Handling Pure Camera Rotation in Keyframe-Based SLAM

Christian Pirchheim, Dieter Schmalstieg, Gerhard Reitmayr [*]
Graz University of Technology

Abstract

Handling degenerate rotation-only camera motion is a challenge for keyframe-based simultaneous localization and mapping with six degrees of freedom. Existing systems usually filter corresponding keyframe candidates, resulting in mapping starvation and tracking failure. We propose to employ these otherwise discarded keyframes to build up local panorama maps registered in the 3D map. Thus, the system is able to maintain tracking during rotational camera motions. Additionally, we seek to actively associate panoramic and 3D map data for improved 3D mapping through the triangulation of more new 3D map features. We demonstrate the efficacy of our approach in several evaluations that show how the combined system handles rotation only camera motion while creating larger and denser maps compared to a standard SLAM system.

Keywords: monocular visual SLAM, hybrid 6DOF and panoramic mapping and tracking, general and rotation-only camera motion, mobile phone

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; I.4.1 [Image Processing and Computer Vision]: Scene Analysis—Tracking

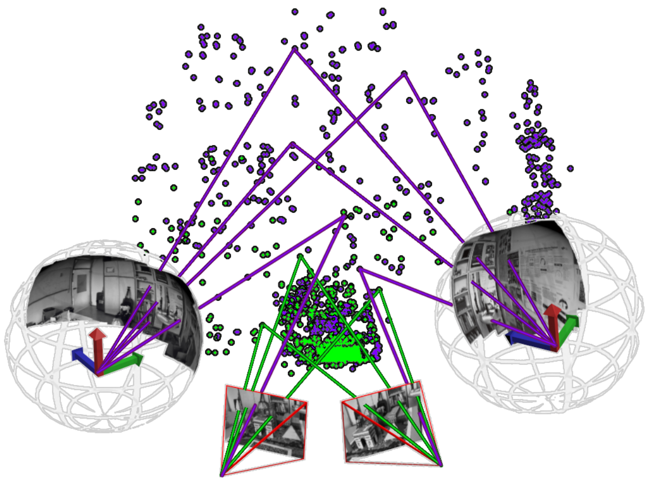


Figure 1: Rotation-only camera movements are handled by tracking and mapping local panorama maps registered within a global 3D map. The information contained in the panorama maps is also used for 3D reconstruction.

1 Introduction and Related Work

Two essential approaches to the monocular visual simultaneous localization and mapping (SLAM) problem have been demonstrated in augmented reality applications. Filter-based approaches adopt probabilistic filtering algorithms such as the extended Kalman filter (EKF) [5]. Localization of the camera and mapping of landmarks is tightly coupled with estimating all visible landmarks in each frame. Due to real-time constraints, only a small number of landmarks can be mapped. In contrast, optimization-based approaches [11] decouple tracking from mapping and execute these tasks asynchronously in separate threads. This allows for increasingly robust frame-rate camera tracking from a map which is optimized in the background with bundle adjustment.

Based on either approaches, SLAM systems have been presented which specialize in mapping and tracking either general or rotation-only camera motion. SLAM systems with six degrees of freedom (6DOF) [6, 7, 11, 15] assume general camera motion and apply structure-from-motion techniques to create 3D feature maps. Robust triangulation of 3D map features from observations of multiple camera viewpoints requires sufficient parallax induced by translational or general camera motion. In contrast, panoramic SLAM systems [4, 14, 17] assume rotation-only camera motion and track the camera in 3DOF. Because no parallax is observed, feature points are not triangulated and consequently can only be thought of as rays. In the following, we call such rays *infinite features*, while 3D points from 6DOF SLAM are called *finite features*.

Panoramic and 6DOF SLAM have complementary strengths and weaknesses: 6DOF SLAM cannot handle pure rotational camera movements well. Tracking may be lost, and in unfortunate situations, erroneously measured finite features may corrupt the map. In contrast, panoramic SLAM can only handle rotational motion. Any translational motion component may be encoded as additional rotation, also leading to a degradation of map quality.

The problem of handling both general and rotation-only camera motion has been little addressed in the literature except in some recent papers. Civera et al. [3] integrated a Bayesian motion model framework and inverse feature depth parameterization [2] into their EKF-based system. The detection of the current camera motion (stationary, rotation-only, general) allows for adopting feature depth and confidence parameters in the filter such that smooth transitions are possible. Features are restricted to infinite depth until parallax-inducing camera motion is detected. With features becoming finite, cameras become estimated in 6DOF rather than in 3DOF. However, filtering is computationally expensive and allows for mapping a small number of features only which goes at the expense of robustness. Additionally, the system has not been shown to run in real-time.

Recently, Gauglitz et al. [9] presented a keyframe-based real-time mapping system that applies a generalized GRIC model selection algorithm [16] to explicitly distinguish homography/rotation and essential motion models between keyframe pairs during frame-to-frame tracking. Depending on the selected model, panoramic or 3D mapping is performed. With each model selection context switch, a new panorama/3D submap is created. The system attempts to merge submaps in the background. However, due to the lack of a global map, the system does neither perform global camera tracking nor relocalization. It also does not use the 3D map information to inform and stabilize the tracking. Therefore, while their system in principle handles more motion sequences compared to our proposal, it does so in a way that is not robust enough for practical applications. Section 7.2 contains a detailed discussion of

[*]e-mail: {pirchheim,schmalstieg,reitmayr}@icg.tugraz.at

differences between Gauglitz et al. and our approach.

The inherent problem of dealing with rotation-only camera motion is partly created by the real-time processing constraint of SLAM systems. As incoming video frames can only be triangulated with respect to the map known up to the current moment, it is rather straightforward to discard frames that cannot be triangulated yet. However, keeping more potential keyframes around would increase the likelihood of matching observations in later frames. This is in contrast to full structure-from-motion systems that have complete information available and therefore can match with both past and future frames.

We propose to combine the advantages of 6DOF and panoramic SLAM into a hybrid keyframe-based system that accepts both fully triangulated keyframes for normal 6DOF operation as well as keyframes with only rotational constraints. This combination contributes in several ways to an optimization-based SLAM system:

- Tracking can cope with pure rotation and provide a more seamless experience to the user.
- Mapping has more keyframes available to estimate new parts of the 3D SLAM map.
- As we extend state-of-the-art approaches, we obtain a system that performs as least as well as a normal 6DOF SLAM.

The tracking component can dynamically and seamlessly switch between full 6D and panoramic tracking modes, depending on current motion performed by the user. It is designed to handle temporary rotations away from the mapped part of the scene that users often make in practice. We detect these rotations and select special "panorama" keyframes that are used to build up local panorama maps. The local panorama maps are registered in a single consistent 3D map. The observed finite and infinite map features allow for robust tracking of alternating phases of general and rotation-only motion with a unified pose estimator. Additionally, we support re-localization.

The transition from a panorama map to another 3D sub-map is not intended, since global map scale consistency would be lost. However, we explicitly exploit observations of infinite features measured in panorama keyframes in the construction of the global 3D map, if they can be combined with observations in later keyframes allowing for full triangulation of the feature.

2 System overview and map representation

The system architecture of our hybrid approach follows the standard two-component design of optimization-based SLAM. Tracking and mapping components are executed in separate threads and synchronize via dedicated keyframe and map interfaces. The keyframe interface allows the tracker to submit keyframes which are asynchronously processed by the mapper. The map interface allows the tracker to retrieve a snapshot of the current map.

