single position in a compact image-based format. As such, they can be used for more seamless integration of virtual objects into the physical scene by supplying realistic image-based lighting for virtual geometry [3], [4].

Additionally, we wish to allow for a low-computational cost orientation tracking solution. While our solution does not currently extend to six degrees of freedom (6 DoF) position and orientation tracking, our work could be used as a component to a larger tracking system. So that, when users transition from movement, as tracked by some other system, to examining their surroundings, we can switch to our orientation tracking solution, Envisor [5], as a reliable low-cost method for orientation tracking while constructing environment maps at the same time. If we assume that scene geometry is relatively planar or sufficiently far away from a user, virtual keyframes can also be used for simultaneously generating and tracking over large-scale planar photo-panoramas from a panning camera in real-time. Another possible scenario is to capture an environment map of a scene while subsequently updating interesting sections with live video. For an important event, such as a sporting game or concert, using a mobile phone with a small field of view, we could capture a panorama containing the audience and the surrounding buildings, and then capture live data being performed on the stage.

Ideally, we would like to allow a panorama, which has been captured by one user, perhaps at a famous site, to be used for generating keyframes for a later user, thereby providing more robust tracking. Our work on improving the robustness of virtual keyframes with respect to lighting takes an important step in that direction.

Up to now, a wide variety of tracking technologies, employing various sensors, have been investigated for AR systems. In general, relocalization has been performed by using corner-like features or training a classifier with features. Subsampled images are also adopted as descriptors for relocalization. The methods are, however, only able

• *The authors are with the UCSB Four Eyes Lab., Department of Computer Science, Harold Frank Hall (Eng I Bldg), Santa Barbara, CA 93106-5110. E-mail: {skim, ccoffin, holl}@cs.ucsb.edu.*

Manuscript received 16 Mar. 2010; revised 24 June 2010; accepted 29 June 2010; published online 29 Oct. 2010.
Recommended for acceptance by D. Thalmann and B. Lok.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2010-03-0070.
Digital Object Identifier no. 10.1109/TVCG.2010.243.

to generate keyframes from video frames which have previously been acquired by a camera.

In this paper, we propose a novel camera pose relocalization approach for orientation tracking using virtual keyframes which are created from an environment map in the form of a cube map instead of incoming video frames, allowing for keyframes at novel view orientations and making it possible to provide a wider relocalization range.

The ideal number of virtual keyframes is influenced by both system performance and convenience of use. Generating virtual keyframes is a relatively quick process, however generating a large number of them can still affect the speed of the system, mostly due to the cost of relocalization. At the same time, a small number of keyframes may cause a user some difficulty in finding one of the keyframe locations for camera relocalization. Note that the number of keyframes needed for easy use is dependent not only on the distance between neighboring keyframe locations on a unit sphere, but also the number of virtual keyframes in the roll direction [6]. The total number of virtual keyframes we need is 2,000 which are (400 virtual keyframe locations) $\times$ (5 directions in roll), which is reasonable for real-time frame rates on a common laptop. See [7] for a detailed derivation of the number of keyframes.

We present the following contributions in this paper. First, we suggest how to generate virtual keyframes from an environment map on the fly using a fragment shader [8] and present a pose relocalization method for a camera which has temporarily lost tracking (due to occlusion, motion blur, lack of visual features, or other reasons) using the generated virtual keyframes. Second, we improve the system speed by accomplishing the relocalization only in the case of tracking failures and using inexpensive image-patch descriptors. Third, we compare the performance of the orientation tracking among four different systems including the original Envisor [5] based on ground truth data which were recorded using a pan tilt unit [9] and based on expert evaluations. Finally, we reduce the effects of different lighting conditions for a robust relocalization in indoor and outdoor environments.

The rest of this paper is structured as follows: After a discussion of related work in Section 2, our relocalization approach using virtual keyframes is discussed in Section 3. In Section 4, we introduce how to improve the tracking speed by using image-patch descriptors and by detecting tracking failures. Then, in Section 5, we evaluate four different variants of Envisor for the purpose of tracking performance comparison based on three different evaluation methods. In Section 6, we discuss how to improve relocalization robustness with respect to lighting changes. Finally, in Section 7, we present our conclusions and ideas for future work.

2 Related Work

In recent years, there has been steady research on camera pose relocalization. Pupilli and Calway propose a system which deals with short tracking failures in a monocular SLAM context [10]. This is accomplished based on multiple hypotheses with a particle filter. The system is bootstrapped by a set of known 3D points to build up an initial particle distribution. As tracking progresses, new 3D points are introduced by identifying salient points and estimating their depths by triangulation of the camera particles. Se et al. [11] focus on a global approach to relocalization for a moving robot. They use SIFT visual landmarks in unmodified environments to find matches to image features with map features, and build a 3D map of the environment by tracking the landmarks over time [12]. These 3D landmarks are used to find the pose using RANSAC or a hough transform [13]. Reitmayr and Drummond deal with camera pose relocalization using keyframes which are saved during tracking [14]. In the case of failure, they try to find a best-matching keyframe in the stored selection of older frames with the current video frame. They propose a statistical test to detect when the edge-based tracking system fails. They recover the camera from the proximity of a finite section of the previously traversed path. Williams et al. [15] carry out relocalization by using a randomized list classifier to establish feature correspondences in an image. Then, these correspondences are quickly detected for robust pose relocalization using RANSAC when tracking fails. On the other hand, Klein and Murray employ subsampled blurry images as descriptors instead of extracting some form of interest features from keyframes [16], [17]. When tracking is lost, they subsample an incoming video frame, and apply a Gaussian blur. Then, they compare the incoming video frame with all of the virtual keyframes and find the keyframe with the smallest sum-squared difference. The final camera rotation is estimated using Efficient Second-order Method (ESM) Visual Tracking [18] and best-fit 3D camera rotation estimation using virtual sample points. Most of the methods are, however, based on features in previously captured images, and therefore keyframes are limited to previous camera poses. In this paper, we propose a method of generating virtual keyframes for relocalization even at camera orientations not present in the live video. We also discuss how these virtual keyframes provide a wider range of relocalization.

Previous real-time environment mapping approaches (e.g., [19], [5]) rely on restarting the mapping process when tracking is lost. The ideas of Klein and Murray's PTAM approach [16] and their keyframe-based relocalization method [17] are also used by Wagner et al. [20] in their recent cell-phone based environment mapping software. Whereas they generate keyframes from incoming video frames, we generate virtual keyframes from an environment map, enabling more flexible relocalization. Users are freed from having to move their camera back to the exact same locations visited before. Furthermore, with a physical keyframe approach, camera pose cannot be recovered if the roll of the camera is different from that of the saved keyframe. Using the virtual keyframe concept, we can easily generate a virtual keyframe with any camera roll at any keyframe location.

Irschara et al. [21] propose a fast location recognition approach based on structure-from-motion point clouds. They address a more difficult problem, resulting in much higher computational and data requirements. Their construction of synthetic views is similar to our virtual keyframe synthesis, however their system requires a set of 3D points while ours takes advantage of the limited domain of orientation tracking to provide a faster and more accurate construction of keyframes from novel angles. Additionally, our work on improving lighting invariance is generally

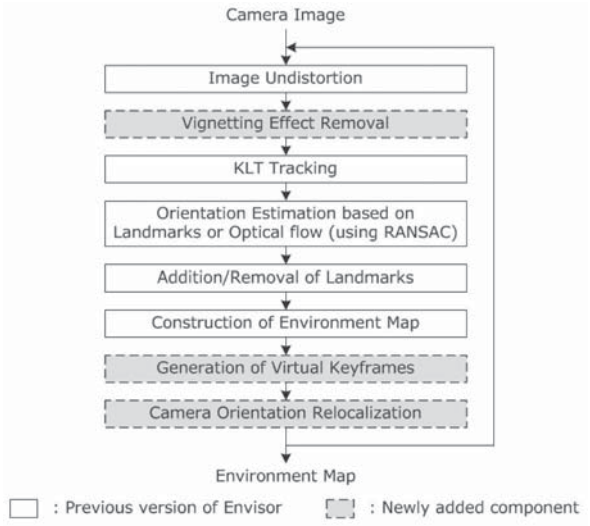

Fig. 1. An overall flow diagram.

applicable to keyframe-based approaches including theirs as well as 6 DoF tracking systems such as PTAM [16].

There has been active research on illumination changes for visual tracking, but we are not aware of research specifically addressing relocalization under illumination changes. Pilet et al. [22] deal with photo-realistic augmentation on 3D surfaces under complex illumination conditions. However, they only deal with ambient diffuse lighting and assume correspondences between model and input images. In [23] and [24], even though they consider specularities/saturation, and arbitrary illumination changes, a reference image needs to be compared with input images. Tian et al. [25] propose a more elaborate shadow detection technique which can generate a nonshadow image. However, its computational complexity is not appropriate for a real-time system.

