# Model Estimation and Selection towards Unconstrained Real-Time Tracking and Mapping

Steffen Gauglitz, *Student Member, IEEE* , Chris Sweeney, *Student Member, IEEE* ,
Jonathan Ventura, *Member, IEEE* , Matthew Turk, *Fellow, IEEE* , and Tobias Höllerer, *Member, IEEE*

**Abstract—**We present an approach and prototype implementation to initialization-free real-time tracking and mapping that supports any type of camera motion in 3D environments, that is, parallax-inducing as well as rotation-only motions. Our approach effectively behaves like a keyframe-based Simultaneous Localization and Mapping system or a panorama tracking and mapping system, depending on the camera movement. It seamlessly switches between the two modes and is thus able to track and map through arbitrary sequences of parallax-inducing and rotation-only camera movements. The system integrates both model-based and model-free tracking, automatically choosing between the two depending on the situation, and subsequently uses the "Geometric Robust Information Criterion" to decide whether the current camera motion can best be represented as a parallax-inducing motion or a rotation-only motion. It continues to collect and map data after tracking failure by creating separate tracks which are later merged if they are found to overlap. This is in contrast to most existing tracking and mapping systems, which suspend tracking and mapping and thus discard valuable data until relocalization with respect to the initial map is successful. We tested our prototype implementation on a variety of video sequences, successfully tracking through different camera motions and fully automatically building combinations of panoramas and 3D structure.

**Index Terms—**Visual tracking, simultaneous localization and mapping, panorama mapping, model selection, GRIC score, keyframe-based, initialization-free, augmented reality

## 1 Introduction

Over the past decade, there has been a tremendous amount of work on real-time monocular vision-based tracking and mapping (T&M) systems, that is, systems that simultaneously determine the position and/or orientation of the camera with respect to a previously unknown environment and create a model of this environment. In addition to other applications, T&M is an important enabling technology for Augmented Reality (AR) in unprepared environments.

Two important characteristics of a T&M system are the type of camera motion and the geometry of the environment that it supports. For example, a system may assume a planar environment [31] or a camera that is rotating around its optical center [7], [42]. Simultaneous Localization and Mapping (SLAM) systems such as [6], [8], [17], [27] can deal with environments of arbitrary geometry and any camera motion that induces parallax. However, with few exceptions [4], they do not support rotation-only camera motion: Their mapping is intrinsically built upon triangulation of features; thus, they require that each feature be observed from two distinct camera locations and may produce degenerate maps or fail completely if the camera rotates from one part of the environment to another.

Therefore, most SLAM systems need to be initialized with a distinct "traveling" movement of the camera *for each newly observed part of the environment*, and the required travel distance is directly proportional to the distance to the environment. This restriction is acceptable for vehicle navigation or if building a model of the environment is the user's main intent. However, it is a major limitation for the use of SLAM systems in AR, where the environment modeling is assumed to be done in the background and ideally transparent to the user, who should not be required to move a certain way in order to make the system work. Moreover, rotation-only "looking around" is a very natural motion and may occur in many AR applications [21], [42].

Our particular motivation is the use of a T&M system for remote collaboration [10], where the emerging model of the environment is used to allow a physically remote user to view and navigate the environment, and AR annotations that are registered to this model are used to communicate visual/spatial information. For this and many other applications, the paradigm for modeling the environment should be to make the best possible use of all data that can be casually collected and to enable viewing and placement of annotations for as much time as possible. In particular, this means not forcing the user to concentrate on model building, and not discarding all frames that stem from rotation-only movements (as in most SLAM systems) or translations (as with panorama mapping).

In this paper, we present an approach and prototype implementation that fulfills these criteria: We describe a real-time tracking and mapping system that explicitly

- *S. Gauglitz, C. Sweeney, M. Turk, and T. Höllerer are with the Department of Computer Science, University of California, Santa Barbara, Santa Barbara, CA 93106. E-mail: {sgauglitz, cmsweeney, mturk, holl}@cs.ucsb.edu.*
- *J. Ventura is with the Institute for Computer Graphics and Vision, Graz University of Technology, Inffeldgasse 16, 8010 Graz, Austria. E-mail: ventura@icg.tugraz.at.*

*Manuscript received 22 Feb. 2013; revised 5 Aug. 2013; accepted 3 Oct. 2013; date of publication 17 Oct. 2013; date of current version 25 Apr. 2014. Recommended for acceptance by M. Gandy, K. Kiyokawa, and G. Reitmayr. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TVCG.2013.243*

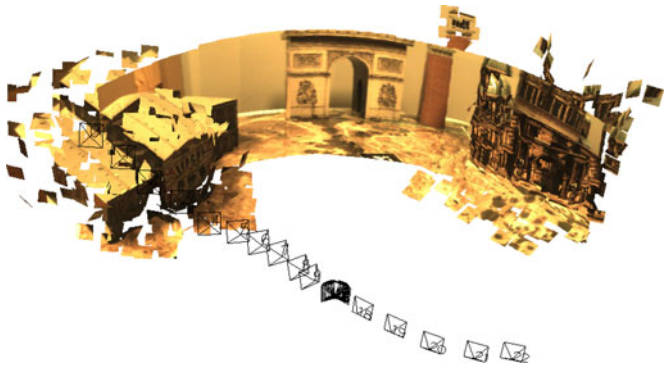


Fig. 1. Our system supports both parallax-inducing and rotation-only camera motion. In the former case, it acts as a SLAM system and models 3D structure in the environment; in the latter case, it acts as a panorama mapper. It seamlessly switches between the two modes, thus being able to track and map through arbitrary sequences of parallax-inducing and rotation-only camera movements and—fully automatically and in real time—creating combinations of 3D structure and panoramic maps, as shown here.

supports both parallax-inducing and rotation-only camera motions in 3D environments, does not need a separate initialization step, and continues to collect data despite intermittent tracking loss (see Fig. 1). In the case of intermittent tracking loss, it creates several disjoint maps which are later merged if possible. One key element of our approach is the use of the "Geometric Robust Information Criterion" (GRIC) by Torr [41] (adapted to support large search regions (Section 6.1) and absolute pose models (Section 6.2)) to decide whether the current camera motion can best be represented as a parallax-inducing motion or a rotation-only motion.

An earlier version of our system was presented at ISMAR 2012 [11]. We have significantly extended this work, incorporated new features and improvements, obtained new quantitative and qualitative results, and added discussion. Most notably, we have generalized the model estimation and selection to include both two-view relations as well as 3D-model-based pose estimation, thus showing that an important core component of state-of-the-art SLAM systems can be integrated in our framework.

## 2 Related Work

### 2.1 Monocular Vision-Based SLAM

SLAM is the problem of determining the pose of the observer relative to an unknown environment while concurrently creating a model of the environment (which may have arbitrary geometric complexity) as the observer moves around. In the case of monocular vision-based SLAM [6], the only sensor used to accomplish this task is a single camera.

Filter-based SLAM systems [3], [6], [8] maintain estimates of both camera and feature positions (i.e., the map) in a large state vector which is updated using Kalman filters in each frame. In contrast, keyframe-based systems [17], [18], [26] track features in each frame, but use only selected frames to update their map, typically using bundle adjustment for the latter. While all aforementioned systems are based on sparse features, Newcombe et al. [27] presented a keyframe-based system that uses dense mapping. They report extremely robust results, but (in contrast to the above systems) require a GPU for real-time operation.

In both types of systems, the map is designed to store structure-from-motion (SfM) data, that is, feature positions in 3D that have been triangulated using observations from multiple viewpoints. Thus, they require parallax-inducing camera motion in order to bootstrap their map [6], [17], [27], as otherwise, the features cannot be triangulated and integrated into the map. In some systems [17], [27], the initialization is performed as a dedicated separate step, and tracking quality crucially depends on the quality of this initialization. Rotation-only motions are supported only if they are constrained to the already observed part of the scene.

For filter-based systems, an alternative, 6D parametrization of the feature locations [3] can provide a remedy: Here, rotation-only motions are supported by admitting features with a depth prior that represents extreme uncertainty and filtering the features through multiple motion models [4] to constrain their uncertainty. However, this support comes at a high computational cost: The already high cost of filtering of each feature point is further increased by doubling the dimensionality of the feature state vector as well as computing the results for multiple motion models (note that Civera et al. [4] use seven models in each frame). As a result, the number of features that can be tracked in real time, which is typically already smaller for filter-based SLAM than for keyframe-based SLAM,[1] is further decreased: With just one motion model, Civera et al. [3] mention map sizes of up to one hundred features, compared to several thousand for PTAM [17]. For the approach with multiple models [4], no real-time implementation is described. In the case of long sequences of camera rotation, many of those computation cycles are spent filtering data where no gain is to be expected (namely, on re-estimating the (still undefined) feature depth).[2]

Therefore, we consider it an advantage of our approach that we explicitly switch to panoramic mapping if supported by the observations, thus taking advantage of some of the advantages that panoramic mapping offers, such as a robust outlier-resilient model (homography) and a straightforward mapping of the entire frame instead of sparse features, both of which are especially important for AR. On the other hand, the approach of Civera et al. [4] may be preferable when modeling the transition between two types of movement or when only some of the features exhibit parallax.

While admitting features without depth could in principle be adopted for keyframe-based SLAM (in [19], this approach is employed to admit features before their depths are known), the ability to rely on them exclusively would require fundamental and possibly costly changes to the underlying mapping and bundle adjustment. We are not aware of any existing keyframe-based SLAM system which explicitly supports rotation-only movements.

1. For an interesting analysis of the relative computational cost of the two approaches see Strasdat et al. [39].

2. Conceptually, whether or not existing features can be triangulated using a new camera view depends more on the camera movement than on each individual feature (particularly since feature depths are likely to be correlated), and could thus be decided once per frame. In [3], [4], however, the question is answered for each feature individually.

