Improving Localization Robustness in Monocular SLAM Using a High-Speed Camera

Peter Gemeiner
Automation and Control Institute
Vienna University of Technology
gemeiner@acin.tuwien.ac.at

Andrew J. Davison
Department of Computing
Imperial College London
ajd@doc.ic.ac.uk

Markus Vincze
Automation and Control Institute
Vienna University of Technology
vincze@acin.tuwien.ac.at

***Abstract*— In the robotics community localization and mapping of an unknown environment is a well-studied problem. To solve this problem in real-time using visual input, a standard monocular Simultaneous Localization and Mapping (SLAM) algorithm can be used. This algorithm is very stable when smooth motion is expected, but in case of erratic or sudden movements, the camera pose typically gets lost. To improve robustness in Monocular SLAM (MonoSLAM) we propose to use a camera with faster readout speed to obtain a frame rate of 200Hz. We further present an extended MonoSLAM motion model, which can handle movements with significant jitter. In this work the improved localization and mapping have been evaluated against ground truth, which is reconstructed from off-line vision. To explain the benefits of using a high frame rate vision input in MonoSLAM framework, we performed repeatable experiments with a high-speed camera mounted onto a robotic arm. Due to the dense visual information MonoSLAM can faster shrink localization and mapping uncertainties and can operate under fast, erratic, or sudden movements. The extended motion model can provide additional robustness against significant handheld jitter when throwing or shaking the camera.**

I. Introduction

For mobile robotics it is essential to continuously localize and estimate 3D positions of new landmarks in an unknown environment. The *localization* and *mapping* can be addressed with an incremental probabilistic approach, which is known as SLAM (for an overview please refer to [1]).

As input for SLAM different kinds of sensors (e.g. laser [2], sonars [3]) can be used. One of the most interesting (cost, weight, etc.) and challenging sensors is a single perspective-projective camera. When observing the environment with a camera, the depth information of new landmarks can not be directly acquired. To recover this depth information the camera has to move, and observe these landmarks from different viewpoints.

Davison et al. introduced the first real-time Monocular SLAM (MonoSLAM) (recently summarized in [4]) algorithm. The camera motion estimation and incremental map building (from new landmarks) are computed within a standard Extended Kalman Filter (EKF) SLAM framework. An alternative SLAM framework is typically based on FastSLAM-type particle filter algorithms.

One of the underlying assumptions in MonoSLAM is that the camera is expected to move smoothly. This classical EKF SLAM framework is prone to fail as soon as sudden or erratic camera movements occur, and can not reliably recover when the pose is lost. The smooth motion assumption attracted recently attention, and several authors proposed solutions to this problem.

Williams et al. presented in [5] an additional relocalization algorithm, which can operate parallel to MonoSLAM, and increases robustness against camera shakes and occlusions. A recent improvement of this algorithm is using randomized lists classifier and `RANSAC` to determine the pose robustly [6].

To handle erratic camera motion and occlusions, Pupilli and Calway [7] presented a visual SLAM framework based on a FastSLAM-type particle filter. This work tackles mainly the problem of camera robust localization, and Chekhlov et al. [8] extended this SLAM framework to operate over a large range of views using a SIFT-like spatial gradient descriptor.

Another visual SLAM framework based on a FastSLAM-type particle filter introduced by Eade and Drummond [9] can incorporate hundreds of features in real-time. However, the filter needs to be adapted for closing loops over large trajectories.

The monocular SLAM algorithms discussed above [5, 6, 7, 8, 9] are using a standard 30Hz camera.

Interesting high-speed applications already enabled to e.g. measure the motion of a waving flag or a flying ball [10]. Komuro and Ishikawa proposed in [11] a method for three-dimensional object tracking from noisy images. The noise typically produced by high-speed cameras can be tackled with a proper noise models.

