"A Novel Feature-Based Approach for Indoor Monocular SLAM"

KABIRI, Peyman <http://orcid.org/0000-0001-5143-0498> and HOSEINI, Seyyed Ali Hoseini

electronics

MDPI

Article

A Novel Feature-Based Approach for Indoor Monocular SLAM

Seyyed Ali Hoseini and Peyman Kabiri *

School of Computer Engineering, Iran University of Science and Technology, Tehran 16846-13114, Iran; sa.hoseini@birjand.ac.ir

* Correspondence: peyman.kabiri@iust.ac.ir; Tel.: +98-217-302-5341

Received: 14 August 2018; Accepted: 22 October 2018; Published: 7 November 2018

Abstract: Camera tracking and the construction of a robust and accurate map in unknown environments are still challenging tasks in computer vision and robotic applications. Visual Simultaneous Localization and Mapping (SLAM) along with Augmented Reality (AR) are two important applications, and their performance is entirely dependent on the accuracy of the camera tracking routine. This paper presents a novel feature-based approach for the monocular SLAM problem using a hand-held camera in room-sized workspaces with a maximum scene depth of 4–5 m. In the core of the proposed method, there is a Particle Filter (PF) responsible for the estimation of extrinsic parameters of the camera. In addition, contrary to key-frame based methods, the proposed system tracks the camera frame by frame and constructs a robust and accurate map incrementally. Moreover, the proposed algorithm initially constructs a metric sparse map. To this end, a chessboard pattern with a known cell size has been placed in front of the camera for a few frames. This enables the algorithm to accurately compute the pose of the camera and therefore, the depth of the primary detected natural feature points are easily calculated. Afterwards, camera pose estimation for each new incoming frame is carried out in a framework that is merely working with a set of visible natural landmarks. Moreover, to recover the depth of the newly detected landmarks, a delayed approach based on linear triangulation is used. The proposed method is applied to a realworld VGA quality video (640 $\times$ 480 pixels) where the translation error of the camera pose is less than 2 cm on average and the orientation error is less than 3 degrees, which indicates the effectiveness and accuracy of the developed algorithm.

Keywords: camera tracking; visual simultaneous localization and mapping; feature matching; particle filter; triangulation

1. Introduction

The purpose of vision-based camera tracking is to estimate the camera pose from a sequence of input images often in the form of video frames. Vision-based camera tracking has a close relation with some fundamental problems in computer vision, e.g., 3D reconstruction, image registration, and AR. Visual SLAM aims to estimate the camera trajectory and at the same time, to construct a sparse or dense representation of the scene. Visual SLAM solutions incrementally construct a map of the observed scene and then use this map to locate the camera position. Once the visual sensor embedded in the camera is a range scanner or an RGBD sensor, the measured depth of extracted feature points with a certain precision is available and no feature initialization is needed for newly detected features. However, unlike cameras equipped with active sensors, a monocular camera has a passive and bearing-only sensor that can only produce 2D measurements of the 3D observed scene. On the other hand, for monocular cameras due to the lack of depth data for newly observed features, it is necessary to initialize them. Feature initialization is an erroneous task, because the associated computations are performed using noisy feature measurements and camera poses.

The purpose of AR is to dynamically enhance the real world with computer-generated virtual objects. In order to achieve neat and clear integrated videos for the output of the AR system, objects in the real world and overlaid virtual objects should be properly aligned with respect to each other. A key requirement to achieve this goal is to provide a camera tracking system that can accurately calculate the position and orientation of the camera with respect to its surrounding environment. Primary AR systems exploit some information from the environment. This information may include fiducial markers in the scene or appear in the form of Computer-Aided Design (CAD) models. However, recent research in this field has employed natural landmarks to estimate camera pose parameters.

In this paper, a feature-based method for monocular SLAM that operates in unknown environments is proposed. The main contributions of this work are summarized as follows:

- A coarse to fine feature tracking scheme that is highly reliable and robust against quick camera movement;
- Depth information propagation that helps the proposed system to estimate camera pose parameters as the camera moves;
- A delayed feature initialization approach that only estimates the depth of new features when the observed parallax exceeds a threshold.

The structure of this paper is as follows: Related work is discussed in Section 2. In Section 3, the proposed approach will be explained in detail. Experimental results are presented in Section 4. The conclusion and future works are included in Section 5.

2. Related Work

If a visual SLAM system does not acquire any knowledge about the depth of the observed scene, then it will produce drift in the camera trajectory and therefore the cumulative error of the camera pose will increase. When a monocular camera moves around the environment, the captured video provides information about the structure of the scene with a scale factor. To recover this scale, Ababsa and Mallem [1] placed 2D fiducials in the scene. This enables their algorithm to make a metric map and in addition avoids the accumulation of error. Another approach, which solves the problem of scale ambiguity in the implementation of the visual SLAM algorithms, is the use of calibrated images [2,3]. The advantage of calibrated images is that the Euclidian structure of the scene for some sparse points of the image is available. With reference images, the process of pose estimation reduces to data association between each new image and the reference images.

When the camera moves in an unknown environment and continuously explores new regions, the cumulative error of the camera position and orientation is gradually increasing. This problem reduces the accuracy of initialization for newly detected feature points. If the algorithm continues in this direction, it will finally lose the correct path and the global error of the camera position will experience an overshoot. However, when the camera revisits previously observed regions and the algorithm is capable of detecting loops, then drift growing of the camera trajectory can be controlled. Visual SLAM approaches considering loop closure detection [4,5] significantly alleviate enlargement of the camera trajectory drift. However, detecting loops in the camera path is only desirable when the camera revisits previously explored scenes. In the presented work, detecting loops is not addressed and there is no intent to construct a global map. As a consequence, when a feature is occluded or leaves the camera's field of view, it will be removed from the list of the "to be tracked" features. The reported work creates a metric map, which successively propagates depth information to newly extracted features as the camera explores new areas.