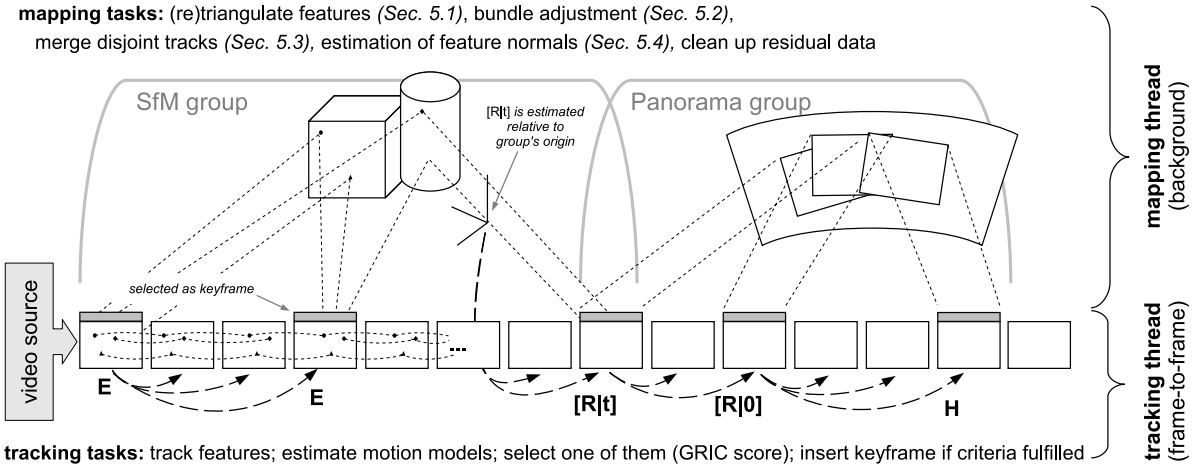



Fig. 2. Conceptual overview of the system's operation.

### 2.2 Panorama Tracking and Mapping

Like a SLAM system, a panorama tracking and mapping system aims at modeling the environment while determining the pose of the camera,[3] but in this case, the camera is assumed to rotate around its optical center, so that only its orientation has to be determined. An early real-time system is Envisor [7]. Wagner et al. [42] describe a system that operates robustly and in real time on a mobile device.

### 2.3 Stereo and Non-Visual Environment Modeling

In theory, using stereo cameras eliminates the problem of requiring the camera to travel, since the baseline required to triangulate features is built-in. In practice, however, using stereo cameras is only a partial remedy, since the baseline has to be significant in relation to the distance to the environment in order to reliably estimate depth. Thus, a wearable stereo system would be unable to map a building across the street without requiring the user to provide additional baseline by traveling (while a panorama system, though unable to provide depth, would produce very usable information).

Further, systems based on alternative sensor types should be considered. In particular, active depth sensors based on time-of-flight or structured light [22], [28] have recently generated significant interest and can arguably provide for more detailed models and more robust tracking when lighting conditions for vision-based tracking are unfavorable. However, all of these systems have other inherent limitations such as limited range, inability to work in sunlight, power consumption, and need for additional special hardware.

We thus argue that there are both theoretical and practical interests in solving T&M using monocular vision.

### 2.4 Model Selection, GRIC Score and Applications

Model selection is defined as choosing the model that best describes a set of observations. Various metrics (frequently dubbed "information criteria") have been proposed to assess the fitness of a particular model given the data, for example minimum description length [34], AIC [1], and BIC [36]. Torr described both a maximum likelihood [40] and a Bayesian formulation [41] of GRIC, and we use the latter in this work. These information criteria are general in nature and can be applied to various types of models. In SfM, the GRIC score has been applied particularly to detect homographies in order to *avoid* them during keyframe selection [32], [33]. In contrast, we use the GRIC score to select between two models, but we use the data in either case.

## 3 System Overview

Our concept borrows two key ideas from Klein and Murray's PTAM system [17], [18], [19], namely, the central role of keyframes and the splitting of tracking and mapping into two parallel threads.

The split and design of the threads follows two guidelines: 1) the tracking thread should be as fast as possible and leave all tasks that are not imminent in order for the next frame to be processed to the mapping thread, which runs asynchronously in the background; and 2) the first steps in the pipeline should not be dependent on the motion model (parallax-inducing versus rotation-only) that will be selected, in order to minimize redundant computations.

Fig. 2 presents a conceptual overview of the system's operation. Briefly summarized, the system operates as follows: The tracking thread receives a new frame and locates features via template-based matching, which stem either from the previous frame or, if a map of the environment has already been built, were projected from the map into the frame. From these feature correspondences, it then estimates both a model for parallax-inducing camera motion as well as a model for rotation-only motion, and selects the model that better represents the data via comparing their GRIC scores. Under certain conditions, a *keyframe* is inserted. Consecutive keyframes of the same kind (SfM or panorama) are collected into a *keyframe group*. The mapping thread operates on these sets of collected keyframes and creates an appropriate environment model (SfM or panorama).

3. Depending on the exact interpretation of the word "localization," one may argue that a panorama tracking and mapping system *is* a SLAM system, as advocated by Lovegrove and Davison [24]. This interpretation would render the term "tracking and mapping," which we chose to refer to the general class following [17], [31], [42], obsolete.

The tracking and mapping threads are described in detail in Sections 4 and 5, respectively. Section 6 describes the problem of model selection and the GRIC score.

### 3.1 Two-View Relations and Model-Based Poses

Conceptually, two two-view relations (namely, the essential matrix $E$ and the homography $H$) are all-encompassing in that they describe all cases of camera movements that can occur in a static environment. Consequently, our previous implementation used only these two models [11]. However, this implementation does not make optimal use of the environment model that is built in the background; by sequentially estimating $E$ from potentially small baselines, tracking remains relatively brittle and jittery (cf. Fig. 10).

Therefore, we have extended our concept to include both two-view relations (for initialization, after tracking loss, or when rotating into uncharted territory) as well as two model-based absolute poses $[R|t]$ and $[R|0]$[4] (whenever the camera observes a part of the environment for which a 3D model has been built). It should be noted that, in each frame, we still estimate only two models (the two-view relations *or* the absolute pose models); thus, the computational load has not increased.

The integration of absolute pose models has two further advantages: First, it allows to distinguish rotation-only movement from planar environments (which is difficult in the model-free case, since in both cases, the resulting transformations are described by the same relation, namely, a homography). Second, it decreases the risk of fragmentation of the model into connected sub-maps of alternating types (which a linear sequence of $E$'s and $H$'s may produce, as discussed in [11]), since the system can connect incoming frames directly to an existing model (rather than only to the previous keyframe).

*Why is estimating* $[R|0]$ *necessary?* Unlike in the case of $E$ versus $H$, when a model is available, the absolute pose $[R|t]$ is well-defined and can be estimated irrespective of the (relative) camera motion. Thus, it is less obvious why estimating $[R|0]$ and the subsequent model selection step is necessary. We do so for the following reason: Consider the case that the camera observes a known scene (thus using model-based tracking), then rotates towards unobserved parts of the scene. Due to the rotation-only movement, no new features can be triangulated. If $[R|t]$ is the only model-based pose that is estimated, and has priority as long as enough triangulated features are visible, the system will switch to $H$ only when very few (if any) of the existing features are left, risking tracking loss in between and generating a panorama that has very little overlap with the existing model. By estimating $[R|0]$ and thus explicitly switching to panorama mode (if $[R|0]$ proves to better represent the data), we can start building the panorama immediately, ensuring that it is well-connected to the existing structure by the time the system switches to $H$, seamlessly continuing the panorama as it extends into newly observed parts of the scene.

4. More precisely, $[R|t]_{\text{prev}} \circ [R|(0,0,0)^T]$, where $[R|t]_{\text{prev}}$ is the (fixed) pose of the previous keyframe.

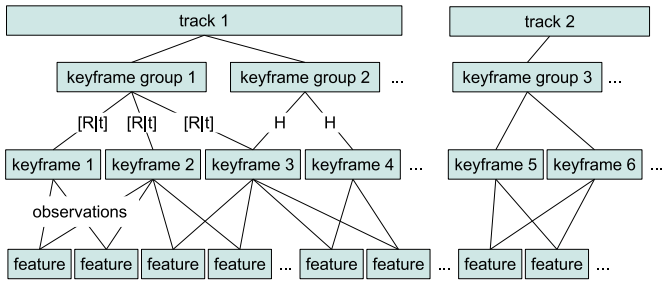



Fig. 3. Schematic overview of the main data structures used to store the emerging map(s).

### 3.2 Data Structures

Fig. 3 visualizes the main data objects that store the current system state and the emerging map as well as their relations.

The most central element is the keyframe group: each keyframe group governs one sub-map consisting of a set of keyframes which are either all linked by 3D pose information (SfM group) or all linked by homographies (panorama group). The keyframe group also determines the frame of reference, with respect to which all pose information is stored. A keyframe may be part of a panorama group as well as a SfM group (e.g., keyframe 3 in Fig. 3), in which case it gets assigned a pose in both groups. The set of keyframes and keyframe groups can be thought of as a graph, similar to the representations by Eade and Drummond [8] and Klopschitz et al. [20]. Initially, this graph is linear, consisting of a chain of $E$'s and $H$'s modeling the (linear) video stream. As soon as the system has built a model of the environment and is using model-based tracking however, new keyframes are linked directly to the model, creating a branching graph.

When tracking is lost, all links to the current map are lost, and the tracker starts a new track. Initially, the new track is completely unconnected to the previous data, but can later be merged (if there is some overlap in what is observed during both tracks) as explained in Section 5.3.

## 4 Tracking

A flowchart of the tracking thread is presented in Fig. 4. This section describes the main operations in detail.

When a new keyframe is added, new keypoints are detected using a corner detector (such as FAST [35] or Shi-Tomasi [37]) in all areas not already covered by keypoints. We enforce a spatially well-distributed set of keypoints, which was shown to improve tracking robustness [9], [13], by overlaying a simple rectangular grid over the image and selecting the strongest keypoint in each cell.

To project existing features into the frame and predict their appearance, as much information about the feature as possible is used. That is, if the 3D position and the normal (cf. Section 5.4) of the feature are available, it is sampled from the original keyframe using a fully qualified homography (representing a 3D plane rotation). If only the 2D location of the feature is known, the patch is sampled in 2D.