Another way to improve the robustness of tracking and estimating features is to increase the sampling rate of the camera. This has been beautifully demonstrated by Ishii et al. [12]. They introduced a 1ms visual feedback system using massively parallel processing. Since image acquisition time is countered with equally fast image processing (both operate at 1kHz), very fast motion of a bouncing ball can be tracked. However, objects need to be bright in front of dark background.

Our idea is to show that monocular SLAM can operate at a frame rate of 200Hz and that this improves the robustness of localization considerably. The intention is to show that with a higher frame rate a performance similar to the recent improvements discussed above [5, 6, 7, 8, 9] can be achieved, while additionally all these measures could again be used to even further improve performance.

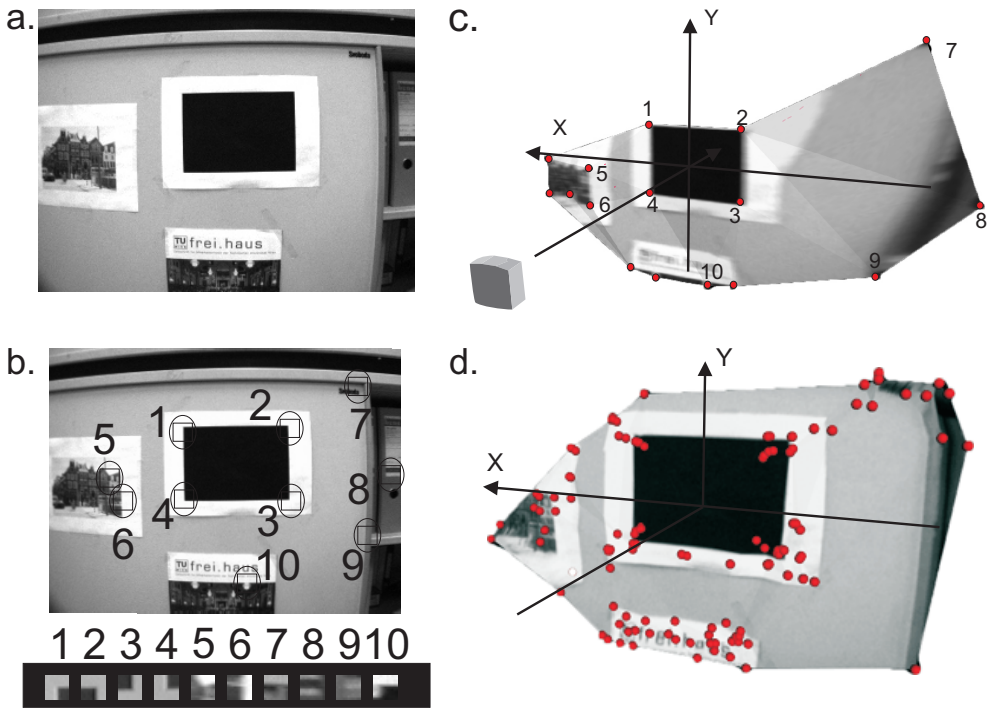

Fig. 1. The only sensor input in MonoSLAM are images from a single camera (a). As the camera moves, new distinctive features are detected (b). The output of MonoSLAM is the camera pose and a sparse three-dimensional map of these distinctive features (c). An alternative to MonoSLAM sparse mapping is a more dense structure reconstruction using nonlinear optimization techniques (d).

From the variety of state-of-the-art monocular SLAM frameworks mentioned before [4, 7, 8, 9] we are using in this work MonoSLAM algorithm, because we think that it is the most advanced.

The contribution of this paper has three parts:

- using a high-speed camera for monocular SLAM for the first time,
- more robust MonoSLAM localization using an extended motion model and
- experiments exploiting the repeatable motion of a robotic arm compared with accurate ground truth from off-line vision and an experiment to throw the camera to indicate robustness.

In this paper the standard MonoSLAM algorithm and the ground truth calculations are briefly introduced in Section II. Details and discussions related to the proposed more robust motion model are introduced in Section III. Section IV presents experimental results, performing well-controlled motions with the camera mounted on the robotic arm and handheld sequences. Section V closes with a discussion and an outlook to the future work.