3 Virtual Keyframes for Relocalization

We are working toward acquisition of surround-view panoramas using a single camera with real-time camera orientation tracking which is robust to fast smooth and abrupt orientation changes and deviations from the observer's position. In effect, we want to make the acquisition of surround view panoramas using a hand-held or head-worn camera as fast, convenient, and robust as possible. We propose methodology for camera orientation relocalization using virtual keyframes for online environment map construction. We also provide an online procedure for generating them and recovering camera orientations based on them. The flow diagram for the procedure is depicted in Fig. 1.

Let us compare the proposed virtual keyframe to a conventional keyframe. As shown in Fig. 2a, in the conventional methods, a keyframe is generated from an actual camera image. However, with virtual keyframes, we can create keyframes independently from camera paths. In addition, we can easily generate any number of virtual keyframes at any arbitrary yaw, pitch, or roll as indicated in Fig. 2b and Fig. 2c.

3.1 Environment Map Construction

Our proposed relocalization method relies on the generation of an environment map. Our system uses the

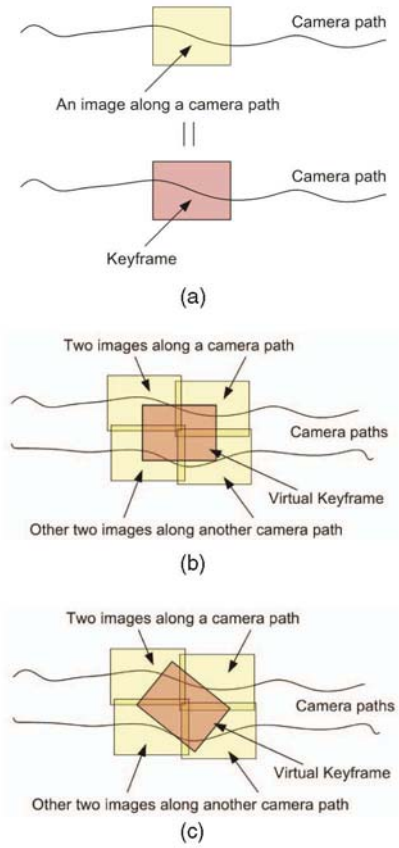

Fig. 2. Using the conventional method a keyframe (a) is generated only along the camera path. In the proposed method virtual keyframes (b), (c) are generated independently from the camera paths.

environment map generation method present in the original version of Envisor, which is a system for online construction of environment maps in new locations [5], [26], [27]. The tracking mechanism consists of two phases, one of which is frame to frame relative rotation tracking using Shi and Tomasi's feature detector and Lucas and Kanade's optical flow algorithm [28], [29]. The second phase is absolute orientation tracking using SURF-based landmarks [30]. We also require an additional correction step before the texture mapping is performed, in order to eliminate distortion effects such as vignetting. Currently, using auto-exposure may create artifacts in the environment map and resulting virtual keyframes. We therefore suggest that all exposure settings be locked. We are working to address this issue.

3.2 Online Virtual Keyframe Generation

In our approach, the number and locations of virtual keyframes are determined in advance. But, in order to generate a new virtual keyframe, we have to wait until the whole area of each virtual keyframe is fully filled with valid pixel values. Instead of checking each pixel within a virtual keyframe, we use only the four corners of each virtual keyframe to decide whether a planned virtual keyframe can now be created. We calculate its four corners by projecting the view frustum of the camera orientation corresponding to the virtual keyframe onto the cube map. Because we already have an ideal constellation of virtual keyframes on the sphere with yaw and pitch information for each location, we can calculate the position of all four corners

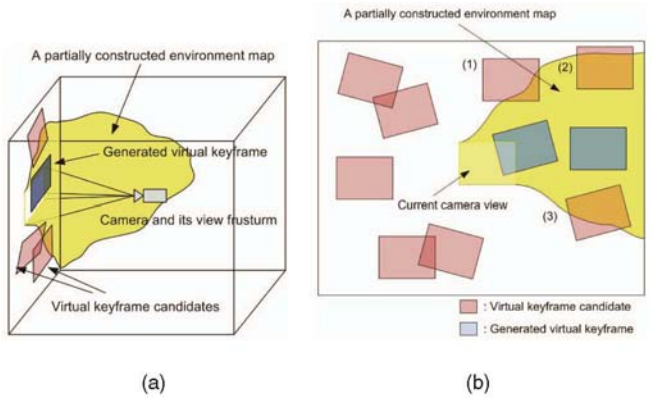

Fig. 3. Generation of virtual keyframes (a) illustration of constructing an environment map by rotating a camera (b) an instance of generating virtual keyframes. The virtual keyframe candidates (1), (2), and (3) are examples of candidates whose valid corners are one, two, and three, respectively.

for all of the virtual keyframes. If the four corners become valid, the corresponding virtual keyframe image is immediately generated from the environment map and saved into the database. A fragment shader is used to detect four corners for all of the virtual keyframes in real time. While sampling only the four corners of the keyframe location may cause some keyframes to be generated without being completely filled with video information, it is generally sufficient to check only the four corners. Users specifically constructing a panorama will generally avoid holes during the construction, and AR users generally have a more restricted motion path not involving large loops. Occasional invalid keyframes can be tolerated as long as there is a sufficient number of functional keyframes.

In Fig. 3, we show a conceptual diagram of how to generate virtual keyframes using their four corners. Fig. 3a illustrates that a camera rotates around the camera center, constructing an environment map, and Fig. 3b depicts an instance of generating virtual keyframes. The figure shows that there are several virtual keyframe candidates whose locations have been determined beforehand. If the four corners of each candidate become valid, it is changed to a valid virtual keyframe. The virtual keyframe candidates (1), (2) and (3) in Fig. 3b are examples of candidates whose valid corners are one, two, and three, respectively.

Following [17], we subsample the virtual keyframe down to 40×30 pixels, apply a Gaussian blur of $\sigma = 2.5$ pixels, and subtract the mean image intensity. This zero mean "small blurry image" is saved as the virtual keyframe's descriptor and used to calculate the sum of squared difference (SSD) with an incoming video frame later. The corresponding absolute orientation information is saved as well, to be used for recovering the camera pose when the camera gets lost.

3.3 Pose Relocalization

Pose relocalization follows the method proposed by [17]. That is, the subsampled and blurred images of the virtual keyframe and the incoming frame are compared constantly for camera orientation relocalization. In the matching step, we subsample and blur the incoming video frame following the same procedure we apply to the virtual keyframe. Then, in the alignment step, we compare the incoming video frame with all of the virtual keyframes and find the virtual keyframe with the smallest SSD. The final camera rotation is estimated with the help of ESM Visual Tracking and best-fit 3D camera rotation estimation by considering the motion of a few virtual sample points placed about the image.

3.4 Experimental Results

Images are captured from a PointGrey DragonFly2 video camera which delivers 640×480 pixel RGB frames at 30 Hz. These frames are converted to 8 bpp grayscale for tracking and an RGB image for display. We use Zhang's calibration technique to measure the camera's intrinsic parameters in a one-time off-line calibration procedure [31], [32]. In addition to the focal length and principal point, lens distortion parameters are also measured which are used to correct the position of features in the image and to undistort each frame. For testing against the orientation ground truth, we used a D46-17 pan tilt unit (PTU) from FLIR Motion Control Systems [9]. The step size of the device is 0.0514 degrees, 300 degrees per second.

To show that the proposed orientation relocalization based on virtual keyframes works well, we compared the current system with the previous version of Envisor [5]. Each of our tests consisted of an initialization stage, where the camera is moved around the scene to generate virtual keyframes. We then tested various speeds of rotation over a range of 110 degrees around the vertical axis. While it is possible to define a line or path avoiding the virtual keyframes, our tests make the likely assumption that the camera comes across some of these keyframes, so without loss of generality, we moved the camera in the yaw direction only. In practice, the time it takes a user to find a keyframe may vary, so as a control for the number of keyframes encountered, we stationed virtual keyframes along the horizontal line at every 10 degree. Since we are using 400 virtual keyframe locations, the angle between two points with a minimum distance is about 10 degrees, our experimental setup is reasonable, if idealized. Envisor uses a hybrid vision-based tracking approach which combines a frame to frame relative rotation tracking and landmark-based absolute orientation tracking. For our comparative analysis between the two approaches, we turned off the landmark-based tracking module when testing the virtual keyframe system. In practical deployment, however, landmarks would still be used to estimate the camera orientation simultaneously with virtual keyframes.