Camera tracking or the estimation of camera pose parameters for a sequence of video frames is widely studied. In this research area, two main categories of solutions are employed to address the problem, i.e., Structure from Motion (SfM) and filtering. The reported works developed on the basis of an SfM solution mainly use epipolar geometry principles to estimate the camera pose [6]. Early works extended according to the SfM solution were mainly implemented on a small set of images.

However, there are some reported studies where longer image sequences are used [7,8]. Moreover, reported works are often implemented in an offline manner. Camera extrinsic parameters and the constructed map obtained through the SfM solution are often optimized using Bundle Adjustment (BA) [9]. However, recently, some real-time works have been reported in this context. Parallel Tracking and Mapping (PTAM) [10] is a leading work proposed for small workspaces. Camera tracking and map construction in PTAM are two separate tasks that are processed in two parallel threads. This enables the algorithm to perform the expensive batch optimization routine of BA in real-time. Some visual SLAM solutions optimize a pose graph to obtain an optimized, globally consistent camera trajectory. RGBD-SLAM [11], which is developed for RGBD sequences, uses the g2o [12] framework to optimize a pose graph for the creation of a globally consistent camera trajectory.

In filtering solutions, the problem is in the shape of a discrete dynamic system. Camera pose parameters are contained in the internal state of this system. Mostly, due to the nonlinear nature of the observation model, nonlinear variants of the Kalman filter are used for camera pose estimation [13]. MonoSLAM [14] is a prominent work, which uses a full covariance Extended Kalman Filter (EKF) approach for camera tracking. Moreover, it uses a probabilistic map and a top-down method to obtain feature correspondences along successive frames. Some researchers employed PF as an internal state estimator for nonlinear dynamic systems to estimate the camera pose parameters [15,16].

According to the adopted strategy employed for data association, visual SLAM methods may be classified into two groups, i.e., feature-based methods and direct methods. In feature-based methods, scene geometry and camera pose parameters are estimated using a sparse set of features extracted and tracked in consecutive frames. PTAM, MonoSLAM, and RGBD-SLAM are prominent feature-based algorithms presented for the monocular visual SLAM problem. In contrast, optimizing a photometric error function, direct methods try to estimate dense or semi-dense geometry of the scene. DTAM [17] is an example of a direct visual SLAM algorithm, which constructs a dense map of the scene. LSD-SLAM [18] is a monocular SLAM solution that builds a semi-dense large-scale and consistent map of the environment viewed by the camera. Similar to PTAM, the proposed PF-based visual SLAM algorithm employs a pyramidal approach for tracking FAST feature points. However, the RANSAC algorithm is performed to ensure a robust feature tracking process.

3. Proposed Method

An overview of the proposed approach is illustrated in Figure 1. A data association step is performed after the arrival of each new incoming frame. In the data association step, the intention is to find matched feature points between the current and previous frames. Among all matched feature points, those already initialized are employed for the estimation of camera pose parameters in a PF framework. Once camera pose parameters are obtained, any feature point with parallax exceeding a certain threshold is initialized. Finally, the algorithm proceeds with the extraction of new features if necessary.

3.1. PF Implementation

In the proposed approach, a PF framework is employed for the estimation of 6-DOF camera pose parameters. Actually, the internal state of the system constitutes the camera position and orientation with respect to the world coordinate frame, as defined in Equation (1). PF estimates this internal state given feature matchings that are acting as noisy observations of the system.

$$x = [t \quad \omega]^T, \qquad t, \omega \in R^3. \tag{1}$$

where $t = [t_x \; t_y \; t_z]$ is the coordinate of the camera center and $\omega = [\omega_x \; \omega_y \; \omega_z]$ is a vector denoting camera orientation in axis/angle representation. Given ω, the rotation matrix is obtained using Rodrigues' formula, i.e., Equation (2) [19].

$$R = I + \frac{[\omega]_\times}{|\omega|}\sin(|\omega|) + \frac{[\omega]_\times^2}{|\omega|^2}(1 - \cos(|\omega|)). \tag{2}$$

where $I \in R^{3\times3}$ is the identity matrix and $[\omega]_\times$ is the skew-symmetric matrix corresponding to ω obtained using Equation (3).

$$[\omega]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{3}$$

The state of the developed PF at frame k is denoted by x_k. In the PF developed for estimation of the camera pose, a constant position and rotation model is employed for state transition between consecutive frames. Equivalently, the pose of the camera only undertakes a Gaussian random walk, as described in Equation (4).

$$p(x_k|x_{k-1}) \propto N(x_{k-1}, \Sigma_x). \tag{4}$$

where x_{k-1} and Σ_x are the mean and covariance matrix of the camera pose in the previous frame, respectively. Let $Z_k = \{z_1, z_2, \ldots, z_n\}$ be a set of 3D points in the scene, which were added to the map in previous frames. Since the camera is freely moving in 3D space, the observed features are consistently changing. In the other words, some parts of the scene leave the camera's field of view and hence their associated feature points should be removed from the map. On the other hand, the camera views new regions and consequently new feature points are detected. Once the detected features are tracked successfully, they can be included in the map. Therefore, Z_k needs to be updated constantly. It is assumed that $U_k = \{u_1, u_2, \ldots, u_n\}$ represents the associated observed feature points in frame k obtained through the feature tracking procedure. Furthermore, according to the standard pinhole camera model with focal length (f_x, f_y) and camera center (c_x, c_y) for camera state x_k, the projection of every 3D point $z_i \in Z_k$ on frame k is computed using Equation (5).

$$y_k^i = \begin{pmatrix} c_x + f_x \frac{R_{k,1}^T z_i + t_x^k}{R_{k,3}^T z_i + t_z^k} \\ c_y + f_y \frac{R_{k,2}^T z_i + t_y^k}{R_{k,3}^T z_i + t_z^k} \end{pmatrix} \in R^2. \tag{5}$$

where $R_{k,i}^T$ denotes the i-th row of the rotation matrix R_k. With the particle filter, the intention is to successively estimate the posteriori density $p(x_k \mid y_k, Z_k)$. This density is approximated as a weighted average of sample particles drawn from state space.