II. Localization and Mapping Algorithms

Before explaining the proposed more robust localization and benefits of using a high-speed camera, a short outline of the top-down Bayesian MonoSLAM system (see Fig. 1) is given in two parts. Firstly, we summarize MonoSLAM from the system perspective (input, initial assumptions and output). Secondly, a short introduction to additional existing modules used in this work is presented. This section concludes with a brief description of offline computer vision algorithms used in this work for the ground truth acquisition.

A. MonoSLAM and Additional Modules

The only sensor input in MonoSLAM used in this work are images from a single perspective-projective camera as displayed in Fig. 1-a. When the camera moves new distinctive features can be detected (e.g. corners with numbers 5-10 in Fig. 1-b and Fig. 1-c). The output of MonoSLAM is the camera poses and a sparse map of recovered features, as depicted in Fig. 1-c. Due to the real-time demand *mapping* in MonoSLAM is not playing a crucial role, and should rather support *localization*.

In MonoSLAM, the following conditions are assumed:

- a well-calibrated camera,
- rigid scene with textured objects (not moving),
- constant lighting conditions,
- one initial object with known geometry (e.g. features with numbers 1-4 in Fig. 1-b and Fig. 1-c) and
- camera is expected to move smoothly.

Three additional modules have been used in this work to improve the performance of MonoSLAM.

- To permit initialization of features of all depths, Montiel et al. introduced an *inverse-depth* [13] parameterization.

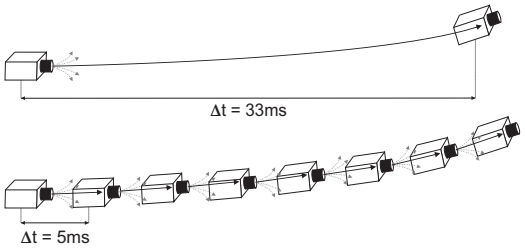

Fig. 2. MonoSLAM can handle smooth camera movements using the standard camera (difference of timestamps - Δt is equal 33ms), and the high-speed camera (Δt is equal 5ms) is not needed.

This parameterization can cope with features, which are very distant from the camera.

- Instead of detecting Shi and Tomasi [14] features, the *Fast* feature detector [15] is applied. The best *Fast* features are then found using the Shi and Tomasi [14] cornerness measure.
- Joint Compatibility Branch and Bound (JCBB) [16] is employed to search for the largest number of jointly compatible pairings.

B. Ground-truth Computation

To precisely evaluate experiments with MonoSLAM motion models and the high-speed camera, we need accurate *localization* and *mapping* ground truth. For the computation of accurate ground truth we used two offline, iterative and *computationally intensive* algorithms from the field of computer vision. The first is a camera pose algorithm presented by Schweighofer and Pinz [17]. This algorithm needs a object with known geometry (black rectangle in Fig. 1-b), which is visible in every frame. The second is a structure reconstruction algorithm based on nonlinear optimization, summarized by Triggs et al. in [18]. An example of the scene reconstruction computed with the combination of these two algorithms is depicted in Fig. 1-d. These computer vision techniques provide more dense and more accurate results then MonoSLAM reconstruction (see Fig. 1-c). The detailed explanation of these algorithms is out of scope of this paper, but in the experimental section an accuracy example is provided.

III. Improving Localization Robustness

To explain the details related to the more robust *localization*, this section comprises four parts. Firstly, the benefits of the *localization* using the high-speed camera are presented. Secondly, the proposed more robust MonoSLAM motion model is explained. Thirdly, a discussion of motion parameters is included. Fourthly, a problem of skipping vision information when using an asynchronous high-speed camera is reviewed.