Fig. 4 shows the comparison results of camera orientation tracking accuracy between the conventional Envisor and the proposed system at different angular speeds of 30 degrees per second, Fig. 4b 60 degrees per second, and Fig. 4c 80 degrees per second. As mentioned, in order to ensure a fair and accurate comparison between the tracking methods, we mounted our camera on our PTU and logged ground truth orientation information for each frame of video.

As shown in Fig. 4a, the results are similar in both cases when the camera motion is sufficiently moderate. However, when the speed exceeds the previously mentioned threshold, the conventional Envisor always failed as shown in Fig. 4b and Fig. 4c. However, with the proposed orientation relocalization approach, we can recover the

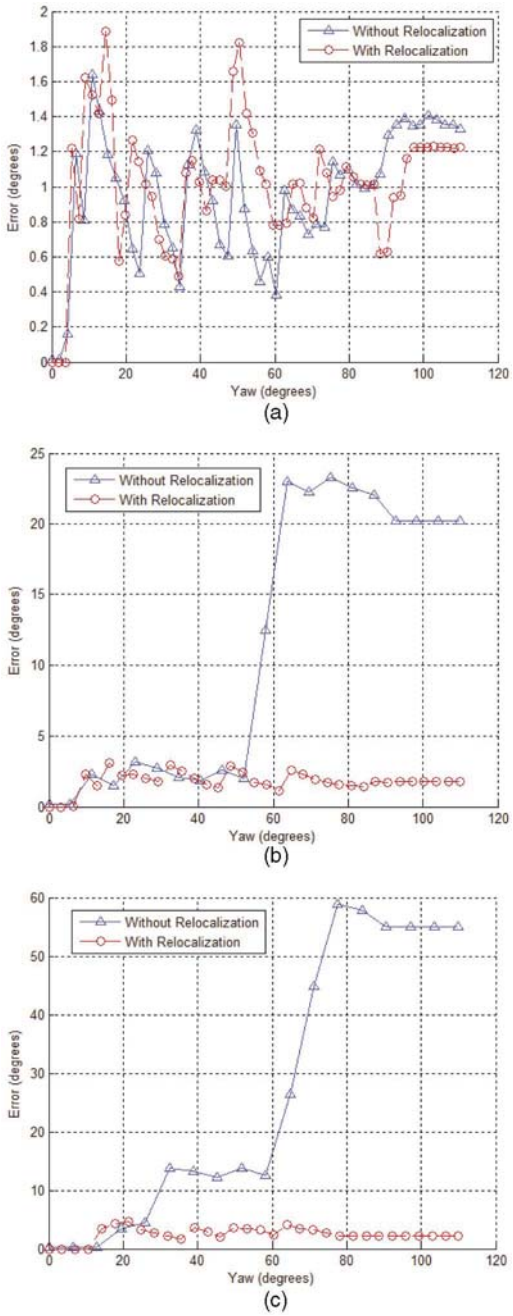

Fig. 4. Comparison of camera orientation tracking accuracy between previous version of Envisor and proposed system at the angular velocity of (a) 30 degrees per second, (b) 60 degrees per second, and (c) 80 degrees per second, (Note: Different scales of the y axes).

camera orientation and continue tracking. If we compare Fig. 4b and Fig. 4c, we can see that the error for Envisor increases with higher angular speed. Further examples of relative tracking performance and illustration of resulting environment maps can be found in [7].

A performance comparison in an outdoor scene is shown in Fig. 5. In this case, we did not use an image sequence for generating virtual keyframes, but instead generated them in a live fashion while moving a hand-held camera around the scene. We purposefully moved our camera very quickly five times to trigger tracking failure. Fig. 5a and Fig. 5b show tracking results without and with the relocalization capability. Whereas there are

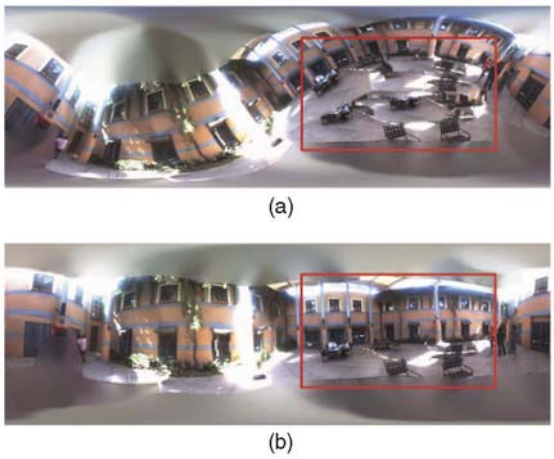

Fig. 5. Comparison of environment maps between the previous version of Envisor and the proposed system (a) without relocalization (b) with relocalization.

many misalignments in Fig. 5a, Fig. 5b exhibits only small errors thanks to virtual keyframe relocalization.

3.5 Performance

While our camera captures at 30 fps, Envisor runs at approximately 18 fps on our system (Dell XPS M1210, 2.0 GHz CPU) as shown in Table 2. On the other hand, the processing time required to find the best-match keyframe depends on the number of virtual keyframes. As shown in Table 1, the processing time increases almost linearly. Since we use 2,000 virtual keyframes, we spend about 7 ms on relocalization which is not a big overhead for our system. For a detailed evaluation of the performance of the proposed keyframe relocalization method, relative to the performance of the other invariants of Envisor, see Section 5.

4 Improved Tracking Speed

In this variant, for speed improvement, we replace SURF with an image patch descriptor [5]. For panorama creation, a user's viewpoint is static, so less complex descriptors can be used. In our application, image-patch descriptors yield tracking performance on par with SURF descriptors, and provide a speed increase.

One of the weaknesses of the proposed version of Envisor is that it attempts to recover the camera pose at every frame, slightly reducing the tracking speed. Our intention with a new variant, Envisor with selective relocalization, is to eliminate the constant relocalization attempts, thereby improving the system speed. That is, the proposed method

TABLE 1
Processing Time Required to Find the Best-Match Keyframe According to the Number of Virtual Keyframes

Number of virtual keyframes	Time (ms)
400	1.614
800	3.053
1,200	4.398
1,600	5.900
2,000	7.247
2,400	8.420

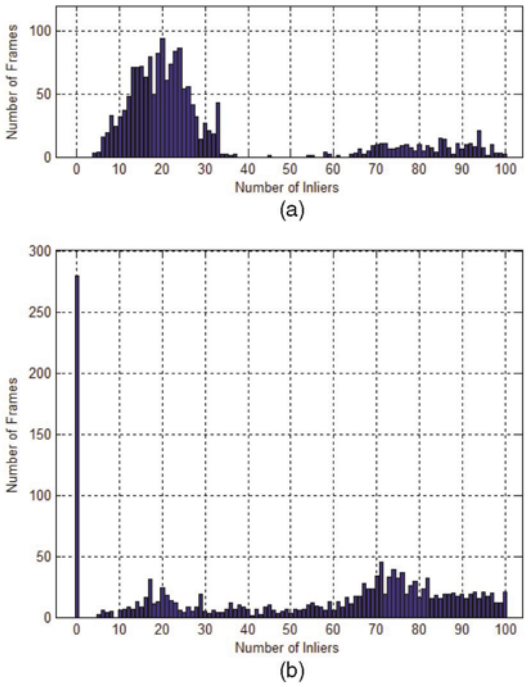

Fig. 6. Distribution of the number of inliers (a) slow motions (b) fast and abrupt motions.

detects when tracking is lost before attempting relocalization. To estimate tracking failure, we measure the number of inliers after applying RANSAC at every frame. Once tracking is lost, it finds the virtual keyframe which is most similar to the current frame to relocalize the camera orientation.

To determine a threshold for estimating tracking failure, we make use of the number of inliers after applying RANSAC. Fig. 6 shows a typical distribution of the number of inliers when freely moving a camera over a period of a minute using (1) slow continuous and (2) fast and abrupt camera motions. First, we rotate the camera slowly to understand the relationship between the two tracking approaches (landmark and frame to frame) of Envisor and the corresponding distributions of inliers. In Fig. 6a, most of the frames from the landmark based absolute orientation tracking have between 10 and 40 inliers. Most frames employing frame-to-frame feature tracking have between 70 and 100 inliers. Once Envisor finds landmarks in a camera's FOV, it takes advantage of the landmarks even though their number is not high. However, when a camera moves into unvisited regions, Envisor depends on the frame to frame tracking, and generates many features for tracking. As shown in Fig. 6b, we can observe that a number of frames have high number of inliers with fast camera motion, and there are many tracking failure cases (i.e., the number of inliers is zero.). Based on this, we determined the threshold for tracking failure to be zero.