The tracking component performs robust frame-rate map tracking of 6DOF or rotation-only camera motion and selects new keyframe candidates. In our system, keyframes can either be fully localized with a 6DOF pose, or panorama keyframes that are described with a rotation relative to a reference pose. The mapping component adds keyframes to a single global map comprising both types of keyframes and finite and infinite features. Additionally, it also refines the map through establishing new data associations and bundle adjustment optimization.

Our system builds and maintains a single global map that has a consistent scale. The map is composed of finite and infinite point features which have 2D image observations in regular 6DOF and panorama keyframes. Figure 2 shows the relationships within our hybrid map representation.

The map is represented as a collection of keyframes that store the camera pose C_i of the keyframe and an image pyramid for tracking

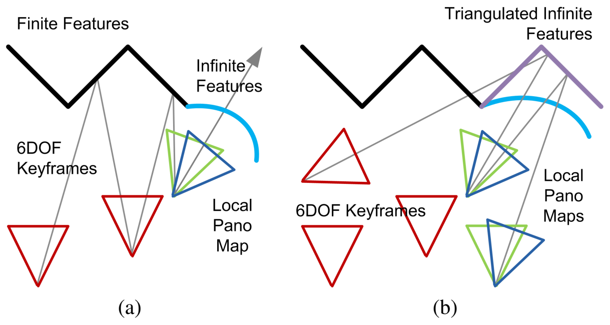


Figure 2: Relationships in our hybrid map representation between keyframes and features depicted in two stages. In stage (a) 6DOF keyframes observe finite map features. Local panorama maps are registered in the 3D map via reference panorama keyframes (green) that have finite and infinite feature observations, while the remaining dependent panorama keyframes (dark blue) observe infinite features only. In stage (b) infinite features are triangulated from corresponding observations matched between additional 6DOF keyframes and/or localized panorama keyframes from different local panorama maps. Note that the additional features enable the localization of further dependent panorama keyframes.

points and finding new correspondences. We use the pose C_i to denote the keyframe itself. Keyframes generally fall into two categories, either regular full 6DOF frames or panorama keyframes. Full 6DOF keyframes are denoted by the set $K = \{C_k\}$. Panorama keyframes are organized in local panorama maps $P_j = \{C_{i,j}\}$ that are registered in the 3D map with its center of rotation. The center of rotation is determined by a dedicated reference panorama keyframe C_j^r which is both a 6DOF frame and part of the panorama map, thus $\{C_j^r\} = K \cap P_j$. The remaining panorama keyframes $C_{i,j}$ are dependent and effectively have only a 3DOF rotation pose relative to the reference keyframe C_j^r.

Point features are represented as homogeneous 4-vectors $X_i = (x,y,z,w)^T$, where $w = 1$ for finite points with known 3D location and $w = 0$ for infinite points that were only observed in panorama keyframes. Infinite points are observed in one or more panorama keyframes of a single local panorama map.

3 Tracking

The tracking component processes the video stream of a single calibrated camera at frame-rate and tracks both general and rotation-only camera motion with respect to a global map that consists of finite and infinite features. The pose estimation combines measurements of both finite (known 3D location) and infinite features, and automatically computes either a 6DOF or 3DOF pose update. In case of incremental pose estimation failure, we provide a relocalization method (see Section 3.2) based on small blurry images [12].

3.1 Incremental pose tracking

Starting with a known pose from the previous input frame, the current camera pose is predicted by a simple decaying constant-velocity motion model.

We select a feature set for matching from all map features by filtering features for (1) visibility from the predicted camera pose, (2) only infinite features of the currently enabled panorama map, (3) overlapping feature re-projections where we prefer finite over infinite features. At any point in time, either none or exactly one panorama map is enabled for tracking. Thus, we are only considering infinite features if the center of rotation of the corresponding panorama map is located close to the camera. Note that panorama maps and keyframes are always used for mapping.

Then, the following steps are executed for each image pyramid level of the current frame, starting at the lowest level. We actively search for each selected feature in the current frame using NCC as score function. Matches with a sufficiently high NCC score are added to the correspondence set that is processed by our unified relative pose refiner. This yields a set of 2D observations O_i for map features X_i.

Given a pose prior and the set of observations, we iteratively estimate incremental pose updates. We optimize the re-projection error

$$E(C) = \sum_i W_i \|\operatorname{Proj}(C \cdot X_i) - O_i\|^2 \tag{1}$$

using standard Gauss-Newton iteration both for finite and infinite map points, where $\operatorname{Proj}(\cdot)$ is the camera projection including radial distortion.

For a given camera pose $C = \begin{pmatrix} R & T \end{pmatrix}$, the transformation of a map point $X = (x, y, z, w)$ is

$$C \cdot X = R \begin{pmatrix} x \\ y \\ z \end{pmatrix} + T \cdot w. \tag{2}$$

Thus, finite points add a constraint on the camera translation ($w = 1$), while infinite points do not ($w = 0$). To ensure that the system is stable even if no finite points are observed (as in a pure panoramic tracking mode), we add a small regularization term to the linear system.

Additionally, we also apply a weight W_i to each measurement to balance the influence of infinite and finite features. Typically, we want to rather follow finite features (weighted with $W_i = 1$), therefore we weight infinite features with a factor $W_i = 0.01$ that was empirically determined.

3.2 Relocalization

We perform relocalization based on small blurry images (SBIs) that work with both 6DOF and panorama frames. A small blurry image is a small downscaled version of a video frame (40x30) with Gaussian blur applied. A history of SBIs is recorded with frames added at regular time intervals together with their 6DOF tracking poses. We query the history with an SBI computed from the current frame, resulting in a sorted set of similar SBI candidates. Each candidate is verified: First, the stored candidate pose is updated by estimating a relative 3DOF rotation between candidate and current frame with ESM [1]. Next, we execute the active search map tracker using the updated 6DOF pose as prior. If tracking succeeds for any of the candidates, the resulting pose is used to re-initialize relative pose tracking.

4 Keyframe Selection and Insertion

Selecting and inserting new keyframes into the map is an essential step in an efficient optimization-based SLAM system. To keep processing requirements low, only important keyframes should be chosen from the video stream. Important keyframes have two properties: (1) they image new parts of the scene, or known parts from different directions; (2) they have enough observations of known structure to ensure good connectivity in the map. Enabling the system to take more keyframes is essential to have a more detailed or expansive map. Our system is able to record more keyframes by relaxing the second requirement. Keyframes that are not well constrained in 6DOF through known 3D map features, but only through 2D-2D observations are recorded as well, if they image substantially new parts of the scene.