$$p(x_k|y_k, Z_k) = \sum_{i=1}^{m} \pi_k^i \delta(x - x_k^i). \tag{6}$$

where m is the number of particles and x_k^i is the i-th particle drawn from state apace. π_k^i is the weight of x_k^i that is proportional to the conditional likelihood $p(y_k|x_k, Z_k)$. Actually, each particle is an estimate of the filter state and its weight is the probability of the particle being correct. Given x_k^i, the i-th particle in frame k, π_k^i is computed using Equation (7).

$$\pi_k^i = p(y_k|x_k^i, Z_k) \propto exp(-\sum_{p=1}^{n} (u_p - y_k^p)^T (u_p - y_k^p)). \tag{7}$$

The value of π_k^i is representing the closeness of the associated particle to the correct likelihood density.

At the outset of the system, weights of the particles are initialized with equal values (i.e., set to $1/m$). In the developed PF for frame k, at first, particles are resampled in accordance to their weights. In the resampling process, since the particles are sampled with replacement, particles with higher

weights are more likely to be drawn and hence are multiplied. On the other hand, particles with lower weights tend to be deleted and are hence replaced by particles with higher weights. Therefore, over time, the weighted particles tend to approximate the correct posteriori density. The resampled particles are then diffused using the state transition model, as described in Equation (4). Thereafter, the weights of the diffused particles are obtained using Equation (7). The new particles' weight is normalized such that $\sum_{i=1}^{m} \pi_k^i = 1$. The new state of the filter is computed as the weighted average of diffused particles.

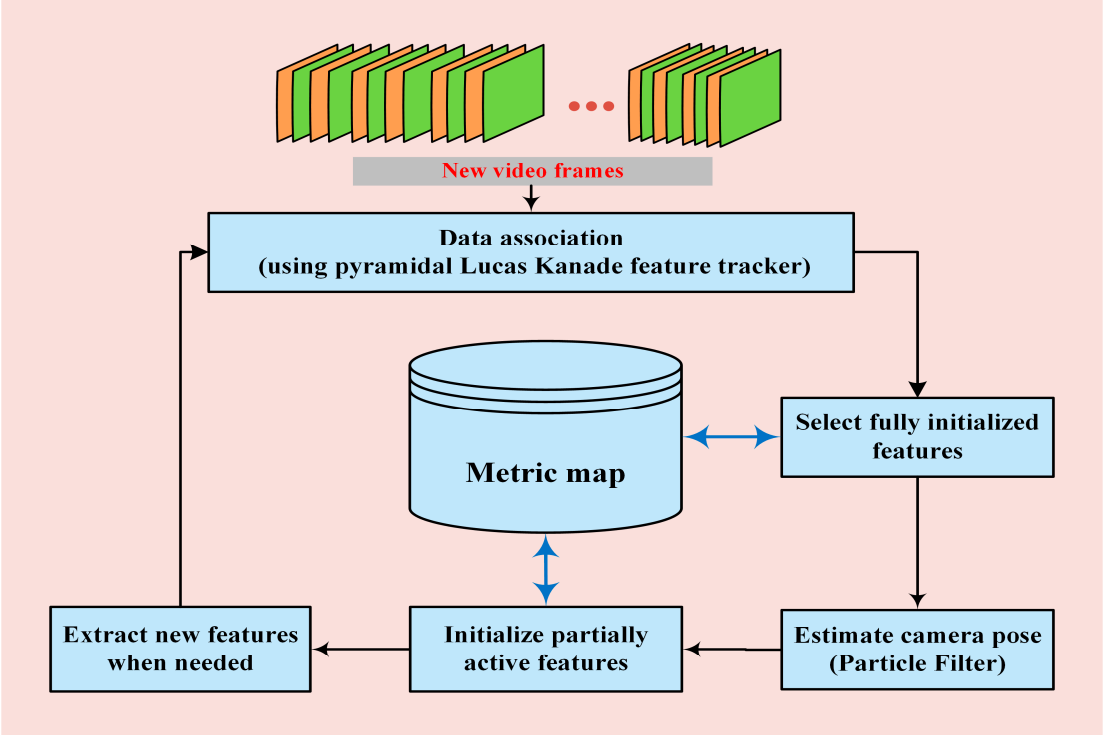


Figure 1. Overview of the proposed approach.

3.2. Feature Extraction and Tracking

In the presented algorithm, a data association step is carried out with the help of extracted FAST feature points [20]. However, in the experimental section, the performances of some known feature point detectors are compared. In any feature-based visual SLAM method, data association is indeed the procedure of feature tracking, which highly affects the accuracy of camera localization and map extension. Early feature tracking methods used a correlation window to find corresponding features in successive frames. A simple correlation window cannot produce favorable results for camera quick motions. In this work, a pyramidal feature tracking technique proposed by Bouguet is used [21]. As showed in Figure 2, employing this technique for tracking a feature, at first, a pyramid of a window centered at a feature point is constructed. Then, using the popular Lucas-Kanade method [22], the optical flow at the coarsest level is computed. The result of this level is then propagated to the next level as an initial guess for the Lucas-Kanade algorithm. This process continues until it reaches the finest level. Computed optical flow at the finest level indicates the feature displacement in the next frame. Meanwhile, a tracking score is assigned to the acquired matched feature points in two successive frames. This tracking score is a normalized value in the range [0, 1], indicating the quality of obtained matched feature points. Higher values for the tracking score indicate more reliable feature matchings. Selecting a proper number of pyramid levels usually depends on the resolution of the video frames. Generally, for VGA quality videos, constructing pyramids with three or four levels is a convenient selection. Often, due to image blurriness or repetitive textures, matched features thus

obtained may contain noisy or wrong correspondences. The RAndom SAmple Consensus (RANSAC) robust estimator [23] is used to remove noisy and incorrect correspondences.