In Table 2, we show average times of Envisor with selective relocalization compared to Envisor with constant relocalization. We can observe that the results are very similar between the two Envisors except in the stages, *Landmarks* and *Relocalization*. The first gain is from that fact that we replaced SURF with an image patch descriptor, and the second gain is due to the tracking failure detection.

TABLE 2
Average Times of the Various Stages of Envisor

	Constant Relocalization	Selective Relocalization
Stage	Time (ms)	Time (ms)
Video decoding, preprocessing	18.998	19.064
Vignetting removal	6.948	6.962
Undistortion	0.016	0.017
Preprocessing total	25.963	26.042
KLT tracking	7.947	6.871
RANSAC	0.334	0.484
Landmarks	11.170	1.518
Relocalization	5.018	2.801
Tracking total	24.469	11.675
Cubemap update	3.056	2.611
Total	53.488	40.328

The preprocessing and tracking are broken up into their component stages and timings are presented for each stage as well as the frame total. The final total is the start to finish for each frame of the test application.

In addition to the speed gain, Envisor with selective relocalization generates better panorama images since it pauses rendering onto the cube map when tracking fails. When the camera pose is recovered again, it resumes tracking and rendering.

5 Evaluation

This section presents the details of the methods used to evaluate the performance of the original version of Envisor as well as the two proposed variations: Envisor with Constant Relocalization and Envisor with Selective Relocalization. In addition, we add a best case variant (Envisor with pre-scanning) to the analysis in order to gain insight into the performance possibilities of the relocalization method itself. The first step in our analysis is the collection of a set of meaningful orientation data. Using these orientation paths, we then obtain video data which is used as input to each system in order to generate panoramas which are then evaluated by experts. Two additional evaluations based on pure tracking performance (not generated panoramas) are also presented. The first is a quantitative analysis of the average distance to ground truth over the collection of input videos. The second is a qualitative analysis of a live demo of each system by expert users.

5.1 Envisor with Pre-Scanning

In these evaluations, we compare against a version of Envisor with prescanned environment maps for two reasons. First, it allows for a better analysis of how the system will perform with a complete set of virtual keyframes. Second, we wish to isolate one problem of the proposed relocalization methods, which is that if they cannot generate valid virtual keyframes, relocalization might not produce a good panorama image. Using pre-scanned data allows us to focus on the accuracy of relocalization apart from the errors introduced during the panorama construction. Additionally, while pre-existing environment maps are not generally available in sufficient density, in the future, it may be possible to rely on Google StreetView or Microsoft Bing StreetSide, or to generate them using models of surrounding buildings. Practically, this is akin to generating an a priori environment model for model-based tracking.

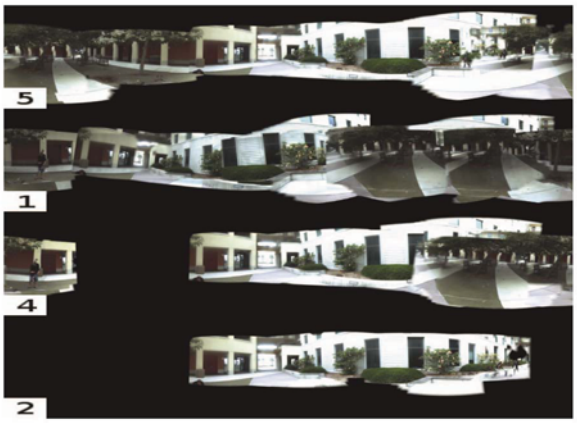

Fig. 7. Users were asked to rank panoramas generated by each of the four methods used, and they were able to click on an item to compare it to a ground truth panorama.

5.2 Preliminary User Study

We based our evaluation of the systems presented on sets of recorded head orientations collected from 23 participants over a set of two tasks. The participants were campus students with no familiarity with our project. Each participant was given a small monetary compensation for their participation. The participants were asked to perform their tasks while wearing a hat with an attached orientation tracker (InterSense InertiaCube2 [33]). Between each run the tracker was calibrated in order to ensure that the motion data collected accurately matched the view of the participant. The first task was a simple examination of their surroundings, motivated by a series of questions concerning their environment, which were asked after providing a minute for observation. The second set of data collected was the last in a series of search tasks. Being the last, this search task was limited to one minute as the object was not present in the environment. The unannounced purpose of this study was to collect a realistic set of motion data on how a person typically searches through or examines an environment. This data is important as these are two common tasks users may perform while using a tracking solution such as Envisor to capture the environment or overlay augmentations.

5.3 Capturing Ground Truth

We used a camera mounted on the PTU (see Section 3.4) in order to precisely replay the orientation information from the user study and capture video feeds for environment map construction. For our analysis, we replayed randomly chosen sequences from our orientation information in both indoor and outdoor environments, yielding datasets with ground truth orientation information. We randomly divide the motion paths into two sets collected at one indoor location and one outdoor location.

5.4 Panorama Evaluation

The recorded video data then serves as input for each of the proposed methods. We used a total of 45 video sequences, 25 of which were recorded indoors and 20 recorded outdoors. The outdoor sequences were purposefully recorded at two different times of day, to capture different lighting conditions. These panoramas were presented to experts in the field of computer vision, AR, and visualization using a ranking program, the interface for which is shown in Fig. 7. In the evaluation, the experts were presented with all four panoramas generated from a given input video sequence (by each of our four methods: Envisor, Envisor with constant relocalization, Envisor with selective relocalization, and Envisor with pre-scanning). The experts selected and ranked the perceived accuracy and reliability of each system on a scale from 1 (worst) to 7 (best). The selected scores were displayed on the left-hand side of the panorama. In addition, users were able to compare each panorama to the ideal panorama generated by using the orientation data from the PTU directly. The panoramas displayed were randomly ordered, and each evaluation was repeated two additional times in order to control any initial bias and to ensure consistency.

TABLE 3
Comparison of Panorama Evaluation and Live Evaluation

	CRS	SR	CR	NR
Panorama evaluation	0.46	0.77	0.84	1.00
Live evaluation	0.44	0.68	0.72	1.00

First row, the mean measurement of robustness for each system over all users and over all sets. Lower values are better with values in each row scaled by the worst result. Second row, the robustness ratings from the live evaluation. (CRS: Envisor with prescanning, SR: Envisor with selective relocalization, CR: Envisor with constant relocalization, NR: original version of Envisor (no relocalization)).

We found that the evaluations were very consistent between trials. Then, we normalized the results of each user's data by subtracting each vote by the minimum vote for that user and dividing the difference by their overall range of votes. From this data we were then able to determine an average ranking over all users, for each method applied to each data set (cf. Table 3).

Regarding the evaluation of system robustness by judging the final outcome (the completed environment map) we would like to point out that with the current rendering of the environment map, it is possible that places where the system failed or showed bad performance stayed unnoticed because the area was correctly covered later on, for example, if there was a sudden loss of tracking early on (or vice versa in the case of late tracking failures). This happened in a small set of panoramas however, and the effects were still noticeable as the errors were not completely corrected. Therefore, we do not believe that it influenced the results of the panorama evaluation unduly. The addition of live system evaluation was prompted by considerations that judging a static result image would not reveal all aspects of such a difficult dynamic concept as robustness, but we ended up with good correlation between these evaluation methods.