The top priority of keyframe insertion is to avoid map corruption by carefully selecting 6DOF keyframes and robustly localizing

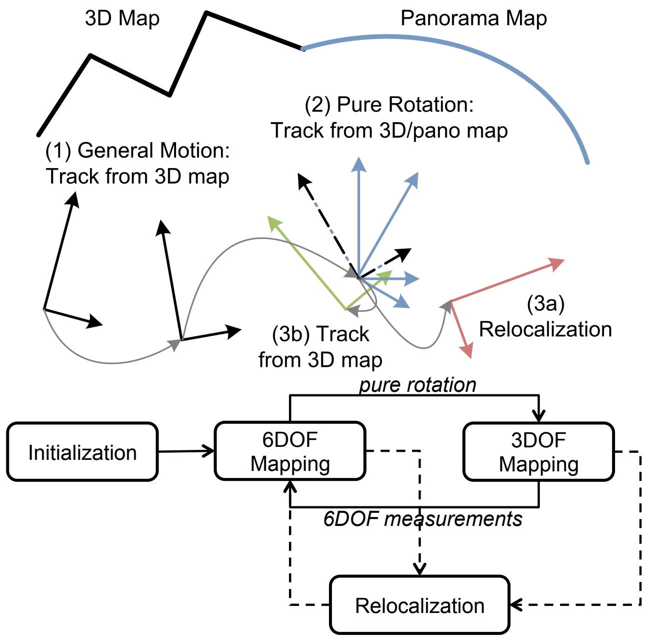


Figure 3: State diagram for the different states of keyframe selection during mapping. After initialization, the system starts to operate in full 6DOF mapping mode (1). If pure rotation motion is detected, the system switches to 3DOF mapping mode and creates a new panorama map (2). 6DOF measurements move the system back to full 6DOF operation (3b). In case of tracking failure, relocalization always recovers a full 6DOF pose (3a).

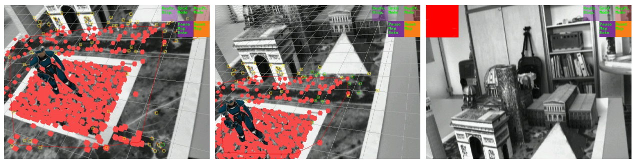
panorama keyframes. For 6DOF keyframe selection, we require a large number of correspondences for pose estimation and a conservative threshold for parallax. For panorama keyframe selection, we relax the rules: the pure rotation tracking may drift due to small camera translations. However, the relaxation does not harm the 3D map, as panorama keyframes are used for mapping only after they have been robustly localized later on. For localization, we again enforce strict thresholds on the pose estimation.

Similar to PTAM [11], our method uses heuristics for keyframe selection. We have adopted parallax and tracking quality criteria, and combined it with coverage and camera view angle criteria. In practice, distinguishing between true rotation-only motion and small translations is impossible due to measurement noise. Therefore, we use the parallax criterion to detect camera motion that allows for robust triangulation and 3D reconstruction. In the following, we describe the selection of keyframes in more detail. Figure 3 shows an overview of the different keyframe selection modes during operation.

4.1 6DOF mapping

6DOF keyframes can be selected whenever we have a camera pose that is fully constrained in 6DOF. This is the case if enough finite feature points are part of the pose estimation as described in Section 3.1. Furthermore, we select regular 6DOF keyframes when they generate enough parallax to existing keyframes while imaging a new part of the scene. Parallax is required for robust feature triangulation. New parts of the scene are characterized by areas in the image where few or no known features project into. Thus, we say that there are areas that are not well covered by the current map which indicates that the camera is observing unmapped scene regions.

Parallax is the angle α between the viewing directions of two camera frames (e.g. between the current frame and an existing


(a)

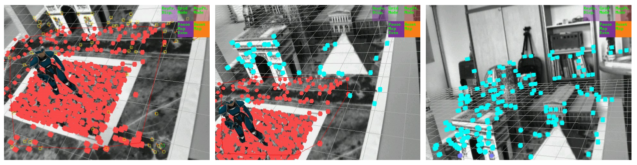

(b)

Figure 4: Handling pure-rotation camera motion at the map boundary. The same three non-consecutive frames as processed by (a) 6DOF SLAM and (b) hybrid SLAM (our approach). 6DOF SLAM discards low-parallax candidate keyframes, resulting in tracking failure due a lack of new finite map features (rendered in red). Hybrid SLAM detects the pure-rotation camera motion, creates a local panorama map, and continues camera tracking from infinite map features (rendered in cyan).

keyframe) onto the commonly viewed scene geometry. It can be approximated as $\alpha = 2\arctan(d/(2f))$ where f is the mean depth of the finite map features observed in both frames and d is the relative distance between the 3D locations of the frame pair. Note that in contrast to distances, the parallax angle is scale-independent. We empirically determined a parallax angle of $\alpha > 5\,\text{deg}$ as sufficient for 6DOF keyframe selection.

To estimate low coverage, we compute the frame area ratio that is covered with finite feature projections. We divide frames into a regular grid with 4*x*3 cells and project finite map features. Grid cells with a minimum number of contained features are considered covered. The coverage is the ratio c of the number of covered vs. all grid cells. We empirically determined that a frame is not well covered if the coverage ratio $c < 0.75$.

When inserting the 6DOF keyframe into the map, we add new finite map feature observations and new finite map features. New observations are added to features which have been successfully tracked in the keyframe. New finite 3D map features arise from frame-to-frame 2D feature tracking between the previous and the current 6DOF keyframe. With the insertion of the current keyframe, the new correspondences are added as new finite features having triangulated 3D positions $(x, y, z, 1)$ and two observations.

4.2 Localized panorama keyframe insertion

When the system detects low coverage (e.g. $c < 0.75$) but not enough parallax (e.g. $\alpha < 5\,\text{deg}$) between the current frame and existing keyframes, then tracking may fail if all known features become invisible. Low coverage indicates that the camera points towards unmapped scene regions. However, a regular 6DOF keyframe cannot be taken due to low parallax of pure-rotation camera motion. Thus, we select a panorama keyframe that is localized with respect to the 3D map. Figure 4 illustrates the different behaviour of standard 6DOF SLAM and our approach.

We detect pure rotation camera motion based on the history of tracked 6DOF poses. Tracked 6DOF poses are stored chronologically in a history. We compute the parallax angle between the current pose and the history and discard all poses with a sufficiently high parallax (e.g. $\alpha > 5\,\text{deg}$). The remaining history poses have similar 3D locations as the current frame. Finally, we compute the angles between viewing directions and detect pure rotation, if we find a pose in the history that has low parallax and a sufficient angle with respect to the current frame. The view difference angles β are normalized with the field-of-view angle γ of the calibrated camera, resulting in a angle ratio $r = \beta/\gamma$. We empirically determined a ratio $r > 0.2$ as sufficient for pure-rotation detection.

The selection of a localized panorama keyframe marks the beginning of a local panorama map. The system creates a new local panorama map P_j and assigns its reference keyframe C_j^r that defines the center of rotation.

When inserting the localized panorama keyframe into the map, we add new observations and new infinite features. New observations are added to finite features which have been successfully tracked in the keyframe. New infinite features are initialized from 2D image features. We apply a corner detector on the keyframe resulting in a 2D image feature set and subtract the projections of existing map features. The remaining 2D image features (u, v) are converted into rays $(x, y, z, 0)$ in world space and added as new infinite map features with a single observation.

4.3 Transition to 3DOF mapping

As soon as we do not observe sufficient finite features in the current frame, the hybrid pose estimation only updates the 3D orientation from infinite map features around a fixed 3D position using the local panorama map P_j. Slight camera translation may be estimated as additional rotation into the 3DOF poses.