It is noteworthy that all features, both those with known 3D coordinates (fully active) and those not yet initialized (partially active), will be tracked. Nevertheless, for the partially active features, it is necessary to store their pixel coordinates until they are triangulated. If a partially active feature cannot be initialized after a predetermined number of frames, then it is removed from the list of partially active features. More detailed information on feature initialization and management is explained in subsequent sections.

3.3. Pose and Map Initialization

In the proposed framework, there is no way to recover the depth of newly added features, except using the structure of features with a determined 3D position. From a set of 2D-2D feature correspondences in two or more frames, it is only possible to estimate the depth of corresponding features with a scale factor. To construct a metric map, it is necessary to provide some prior information about the geometry of the 3D scene. Generally, a sparse set of feature points with a known 3D location or a known CAD model is sufficient for this purpose. In the reported case, a chessboard with a known cell size that is placed on a desk was used to estimate the camera pose for the initial few frames (about 10–15 frames). As depicted in Figure 3, the origin of the world coordinate system is aligned to one corner of the chessboard. Since the size of the chessboard cells is known, they are selected as feature points with a known 3D position in the world coordinate. These 3D points along with their projections on each frame make a collection of 3D-2D correspondences in each frame that enables the algorithm to estimate camera pose parameters with a high precision.

It is obvious that the estimated camera poses are metric and there is no scale ambiguity. To make the initial map, it is only necessary to track the FAST feature points detected in the first frame. Once the parallax of initially detected features exceeds a predetermined threshold, they can be triangulated to construct the initial map. In the reported work, the camera tracking procedure relies on data association between successive frames. In turn, the estimated camera pose in each frame is used to estimate the depth of newly extracted features. In fact, the estimated depth information of the initial known targets propagates to subsequent images.

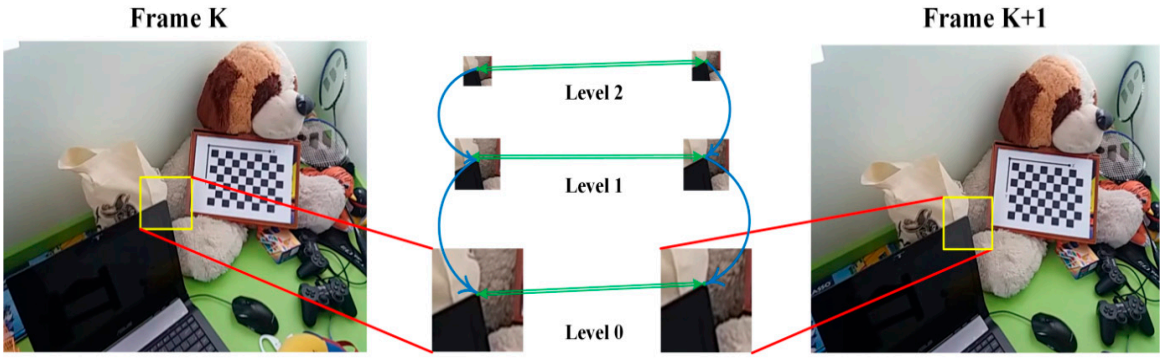


Figure 2. Pyramidal implementation for feature tracking in two consecutive frames.

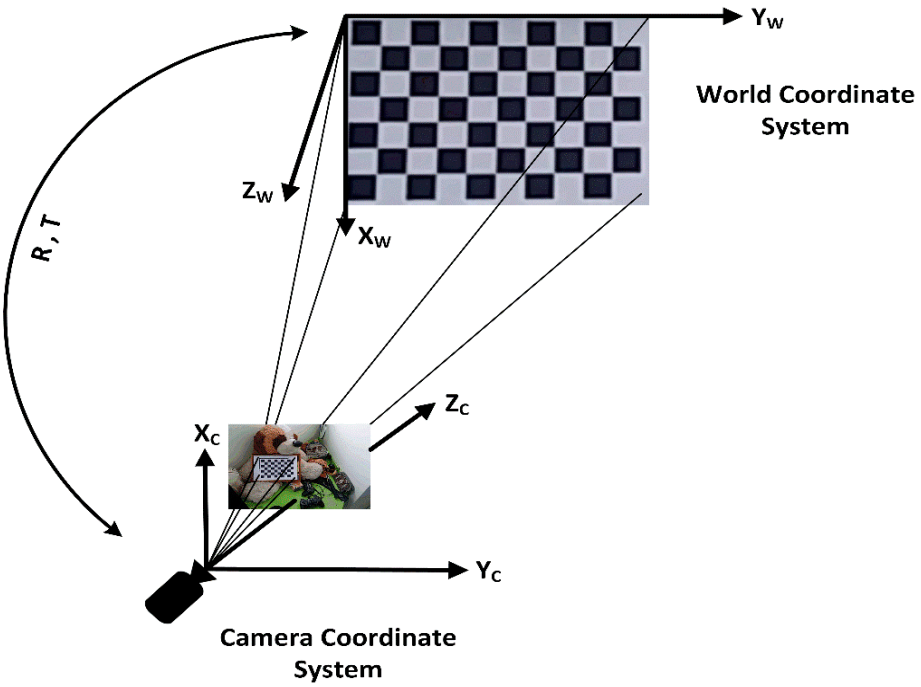


Figure 3. Definition of world and camera coordinate systems in the first image.

3.4. Initialization of New Features

It is known that the estimation of camera pose parameters and scene structure are tightly coupled problems. In other words, the accurate estimation of camera pose parameters leads to more precise triangulation of newly extracted features. Estimating the 3D coordinates of newly detected features is perhaps the most challenging task in extendible camera tracking for a freely moving camera in unknown environments. However, one may use loop closure or known objects in the scene to avoid drift enlargement. The proposed system is capable of adding new detected feature points to the map after their 3D coordinates have been estimated.