5.5 Ground Truth Evaluation

To obtain insight into the respective performance of the four different versions of Envisor, we show a comparison of absolute orientation errors from each of the four methods for two different camera input feeds as shown in Fig. 8. We observed that in general, Envisor with pre-scanning carries out more stable and accurate tracking compared to the other methods. On the other hand, the original version of Envisor

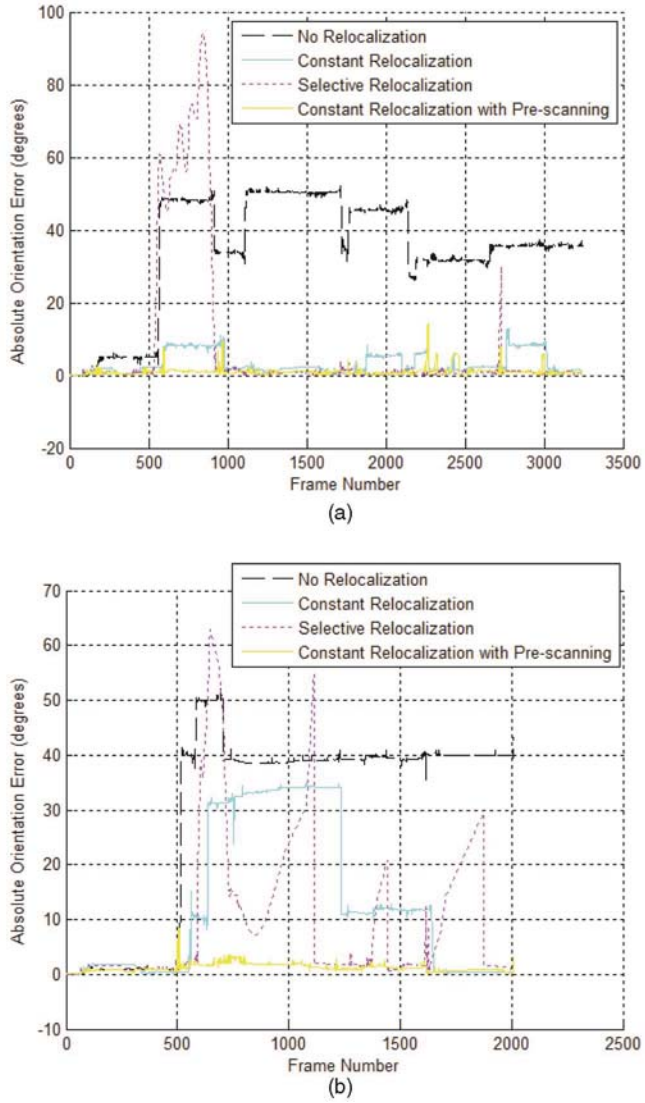

Fig. 8. Comparison of absolute orientation errors from each of the four methods for different sequences (a) example 1 (b) example 2.

shows many erroneous tracking results especially when a user's motion is fast. However, Envisor with selective relocalization sometimes produces high peaks because estimated absolute orientations do not change when tracking is lost. Fig. 8a also shows that Envisor with constant relocalization could have relatively many small errors due to some unstable virtual keyframes.

5.6 Live Evaluation

While environment maps allow for a large number of comparative evaluations to be collected, the performance of the systems under live evaluation is also important. For such an evaluation, five expert users were asked to rank the performance of the tracking of each of the proposed systems. To ensure consistency, each user ranked each method four times for a total of 16 randomly ordered runs. Similarly to the panorama evaluation, the evaluators were asked to rank each system from 1 to 7. The same normalization and averaging was performed from the previous evaluation.

The final averages of the live evaluation for each system are listed alongside the average scores for the panorama

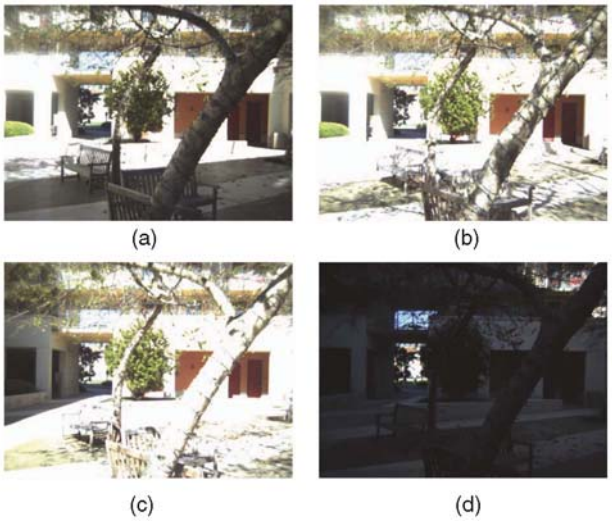

Fig. 9. Example of the same scene at different times of day. (a) 10:00 AM, (b) 11:30 AM, (c) 1:00 PM, and (d) 3:30 PM.

evaluations (cf. Table 3). We observe that the live user evaluation matches closely to panorama evaluation results.

It is worthwhile to note that two distinctive phenomena of Envisor with selective relocalization had an effect on the final evaluation. As we described, this version of Envisor detects when tracking is lost and halts rendering onto the cube map. Immediately after the tracking is recovered, rendering is resumed. Some evaluators commented that this pause had a negative effect on their evaluation. However, the evaluations indicate that the generated panorama results are better than those acquired from the original version of Envisor and Envisor with constant relocalization.

6 Improved Relocalization Robustness

In the analysis presented in the previous section, we captured both indoor and outdoor scenes. It is worthwhile to note that in some of the outdoor scenes the constantly recovering methods tended to fare worse relatively than the original version of Envisor or the selective relocalization version. One of the central reasons for this problem is a change in lighting conditions over time and at various orientations.

In this section, we discuss the robustness of virtual keyframe-based relocalization in the presence of lighting changes. We then introduce a new method for improved robustness and analyze its performance.

6.1 Robustness with Respect to Lighting Changes

First, we discuss the effects of lighting changes on orientation relocalization performance, and investigate how to improve the relocalization rate. There are several reasons for considering the effects of lighting conditions on tracking. First, lighting can change rapidly, especially in outdoor cases, as can be seen in Fig. 9, in which we can observe the changes of ambient illumination, directional lighting, shadows, and so forth. Additionally, we may want to use data stored from previously captured environment maps of an area for relocalization of orientation in sessions at a later time.

Using the PTU, we took several video sequences of three indoor environments at different shutter speeds, and five

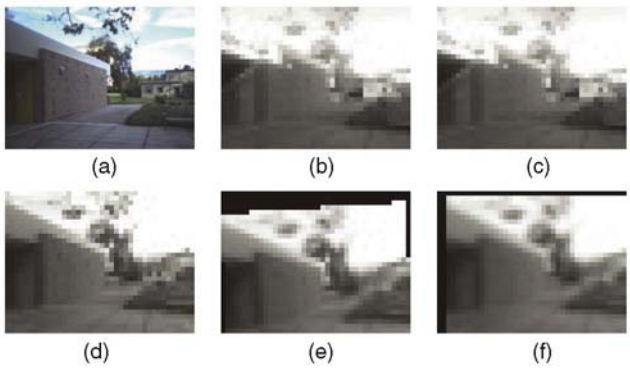

Fig. 10. Example of an alignment failure (a) original 640×480 image, (b) corresponding 40×30 image, (c) best-match virtual keyframe, (d) warped image (wrong) using the basic virtual keyframe approach, (e) best-match virtual keyframe, and (f) warped image using the proposed approach.

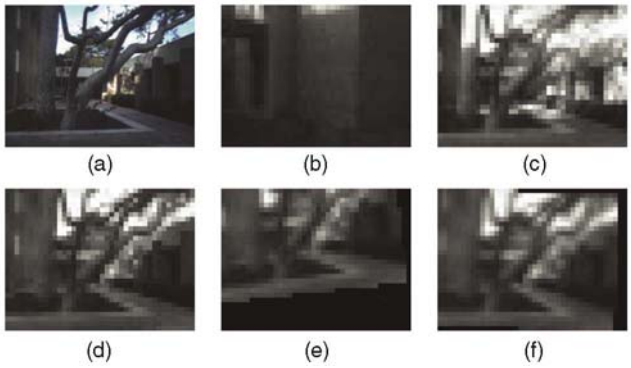

Fig. 11. Example of a matching error of a best-match virtual keyframe (a) original 640×480 image, (b) corresponding 40×30 image, (c) best-match virtual keyframe (wrong), (d) warped image (wrong) using the basic virtual keyframe approach, (e) best-match virtual keyframe, and (f) warped image using the proposed approach.

outdoor environments at different times of day. More detail information will be given later. For each data set, we generated 36 virtual keyframes at every 10 degree along the horizontal direction. That is, we ran Envisor to generate virtual keyframes for each of the video sequences, and tested one reference video sequence from the same location against each set of keyframes set by set.

When the absolute orientation error between the known PTU orientation (i.e., ground truth) and recovered orientation is greater than 3 degrees, we consider the result a "relocalization failure." We derived this limit by observing that the proposed version of Envisor works successfully at angular velocities of up to 56 degrees per second and by using the timing information from Table 2. From this, we determined that the maximum distance per frame is about 3 degrees. To determine matching errors, we used the five virtual keyframes closest to the current camera orientation. We chose five as there is a guarantee of sufficient overlap between the keyframe and the camera frame based on the FOV of our camera and the chosen distribution of virtual keyframes. These five then have a chance to correctly align in the second step. As expected, the relocalization performance was very poor in outdoor cases when the time of the virtual keyframe capture and comparison differed considerably, and when large amounts of information was lost due to under/over exposure, which reduced image contrast. We thus address the issue of how to reduce the effects of changing light conditions and achieve better relocalization capability by analyzing the shortcomings of the existing system and introducing a new approach.