4.4 Panorama keyframe insertion

The system continues to select panorama keyframes based on low coverage (e.g. $c < 0.8$) and sufficient rotation. Low coverage indicates the camera continues to explore unmapped scene regions. Rotation is computed as the difference angle between the viewing directions of the current frame and keyframe poses of the current panorama map. Again, we normalize the view difference angle with the camera field-of-view angle, and determined sufficient rotation if the ratio $r > 0.2$.

When inserting the panorama keyframe into the map, we add new infinite feature observations and new infinite features. New observations are added to infinite features which have been successfully tracked in the keyframe. New infinite features are computed the same way as with localized panorama keyframes (see above).

4.5 Transition back to 6DOF mapping

The system implicitly moves back to the full 6DOF operation, if it observes part of the 3D map again. Then the same criteria as before apply and a new 6DOF keyframe can be created.

With the transition, the panoramic tracking session ends and the current panorama map is disabled. Observations of map features within panorama keyframes of this session are disabled so that they are ignored by tracking future frames. We disable all feature observations of non-localized panorama keyframes. Localized keyframes keep their finite feature observations.

5 MAPPING

The mapping component refines the map through establishing new data associations and bundle adjustment optimization. In particular, it also estimates full 6DOF poses for panorama keyframes and triangulates infinite features to extend the 3D map. As part of data association refinement, we seek new keyframe-feature observations to further constrain existing feature locations and keyframe poses. We apply active search and descriptor matching techniques to establish 2D-2D correspondences.

We robustly localize panorama keyframes with respect to finite map features. Panorama keyframes are initialized with poses from panoramic tracking that are considered unreliable since we cannot estimate the poses in full 6DOF from infinite features and thus cannot measure camera translation. However, by establishing correspondences to existing finite map features, we can estimate full 6DOF poses. Thus, we effectively convert panorama keyframes into regular 6DOF keyframes.

We exploit the information stored in local panorama maps for 3D mapping by triangulating infinite feature observations. We employ descriptor matching to find 2D-2D correspondences between robustly localized keyframes, e.g. in separate local panorama maps that view the same scene regions. Correspondences which pass the verification tests constitute additional finite map features. Thus, we effectively convert infinite to finite features.

Finally, we also optimize the map with bundle adjustment [8]. Bundle adjustment updates the 6DOF poses of localized keyframes, and the 3D positions of finite map features by minimizing again the reprojection error between feature locations and observations in keyframes. Non-localized panorama keyframes and infinite features are not optimized. However, we maintain map consistency by adjusting the registration of panorama maps within the optimized 3D map.

5.1 Panorama keyframe localization

Robust localization of panorama keyframes is an iterative process that finds new correspondences between infinite features in the panorama keyframes and finite features observed in normal keyframes. Once enough such correspondences are established, a dependent panorama frame contained in a local panorama map can be localized with a full 6DOF pose and converted to a normal keyframe. This in turn can lead to further triangulation of infinite feature points, which again may allow for localizing other panorama keyframes.

To find correspondences to finite features, we employ both active search and wide-baseline matching using visual descriptors. The active search technique is borrowed from relative pose tracking. We iterate all finite map features. If a particular feature does not have an observation in the panorama keyframe, we project the feature and perform NCC matching in the neighborhood of the projected 2D image location. If we get a sufficiently high NCC score, we have found a 3D-2D correspondence and add a new feature observation.

Since the active search method relies on a reasonably accurate panorama keyframe pose, we additionally perform wide-baseline descriptor matching. We maintain a database (see Section 5.4) that contains descriptors of all finite feature observation patches in its leaves. The database is queried with a set of input descriptors from a panorama keyframe. These input descriptors are created from 2D image features. The query delivers a set of correspondences between 3D finite map features and 2D image features. We iterate the correspondences and check whether the panorama keyframe already has an inlier observation of the map feature. If not, we add a new observation.

Finally, we attempt to localize the panorama keyframe with its current 3D-2D correspondences. We retrieve all of the keyframes' finite features observations, resulting in a set of 3D-2D correspondences. The correspondences are passed to a RANSAC algorithm which employs a three-point-pose estimator [10]. The output pose is additionally refined with our robust relative pose estimator using the RANSAC inlier correspondences only. If the pose result is valid, we consider the panorama keyframe as robustly localized. Localized panorama keyframes are removed from their native local panorama map and are declared as reference of a new local panorama map.

5.2 Infinite feature triangulation

We triangulate infinite features which have observations in localized keyframe pairs with sufficient baseline. Since infinite features cannot be projected into keyframes outside of their native local panorama map, we apply descriptor matching to find relevant 2D-2D correspondences.

We maintain a second database (see Section 5.4) that keeps descriptors of infinite feature observations contained in localized panorama keyframes. After major modifications, e.g. after descriptors from a new keyframe have been added, the tree is queried with all localized keyframes. For each keyframe, we create a input descriptor set that contains 2D image features that do not coincide with existing finite feature observations. We receive a set of 2D-2D correspondences between two localized keyframes. Using the difference pose, we run an epipolar point-line check and discard outliers. We triangulate the 3D points and check if they are in front of both cameras. Finally, we enforce a minimum triangulation angle to avoid spurious depth estimates.

The remaining correspondences are converted into finite map features. We add the matched observation from the second keyframe and assign the triangulated 3D position.

5.3 3D and panorama map consistency

We do not include non-localized panorama keyframes and infinite map features in bundle adjustment. Since the poses of localized keyframes may be updated, the registration of local panorama maps within the 3D map becomes incorrect. We correct the registration of local panorama maps and its infinite features.

The pose of local panorama maps is defined by their reference keyframes, which are localized within the 3D map and updated in bundle adjustment. For each local panorama map we compute the difference pose of its reference keyframe before and after bundle adjustment. We apply this difference pose to the dependent non-localized keyframes. The infinite feature rays are re-evaluated by transforming the observations into world space with the updated poses and computing the centroid.

5.4 Feature descriptor database

We use offline-trained hierarchical k-means trees for nearest neighborhood matching that contain PhonySIFT descriptors [18] in its leaves. The PhonySIFT descriptor has 36 elements computed from 3x3 subregions with four bins each. For each map feature observation, we compute descriptors on multiple image pyramid levels to increase scale invariance. The trees have a fixed structure (branching factor of 8, 4 levels, resulting in 4096 leaves) and are trained offline from a large set of images. The trees allow adding and removing descriptors efficiently and are synchronized with the map as part of the mapping process.

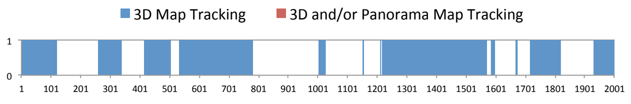
5.5 Map initialization

At system start-up, we employ a model-based detector and tracker [18] to create the initial map. Upon detection of a known planar image target, a first keyframe is created. The system continues to track the camera in 6DOF from the image target and additionally performs frame-to-frame tracking of 2D features. The second keyframe is selected as soon as sufficient 2D-2D correspondences can be robustly triangulated. Thus, two regular keyframes and the resulting finite map features constitute the initial map.

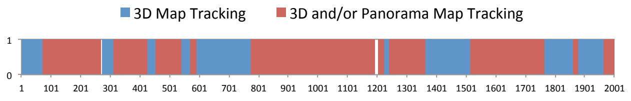
6 RESULTS

The implementation of our method builds upon a state-of-the-art optimization-based 6DOF SLAM system that runs in real-time on modern mobile phones.