Once a new feature is detected, it cannot be initialized immediately. This is due to the fact that having the position of any detected feature on a single monocular image does not provide any information about its depth. In other words, to acquire information about the depth of a feature point, the position of feature points within subsequent frames has to be located. Moreover, due to the narrow-baseline nature of successive frames in a given video, linear triangulation of newly extracted features in two successive frames leads to a large error in the depth calculation. To deal with this problem, it is usual to initialize the depth of newly detected features in the reference frame with some sort of ambiguity. Handling this ambiguity, Kwok and Dissanayake [24] proposed a multiple hypothesis distribution along the semi-infinite ray from the measured feature to infinity. The validity of each hypothesis is evaluated in subsequent frames until the best approximation to the feature location is retained. Davison [25] proposed a particle filter-based approach to represent the initial depth of new features. The observation of associated features in subsequent frames is used to update the initial depth distribution until a sharp Gaussian posteriori is obtained. Eade and Drummond [26] employed a similar strategy; however, they used inverse depth parameterization incorporated in the initial depth distribution. The accuracy of the foregoing methods is significantly decreased in a low parallax situation. In the method presented in this manuscript, the linear triangulation method is used to initialize features, but it is deferred until the parallax exceeds a predefined threshold, as given in Equation (8).

$$|x_i - x_j| > \rho, \quad i < j. \tag{8}$$

where x_i, x_j are the pixel coordination of a given feature point in frames i, j, and ρ is the foregoing threshold (in the reported case $\rho = 25$ pixels). This delay for the depth estimation of newly extracted features is because quick initialization of newly extracted features is not necessary whilst a high enough number of features exist in a processing frame.

3.5. Feature Management

Automatic management of the extracted features is an important problem in the proposed framework. In this context, it should be decided when the detection of new features is required, as well as when it is necessary to remove a feature from the constructed map. It is clear that as the camera moves around, new regions of the scene are viewed, and it is necessary to detect new features for tracking. Another issue that should be taken into account is the optimized number of features in the map of the world. Although only four non-coplanar fully active features is enough for the estimation of camera pose parameters in the PF framework, it is strongly suggested that more features are used to obtain more reliable results. It is clear that tracking many features in each frame will improve the accuracy of the estimated pose at the cost of efficiency reduction. In the experiments, minimum and maximum numbers of the tracked features are set to 30 and 80, respectively. In addition, the selected FAST features are distributed evenly over the frame to better represent most regions of the observed scene.

There are two conditions where the algorithm is forced to remove a feature from the map. The first condition is when a feature disappears from the camera field of view and consequently goes outside the image boundaries. The second condition occurs once a feature point cannot be robustly tracked. The latter one occurs whenever, due to occlusion or a poor tracking score, the position of a specified feature point in subsequent frames cannot be found. Practically, we found that any tracking score less than 0.9 is not suitable and the associated feature point is removed.

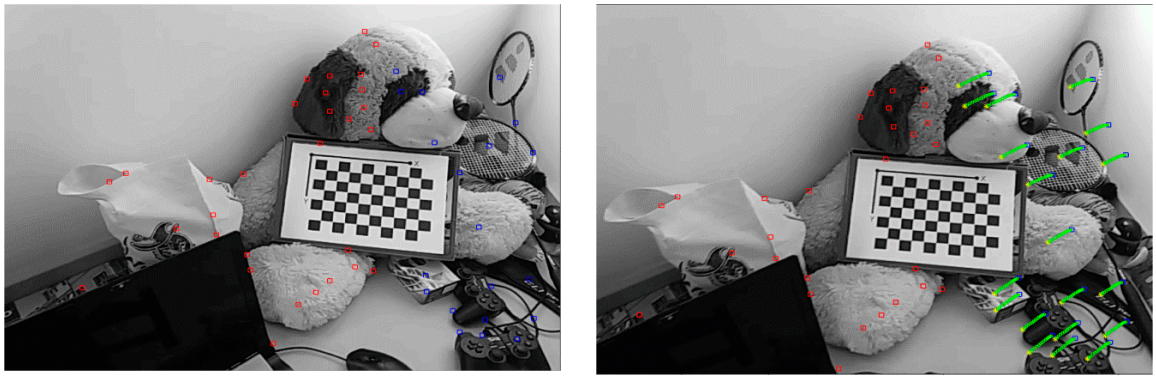
4. Experimental Results

In the reported experiments, using a freely moving handheld camera, a video sequence of 1185 frames is recorded. The resolution of the video frames is 480×640 pixels and the video is captured at the rate of 30 fps. Furthermore, the experimental scene is a collection of cluttered objects placed on a desk. A planar chessboard pattern is placed on the desk that is used for calculation of the ground truth camera pose. It is worth noting that the foregoing marker is solely employed to calculate the ground truth camera pose. Hence, any extracted corner point on the marker will be considered as an ordinary natural feature point and upon successful tracking in subsequent frames, it needs to be initialized regardless of its known 3D coordinates. Additionally, it is assumed that the camera is calibrated in advance. To do so, a collection of images of a chessboard with a known cell size taken from different viewpoints was used to estimate the intrinsic parameters of the camera. This routine is performed using a technique reported in a work by Zhengyou [27].

Figure 4 shows the result of the feature tracking routine as described in previous sections. Fully active features are marked with red squares and newly detected features are marked with blue squares. New features are tracked for some few frames (green trails in Figure 4b) and are finally triangulated (yellow asterisks in Figure 4b). Figure 5 shows the estimated camera trajectory against ground truth in 3D space, as well as its projection on the XY plane. As shown in Figure 5, the position of the camera center in 3D space is tracked with high precision in comparison to the ground truth camera path.