### 4.1 Coarse-to-Fine Feature Tracking

Frame-to-frame feature correspondences are created using a multi-level (i.e., coarse-to-fine), active search patch tracker

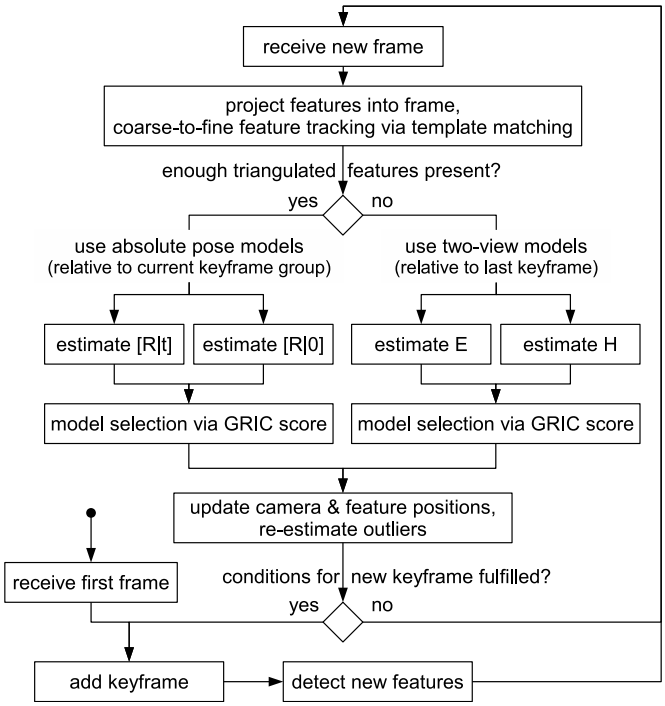



Fig. 4. Flowchart of the tracking thread.

with normalized cross-correlation (NCC)-based template matching. On the full-resolution image, the feature location is refined to subpixel accuracy by using a quadratic fit to neighboring scores.

This is similar to the keypoint tracking by other systems [17], [42], [43]; however in contrast to these system, the multi-level tracking is executed on a per-feature basis (instead of interleaved with the pose estimation), since we do not know which type of camera motion to expect (and thus which model to enforce) until after the tracking step. We have designed, but not yet fully integrated, a more sophisticated approach that retains the advantages of the interleaved pose estimation, and discuss this approach in Section 8.2.

### 4.2 Model Estimation and Outlier Re-Estimation

After all features are located in the full-resolution image, the motion model is estimated. If a 3D model of the observed scene is available (that is, there exists a SfM group which contains sufficiently many of the successfully tracked features), we estimate $[R|t]$ and $[R|0]$ using iterative gradient descent of an M-estimator (cf. Table 1 bottom). Otherwise, we instead estimate both a homography $H$ and an essential matrix $E$ between the previous keyframe and the current frame using MAPSAC (cf. Table 1 top).

The probability density function that is assumed for inliers and outliers is an important part of the model selection process. Here, we make the common assumption that inliers are distributed normally with measurement error $\sigma$, and outliers are distributed uniformly across the search region. Thus, after model estimation, the measurement error $\sigma$ and inlier ratio $\gamma$, which are needed for the model selection step, are estimated using expectation-maximization.

Next, the GRIC score is computed for both models, and the better model (i.e., the one with lower GRIC score) is selected. The model selection and the GRIC score will be explained in more detail in Section 6. If $E$ is determined to be the better fit, it is decomposed into a relative pose $[R|t]_{\text{rel}}$.

TABLE 1
Estimation Algorithms for the Four Models Considered Here

| Model | Estimation algorithm |
|---|---|
| Essential matrix $E$ | MAPSAC [41] with five-point algorithm [29] |
| Homography $H$ | MAPSAC [41] |
| Abs. pose $[R\|t]$ | Gradient descent with M-estimator (cf. [17]) |
| Abs. orientation $[R\|0]$ | Gradient descent with M-estimator (cf. [42]) |

*In each frame, only two models are estimated.*

In an attempt to retain individual features as long as possible, outliers are re-estimated after the model selection. This is trivial in the cases of $H$, $[R|t]$, and $[R|0]$, since the model defines a unique mapping for each point. In the case of $E$, each outlier is re-estimated by running the NCC-based template matching again on a thin rectangular matching area that was rotated to align with the epipolar line. Features that prove unreliable (i.e., that repeatedly are outliers and need to be re-estimated) are removed.

If no keyframe is added (cf. next section), processing of this frame is completed.

### 4.3 Inserting a New Keyframe

The current frame is added as a new keyframe $k_{\text{new}}$ when several conditions (similar to the ones suggested by Klein and Murray [17]) are met: 1) tracking quality is good (as determined by the fraction of inliers that MAPSAC finds); 2) enough time has passed since the last keyframe insertion; and 3) in the case of rotation-only motion ($H$ and $[R|0]$), when the median 2D distance that the keypoints "traveled" since the last keyframe is large enough, and in the case of parallax-inducing motion ($E$ and $[R|t]$), when the median feature triangulation angle is large enough.

If the estimated motion model connects $k_{\text{new}}$ to an existing keyframe group of the respective type (i.e., a SfM group for $E$ or $[R|t]$, and a panorama group for $H$ or $[R|0]$), $k_{\text{new}}$ gets merged into the existing group (detailed below for the case of $E$, and straightforward in all other cases).

Otherwise, a new keyframe group consisting of $k_{\text{new}}$ and the previous keyframe $k_{\text{prev}}$ gets created. Insertion of a new group marks the beginning of a new sub-map, but it does not invalidate any of the existing map data: particularly during model-based tracking, where both feature and camera positions are known in 3D, the tracker can still refer to all currently registered features and project them into incoming frames. Note that a switch from $[R|0]$ to $[R|t]$ does *not* cause creation of a new SfM group: coming from $[R|0]$ implies that 3D features are visible and thus that there is an SfM group to which the new keyframe is connected even after intermittent rotation-only motion.

Lastly, new features are detected in all uncovered image regions by applying the same grid as in the first frame and choosing new features for each cell that is not covered by currently tracked features.

*Merging essential matrices.* While $E$ can be decomposed into relative pose information $[R_{\text{rel}}|t_{\text{rel}}]$ between the new frame and the previous keyframe $k_{\text{prev}}$, the scale of $t_{\text{rel}}$ is

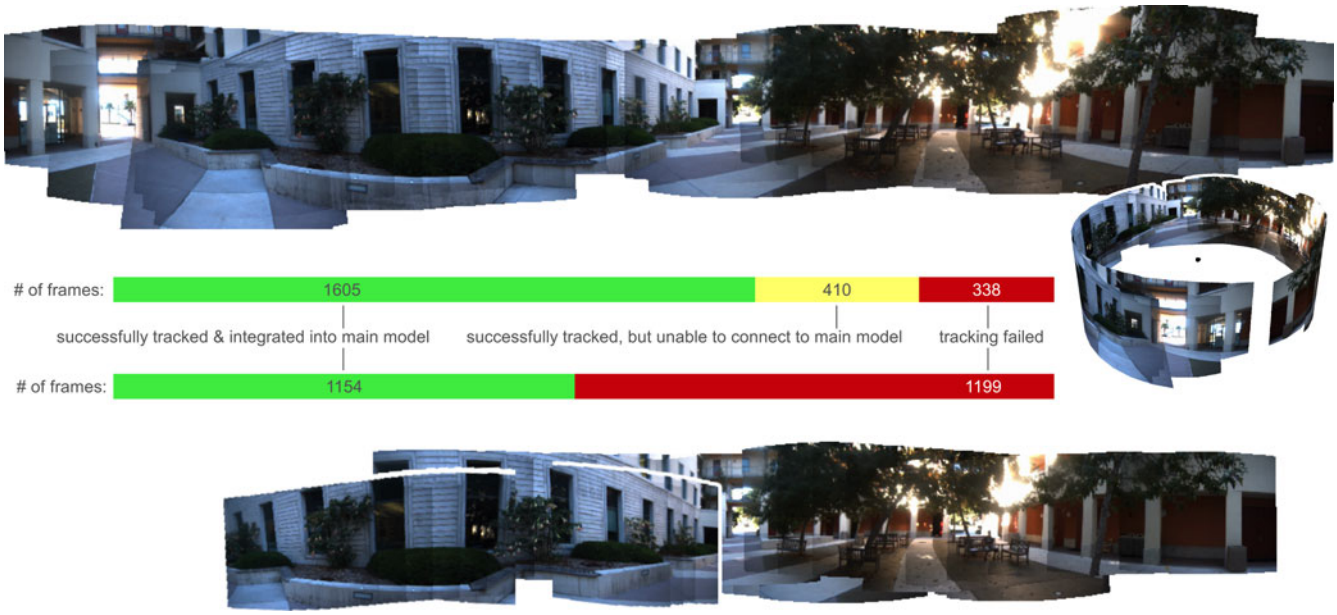



Fig. 5. Merging of tracks (top) versus relocalization (bottom). The rotation-only input video for the data shown here contains several rapid camera motions (cf. description of the dataset in [5]), resulting in intermittent tracking loss. After each loss, our system starts a new partial panorama, and, while the incoming live frames are being processed, stitches them together in the background. The final panorama (top and right) was stitched together from 11 partial panoramas. In comparison, with the same tracking system but using relocalization, only the model at the bottom gets created. (Here, two pairs of keyframes were estimated to have a small non-zero baseline, thus two parts of the panorama—displayed with a white border—are offset slightly from the rotation axis.)

arbitrary. Before it can be integrated into an existing SfM group, a common scale has to be found. In order to do so, we use the set of all features that have been observed (and thus have a triangulated position) in both the existing SfM group as well as with respect to $[R_{\text{rel}}|t_{\text{rel}}]$, and calculate the ratios of their distances to $k_{\text{prev}}$ in both coordinate systems. We then take the median of those ratios as a robust measure of the scale between the two coordinate systems and scale $t_{\text{rel}}$ accordingly.

### 4.4 Relocalizing versus Starting a New Track

When tracking gets lost—i.e., when the number of inliers for the selected model falls below a set threshold—the standard strategy employed by most T&M systems (e.g., [17], [31], [42]) is to continuously try to *relocalize* the camera with respect to the current map with each new frame until successful. However, this means that tracking and mapping are suspended and no data is collected until relocalization is successful.