We compared our hybrid SLAM method with standard 6DOF SLAM on several image sequences. The two methods perform equal on image sequences that show general camera motion such that the scene can be mapped with 6DOF keyframes only. This is no surprise, since tracking and mapping procedures are identical when operating with 6DOF keyframes and finite features only. Thus, to demonstrate the additional capabilities of our method, we



(a)



(b)

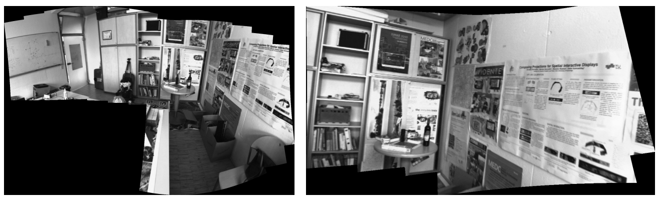
Figure 5: Tracking status timelines of (a) 6DOF and (b) hybrid SLAM. Each timeline depicts the tracking state for each frame. Filled bars indicate successful tracking or relocalization, empty bars indicate tracking failure. For hybrid SLAM, we distinguish between camera tracking from only finite 3D map features (rendered in blue) and camera tracking from infinite panorama map features (rendered in red), including the hybrid case of camera tracking from both finite and infinite map features. For standard SLAM, the camera is tracked from finite features only.

recorded a 2000-frame image sequence that shows several rotation-only camera pans that, in order to maintain tracking, require the selection of panorama keyframes.

The image sequence used for the comparison was recorded with a handheld camera and captures a well-textured room-sized indoor scene. The scene comprises a table-sized AR workspace in the foreground and the walls of the room in the background. We processed the image sequence with both hybrid and standard 6DOF SLAM methods and logged tracking and mapping statistics. We provide the visual output of both methods as full-length videos as part of the supplementary material. The evaluation was performed on a laptop PC equipped with a quad-core 2.5GHz CPU and 8GB of RAM. We used a PointGrey Firefly MV handheld camera and recorded the images with a resolution of 640x480 pixels.

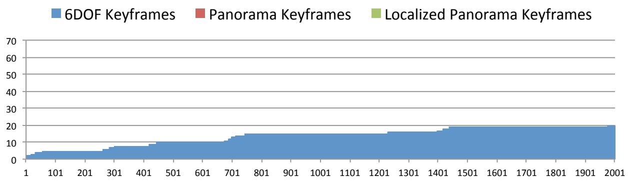
We present several timelines that depict tracking and mapping statistics for each frame of the image sequence. The timeline graphs do not consider the common map initialization phase and start with the first frame tracked from the initial 3D feature map having two 6DOF keyframes.

Tracking timeline. In Figure 5 we observe that the hybrid SLAM method tracked 98% of the frames, while the standard SLAM method only tracked 53% of the frames. Hybrid SLAM detected nine pure-rotation camera pans, resulting in an equal number of local panorama maps. While our method continues tracking

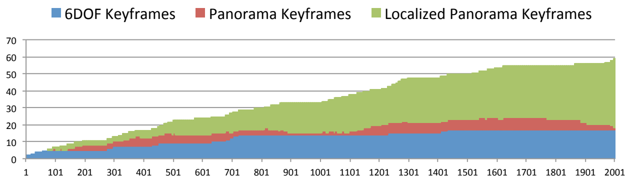


(a) (b)

Figure 6: Panoramas generated from keyframes of two local panorama maps. The local panorama map (a) was created in the period during frame 800 and 1200. The local panorama map (b) was created in the period during frame 1500 and 1800. Both panoramas depict the scene from two different view points with sufficient parallax for triangulation (e.g. the wall on the right-hand side).



(a)



(b)

Figure 7: Keyframe timelines of (a) 6DOF and (b) hybrid SLAM. Each timeline shows the number of mapped keyframes over the sequence. For hybrid SLAM, we distinguish between 6DOF (blue), panorama (red) and localized panorama keyframes (green). Standard SLAM selects only full 6DOF keyframes in blue.

from infinite features, standard SLAM tracking fails. For example, consider the period between frames 100 and 300: earlier, between frames 50 and 100, a pure-rotation camera movement started that was detected by hybrid SLAM around frame 80, resulting in the creation of a local panorama map. In contrast, standard SLAM discards keyframe candidates and runs out of map features around frame 110. The camera returns to the 3D mapped region around frame 270. Hybrid SLAM smoothly transitions from the local panorama map back onto the 3D map, while standard SLAM resumes finite map feature tracking after successful relocalization.

We conclude, that the detection of temporary pure-rotation camera movements and their mapping as local panorama maps improves the tracking performance of the hybrid SLAM method compared to the standard 6DOF SLAM method.

Keyframe timeline. In Figure 7 we observe that hybrid SLAM selects about three times as many keyframes as standard SLAM over the sequence. Even if we consider that standard SLAM only tracked half of the sequence and thus could have potentially selected twice as many keyframes, that is an increase of about one third (0.03 vs. 0.02 keyframes per successfully tracked frame). The increase mostly comes from additional panorama keyframes, while the number of 6DOF keyframes is even slightly lower.

We also see that hybrid SLAM quickly localizes panorama keyframes within the 3D map. The number of non-localized keyframes stays very low over the entire sequence. Furthermore, the localization of panorama keyframes enables the triangulation of infinite map features.

Map feature timeline. As we can see in Figure 8, hybrid SLAM maps contain about three to four times as much features as standard SLAM maps. Considering the finite map features only, hybrid SLAM maps are still two times larger than standard SLAM maps.

The ratio between finite and infinite features in hybrid SLAM maps is about 1:1. From Figures 8(b) and 8(c) we observe that about every third infinite feature was triangulated and thus contributed to 3D mapping. We assume some redundant infinite map features since in our current implementation we are not merging finite and infinite feature correspondences.

Figures 9 and 10 depict the reconstructed 3D maps of hybrid and standard SLAM, respectively. We see that the hybrid SLAM map reconstruction is considerably larger. More importantly, the hybrid

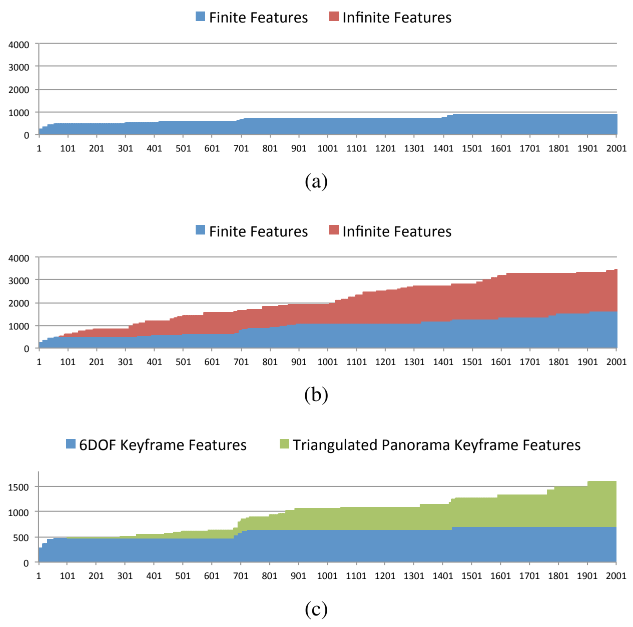


Figure 8: Map feature timelines of (a) 6DOF and (b,c) hybrid SLAM. Each in shows the number of map features over the sequence. Standard SLAM maps consist of finite features only (blue). For hybrid SLAM, we distinguish between finite (blue) and infinite (red) map features. Additionally, in (c) finite map features are split into finite features triangulated from 6DOF keyframes (blue) and converted infinite features triangulated from panorama keyframes (green).