As described in Section 3.3, the relocalization process consists of two steps: 1) matching step—finding the closest matching virtual keyframe to the current image and 2) alignment step—warping the current frame to the found best-match virtual keyframe using ESM. Over/undersaturated pixels in significantly bright or dark images can cause problems in both of the steps.

Fig. 10 and Fig. 11 demonstrate examples of an alignment failure and a matching error, respectively. In the first case, the sky and the oversaturated area on the building (in the virtual keyframe) due to the direct illumination from the sunlight make the alignment difficult. Even though finding the correct virtual keyframe was successful, warping the current image to the correct virtual keyframe failed using the basic approach (cf. Fig. 10c and Fig. 10d). However, with our new approach, we can get a correct warping result (cf. Fig. 10e and Fig. 10f).

In the second case, even finding the closest matching virtual keyframe did not work due to the change in lighting from when the images were acquired (cf. Fig. 11c and Fig. 11d). However, with our new approach, we can get correct matching and warping results (cf. Fig. 11e and Fig. 11f).

Our new approach is meant to cope with ambient illumination changes and some of the effects of changes in direct illumination caused by changes in lighting at different times of day. Thus, the following two methods are adopted. First, for an illumination normalization, we apply a histogram equalization [34] algorithm to a small 40×30 image before we apply a Gaussian convolution. Histogram equalization increases the global contrast of an image, especially when the data of the image is represented by a narrow band of pixel values. This step normalizes each image with different ranges of luminance values, reducing the effect of ambient illumination changes. This is a low-cost operation as the image size is very small. We also tested applying histogram equalization to incoming 640×480 frames and found that the results were very similar in terms of relocalization performance, but took 0.8028 ms versus 0.0288 ms in the case of the 40×30 image. We thus decided to apply histogram equalization to small images.

Second, we remove over/undersaturated areas from the histogram-equalized image. We consider a pixel over/undersaturated, respectively, if their values are within 2 percent of the maximum and minimum. We then calculate the mean of the image avoiding values from over and undersaturated regions. Resuming the normal relocalization process, we subtract this mean value from the image.

In order to evaluate our proposed approach, we took several surround-view video sequences of indoor environments at different shutter speeds and outdoor environments at different times of day, each with corresponding ground truth orientation data from a PTU on which the camera was rotating around its vertical axis. Indoors, we took three

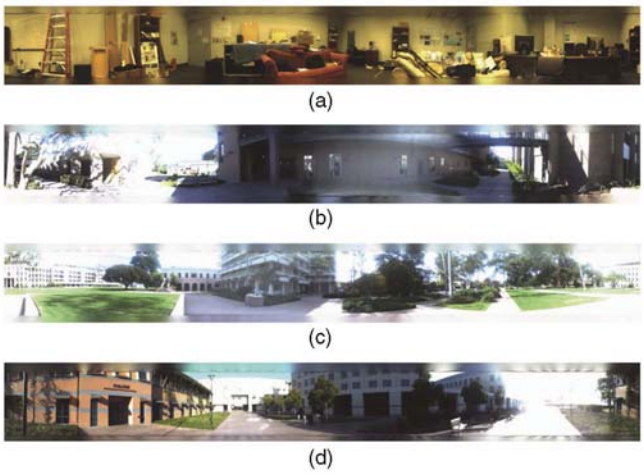

Fig. 12. Example locations for indoor and outdoor datasets used in our experiments (a) "Laboratory," (b) "Footbridge," (c) "Quad," and (d) "Courtyard" datasets.

different datasets, one of which consisted of 15 different sequences according to different shutter speeds ranging from 15 ms to 120 ms, each sequence containing 2,907 frames. Another indoor data set ("Laboratory") contains 13 different sequences (13 different shutter speeds ranging from 1.17 ms to 133.29 ms), and each sequence contains 939 frames. Outdoors, we took five datasets including these three: A "Courtyard" data set constructing a panorama every 30 minute from 7:00 AM to 12:30 PM with different shutter speeds at 4 ms, 5 ms, and 6 ms. Each sequence contains 2,627 frames and ground truth. The "Quad" and "Footbridge" datasets were captured with similar timing and parameters in different locations on different days. Fig. 12 shows panorama examples which were generated using one sequence belonging to each dataset.

We compare our method with a version of the feature-based approach [35] currently under development. It performs a corner-based image alignment on a 80×60 image subsampled four times from a 640×480 input image. A Harris corner detector is employed to detect features and only good features are chosen by comparing corner scores with a threshold. We tested this approach and found that the pose estimation requires 5.77 ms. On the other hand, our approach takes just 0.45 ms.

6.2 Relocalization Performance Indoors

We consider both ambient and directional lighting changes. In order to simulate ambient illumination changes in a controlled environment, we captured several indoor panoramas using a range of shutter speeds, which has the effect of giving a precise metric for the ambient illumination change. Fig. 13 illustrates our resulting test data which also includes histogram equalized 40×30 images.

Using the "Laboratory" dataset, we carried out experiments with a video sequence of a 30 ms shutter speed as a reference for a relocalization test against other video sequences. That is, we generated an environment map and virtual keyframes from each video sequence (all taken at different shutter speeds), and tried to match each frame of the 30 ms shutter speed sequence against the virtual keyframes of each sequence. The 30 ms exposure time was chosen as we

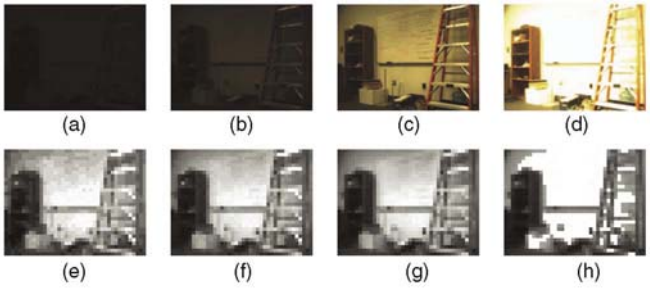

Fig. 13. Indoor scene images for different shutter speeds at (a) 1.77 ms, (b) 5.93 ms, (c) 30 ms, (d) 133.29 ms, and histogram equalized 40×30 images at (e) 1.77 ms, (f) 5.93 ms, (g) 30 ms, and (h) 133.29 ms.

estimate it to be the optimal shutter speed for the indoor scene based on the amount of light entering the camera.

Fig. 14 shows experimental results in the indoor case ("Laboratory" dataset). The graph shows that the basic virtual keyframe approach is very vulnerable to ambient illumination changes and over/underexposed areas induced by shutter speed changes. In this case, a zero-mean image does not help since a large difference in the luminance range still remains even after a zero-mean operation. It is worthwhile to note that there are not many matching errors, but most errors came from misalignment between the current image and a best-match virtual keyframe. The reason

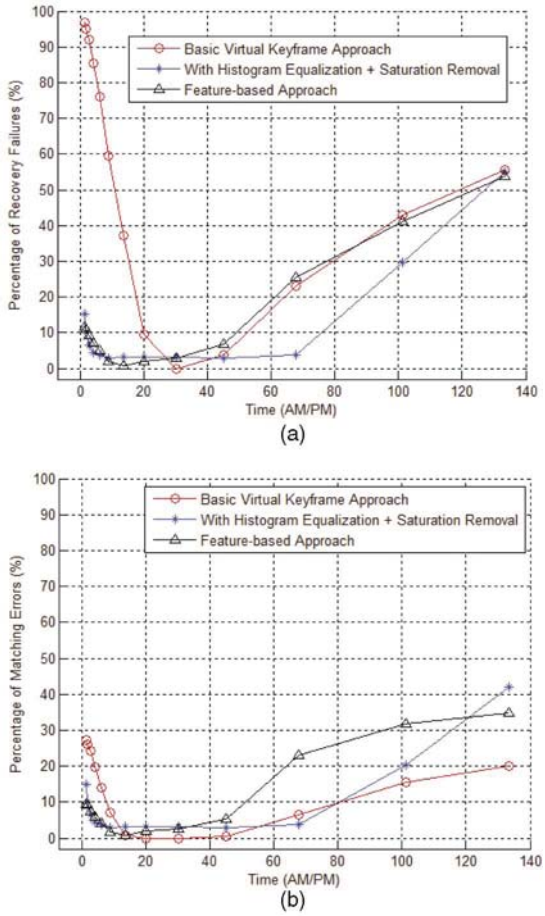

Fig. 14. Comparison of relocalization performance according to different shutter speeds for an indoor case (a) relocalization failures (b) matching errors.