Here, we employ an alternative strategy proposed by Eade and Drummond [8]: instead of trying to relocalize, we start a new track immediately, and leave it to the background thread to merge tracks if possible (cf. Section 5.3). The benefit of this method, illustrated in Figs. 5 and 11, is that the system continues to collect data even after tracking failure occurs, and, if the tracks are later found to overlap, merges the created maps. If they do not overlap, the maps remain separate. (Note that in this case, a recovery-based system would never recover.)

## 5 Mapping

The mapping thread runs in parallel to the tracking thread and is responsible for the following tasks:

1. triangulate new features,
2. run bundle adjustment,
3. merge disjoint tracks,
4. estimate feature normals, and
5. clean up residual data.

These tasks are allowed to be more computationally intensive than the tracker's tasks, since the system does not depend on them in order to process the next frame.

Each of the tasks gets assigned a priority which depends on the time it was last executed and the system's current state. In each iteration, the task with highest priority is executed. For example, after a new keyframe is inserted, triangulation of new features becomes important; if tracking got lost and thus a new track is started, merging of tracks is prioritized to enable quick merging of tracks. Thus, assuming that the two threads run on two independent cores, the latter can happen as quickly as conventional relocalization.

### 5.1 Triangulating Features

A feature that was observed in at least two keyframes within the same SfM group, but not bundle adjusted yet, is triangulated and gets assigned a 3D location with respect to this group's frame of reference. When the tracker adds a new keyframe with a new observation of a feature $f$, $f$ is retriangulated using all information when the mapper cycles through this step the next time.

### 5.2 Bundle Adjustment

SfM groups with at least three keyframes get passed through a standard bundle adjuster [23] that globally optimizes all keyframe (i.e., camera) poses and feature positions in this group's frame of reference. If there are multiple such groups, the group with the newest

keyframe (that is, the keyframe that got adjusted the least number of times) is processed.

More sophisticated bundle adjustment strategies with local and global adjustment steps [17] could be integrated as needed.

### 5.3 Merging Disjoint Tracks

When tracking gets lost, the tracker immediately starts a new, independent *track*, rather than continuously trying to relocalize with respect to the existing map. In doing so, the tracker continues to collect and "stitch together" data even though the spatial relation to the first map is initially unknown.

The algorithm that merges tracks is similar to the keyframe-based recovery by Klein and Murray [18] (also used in [31], [42], among others): as observed by Eade and Drummond [8], recovery, loop closure, and (here) merging of tracks are effectively similar to each other; the main difference lies in when the algorithm is executed and how its result is used.

Whenever a new keyframe is taken, the system stores a downsampled, blurred copy of the image (here: $80 \times 60$ pixels, blurred with a Gaussian with $\sigma = 1.5$px), dubbed small blurry image (SBI).

Merging of tracks is implemented as follows: The algorithm chooses a keyframe $k_1$ and computes the NCC of its SBI with the SBI of all other keyframes. Keyframes on the same track as $k_1$ are omitted, as are keyframes to which a previous merge attempt failed. The keyframe $k_2$ with the highest NCC score is selected, and the SBIs of $k_1$ and $k_2$ are aligned to each other using inverse compositional image alignment [2] of an affine homography $H_A$. The features of $k_1$ are then projected into $k_2$ using $H_A$, and a regular "tracking" step (cf. Section 4) is executed.

If the tracking step fails, $k_2$ is "blacklisted" in $k_1$ as a failed attempt (so that the algorithm does not attempt the same combination again), and $k_1$ stores a timestamp of when this merge attempt occurred. The next time the algorithm tries to merge tracks, the keyframe that has not been chosen as $k_1$ the longest is chosen as $k_1$.

If the tracking step succeeds in estimating a model that is supported by a sufficient fraction of feature correspondences, the two tracks are considered successfully merged. Several different cases have to be considered in actually merging the environment models, depending on the type of transition connecting the two tracks, and whether or not $k_1$ and $k_2$ are already part of a keyframe group of the respective type. If available, the transition is preferred that would not introduce a new keyframe group. (For example, if $k_1$ and $k_2$ are both part of a panorama group, and could be connected with either $H$ or $E$, $H$ is preferred.) Adding of a group (if needed) as well as merging of panorama groups is straightforward (the latter only requires concatenating the homographies accordingly). To merge SfM groups, features that are observed in both groups are needed. To generate those, we track the features that are connecting $k_1$ and $k_2$ one frame "into" $k_2$'s group via epipolar search. Then, a common scale is computed as described in Section 4.3.

The benefit of merging of tracks is visualized in the case of panorama data in Fig. 5. In this particular case, the map of the merged tracks consists of an almost complete horizontal panorama, to which 1,605 frames (68 percent of the input data) are registered. Another 410 frames are registered to partial panoramas (not shown) which the system was unable to connect to the main model (whether these should be counted as success or failure depends on the application). In comparison, with the same tracking system but using relocalization, while the system is able to recover and expand the initial map several times, only 1,154 frames (49 percent) are tracked; all other data (which "passed by" while the system unsuccessfully tried to relocalize) are discarded. A similar result for 3D data is presented in Fig. 11.

### 5.4 Estimation of Feature Normals

To properly predict a feature's appearance from an assumed camera pose, one needs to know not only its 3D position, but also its local structure. Assuming that sufficiently many features are located on locally approximately planar surfaces, this structure can be defined by a normal vector $n$, and the change in appearance between two views of the feature is given by a homography $H_\perp$, which can be expressed in terms of $n$, the relative camera pose $[R_{\text{rel}}|t_{\text{rel}}]$, and the feature's 3D position $x$ [25], [44]:

$$H_\perp = R_{\text{rel}} + \frac{t_{\text{rel}} \cdot n^T}{n^T x} \quad \Leftrightarrow \quad H_\perp = R_{\text{rel}}\left(n^T x \cdot I_{3\times 3} + t_{\text{rel}} \cdot n^T\right). \tag{1}$$

(Note that $H_\perp$ is a homogeneous quantity, i.e., its scale is arbitrary.) The two views will be reasonably similar, as otherwise the tracker would have failed to locate the feature. Yet, small differences between them allow to estimate $H_\perp$ and thus $n$ using image alignment [25], [44].

If we assume $n$ to be the only unknown, $H_\perp$ has only two degrees of freedom, and it is possible to parametrize it accordingly [25]. However, as Molton et al. [25] note, noise in other variables will cause the projections to not coincide precisely, such that one has to allow for at least two additional degrees of freedom for shift. We have had more success with using an affine homography (as in [44]) with a straightforward six-degree-of-freedom parametrization, which we use in a standard inverse-compositional image alignment framework [2], and afterwards extracting $n$ from the over-determined system given by Eq. (1) using singular value decomposition.

Executing the image alignment step for each feature is fairly expensive; however, it should be noted that it can be run feature-by-feature in the background and is highly parallelizable, and none of the other steps are immediately dependent on it. (While no normal estimate is available, the tracker will continue to use a 2D projection of the feature.)

To maintain and refine the normal vector $n$ over time, instead of feeding it through a Kalman filter [25], [44], we simply collect independent estimates of $n$ from random pairs of views, the average of which is taken as the currently assumed value of $n$. This allows us to adapt elastically to the amount of processing power available. For well-textured and indeed planar features, the set of estimates is highly consistent, while less textured and highly non-planar features cause the image alignment to fail to converge or produce a set of wildly diverging estimates. This information

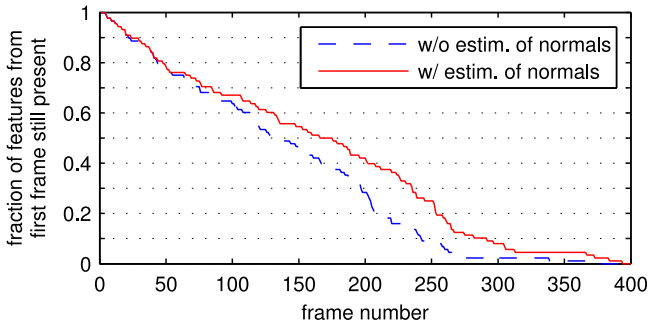



Fig. 6. Effect of normal estimation on lifespan of features. The data in this figure stems from the first part of the video to Fig. 1, in which the camera slowly descends around the building on the left.

could be used to further characterize the features and potentially remove them; this idea is left to future work.

In our current implementation, apart from the added value of having a richer environment model, estimation of normals extended the lifetime of individual features (that is, the number of frames that they were correctly tracked) by a moderate but noticeable amount, as shown in Fig. 6.

## 6 Model Selection

If observed data may have arisen from several different models, one faces the problem of selecting the right model in addition to the common problem of estimating the model parameters. Model selection is a complex problem in particular if the models in question are of fundamentally different kind, as is the case here: a homography is a bijective 2D map, and thus the observed residual between an estimated and measured feature location is 2D, whereas the essential matrix maps a 2D point to a line in 2D, and thus the residual is 1D (perpendicular to the line). This is analogous to trying to determine whether an observed set of 2D points can best be represented by a point or a line (see Fig. 7). It becomes apparent that residuals alone are not sufficient to select the model.

For this reason, several different metrics have been proposed [1], [34], [36], [40], [41]. Here, we use the Bayesian formulation of the GRIC by Torr [41], which is based on the Bayesian probability that a given model generated the observed data. In this section, we first describe the GRIC score for the case of the two-view relations $E$ and $H$, then its application to the estimation of the absolute poses $[R|t]$ and $[R|0]$. We use the same notation and variable names as Torr [41]. The subscript $m$ is used to indicate that a quantity is dependent on the models that are being compared. $\log(x)$ denotes the natural logarithm.

### 6.1 Generalized GRIC Score for Two-View Relations

The generic formula of Torr's Bayesian GRIC score is

$$\mathrm{GRIC}_m = -2\mathcal{L}_{\mathrm{MAP},m} + k_m \log n, \tag{2}$$

(cf. [41, Eq. (46)]) where $k_m$ is the number of parameters of the model, $n$ is the number of observations, and $\mathcal{L}_{\mathrm{MAP},m}$ denotes the maximum a-posteriori log likelihood of model $m$ given the data, which is based on the probability density functions (PDFs) that the data are assumed to stem from. (Note that the GRIC score is a *cost* function, i.e., lower score indicates better model fit.)