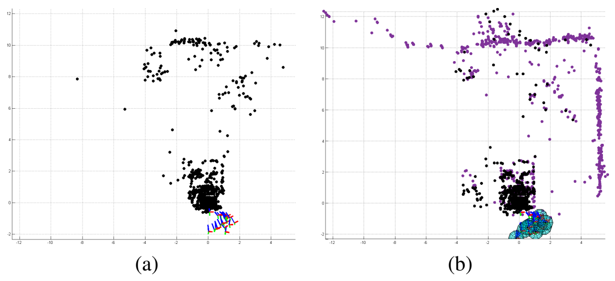


Figure 9: Final reconstructed 3D point feature maps of (a) 6DOF and (b) hybrid SLAM projected onto the XY-plane. Finite features triangulated from 6DOF keyframes are rendered in black, converted infinite features triangulated from panorama keyframes are rendered in purple. The keyframes are located south of the table that hosts the AR workspace.

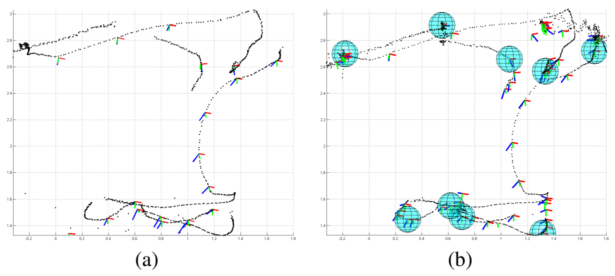


Figure 10: Reconstructed camera trajectories and keyframe locations of (a) 6DOF and (b) hybrid SLAM projected onto the XZ-plane. Local panorama maps of hybrid SLAM are rendered as spheres.

mapping approach was able to reconstruct far more background detail, e.g. the wall on the right is almost entirely reconstructed from infinite features, but not reconstructed at all by standard SLAM. Many of these infinite features have been triangulated with correspondences between keyframes two local panorama maps that have sufficient parallax, as can be seen in Figure 6.

Mapping time requirements. The increased map data volume and the additional operations for panorama keyframe localization and infinite feature triangulation require additional computational resources for mapping. Figure 11 shows the time requirements of (a) bundle adjustment, (b) panorama keyframe localization and (c) infinite feature triangulation mapping tasks along the image sequence. Each timing slot refers to either a (1) 6DOF or (2) panorama or (3) localized panorama keyframe event that triggers the execution of the mapping thread. Depending on the keyframe event type, the mapping tasks behave differently. Currently running tasks may be interrupted by an upcoming high-priority keyframe event.

The effort for bundle adjustment increases quadratically with the number of localized keyframes. Non-localized panorama keyframes are not included in the optimization. Note that bundle adjustment is executed with a varying number of iterations. The effort for panorama keyframe localization depends on the number of non-localized panorama keyframes. The effort for infinite feature matching and triangulation increases linearly with the number of localized keyframes which are used to query the descriptor database. The number of query keyframes depends on the type of keyframe event. For a new 6DOF keyframe, the database is queried with this single keyframe only, while for a new localized panorama keyframe, the database is updated with this keyframe and queried with all available localized keyframes. For new panorama keyframes, the database is not queried at all.

In our current implementation, the effort for infinite feature matching and triangulation sometimes exceeds the effort for bundle adjustment optimization. However, this can be improved with better mapping task scheduling and selection of input features for descriptor matching. We also consider a dedicated mapping task that removes redundant 6DOF and localized panorama keyframes with a similar method as described in [13].

We conclude that the information contained in local panorama maps can be fruitfully used for 3D reconstruction resulting in SLAM maps with an increased number of 3D features. Furthermore, we see that the map is build quicker and covers a larger extend, both due to the possibility for delayed matching of infinite features. In return, larger and denser finite feature maps allow for continued and more robust camera tracking.

6.1 Mobile phone application

Our hybrid SLAM system also runs in real-time on modern mobile phones. We used a Samsung Galaxy S2 equipped with a dual-core 1.2GHz ARM CPU and 2GB RAM running Android OS 4 for our tests. As expected, we found the overall robustness and performance restricted in comparison to the PC version. However, we applied the system in indoor and outdoor environments and created maps with e.g. about 100 keyframes and 4000 map features, including finite map features of 25 local panorama maps. Irrespective of the map size, tracking is mostly running with 20 to 30Hz. However, with increasing map sizes, we noticed congestion symptoms in the mapping thread, resulting in delayed map updates and consequent incremental pose tracking problems. Our implementation is

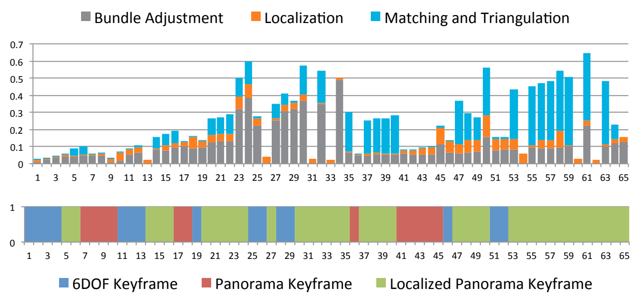


Figure 11: Major mapping tasks and its time requirements in seconds along the sequence. The tasks are triggered by diverse keyframe events. See text for further explanations.

not fully optimized yet, and we did not adjust all system parameters to the mobile phone platform. Figure 12 shows the mobile application handling a pure-rotation camera motion. Please also refer to the accompanying video.

7 Discussion

We provide comparisons with the original PTAM system and our closest related work. Furthermore, the effects of camera translation on the accuracy and robustness of panoramic SLAM are discussed.

7.1 Comparison with Klein et al.

We processed the 2000-frame image sequence from Section 6 with the publicly available PTAM software[1] described in the seminal paper of Klein et al. [11]. We modified the software to automatically select first and second keyframes and to log mapping and tracking statistics. The results are depicted in Figure 13.

From the tracking timeline in Figure 13(a), we see that PTAM tracks 63% of the frames successfully. The image sequence contains five major rotation movements as apparent in the tracking timeline of hybrid SLAM in Figure 5(b). PTAM fails to track two of the rotation movements completely, but manages to track three movements at least partially. Note that the comparison between PTAM and our SLAM system may not be entirely fair, since the tracking success criteria of PTAM is less strict than ours (e.g. PTAM expects to track less map features to report tracking success).