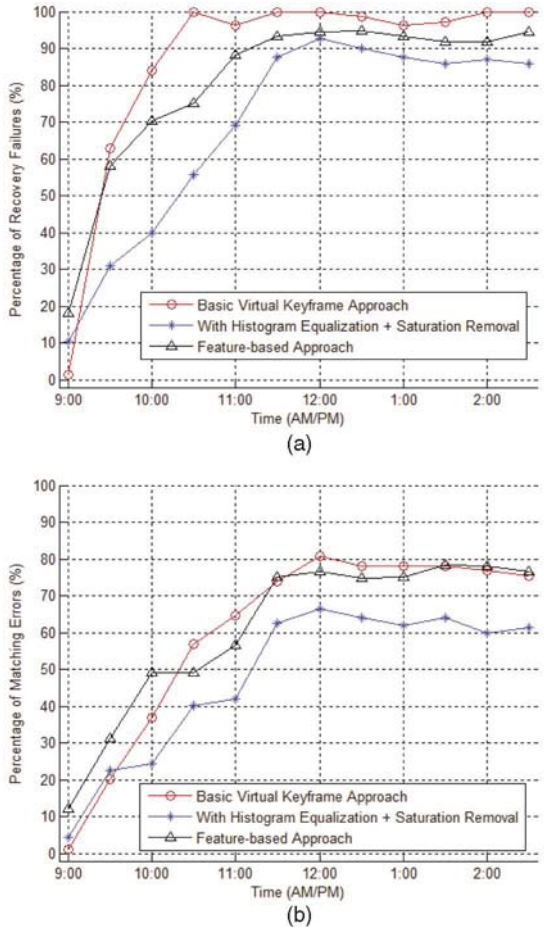

Fig. 15. Comparison of relocalization performance according to the time of day for an outdoor case (a) relocalization failures (b) matching errors.

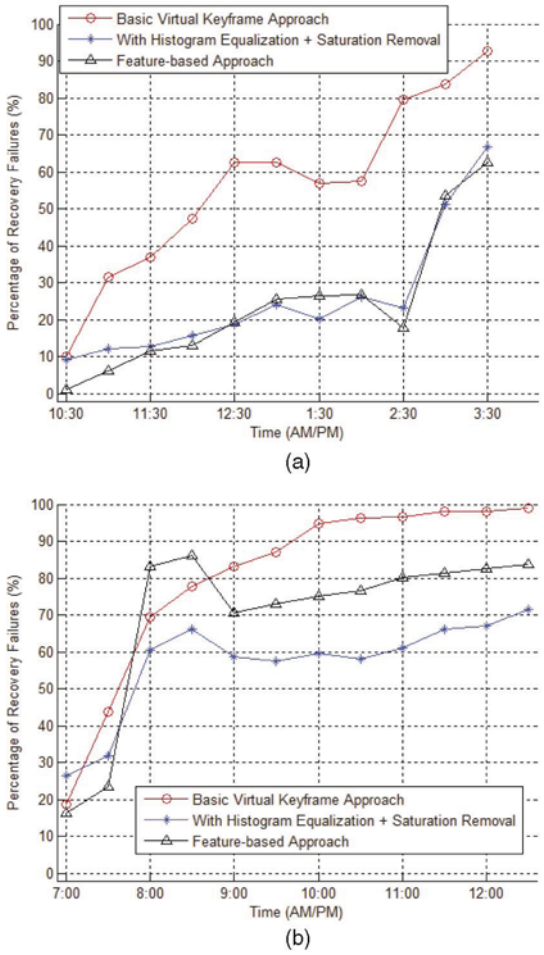

Fig. 16. Comparison of relocalization failures according to the time of day for outdoor cases (a) "Quad" and (b) "Courtyard" datasets.

is that large luminance differences between the two images have a bad effect on the ESM optimization. When the shutter speed is low, it is difficult for ESM to find a proper transformation between the current image and very dark virtual keyframes. When the shutter speed is high, over-exposed white areas cause misalignment between the two images.

From Fig. 14, we can observe a significant improvement over the conventional approach with the exception of a few sequences which have very big shutter speed differences compared to the reference sequence. The reason that the proposed approach is worse than the basic virtual keyframe approach above 100 ms shutter speeds is that after histogram equalization is applied to the current and keyframe images, corresponding luminance values have been changed very differently. In the graph, the vertical axis represents the number of relocalization failures.

6.3 Relocalization Performance Outdoors

This section illustrates relocalization performance for three different outdoors video sequences. In the indoor case, using only histogram equalization provides better performance as saturation removal removes some useful information. However, in the outdoor case, saturation removal is needed as there are locally-saturated areas due to directional lights. Fig. 15 shows a comparison of results of relocalization performance in outdoor sequences ("Footbridge" dataset) at various times of day ranging from 9:00 AM to 2:30 PM with samples taken every 30 minutes. We use the sample at 9:00 AM as a reference. Note that this particular location is exceptionally difficult. A large building blocked much of the directional light for the early sequences, and created a sudden and large set of differences after the sun had risen over the building.

The main error-causing issues in the outdoor environment are ambient illumination changes and directional light changes as shown in Fig. 9. For example, due to ambient illumination changes, a luminance value on the same spot of a building changes even after 30 minutes even though there is not a directional light cast on the building. In addition, due to the directional light changes, a luminance value on the same spot of a building changes very significantly from very bright to very dark. Furthermore, very similar virtual keyframes could be created owing to repetitive structures of a building, and the same color on a very big building.

Two more results, for the "Quad" and "Courtyard" datasets are shown in Fig. 16. We used the first samples as references (10:30 AM and 7:00 AM, respectively).

In each of the outdoor sequences there is a large change in illumination throughout the day. In some cases, the resulting differences remain too large for our solution to perform well

as seen in Fig. 15. From 11:30 AM onward there is an 85 percent or higher likelihood of recovery failure.

Note that while in some cases the errors is large, the qualitative performance may not be prohibitive. In terms of performance, the graphs in Figs. 15 and Fig. 16 can be interpreted as average time for relocalization. For example, a system with a relocalization failure rate of 50 percent will be able to resume frame to frame tracking after two attempts, on average, whereas a system with a relocalization failure rate of 90 percent will recover after on average 10 attempts. Therefore, even small improvements to the percentage of relocalization failures may result in a much shorter relocalization time, and even relative high percentages of relocalization failure. Finding a solution which will work even better despite some of these larger changes in illumination is left to future work.

7 Conclusion

A robust tracking relocalization method is necessary for real-time surround-view panorama acquisition to prevent the need for a system restart. We presented a relocalization method using virtual keyframes for online environment map construction. We first discussed how to generate virtual keyframes on the fly while constructing the environment map using a fragment shader. Our results demonstrate that a user can create an environment map on the fly by rotating a camera and recover from tracking failure with enough accuracy to continue generating a panoramic map. In addition, we presented a faster variant of Envisor, which recovers camera pose only after tracking failure occurs instead of attempting relocalization at every frame. We also evaluated different versions of Envisor in a practical sense. Finally, we introduced novel ways to cope with lighting changes to improve relocalization performance.

In the original version of Envisor, abrupt motion changes cause errors to accumulate from the beginning, harshly affecting the rendering of incoming frames. Similarly, Envisor with constant relocalization is seriously affected by early abrupt changes as they induce some errors in generating reliable virtual keyframes, making relocalization based on virtual keyframes inaccurate. On the other hand, Envisor with selective relocalization recovers camera orientation relatively accurately even though tracking is lost as it generates virtual keyframes only after tracking is back to normal and reliable.

Note that any qualitative analysis of the relocalization of a system would depend on the quality of the frame to frame tracking. If the frame to frame tracking is very poor and relocalization is in constant use then the performance of the relocalization needs to be very good. If the tracking is such that failures happen relatively infrequently, then a quick relocalization may be less important.

As a reference, we tested the performance of the constant relocalization with pre-scanning, as discussed earlier. Live expert evaluation indicated that this system was robust. An analysis of the average number of frames lost before relocalization proceeds indicates around 4.5 frames to be a tolerable loss while maintaining a robust tracking system. Therefore, between a 50 and 60 percent failure rate can still be considered robust for a relocalization system.

One further drawback of our system is that, in order to use the keyframe relocalization the user must first construct a partial environment map in the area. However, this is made easier by the fact that relocalization works even during this initial step. During the construction the user is able to quickly move back over any previously covered regions to regain tracking in the case of tracking loss and can now more carefully cover the area where tracking failed.

There are still several remaining challenges. Currently, we distribute the locations of the virtual keyframes statically at the start of the environment map construction. While this works very well if the user intends to build a full model of the environment, an AR user may be interested in only constructing a small section of the scene, or may not be interested in capturing the ground or sky. One possibility for future work would be to dynamically choose and adapt the locations of the virtual keyframes. This would provide a higher density of keyframes where needed and could result in better relocalization especially at the start of the environment map construction.