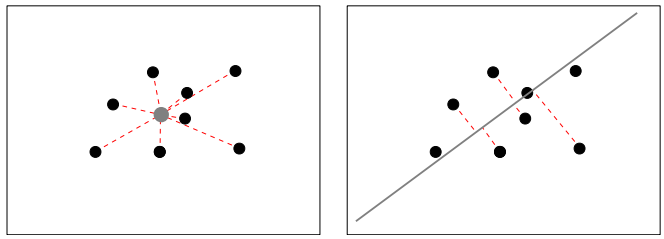


Fig. 7. Intuition of why model selection is difficult: Fitting a 2D point and a 2D line to a set of noisy measurements. Even though both models have the same number of degrees of freedom (2), the dimensionality of the error is different (2 versus 1), and the sum of errors is guaranteed to be smaller in the case of the line, regardless of how "point-like" the distribution may appear. Cf. [40, Fig. 16].

For the problem at hand, we assume a mixture of inliers affected by Gaussian noise, and outliers sprinkled uniformly in the search region. Specifically (cf. [41, Eq. (15)]):

$$\mathcal{L}_{\mathrm{MAP},m} = \sum_i \log(\gamma_i \cdot p_{\mathrm{in}} + (1-\gamma_i)\cdot p_{\mathrm{out}}) \tag{3}$$

$$\text{with} \quad p_{\mathrm{in}} = \frac{\sqrt{2\pi\sigma^2}^{d_m - D}}{c_m} \cdot \exp\left(-\frac{e_{i,m}^2}{2\sigma^2}\right), \tag{4}$$

$$p_{\mathrm{out}} = 1/v\,, \tag{5}$$

$\gamma_i \in \{1, 0\}$ indicates if correspondence $i$ is an inlier, $e_{i,m}$ are the individual reprojection errors, $\sigma$ is their standard deviation, $D$ is the dimensionality of each datum (here: a pair of 2D points, i.e., $D = 4$), $d_m$ is the dimensionality of the model manifold, $D - d_m$ is the dimensionality of the error (for $H$, $D - d_m = 2$, while for $E$, $D - d_m = 1$ (perpendicular to the epipolar constraint, in analogy to Fig. 7)), and $c_m$ and $v$ are the volumes of the spaces from which inliers and outliers, respectively, arise (cf. [41]).

Let $\gamma$ denote the expected inlier ratio $E(\gamma_i)$, and maximize over $\gamma_i$:

$$\begin{aligned}\Rightarrow\ \mathrm{GRIC}_m &= -2\sum_i \log(\max\{\gamma\cdot p_{\mathrm{in}}; (1-\gamma)p_{\mathrm{out}}\}) + k_m \log n \\ &= \sum_i \min\{\underbrace{-2\log(\gamma\cdot p_{\mathrm{in}})}_{(*)}; \underbrace{-2\log((1-\gamma)p_{\mathrm{out}})}_{(**)}\}) \\ &\quad + k_m \log n\end{aligned} \tag{6}$$

$$(*) = -2\log\left(\gamma\frac{\sqrt{2\pi\sigma^2}^{d_m-D}}{c_m}\cdot\exp\left(-\frac{e_{i,m}^2}{2\sigma^2}\right)\right) \tag{7}$$

$$= \frac{e_{i,m}^2}{\sigma^2} + (D-d_m)\log\,2\pi\sigma^2 + 2\log\frac{c_m}{\gamma} \tag{8}$$

$$(**) = -2\log\left(\frac{1-\gamma}{v}\right) \tag{9}$$

$$= T_m + (D-d_m)\log\,2\pi\sigma^2 + 2\log\frac{c_m}{\gamma} \tag{10}$$

TABLE 2
GRIC Score Parameters for the Four Models Considered Here

| Model $m$ | $k_m$ | $D-d_m$ | $c_m$ | $v$ |
|---|---|---|---|---|
| $E$ | 5 | 1 | $L\times L\times\Delta$ | $L\times L\times S\times S$ |
| $H$ | 8 | 2 | $L\times L$ | $L\times L\times S\times S$ |
| $[R\|t]$ | 6 | 2 | $V$ | $V\times S\times S$ |
| $[R\|0]$ | 3 | 2 | $V$ | $V\times S\times S$ |

*$L\times L$ is the area of one camera frame, $S\times S$ is the area of the constrained search region, $\Delta$ is the range of disparity. See Section 6.2 for discussion of $V$.*

$$\text{where}\quad T_m = 2\log\left(\frac{\gamma}{1-\gamma}\cdot\frac{v}{c_m}\right) - (D-d_m)\,\log\,2\pi\sigma^2\,. \quad (11)$$

Thus, the final formula is given by:

$$\mathrm{GRIC}_m = \sum_i \rho_2\left(\frac{e_{i,m}^2}{\sigma^2}\right) + n\left((D-d_m)\log\,2\pi\sigma^2 + 2\log\frac{c_m}{\gamma}\right) + k_m\log\,n \quad (12)$$

with $\rho_2(x)=\min\{x,T_m\}$.

*Difference to Torr's formula.* It should be noted that our formula differs from that given by Torr ([41, Eq. (48)]), even though both are based on PDFs of the form of Eqs. (4) and (5). The difference is that in Torr's formula, it is assumed that the inliers are uniformly distributed in disparity across the entire search region,[5] i.e., $\Delta = S$ ($\Delta$ and $S$ are contained in $c_m$ and $v$, cf. Table 2). For large search areas, this is not the case: the search region is large because we want tracking to be robust to fast camera movement (including camera rotations), but we expect the (2D) movement of all features to be strongly correlated; the range of disparity $\Delta$ is still likely to be only a few pixels. Thus, setting $\Delta = S$ causes the log likelihood for $E$ to decrease (because the observations do not match the model's assumption) and creates a significant bias towards selecting $H$. Hence, to alleviate this bias, we keep $\Delta$ independent of $S$. A more detailed discussion and illustration of this issue, as well as a proof that our formula is equivalent to Torr's for the special case $\Delta = S$, is given in [11].

### 6.2 GRIC Score for Absolute Pose Models

With appropriate model-specific parameters as given in Table 2, Eqs. (11) and (12) are applicable to $[R|0]$ and $[R|t]$ as well, since the PDFs on which they are based follow the shape of Eqs. (4) and (5). Arguably, this case is actually less complex than the two-view relation case, since the dimension of the error $D-d_m$ is the same for both models (see Table 2).

Two aspects are worth noting: First, in analogy to the two-view case, $c_m$ and $v$ are defined as the volumes of the spaces from which observations arise. This appears to necessitate to define the "volume" $V$ of the 3D model, and to raise the question whether one assumes it to be part of the noisy estimation process, or (in comparison with the current measurement error), to be noise-free. However, a look at the GRIC score equations (Eqs. (11) and (12)) and Table 2 reveals that, in the comparison of $[R|t]$ versus $[R|0]$, only the quotient $c_m/v$ and difference $D-d_m$ are needed, and the values of $V$ and $D$ fall out as constant terms.

Second, in the case of $E$ versus $H$, one parameter to the GRIC score is the range of disparity $\Delta$ that we expect for $E$. Conveniently, this parameter gives some control over how the process handles motions that are very close to rotation-only, i.e., have a very small baseline and thus very little parallax: the bigger $\Delta$, the more likely $H$ is selected in those cases.

When comparing $[R|t]$ and $[R|0]$, this free parameter does not exist: all features are mapped to a unique image point. Further, especially for a good model and slow camera motion, the measurement noise $\sigma$ becomes very small, and we observed that consequently, the GRIC score becomes extremely accurate in discerning even small shifts in baseline. Thus, $[R|0]$ is rarely selected (despite the smaller value of $k_m$, which gives it a slight advantage in case of close-to-equal error sums). Probabilistically, this makes perfect sense: the likelihood for $\|t\|$ exactly equal to $0$ is virtually non-existent, and whenever the error distribution hints to even the slightest shift, $[R|t]$ should be selected.

Practically, however, motions with very small $t$—imagine a camera mounted on a tripod with the optical center and the rotation axis in slight misalignment—are arguably better modeled as a panorama: For a panorama, a small shift of the optical center causes minor distortion artifacts, but no fatal harm to tracking and mapping. (DiVerdi et al. [7] analyze this type of error in detail.) For structure-from-motion, it results in features with unreliable depth estimates from extremely ill-conditioned triangulations, and, as soon as only those features are visible, likely tracking failure.

Therefore, we adapt the model that is estimated as follows: instead of estimating $[R|0]$, we estimate $[R|t\le t_{\max}]$ (more precisely, $[R|t]_{\text{prev}}\circ[R|t]\mid\|t\|\le t_{\max}$). Practically, we implemented this by estimating $[R|t]$, but constraining $\|t\|$ to $[0;\,t_{\max}]$ after each gradient descent iteration. $t_{\max}$ can be understood as a soft threshold regulating how much baseline shift the system shall observe before preferring $[R|t]$ over $[R|0]$, analogous to $\Delta$ for $E$ versus $H$. By dividing by the distance to the environment (which is available, since we are in the model-based condition), $t_{\max}$ can be transformed into a scale-independent feature observation angle $\alpha_{\max}$, which we use as input parameter to the system.

## 7 Evaluation

We implemented our system prototype in C++, making use of the OpenCV,[6] TooN[7] and libCVD[8] libraries. Our system runs in real time (at about 20-25 ms per frame) on a commodity PC without specific optimizations. Typical timings are presented in Fig. 8. They are, of course, strongly dependent on the hardware and parameter configuration (e.g., number of keypoints per frame) and presented here only as a coarse reference point.

5. Torr [41] explicitly warns that "care should be taken to rederive [this quantity] according to the exact distribution [...] in different scenarios."

6. http://opencv.willowgarage.com/.
7. http://www.edwardrosten.com/cvd/toon.html.
8. http://www.edwardrosten.com/cvd/.