The PTAM system selects 19 keyframes to reconstruct a map containing about 9000 point features. The initial map created from two keyframes already consists of about 5000 features. The PTAM method is well-known to gain its tracking robustness in large parts from assembling large amounts of map features. In particular, PTAM does not verify if landmarks have been mapped already, resulting in many redundant map features. The 3D map view in Figure 13(b) shows that PTAM triangulates many features despite little

[1]http://www.robots.ox.ac.uk/ gk/PTAM/

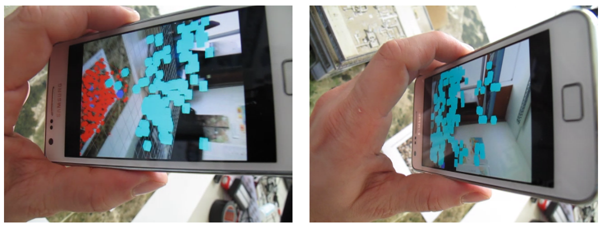
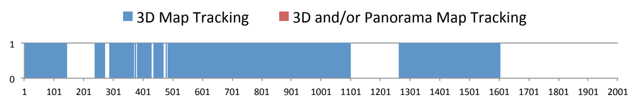
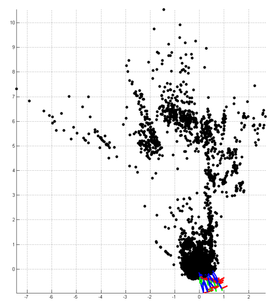


Figure 12: Hybrid SLAM system handling a pure-rotation camera movement in real-time on the mobile phone.



(a)



(b)

Figure 13: Evaluating PTAM on the 2000-frame image sequence of Section 6: (a) tracking status timeline indicating tracking success/failure comparable to Figure 5, (b) final reconstructed 3D point feature map. See text for discussion.

parallax, resulting in spurious depth estimates. For example, consider the scattered features along the camera view rays from bottom right to top left. Instead of unreliably triangulating features from keyframes with insufficient parallax, our hybrid SLAM system inserts these features as rays with infinite depth to local panorama maps.

In [13], Klein et al. presented a heavily modified PTAM method that managed maps with dramatically less features and was thus running in real-time on the iPhone 3G. We argue that even years later, the original PTAM method cannot be applied to modern mobile phones, regardless of their increased computational power. The goal must be to create high-quality maps with an optimized data volume that achieve similar robustness as PTAM. In this respect, we consider our system to be better suited for mobile phones. We provide two arguments: Firstly, comparing the reconstructed 3D maps of hybrid SLAM (Figure 9(b)) and PTAM (Figure 13(b)), we see that our system reconstructed a considerably larger portion of the scene with far less finite map features (1500 vs. 9000). Secondly, we evaluated the total time used for mapping tasks over the entire image sequence: PTAM uses about 36 seconds and thus about two times more than hybrid SLAM, which uses 18 seconds for mapping.

7.2 Comparison with Gauglitz et al.

In the following, we compare our method with Gauglitz et al. [9] - our closest related work. Both methods share the idea of supporting general and pure-rotation camera motion by combining 6DOF and panoramic SLAM methods. However, when looking at the details, the methods differ in many respects and appear as rather complementary approaches.

Mapping. The system of Gauglitz et al. handles arbitrary switches between general and pure-rotation camera motion, resulting in the creation of a new 3D/panorama submap with each context switch. Separate 3D submaps have distinct scales. Within a feature track, successive 3D and panorama maps are linked via a common keyframe. Upon tracking failure, a new feature track is started. Our system handles general camera motion alternated by temporary pure-rotation camera motions, resulting in local panorama maps that are registered within a single consistent 3D map. Given keyframe pairs with sufficient overlap exist, their system merges pairs of 3D-3D or panorama-panorama submaps. However, their system does not merge pairs of 3D-panorama submaps,

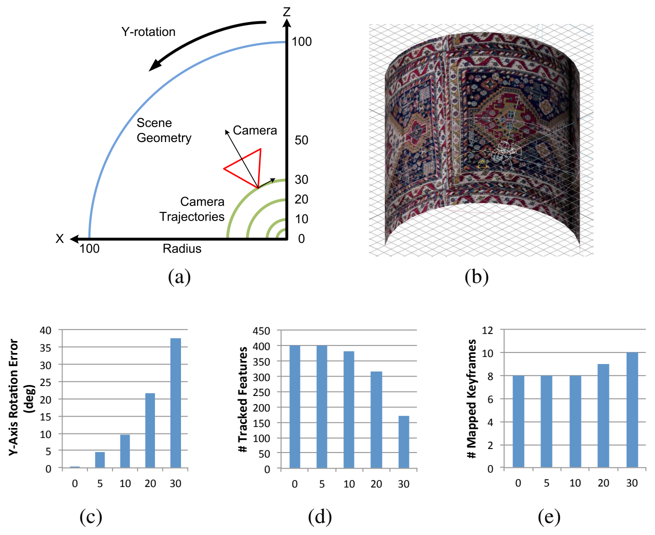


Figure 14: Effects of camera translation on panoramic SLAM accuracy and robustness: (a,b) virtual model used to generate synthesized image sequences. For each camera trajectory radius: (a) Y-axis rotation error in degrees, (b) number of successfully tracked map features, and (c) number of mapped keyframes.

while our system exploits the information contained in registered local panorama maps for 3D reconstruction.

Tracking. The system of Gauglitz et al. performs frame-to-frame tracking between the current frame and the latest keyframe of the current 3D or panorama map. From the resulting 2D-2D correspondences, multiple motion models are estimated. To get a global 6DOF camera pose, the existing keyframe poses are combined with a relative 6DOF pose converted from the motion models. The conversion of the motion models into a relative 6DOF pose is potentially ambiguous. In contrast, our system performs global 6DOF pose tracking by establishing 3D-2D correspondences between the current frame and the projected features of the optimized map, and robustly estimating a 6DOF pose. Thus, we additionally gain notion on whether the camera observes already mapped scene geometry, that allows for more fine-grained keyframe selection.

Keyframe selection. The system of Gauglitz et al. employs the generalized GRIC algorithm to distinguish homography/rotation and essential motion models, which may result in ambiguities, if (1) the scene is planar (2) the camera motion is not a true rotation. Our system employs the robustly tracked camera pose trajectory and scale-independent thresholds for parallax and camera view angles for the detection of pure-rotation camera motion.

Relocalization. The system of Gauglitz et al. performs delayed *loop-closing=recovery* relocalization, which actually refers to the merging of 3D-3D or panorama-panorama submap pairs that is done in the mapping backend. Our system performs immediate relocalization in the tracking frontend with respect to the optimized map.

Summary. The system of Gauglitz et al. aims at rapidly reconstructing the scene with potentially multiple submaps by continuously collecting image data in a visual-odometry-style tracking frontend. Our "classic" SLAM system aims at supporting interactive augmented reality applications by providing a persistent map coordinate system and 6DOF camera poses that allow to register and render virtual content on top of the reconstructed scene geometry.

7.3 Panoramic SLAM accuracy and robustness

With the following experiment we want to access the effects of camera translation onto the accuracy and robustness of 3DOF panoramic SLAM.

We created a virtual model that is used to render synthesized image sequences. The virtual model is depicted in Figure 14(a,b) and consists of a textured cylinder that is observed with a 64deg FOV camera that moves along a circular trajectory with varying radius. With this model, we simulate the real world situation where users perform imperfect camera rotation movements. Starting with the ideal case of a camera rotating around its nodal point, we introduce more and more camera translation to challenge the panoramic SLAM method. The cylinder has a fixed diameter of 100cm and has its center in the coordinate system origin. The circular camera trajectories are located in the XZ-plane and have a varying radius $r \in \{0, 5, 10, 20, 30cm\}$ resulting in the ground truth camera poses $P_i = [R_i|(0,0,-r)^t]$ with $R_1 = I$. We animated the camera to rotate 90deg around the Y-axis. For each camera trajectory radius, we rendered a 100-frame image sequence.