In Figure 6, components of the estimated and ground truth camera position and orientation are depicted. As it is shown in Figure 6, in spite of the lengthy input video, the camera trajectory is estimated with a high accuracy. Translation error of the estimated camera pose in x and y directions is negligible and a small error along the z direction is observed. Estimation error for components of the rotation part of the camera pose at the end of the path tends to increase, which is due to the dead reckoning nature of the proposed algorithm. The statistics of absolute translation and rotation error

over all frames of the input sequence are reported in Table 1. Statistical parameters for absolute camera rotation error in Table 1 are represented in the Euler angle. In the experiments, the average rotation error in all directions is less than 3 degrees.

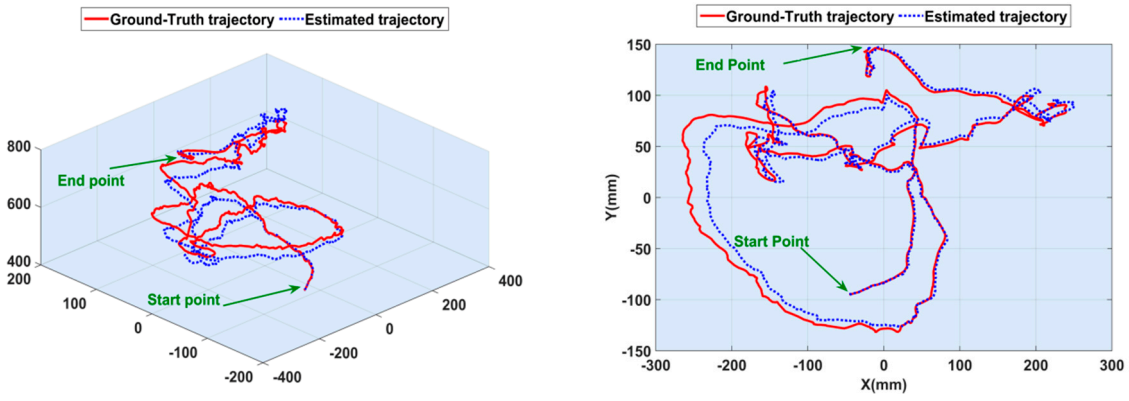


(**a**) (**b**)

Figure 4. Feature tracking until initialization: (**a**) Active features + newly detected features; (**b**) active features + tracking path of new features + initialization point of new features.

Table 1. Absolute translation and rotation errors.

Statistical Parameters	Absolute Translation Error (mm)			Absolute Rotation Error (deg)		
	T_x	T_y	T_z	R_x	R_y	R_z
Mean	3.83	4.49	16.27	2.37	3.41	2.48
Standard Deviation	3.44	3.25	10.51	1.69	2.84	1.36
Minimum	0.004	0.002	0.007	0.001	0.001	0.003
Maximum	15.37	13.71	46.70	7.46	15.06	5.66



(**a**) (**b**)

Figure 5. Camera trajectory estimated using PF versus ground truth data: (**a**) 3D view; (**b**) projections of estimated and ground truth trajectories on the XY plane.

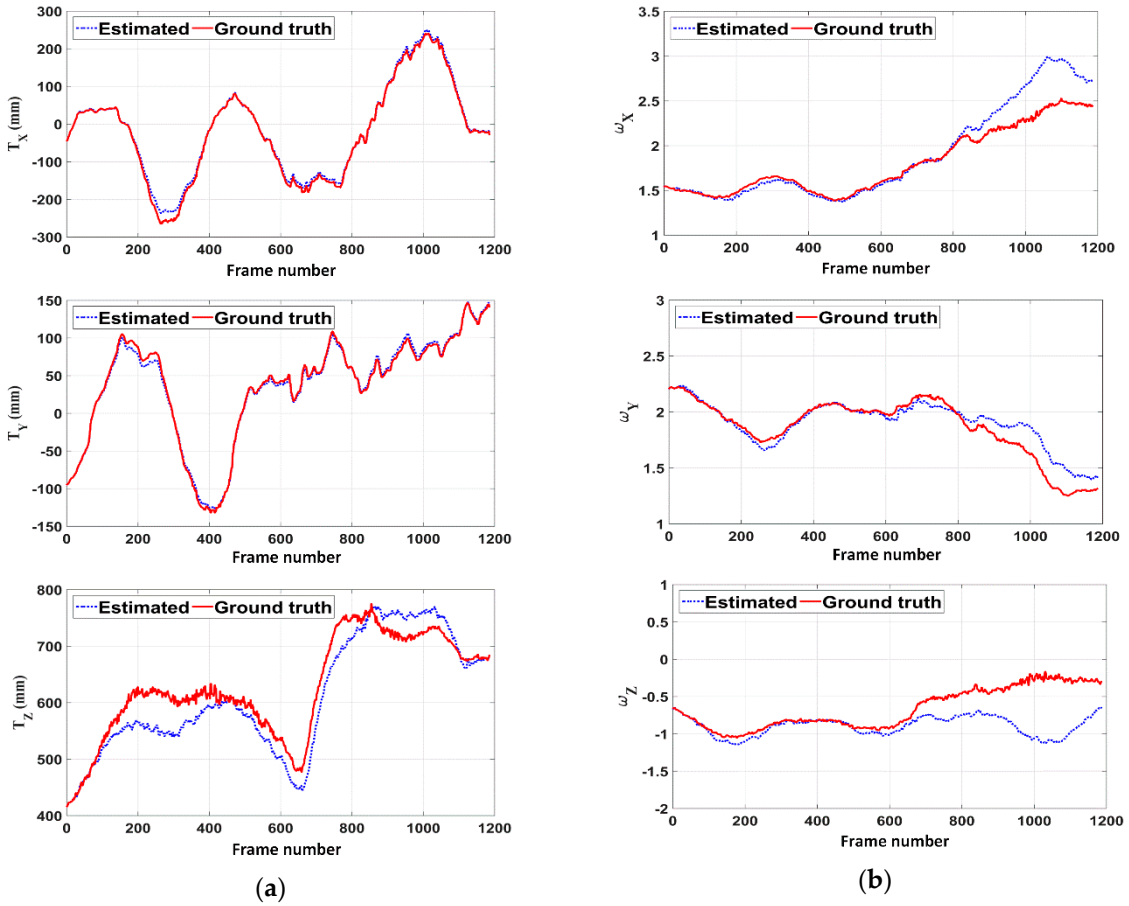


Figure 6. Components of estimated and ground truth camera pose: (**a**) Camera position part; (**b**) camera rotation part.