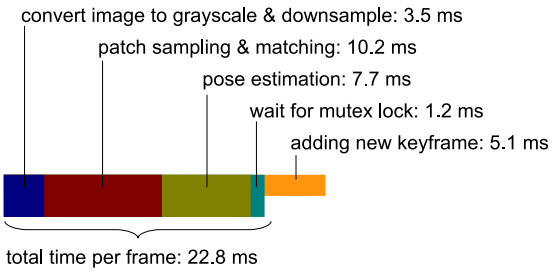



Fig. 8. Breakdown of timings in tracking thread. These times were taken on a commodity PC (Intel i7 Core, 4 GB RAM, Ubuntu 12.04), without specific optimizations or the use of a GPU. All times are averaged per frame over the entire sequence. Since creating a new keyframe is not executed every frame, its contribution to the average time is minimal.

Currently, our implementation is not optimized to run in real time on a mobile device. However, the most expensive parts (in particular, the tracking with NCC-based matching, cf. Fig. 8) are computationally similar to T&M systems that were shown to operate in real time on such devices several years ago [19], [42]. Thus, we argue that with appropriate algorithmic and device-specific optimizations, running an implementation of our concept on a mobile device is feasible.

We tested our system on a variety of video sequences, including rotation-only sequences from a panorama dataset by Coffin et al.[9] [5], sequences from the "City of Sights" repository[10] [12], as well as further self-recorded videos, using models from the "City of Sights" as backdrop. Results from those videos are presented in Figs. 1, 5, 9, 10, and 11.

### 7.1 Tracking Accuracy

To evaluate the benefits of integrating model-based tracking, we ran our system with and without model-based tracking (i.e., for the latter case, the left branch in Fig. 4 is disabled) on 800 frames of video from the "City of Sights" repository for which accurate ground truth of the camera trajectory is provided. The 3D model that our system generates from this video sequence (*with* model-based tracking) is shown in Fig. 9.

Since the reconstructions including camera locations are arbitrary up to a similarity transform, we first aligned the point cloud of the camera locations for both conditions to the ground truth by aligning the first camera position, and choosing scale and rotation such that the overall error was minimal (in a least-squares sense). The results are presented in Fig. 10, showing that with model-based tracking, tracking is highly accurate and very smooth.

### 7.2 Qualitative Comparison with PTAM

In Fig. 11, we present a qualitative comparison of our prototype with PTAM[11] [17], [18] on four representative videos:

For slow, smooth parallax-incuding camera movement (see Fig. 11 far left), both systems produce very similar trajectories. On a parallax-inducing movement followed by extended rotation-only movement (see Fig. 11 second from left), PTAM loses track and unsuccessfully attempts to recover for the remainder of the sequence, while our

9. http://tracking.mat.ucsb.edu.
10. http://cityofsights.icg.tugraz.at.
11. http://www.robots.ox.ac.uk/~gk/PTAM/.

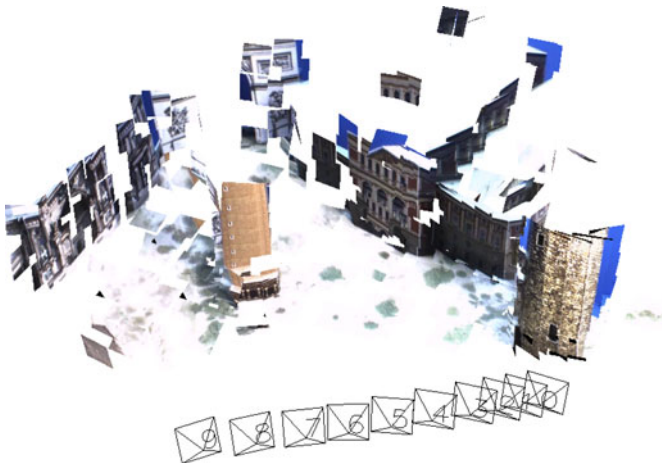


Fig. 9. Our system acting as a SLAM system, reconstructing the scene in 3D and recovering the camera trajectory fully automatically. Each reconstructed 3D feature is visualized as a small image patch (sampled from the frame in which it was first observed), oriented according to its estimated normal vector.

prototype keeps track throughout, attaching a panorama to the triangulated model. In Fig. 11 second from right, both systems lose track due to occlusion. PTAM successfully recovers as the camera moves back to the known part of the scene, but misses out on a long stretch, while our prototype starts a new track and successfully connects the two; thus, its final map covers a larger part of the environment.

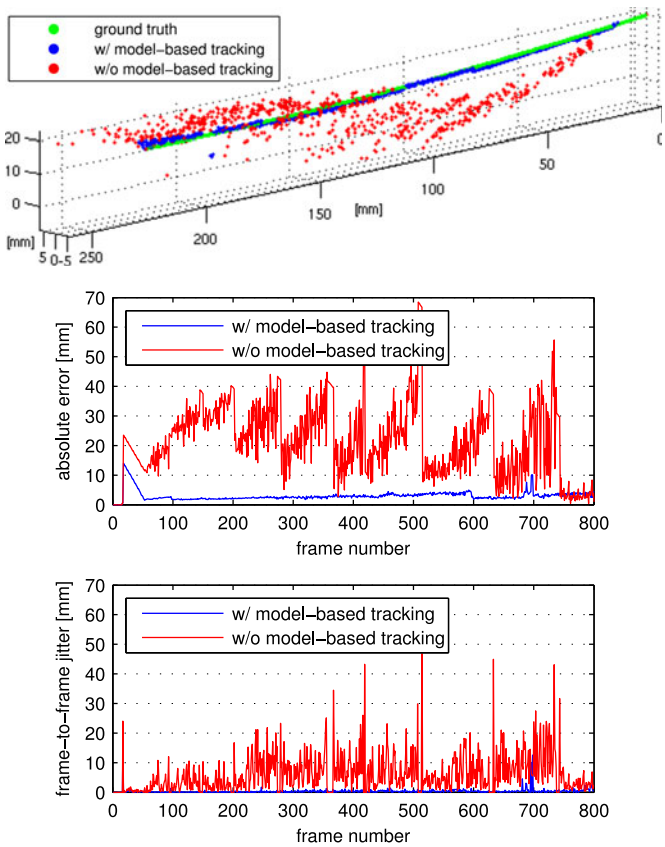



Fig. 10. Tracking with and without model-based tracking. The point clouds are aligned to the ground truth as described in the text. From top to bottom: camera locations in 3D, absolute error per frame, jitter (i.e., camera displacement from one frame to the next) per frame.

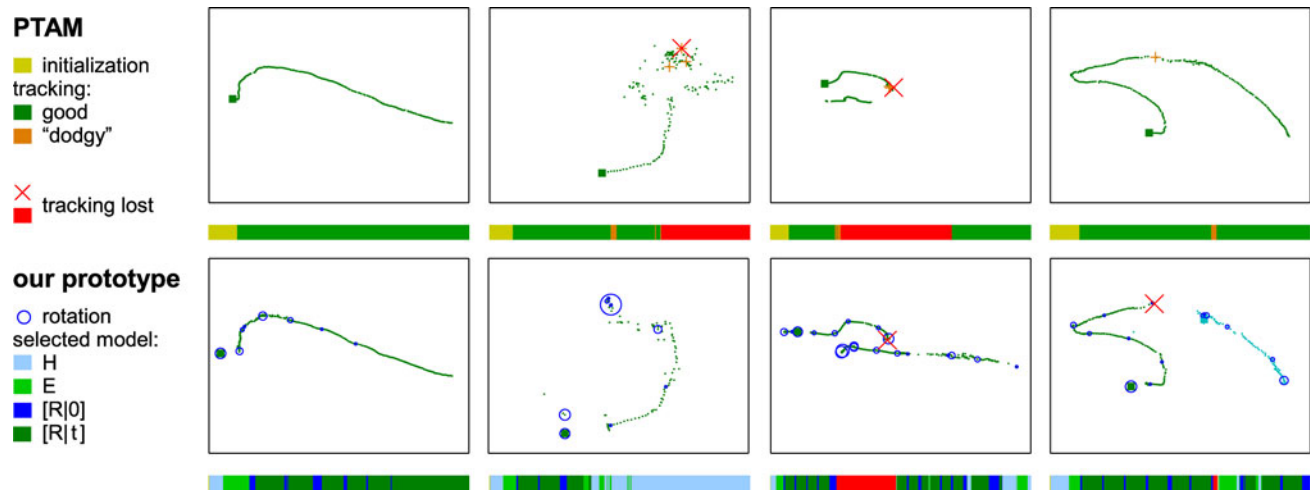



Fig. 11. Qualitative comparison of camera trajectories for PTAM (top) and our prototype (bottom) on selected videos, highlighting a few particular characteristics. Each video is 300-500 frames long, the camera travels a few feet along a scene similar to the one in Fig. 9. The trajectories from both systems were aligned to each other as described in Section 7.1. The color bar underneath each panel shows per-frame tracking/model selection information over time. A blue circle indicates that our prototype selected a rotation-only model (note that this does not have a permanent effect (i.e., there is no effect on the map) unless a keyframe is added at that point). The circle's radius is logarithmically proportional to the length of the rotation-only sequence.

Finally, on a parallax-inducing movement with relatively quick movement in the middle (see Fig. 11 far right), PTAM is able to maintain tracking, while our prototype loses track. It automatically starts a new track (shown in light blue, *aligned separately to PTAM's track*), but is unable to connect the two tracks.

Thus, the results demonstrate the conceptual advantages of supporting rotation-only movement and linking of tracks instead of recovery, but they also underscore that our current implementation is to be regarded a proof-of-concept prototype, and certain aspects are left unoptimized. In particular, tracking during parallax-inducing motion is not as robust as PTAM.

## 8 Discussion: Aspects for Further Investigation, Applications and Limitations

### 8.1 On Estimating the Probability Density Function

Optimally estimating both a motion model and the parameters of the probability density function (PDF) of the measurement error (e.g., standard deviation $\sigma$ of the measurement error and inlier ratio $\gamma$) is a chicken-and-egg problem: either result requires the other. In many T&M systems, knowing the PDF exactly is not crucial: with appropriate cost functions, pose estimation algorithms (both via sample consensus and M-estimators) are reasonably robust to noise in $\sigma$ and $\gamma$, and "bad" values may merely be an indicator for inevitable tracking failure. Model selection however gets more accurate the better the estimate of the PDF is.