We are also working to further mitigate errors in relocalization due to changes in directional lighting. In particular, we are investigating shadow removal techniques (e.g., [25]) in order to reduce matching errors from directional lighting changes.

Even after addressing lighting changes, some issues remain in matching virtual keyframes. If the scene contains largely homogeneous regions, or large sections of a repeating pattern, the matching will be difficult. Additionally, passers-by or nearby plants and leaves moving in the wind make the task of recovering to keyframes more difficult.

We also are currently determining the exact effect of positional changes on relocalization from keyframes. We will also focus on possbile improvements to the relocalization system in order to be used in 6 DoF tracking.

Acknowledgments

This work was supported in part by US National Science Foundation (NSF) CAREER grant #IIS-0747520, ONR grant N00014-09-1-1113, and by a research contract with KIST through the Tangible Space Initiative Project. The authors wish to thank Stephen DiVerdi for his development of the original Envisor system and his continued help. The authors also thank Gerhard Reitmayr at the Graz University of Technology for sample code of Guck (Feature-based Pose Tracking).

References

[1] T. Höllerer, J. Wither, and S. DiVerdi, "Anywhere Augmentation: Towards Mobile Augmented Reality in Unprepared Environments," *Location Based Services and TeleCartography, Series: Lecture Notes in Geoinformation and Cartography*, G. Gartner, M. Peterson, and W. Cartwright, eds., pp. 393-416, Springer-Verlag, Feb. 2007.

[2] M. Uyttendaele, A. Criminisi, S.B. Kang, S. Winder, R. Szeliski, and R. Hartley, "Image-Based Interactive Exploration of Real-World Environments," *IEEE Computer Graphics and Applications*, vol. 24, no. 3, pp. 52-63, May/June 2004.

[3] K. Agusanto, L. Li, Z. Chuangui, and N.W. Sing, "Photorealistic Rendering for Augmented Reality Using Environment Illumination," *Proc. Second IEEE/ACM Int'l Symp. Mixed and Augmented Reality (ISMAR '03)*, p. 208, 2003.

[4] T. Grosch, "PanoAR: Interactive Augmentation of Omni—Directional Images with Consistent Lighting," *Proc. Computer Vision / Computer Graphics Collaboration Techniques and Applications (Mirage '05),* pp. 25-34, 2005.
[5] S. DiVerdi, J. Wither, and T. Höllerer, "Envisor: Online Environment Map Construction for Mixed Reality," *Proc. IEEE Int'l Conf. Virtual Reality,* pp. 19-26, 2008.
[6] Thompson, http://www.math.niu.edu/rusin/known-math/96/repulsion, June 2009.
[7] S. Kim, C. Coffin, and T. Höllerer, "Relocalization Using Virtual Keyframes for Online Environment Map Construction," *Proc. 16th ACM Symp. Virtual Reality Software and Technology (VRST '09),* pp. 127-134, 2009.
[8] R. Fernando and M.J. Kilgard, *The Cg Tutorial: The Definitive Guide to Programmable Real—Time Graphics.* Addison—Wesley Longman Publishing Co. Inc., 2003.
[9] FLIR Motion Control Systems Inc., http://www.dperception.com, June 2009.
[10] M. Pupilli and A. Calway, "Real—Time Camera Tracking Using a Particle Filter," *Proc. British Machine Vision Conf. (BMVC '05),* pp. 519-528, 2005.
[11] S. Se, D.G. Lowe, and J.J. Little, "Vision—Based Global Localization and Mapping for Mobile Robots," *IEEE Trans. Robotics,* vol. 21, pp. 364-375, June 2005.
[12] D.G. Lowe, "Distinctive Image Features from Scale—Invariant Keypoints," *Int'l J. Computer Vision (IJCV '04),* vol. 60, pp. 91-110, 2004.
[13] M. Fischler and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. the ACM,* vol. 24, no. 6, pp. 381-395, 1981.
[14] G. Reitmayr and T.W. Drummond, "Going Out: Robust Tracking for Outdoor Augmented Reality," *Proc. Fifth IEEE and ACM Int'l Symp. Mixed and Augmented Reality (ISMAR '06),* pp. 109-118, Oct. 2006.
[15] B. Williams, G. Klein, and I. Reid, "Real—Time SLAM Relocalisation," *Proc. Int'l Conf. in Computer Vision (ICCV '07),* pp. 1-8, 2007.
[16] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *Proc. Sixth IEEE and ACM Int'l Symp. Mixed and Augmented Reality (ISMAR '07),* Nov. 2007.
[17] G. Klein and D. Murray, "Improving the Agility of Keyframe—-Based SLAM," *Proc. European Conf. Computer Vision (ECCV '08),* pp. 802-815, Oct. 2008.
[18] S. Benhimane and E. Malis, "Homography—Based 2D Visual Tracking and Serving," *Special Joint Issue on Robotics and Vision J. Robotics Research,* vol. 26, no. 7, pp. 661-676, July 2007.
[19] J.M.M. Montiel and A.J. Davison, "A Visual Compass Based on SLAM," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 1917-1922, May 2006.
[20] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg, "Real—Time Panoramic Mapping and Tracking on Mobile Phones," *Proc. IEEE Int'l Conf. Virtual Reality,* 2010.
[21] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof, "From Structure—from—Motion Point Clouds to Fast Location Recognition," *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition ,* vol. 0, pp. 2599-2606, 2009.
[22] J. Pilet, V. Lepetit, and P. Fua, "Retexturing in the Presence of Complex Illumination and Occlusions," *Proc. Sixth IEEE and ACM Int'l Symp. Mixed and Augmented Reality (ISMAR '07),* pp. 1-8, 2007.
[23] J. Pilet, V. Lepetit, and P. Fua, "Augmenting Deformable Objects in Real—Time," *Proc. Fourth IEEE/ACM Int'l Symp. Mixed and Augmented Reality (ISMAR '05),* pp. 134-137, 2005.
[24] G.F. Silveira and E. Malis, "Real—Time Visual Tracking Under Arbitrary Illumination Changes," *Proc. Conf. Computer Vision and Pattern Recognition (CVPR '07),* 2007.
[25] J. Tian, J. Sun, and Y. Tang, "Tricolor Attenuation Model for Shadow Detection," *Trans. Image Processing Archive,* vol. 18, no. 10, pp. 2355-2363, 2009.
[26] S. DiVerdi, J. Wither, and T. Höllerer, "All Around the Map: Online Spherical Panorama Construction," *Computers and Graphics,* vol. 33, no. 1, pp. 835-846, 2009.
[27] S. DiVerdi, "Towards Anywhere Augmentation," PhD dissertation, pp. 835-846, 2007.
[28] J. Shi and C. Tomasi, "Good Features to Track," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR '94),* pp. 593-600, 1994.
[29] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 674-679, 1981.
[30] H. Bay, A. Ess, T. Tuytelaars, and L.V. Gool, "Surf: Speeded Up Robust Features," *Computer Vision and Image Understanding (CVIU '08),* vol. 110, no. 3, pp. 346-359, 2008.
[31] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI '00),* vol. 22, no. 11, pp. 1330-1334, Nov. 2000.
[32] Intel Open Source Computer Vision Library, http://www.intel.com/technology/computing/opencv/, May 2007.
[33] InterSense, http://www.intersense.com/, June 2009.
[34] R.C. Gonzalez and R.E. Woods, *Digital Image Processing.* Addison-Wesley, 1992.
[35] G. Reitmayr, personal comm., June 2010.

Sehwan Kim received the BS degree in electronics engineering from University of Seoul, Korea, and the MS and PhD degrees in the Department of Information and Communications from Gwangju Institute of Science and Technology (GIST), Gwangju, Korea. He is currently a Post-Doctoral Researcher in the Department of Computer Science, University of California at Santa Barbara. His research interests include virtual/mixed reality, 3D computer vision and its applications including attentive AR and HCI. He is a member of IEEE.

Christopher Coffin received the BS degree in computer science at Mount Union College in Alliance, Ohio. He is currently pursuing the PhD degree in computer science at the University of California, Santa Barbara. His research interests include distributed AR and computer vision applications in tracking and modeling. He is a member of IEEE.

Tobias Höllerer received the degree in computer science from the Technical University of Berlin and the PhD degree from Columbia University with a thesis on MAR Systems. He is an associate professor of computer science in the Department of Computer Science, University of California at Santa Barbara, where he codirects the Four Eyes Laboratory, conducting research in the four Is of imaging, interaction, and innovative interfaces. He is a member of the IEEE and the IEEE Computer Society.