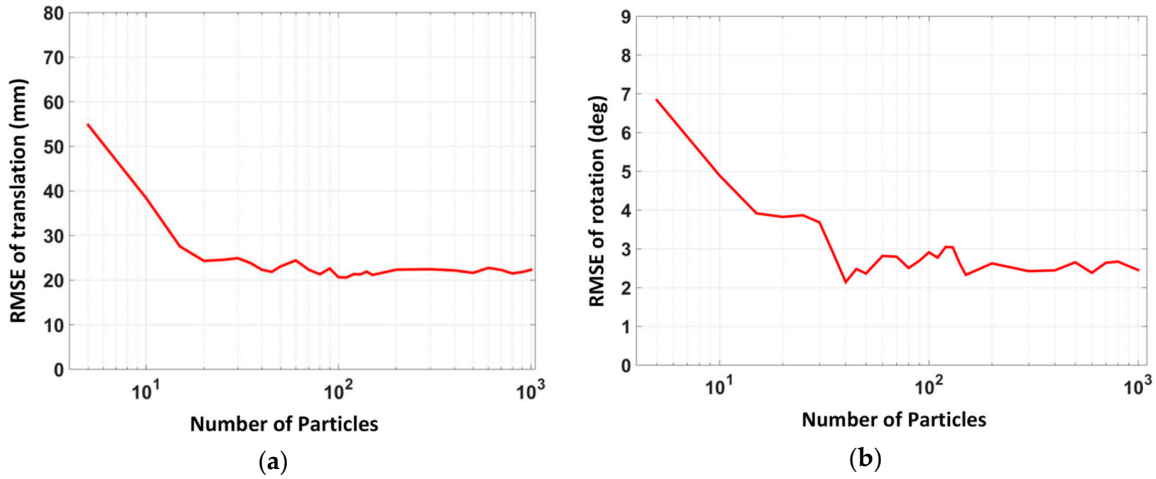
To evaluate the impact of feature type on tracking accuracy, the experiments were repeated using different popular feature extractors. For this purpose, HARRIS [28], MINEIGEN [29], SURF [30], and FAST feature extraction methods were used. For comparison, the Root Mean Square Error (RMSE) criterion was used to evaluate the results for both the translation (i.e., $RMSE_{Trans}$) and the rotation (i.e., $RMSE_{Rot}$) parts of the camera pose. $RMSE_{Trans}$, $RMSE_{Rot}$ quantities are defined in Equation (9).

$$\begin{aligned} RMSE_{Trans} &= \left(\frac{1}{n} \sum_{k=1}^{n} \left| t_{est}^{k} - t_{gt}^{k} \right|^2 \right)^{\frac{1}{2}}, \\ RMSE_{Rot} &= \left(\frac{1}{n} \sum_{k=1}^{n} \left| r_{est}^{k} - r_{gt}^{k} \right|^2 \right)^{\frac{1}{2}}. \end{aligned} \tag{9}$$

where n is the number of video frames. t_{est}^{k} and t_{gt}^{k} are the estimated and ground truth camera position at frame k, respectively. Similarly, r_{est}^{k} and r_{gt}^{k} are the Euler angles representation of estimated and ground truth camera rotation at frame k, respectively. In Table 2, the performance of the proposed approach in terms of $RMSE_{Trans}$ and $RMSE_{Rot}$ for the aforementioned point detectors is presented. As the results show, features extracted using FAST and SURF methods present a significantly better performance in the proposed visual SLAM algorithm. However, due to the higher computational cost of extracting SURF features, FAST features were adopted in the reported work. Furthermore, comparing the performance of the reported approach against other camera tracking algorithms, estimated camera poses using EKF, UKF, and EPnP [31] methods are obtained and the RMSE of camera pose estimation and their computation time are reported in Table 3. One can easily see that the proposed PF-based

approach outperforms the other nonlinear filtering approaches, i.e., EKF and UKF, as well as the EPnP method. As given in the last column of Table 3, the computation cost of the proposed PF-based approach is much lower than other methods.

It is clear that as the number of particles grows, the precision of the proposed approach improves. On the other hand, increasing the number of particles leads to a larger computational cost and it is thus necessary to seek a tradeoff by selecting an appropriate number of particles. Figure 7 depicts camera tracking accuracy versus the number of particles. Experiments show that using 150 particles to approximate the posteriori density is a suitable selection to find equilibrium between accuracy and time consumption.



(**a**) (**b**)

Figure 7. RMSE of (**a**) Translation part; (**b**) rotation part of camera pose against number of particles used in PF.

Table 2. Tracking accuracy using different feature point detectors.

Feature Extraction Method	$RMSE_{Tran}$ (mm)	$RMSE_{Rot}$ (deg)
HARRIS	119.14	17.3
MINEIGEN	112.18	17.7
SURF	23.1	5.1
FAST	22.55	2.5

Table 3. Comparison of different camera pose estimation methods (reported values are average of five executions for each of the methods).

Pose Estimation Method	$RMSE_{Tran}$ (mm)	$RMSE_{Rot}$ (deg)	Computation Time (s per frame)
EKF	37	4.2	0.31
UKF	27.8	3.6	0.35
EPnP	38.2	4.4	0.86
PF	22.5	2.5	0.20

5. Conclusions

This paper reports a novel feature-based monocular SLAM solution. Data association between successive frames of the input video is performed using tracking FAST feature points. Furthermore, to handle the feature tracking problem in the presence of quick movements of the camera, a pyramidal scheme of the Lucas-Kanade feature tracker is used. Using a PF framework in a frame-by-frame sequence, the proposed system estimates the camera position and orientation sequentially. Depth recovering for newly detected feature points is a serious problem in the feature-based visual SLAM problem, which has

a strong influence on the accuracy of the estimated camera pose. In the proposed framework, a delayed method for the depth calculation of newly detected features is adopted.