If the observed error distribution matches the expected shape (i.e., Eqs. (4) and (5)), the described process via expectation-maximization (Section 4.2) works well and appears to produce accurate estimates of its parameters. If however the observed distribution does not match the expected shape, for example due to questionable tracking, expectation-maximization appears to intensify the problem by producing unusable values. This effect can be limited by ad-hoc measures such as filtering over several frames and clamping of the values, but we suggest that further investigation is warranted to find an optimal and principled solution balancing accurate estimates and leniency for tracking.

### 8.2 On Improving Coarse-to-Fine Matching

The standard approach to coarse-to-fine matching (approximately as employed by [17], [42], [43]) is to 1) locate features in a coarse (downsampled) image, 2) estimate the camera pose, 3) reproject features into the higher resolution image using the new pose estimate, 4) refine their position (in a tightly constrained area), and lastly 5) refine the camera pose using the refined features.

Unfortunately, this approach cannot be applied directly here, because we do not know which type of camera motion to expect (and thus which model to enforce) until after the tracking step. As described in Section 4.1, thus far, we have simply omitted the intermediate pose estimation (step 2). This however results in a higher risk of losing individual features, as they must be found independently in an effectively larger search region.

There are several alternative approaches in which this could be solved, for example, executing steps 2 to 5 for each motion model under consideration, or selecting the model after step 2. We have designed another, more sophisticated approach, which retains the advantages of the interleave pose estimation at little additional cost: We propose to estimate a *superset model* which admits all features that are inliers to *either* motion model under consideration (thus, the tracking step remains agnostic to the type of model that is later selected), yet constrains the features such that many outliers can be identified. For model-based tracking, $[R|t]$ is this superset model, as it trivially encompasses $[R|0]$.

For the two-view relations $H$ and $E$, the existence of such a superset model is less obvious. However, a closer look reveals that indeed, $E$ fulfills these criteria: while the epipolar geometry is, strictly speaking, not defined in the case of rotation-only movement, an algorithm used to compute $E$

will return *some* result $\breve{E}$, and *true inliers for either H or E will fulfill* $x_2^T \cdot \breve{E} \cdot x_1 = 0$ (cf. in particular [14, Section 11.9.3]). Thus, while $\breve{E}$ may lack a geometric interpretation (in the case of rotation-only movement), it does provide a necessary condition for any correct correspondence. Assuming that outliers are statistically independent, the chances for a random outlier to satisfy $x_2^T \cdot \breve{E} \cdot x_1 \approx 0$ are small.

We have implemented and verified this approach on several videos (including rotation-only sequences, which, for $\breve{E}$, is the most unfavorable case). However, we have not yet integrated it fully with the rest of our system, so the evaluation of potential robustness and performance improvements remains to be done. One particular challenge is to robustly estimate the PDF parameters (cf. Section 8.1) in the now effectively much smaller active search regions.

### 8.3 On Merging of Maps

As discussed in Section 5.3, the algorithm we use to detect overlap between disjoint tracks and thus initiate their merging is basically the same as what is commonly employed for relocalization, which implies that similar detection performance can be expected. An interesting area for future research is to exploit the fact that we actually have more data than what is available with the relocalization strategy: instead of one individual new frame, we have a new map consisting of multiple frames; thus, overlap detection could potentially yield higher detection performance than relocalization. This could be exploited even within the SBI-based recovery framework: For example, Kim et al. [16] have explored the use of "virtual keyframes," i.e., renderings of the model from strategically distributed viewpoints, rather than actual keyframes. As we now seek to connect two models, this strategy could be applied on both sides.

Similarly, the overlap between the tracks may encompass an area larger than a single frame. While we currently use only features from one pair of keyframes to merge the maps, the registration could be improved by explicitly calculating and then using the entire extent of the overlap. Note however that a much larger overlap is unlikely, since otherwise it would likely have been detected earlier.

### 8.4 On Applications and Limitations of the Hybrid Map

It is clear that hybrid maps consisting of both SfM and panorama data (such as Fig. 1 or the map produced from Fig. 11 second from left) do not possess all qualities that one would look for in an ideal environment map. Most obviously, the panorama part lacks depth information, but furthermore, the individual SfM reconstructions do not share a common scale, since the rotation-only movement cannot propagate scale. We emphasize that this is not a limitation of our design, but an inherent limitation of the input data; the alternative is not to create a fully 3D model, but to lose tracking altogether. If a fully 3D model of the environment is required, one has to either employ additional and/or active sensors, or put constraints on the camera movement.

However, for many AR applications, where the main purpose of the model is to register annotations, the hybrid nature of the map (and lack of depth) may not be a major concern, since the user automatically provides, and thus allows registration of annotations for, exactly those viewpoints that are needed. Specifically, if only a panorama is provided and thus an annotation can be registered to a bearing (but not depth) in this panorama, it implies that the user has been looking at the scene from this viewpoint only and thus mainly needs the annotation's bearing (cf. work on applications of AR annotations based on panoramas [21], [42]). As soon as the user moves and thus the annotation's depth is needed for correct registration, this new viewpoint can be exploited to create 3D structure and register the annotation in 3D. Similarly, annotations in different parts of the scene may be registered to a different SfM group with a different scale factor, but these annotations are (by construction) outside the user's field of view and not visible directly. However, due to the topology provided by the hybrid map, topological or navigational annotations (for example, directions towards an object outside the current field of view) can still be displayed, which is not possible with unconnected local reconstructions.

Further, even if one ultimately aims for a fully 3D model, it is arguably better to collect the panoramas than to discard them. For example, Pan et al. [30] describe how to first collect panoramas, and then use them as single images with wide field of view for 3D reconstruction afterwards.

## 9 Conclusions

We have presented an approach and prototype implementation for initialization-free real-time tracking and mapping that supports both parallax-inducing and rotation-only camera motions in 3D environments, and integrates both model-free and model-based tracking. Our design paradigm was to make use of all data that can be casually collected, and to not require any particular assistance by the user (such as a separate initialization step, or particular types of camera motion).

Our system is able to track and map through motion sequences that neither conventional six-degree-of-freedom SLAM systems nor panoramic mapping systems can process. Depending on the video sequence, the strategy of starting a new track and later merging separate tracks (instead of trying to relocalize with respect to the first map) can significantly increase the amount of data represented in the final environment model.

We believe that our approach is an important conceptual step towards the vision of fully transparent tracking and mapping for "Anywhere Augmentation" [15], making the best use of the input data irrespective of the type of camera motion (or, for example, the distance to the environment). We emphasize that our current implementation is to be regarded a proof-of-concept prototype, and many implementation details are unoptimized. In particular, tracking robustness during general motion is not yet comparable to systems such as PTAM [17] or DTAM [27]. However, we have shown that the core components of state-of-the-art SLAM systems—including model-based pose estimation, which is a significant advancement over our earlier implementation [11]—can be integrated in a general framework (see Fig. 4) such that tracking and mapping can continue through both parallax-inducing and rotation-only motion. From here, techniques that were proven to increase

robustness or scalability—such as estimation of in-plane rotation before the pose estimation [18], more advanced map feature management including filtering of outliers, proactive search for new features in existing frames [17], and more sophisticated bundle adjustment strategies (cf. Section 5.2)—can be integrated as appropriate. Further areas for future work were outlined in Section 8.

There are several other areas with open research questions. For example, while we currently select only one model for the entire frame (which is, theoretically, the correct thing to do, since the motion refers to the camera and thus to the entire frame), there may be cases especially in outdoor scenes in which the foreground exhibits enough parallax to be modeled in 3D, while the background exhibits little parallax and might benefit from the stable, dense mapping that homographies offer. This leads to an interesting problem in which image segmentation and environment modeling interact.

Finally, it remains an open question how a model that consists of a mixture of structural data and (partial) panoramas can best be visualized, presented to, and navigated by the user. This is not a concern if the model is used only as an anchor for AR annotations, in which case the user never actually sees the model, but only the annotations fused with his/her view of the real world (cf. discussion in Section 8.4). However, if the model is to be used in Virtual Reality as well (for example, to allow a spatially remote user to view the scene [10]), the model itself needs to be visualized and navigated. This works very well in the case of panoramic mapping, where the emerging model is easy to interpret and browse. It is inherently more challenging in the case of 3D data (especially if the model is incomplete, so that the viewpoints for which useful views can be rendered are restricted), and, to our knowledge, an open research problem for the case of live, incomplete data that consists of mixtures of structural and panoramic data. By building on large data collections and offline reconstructions, interesting viewing modalities for mixed data like this have emerged from the SfM community [38].

## Acknowledgments

The authors would like to thank their anonymous reviewers for many acute observations and suggestions, and several attendees of ISMAR 2012 for insightful discussions, especially Gerhard Reitmayr, Daniel Wagner, Pierre Fite-Georgel, Andrew Davison, Georg Klein, and Tom Drummond. This work was supported by a Chancellor's fellowship from UCSB for S.G., NSF CAREER Grant IIS-0747520, ONR Grant N00014-09-1-1113, and NSF Grant IIS-1219261.

## References

[1] H. Akaike, "A New Look at the Statistical Model Identification," *IEEE Trans. Automatic Control*, vol. 19, no. 6, pp. 716-723, Dec. 1974.

[2] S. Baker and I. Matthews, "Lucas-Kanade 20 Years on: A Unifying Framework: Part 1," Technical Report CMU-RI-TR-02-16, Robotics Inst., Carnegie Mellon Univ., July 2002.

[3] J. Civera, A. Davison, and J. Montiel, "Inverse Depth Parametrization for Monocular SLAM," *IEEE Trans. Robotics*, vol. 24, no. 5, pp. 932-945, Oct. 2008.

[4] J. Civera, A. Davison, and J. Montiel, "Interacting Multiple Model Monocular SLAM," *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA)*, pp. 3704-3709, 2008.

[5] C. Coffin, J. Ventura, and T. Höllerer, "A Repository for the Evaluation of Image-Based Orientation Tracking Solutions," *Proc. Second Int'l TrakMark Workshop in Conjunction with ISMAR 2011*, 2011.

[6] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052-1067, June 2007.

[7] S. DiVerdi, J. Wither, and T. Höllerer, "All Around the Map: Online Spherical Panorama Construction," *Computers and Graphics*, vol. 33, no. 1, pp. 73-84, 2009.