We processed the synthesized image sequences with our hybrid SLAM system. The system selects the first frame as panorama keyframe, assigns the initial pose $P_1 = [I|0]$, and initializes a map consisting of infinite features. The remaining images are processed in panoramic SLAM mode: further keyframes are mapped while the camera is tracked from infinite features, resulting in poses $P_i = [R_i|0]$ with 3DOF rotation matrices R_i while the camera location stays fixed at the origin.

We recorded tracking and mapping statistics depicted in Figure 14(c,d,e). The hybrid SLAM system tracked and mapped all image sequences successfully. In Figure 14(c) we depict the Y-axis rotation error of the estimated camera pose of the final sequence image for all radii, e.g. the pose estimated for the final 100th image frame with radius $r = 5$ has a relative error of about 5deg. The camera translation encoded as additional rotation by the pose estimator results in a quadratically increasing error.

We observe in Figure 14(d) that the number of successfully tracked map features drops with increasing radius. That means that local active search fails for an increasing number of map features since their projected location in the current frame is too far away from their actual location.

Figure 14(e) shows the number of mapped keyframes which increases with the radius. This behaviour is a consequence of (c) in combination with the "view angle difference" keyframe selection criterion: with increasing translation, more virtual rotation is estimated, the threshold is reached quicker, and keyframes are selected earlier in the sequence.

We conclude that panoramic SLAM can handle a considerable amount of translation. In our system, we select panorama keyframes in a greedy manner as our main priority is to maintain tracking. We accept erroneous poses resulting from camera translation, since panorama keyframes are localized in 6DOF anyway before being used for mapping.

8 Limitations and Future Work

Generally, our system is subject to the same restrictions as any feature-based visual SLAM systems. Tracking and mapping suffers from the lack of measurable point features in the input images which result from conditions such as untextured scene surfaces, motion blur, bad lighting, limited dynamic range of cameras, etc.

Our system combines 6DOF and panoramic SLAM methods and dynamically switches between these operation modes depending on the camera motion. While in panorama mode, we do not support transitions from pure rotation to general camera motion. This is due to the fact that panoramic SLAM only allows for tracking the camera pose as 3DOF rotation. Any camera translation is encoded as additional rotation, resulting in inaccurate poses. While panoramic

SLAM shows quite some robustness against translation (see Section 7.3), there are cases where these inaccuracies result in tracking failure, e.g. due to undetected camera loops. In rare cases, panorama maps may even become corrupted beyond recovery. While this does not harm the global 3D map, tracking requires to be reinitialized by relocalization with respect to the 3D map.

As described above, camera translation cannot be measured during 3DOF panoramic tracking. When returning from the panorama map onto the 3D map, the unrecognised camera translation results in 6DOF pose offsets. If these offsets are sufficiently large, local active search fails to match projected finite map features, resulting in tracking failure due to a lack of correspondences. In these cases, relocalization is required. Otherwise, we iteratively track mixed sets of finite and infinite map features and transition seamlessly.

Infinite feature matching and triangulation generates a moderate number of outliers. We have not found these outliers severely disturbing our system. Nevertheless, we consider a mapping task that removes outlier observations and features, e.g. by verifying reprojection errors.

The runtime behaviour of our mapping method can be improved by revisiting descriptor matching and introducing redundant keyframe removal. The performance of feature matching can be improved with better selection of input features. Removing redundant keyframes and its map feature observations should reduce the computational complexity of bundle adjustment optimization as well as tracking.

The panorama maps estimated by the system are not yet full first class citizens of the SLAM map. During rotation motion a panorama is extended, but once it is closed, the panorama keyframes are not used for tracking or localization any more. In future work, we plan to address this limitation and allow normal tracking as well as relocalization to access the frames and infinite features of closed panorama maps. These maps may then also be extended again, if the camera motion requires this.

A more complex direction is to incorporate translation motion from a panorama map that cannot be referenced back into the single 3D map. This would extend our system to the mapping capabilities of the system described by Gauglitz et al. [9]. In this case the long-term linking of different 3D maps is the main use case to allow creation of a single global 3D map after enough information has been recorded.

9 Conclusions

The presented system demonstrates that the extension of a standard 6DOF optimization-based SLAM system with a dedicated panorama mode yields several improvements. The system is able to track through motions characterized as pure rotations that are difficult in typical monocular SLAM systems. More importantly, the local panorama maps created during such motion phases can contribute substantially to a richer map. They allow estimating more points in less time in the near field of the camera as well as mapping background structure through wide baseline matching between these local panorama maps. All of this is enabled by making these additional panorama map frames available for matching against later incoming frames.

By restricting ourselves to continuous online tracking operations we obtain a robust and well-performing system, because it always has a known map to track from. Therefore, it can use active search combined with motion models which has been demonstrated to work well under fast motions and difficult lighting situations. Overall, we believe that the present hybrid approach is a valuable combination that naturally extends the current design of optimization-based monocular SLAM systems.

Acknowledgements

This work was sponsored by the Christian Doppler Laboratory for Handheld Augmented Reality and Qualcomm Technologies Inc. We would like to thank Clemens Arth for his support.

References

[1] S. Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems 2004*, volume 1, pages 943–948 vol.1, 2004.

[2] J. Civera, A. Davison, and J. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945, Oct. 2008.

[3] J. Civera, A. Davison, and J. M. M. Montiel. Interacting multiple model monocular SLAM. In *IEEE International Conference on Robotics and Automation, 2008*, pages 3704–3709, 2008.

[4] J. Civera, A. J. Davison, J. A. Magallon, and J. M. Montiel. Drift-free real-time sequential mosaicing. *Int. J. Comput. Vision*, 81(2):128–137, Feb. 2009.

[5] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 2, pages 1403–1410. IEEE Computer Society, 2003.

[6] A. J. Davison and N. D. Molton. MonoSLAM: real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1052–1067, 2007.

[7] E. Eade and T. Drummond. Scalable monocular SLAM. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 469–476. IEEE Computer Society, 2006.

[8] C. Engels, H. Stewénius, and D. Nistér. Bundle adjustment rules. In *In Photogrammetric Computer Vision*, 2006.

[9] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Hollerer. Live tracking and mapping from both general and rotation-only camera motion. In *Proceedings IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 13–22, 2012.

[10] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[11] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10. IEEE Computer Society, 2007.

[12] G. Klein and D. Murray. Improving the agility of keyframe-based SLAM. In *Proceedings of the 10th European Conference on Computer Vision: Part II*, pages 802–815. Springer-Verlag, 2008.

[13] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 83–86. IEEE Computer Society, 2009.

[14] S. Lovegrove and A. J. Davison. Real-time spherical mosaicing using whole image alignment. In *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*, pages 73–86. Springer-Verlag, 2010.

[15] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In *2011 IEEE International Conference on Computer Vision (ICCV)*, pages 2320–2327. IEEE, Nov. 2011.

[16] P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *Int. J. Comput. Vision*, 50(1):35–61, Oct. 2002.

[17] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg. Real-time panoramic mapping and tracking on mobile phones. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 211–218, 2010.

[18] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Real-time detection and tracking for augmented reality on mobile phones. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):355–368, June 2010.