In future work, the intent will be to address the problem of camera trajectory drift for long-term video sequences. It is clear that in long-term visual SLAM solutions, the cumulative error for orientation and translation of the camera will increase if the system does not receive any information in the form of calibrated markers or objects with known locations. Increasing cumulative error of camera pose parameters will introduce drift to the camera trajectory and will consequently adversely affect the accuracy of the feature initialization routine. To handle this problem, the plan is to extend the reported work to use known landmarks or to detect loop closures.

Author Contributions: S.A.H. and P.K. contributed to the main idea of this article. S.A.H. performed experiments and data collection. P.K. supervised research activity planning and execution. Both authors contributed to the writing process of the article.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

1. Ababsa, F.; Mallem, M. Robust camera pose estimation using 2d fiducials tracking for real-time augmented reality systems. In Proceedings of the ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry, Singapore, 16–18 June 2004; pp. 431–435.
2. Xu, K.; Chia, K.W.; Cheok, A.D. Real-time camera tracking for marker-less and unprepared augmented reality environments. *Image Vis. Comput.* **2008**, *26*, 673–689. [CrossRef]
3. Dong, Z.; Zhang, G.; Jia, J.; Bao, H. Efficient keyframe-based real-time camera tracking. *Comput. Vis. Image Underst.* **2014**, *118*, 97–110. [CrossRef]
4. Clemente, L.A.; Davison, A.J.; Reid, I.D.; Neira, J.; Tardós, J.D. Mapping large loops with a single hand-held camera. In Proceedings of the Robotics: Science and Systems, Atlanta, GA, USA, 27–30 June 2007; pp. 352–360.
5. Eade, E.; Drummond, T. Unified loop closing and recovery for real time monocular slam. In Proceedings of the British Machine Vision Conference, Leeds, UK, 1–4 September 2008; pp. 136–145.
6. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: New York, NY, USA, 2003.
7. Fitzgibbon, A.W.; Zisserman, A. Automatic camera recovery for closed or open image sequences. In Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany, 2–6 June 1998; pp. 311–326.
8. Pollefeys, M.; Koch, R.; Van Gool, L. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In Proceedings of the Sixth International Conference on Computer Vision, Bombay, India, 4–7 January 1998; pp. 90–95.
9. Triggs, B.; Mclauchlan, P.; Hartley, R.; Fitzgibbon, A. Bundle adjustment—A modern synthesis. In Proceedings of the International Workshop on Vision Algorithms, Corfu, Greece, 21–22 September 1999; pp. 298–372.
10. Klein, G.; Murray, D. Parallel tracking and mapping for small AR workspaces. In Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; pp. 1–10.
11. Endres, F.; Hess, J.; Sturm, J.; Cremers, D.; Burgard, W. 3-D mapping with an RGB-D camera. *IEEE Trans. Robot.* **2014**, *30*, 177–187. [CrossRef]
12. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. G2o: A general framework for graph optimization. In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 3607–3613.
13. Maidi, M.; Ababsa, F.; Mallem, M.; Preda, M. Hybrid tracking system for robust fiducials registration in augmented reality. *Signal Image Video Process.* **2015**, *9*, 831–849. [CrossRef]
14. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. Monoslam: Real-time single camera slam. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
15. Kim, J.S.; Hong, K.S. A recursive camera resectioning technique for off-line video-based augmented reality. *Pattern Recogn. Lett.* **2007**, *28*, 842–853. [CrossRef]

16. Hoseini, S.A.; Kabiri, P. A feature-based approach for monocular camera tracking in unknown environments. In Proceedings of the 3rd International Conference on Pattern Recognition and Image Analysis, Shahrekord, Iran, 19–20 April 2017; pp. 75–79.
17. NewCombe, R.A.; Lovegrove, S.J.; Davison, A.J. DTAM: Dense Tracking and Mapping in Real-time. In Proceedings of the International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2320–2327.
18. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In Proceedings of the 13th European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 834–849.
19. Murray, R.M.; Sastry, S.S.; Zexiang, L. *A mathematical Introduction to Robotic Manipulation*, 1st ed.; CRC Press: Boca Raton, FL, USA, 1994.
20. Rosten, E.; Drummond, T. Fusing points and lines for high performance tracking. In Proceedings of the Tenth IEEE International Conference on Computer Vision, Beijing, China, 17–21 October 2005; pp. 1508–1515.
21. Bouguet, J.Y. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corp.* **2001**, *5*, 1–10.
22. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vancouver, BC, Canada, 24–28 August 1981; pp. 674–679.
23. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
24. Kwok, N.M.; Dissanayake, G. An efficient multiple hypothesis filter for bearing-only slam. In Proceedings of the International Conference on Intelligent Robots and Systems, Shanghai, China, 28 September–2 October 2004; pp. 736–741.
25. Davison, A.J. Real-time simultaneous localisation and mapping with a single camera. In Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; pp. 1403–1410.
26. Eade, E.; Drummond, T. Scalable monocular slam. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; pp. 469–476.
27. Zhengyou, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]
28. Harris, C.; Stephens, M. A combined corner and edge detector. In Proceedings of the 4th Alvey Vision Conference, Manchester, UK, 31 August–2 September 1988; pp. 147–151.
29. Shi, J.; Tomasi, C. Good features to track. In Proceedings of the 9th IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 1994; pp. 593–600.
30. Bay, H.; Ess, A.; Tuytelaars, T.; Van Gool, L. Speeded-up robust features (surf). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359. [CrossRef]
31. Lepetit, V.; Moreno-Noguer, F.; Fua, P. Epnp: An accurate o(n) solution to the pnp problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166. [CrossRef]