[8] E. Eade and T. Drummond, "Unified Loop Closing and Recovery for Real Time Monocular SLAM," *Proc. British Machine Vision Conf. (BMVC)*, 2008.

[9] S. Gauglitz, L. Foschini, M. Turk, and T. Höllerer, "Efficiently Selecting Spatially Distributed Keypoints for Visual Tracking," *Proc. IEEE Int'l Conf. Image Processing (ICIP)*, 2011.

[10] S. Gauglitz, C. Lee, M. Turk, and T. Höllerer, "Integrating the Physical Environment into Mobile Remote Collaboration," *Proc. 14th Int'l Conf. Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, 2012.

[11] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Höllerer, "Live Tracking and Mapping from Both General and Rotation-Only Camera Motion," *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, 2012.

[12] L. Gruber, S. Gauglitz, J. Ventura, S. Zollmann, M. Huber, M. Schlegel, G. Klinker, D. Schmalstieg, and T. Höllerer, "The City of Sights: Design, Construction, and Measurement of an Augmented Reality Stage Set," *Proc. IEEE Ninth Int'l Symp. Mixed and Augmented Reality (ISMAR)*, pp. 157-163, Oct. 2010.

[13] L. Gruber, S. Zollmann, D. Wagner, D. Schmalstieg, and T. Höllerer, "Optimization of Target Objects for Natural Feature Tracking," *Proc. 20th Int'l Conf. Pattern Recognition (ICPR)*, pp. 3607-3610, 2010.

[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. second ed., Cambridge Univ. Press, 2004.

[15] 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*, pp. 393-416, Springer-Verlag, Feb. 2007.

[16] 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)*, pp. 127-134, 2009.

[17] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *Proc. IEEE and ACM Sixth Int'l Symp. Mixed and Augmented Reality (ISMAR)*, 2007.

[18] G. Klein and D. Murray, "Improving the Agility of Keyframe-Based SLAM," *Proc. 10th European Conf. Computer Vision (ECCV)*, pp. 802-815, Oct. 2008.

[19] G. Klein and D. Murray, "Parallel Tracking and Mapping on a Camera Phone," *Proc. IEEE Eighth Int'l Symp. Mixed and Augmented Reality*, pp. 83-86, Oct. 2009.

[20] M. Klopschitz, A. Irschara, G. Reitmayr, and D. Schmalstieg, "Robust Incremental Structure from Motion," *Proc. Int'l Symp. 3D Data Processing, Visualization and Transmission (3DPVT)*, vol. 2, 2010.

[21] T. Langlotz, C. Degendorfer, A. Mulloni, G. Schall, G. Reitmayr, and D. Schmalstieg, "Robust Detection and Tracking of Annotations for Outdoor Augmented Reality Browsing," *Computers and Graphics*, vol. 35, no. 4, pp. 831-840, 2011.

[22] S. Lieberknecht, A. Huber, S. Ilic, and S. Benhimane, "RGB-D Camera-Based Parallel Tracking and Meshing," *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, pp. 147-155, Oct. 2011.

[23] M.A. Lourakis and A. Argyros, "SBA: A Software Package for Generic Sparse Bundle Adjustment," *ACM Trans. Math. Software*, vol. 36, no. 1, pp. 1-30, 2009.

[24] S. Lovegrove and A.J. Davison, "Real-Time Spherical Mosaicing Using Whole Image Alignment," *Proc. European Conf. Computer Vision (ECCV)*, vol. 6313, pp. 73-86, 2010.

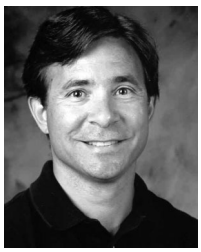
[25] N. Molton, A. Davison, and I. Reid, "Locally Planar Patch Features for Real-Time Structure from Motion," *Proc. 15th British Machine Vision Conf. (BMVC)*, 2004.

[26] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real Time Localization and 3D Reconstruction," *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 363-370, 2006.
[27] R. Newcombe, S. Lovegrove, and A. Davison, "DTAM: Dense Tracking and Mapping in Real-Time," *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, pp. 2320-2327, 2011.
[28] R.A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time Dense Surface Mapping and Tracking," *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, 2011.
[29] D. Nistér, "An Efficient Solution to the Five-Point Relative Pose Problem," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756-770, June 2004.
[30] Q. Pan, C. Arth, G. Reitmayr, E. Rosten, and T. Drummond, "Rapid Scene Reconstruction on Mobile Phones from Panoramic Images," *Proc. IEEE 10th Int'l Symp. Mixed and Augmented Reality (ISMAR)*, pp. 55-64, 2011.
[31] C. Pirchheim and G. Reitmayr, "Homography-Based Planar Mapping and Tracking for Mobile Phones," *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, pp. 27-36, 2011.
[32] M. Pollefeys, F. Verbiest, and L. Van Gool, "Surviving Dominant Planes in Uncalibrated Structure and Motion Recovery," *Proc. European Conf. Computer Vision (ECCV)*, pp. 613-614, 2002.
[33] J. Repko and M. Pollefeys, "3D Models from Extended Uncalibrated Video Sequences: Addressing Key-Frame Selection and Projective Drift," *Proc. Fifth Int'l Conf. 3D Digital Imaging and Modeling (3DIM)*, pp. 150-157, 2005.
[34] J. Rissanen, "Modeling by Shortest Data Description," *Automatica*, vol. 14, no. 5, pp. 465-471, 1978.
[35] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," *Proc. IEEE European Conf. Computer Vision*, vol. 1, pp. 430-443, May 2006.
[36] G. Schwarz, "Estimating the Dimension of a Model," *Annals of Statistics*, vol. 6, pp. 461-464, 1978.
[37] J. Shi and C. Tomasi, "Good Features to Track," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 593-600, 1994.
[38] N. Snavely, S. Seitz, and R. Szeliski, "Photo Tourism: Exploring Photo Collections in 3D," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 835-846, 2006.
[39] H. Strasdat, J. Montiel, and A. Davison, "Real-Time Monocular SLAM: Why Filter?" *Proc. IEEE Int'l Conf. Robotics and Automation (ICRA)*, pp. 2657-2664, 2010.
[40] P.H. Torr, A.W. Fitzgibbon, and A. Zisserman, "The Problem of Degeneracy in Structure and Motion Recovery from Uncalibrated Image Sequences," *Int'l J. Computer Vision*, vol. 32, pp. 27-44, 1999.
[41] P. Torr, "Bayesian Model Estimation and Selection for Epipolar Geometry and Generic Manifold Fitting," *Int'l J. Computer Vision*, vol. 50, no. 1, pp. 35-61, 2002.
[42] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg, "Real-Time Panoramic Mapping and Tracking on Mobile Phones," *Proc. IEEE Virtual Reality (VR)*, Mar. 2010.
[43] D. Wagner, D. Schmalstieg, and H. Bischof, "Multiple Target Detection and Tracking with Guaranteed Framerates on Mobile Phones," *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR '09)*, pp. 57-64, 2009.
[44] H. Wuest, F. Wientapper, and D. Stricker, "Acquisition of High Quality Planar Patch Features," *Proc. Fourth Int'l Symp. Advances in Visual Computing*, pp. 530-539, 2008.

**Steffen Gauglitz** received the Dipl.-Ing. degree in computer engineering from RWTH Aachen University, Germany, in 2008. He is currently working toward the PhD degree in computer science at the University of California, Santa Barbara. His current research interests include augmented reality, real-time computer vision, and in particular their application to interactive and immersive telepresence. He is the recipient of a UCSB Chancellor's Fellowship and has interned at WorldViz LLC, Qualcomm Inc. CR&D, and Google Inc. He is a student member of the IEEE.

**Chris Sweeney** received the BS and BA degrees in computer science and mathematics from the University of Virginia in 2011. He is currently working toward the PhD degree in computer science in the Four Eyes Lab at the University of California, Santa Barbara. His research interests include bridging the gap between large-scale structure from motion and traditional SLAM systems so that large 3D reconstructions can be easily accessible for outdoor augmented reality. He is currently supported by a US National Science Foundation Graduate Research Fellowship. He is a student member of the IEEE.

**Jonathan Ventura** received the BS, MS and PhD degrees from the University of California, Santa Barbara, in 2001, 2005, and 2012, respectively. He has worked as a project scientist at UCSB and as an intern at the Adobe Advanced Technology Lab in San Jose, California. In his doctoral dissertation, he developed methods for individual users to visually model and track outdoor buildings with mobile phone hardware. He is currently a senior researcher at the Institute for Computer Graphics and Vision at Graz University of Technology in Austria. His current research interests include computer vision technologies, user interfaces and application design for wide-area augmented reality. He is a member of the IEEE.



**Matthew Turk** received the BS degree from Virginia Tech, Blacksburg, the MS degree from Carnegie Mellon University, Pittsburgh, Pennsylvania, and the PhD degree from the Massachusetts Institute of Technology, Cambridge. He is a professor of computer science and was a former chair of Media Arts and Technology at the University of California, Santa Barbara, where he co-directs the Four Eyes Lab and is involved in several interdisciplinary centers and programs. He has worked at Martin Marietta Denver Aerospace, LIFIA/ENSIMAG (Grenoble, France), Teleos Research, and Microsoft Research, where he was a founding member of the Vision Technology Group. He is on the editorial board of the *ACM Transactions on Intelligent Interactive Systems* and the *Journal of Image and Vision Computing*. He is a fellow of the IEEE and the 2011-2012 Fulbright-Nokia distinguished chair in Information and Communications Technologies.

**Tobias Höllerer** received the graduate degree in informatics from the Technical University of Berlin, Germany, and the MS and PhD degrees in computer science from Columbia University, City of New York. He is a professor of computer science at the University of California, Santa Barbara, where he co-directs the Four Eyes Lab, conducting research in the four I's of Imaging, Interaction, and Innovative Interfaces. His research interests include the area of human-computer interaction and experimental systems, with a particular focus on augmented and virtual reality, information visualization, 3D displays and interaction, and social and adaptive user interfaces. He is a recipient of the National Science Foundation's CAREER award, for his work on "Anywhere Augmentation", which enables mobile computer users to place annotations in 3D space wherever they go. He is a member of the IEEE.