A. Dense Visual Information

The standard camera (difference of timestamps - Δt is equal 33ms) provides rather sparse vision input, but when a smooth movement is expected (see Fig. 2), it is sufficient for MonoSLAM to robustly *localize*. If an erratic camera motion is performed, EKF based MonoSLAM very likely lose the pose. Observing the scene features with faster readout speed (e.g. Δt is equal 5ms) allows to update the camera state more frequently, and this can prevent MonoSLAM to lose the pose as depicted schematically in Fig. 3.

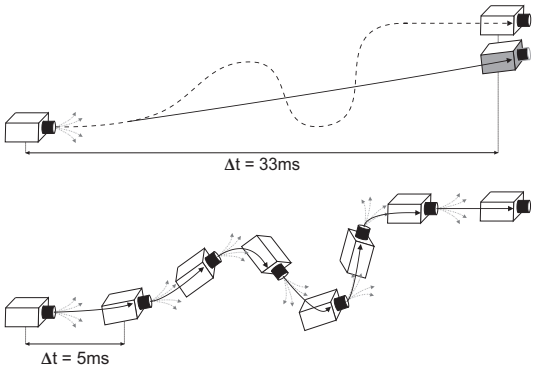

Fig. 3. If erratic camera movements occur, MonoSLAM using the standard camera (difference of timestamps - Δt is equal 33ms) loses the pose. An example is displayed in the upper row, where the camera pose (solid line) drifted away from the true pose (dashed line). A high-speed camera provides more dense visual information, which helps to handle this erratic motion, as depicted in the bottom row.

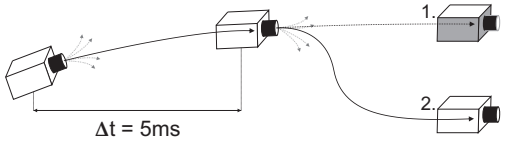

Fig. 4. If erratic or jitter movements occur and dense vision information are available, the second order motion model (2) is assumed to be more robust than the first order model (1).

B. Higher Order Motion Model

In MonoSLAM, it is assumed that the camera linear and angular velocities may change in every frame, but they are expected to be *constant* in average. In other words, the camera movements are approximated using the so-called *constant linear and angular velocity* motion model. This model assumes that in each time step the unknown linear ($\vec{a}^W$) and the unknown angular ($\vec{\alpha}^R$) accelerations cause impulses of linear

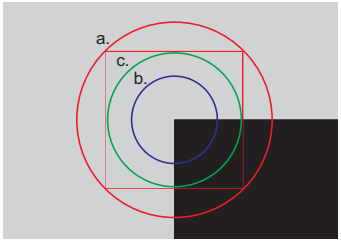

Fig. 5. To update the camera pose in each time step, known scene features' descriptors (square) are matched with predicted features' positions (a). *Localization* using the camera with fast readout speed (e.g. Δt is equal 5ms) causes these predicted features' positions to shrink obviously (b). To handle more erratic movements the motion noise uncertainty P_n should contain larger values, which cause the features' searching regions to enlarge (c).

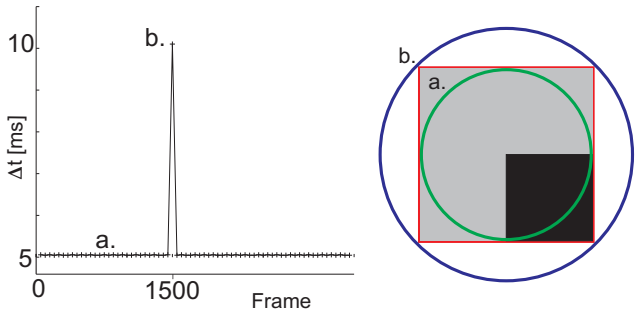

Fig. 6. The asynchronous high-speed camera can not guarantee constant time between consecutive frames. We expect that the readout speed unpredictably decreases for some frames (left image). This Δt diminution causes the features' searching ellipses to grow (right image).

$(\vec{V}^W)$ and angular $(\vec{\Omega}^W)$ velocities. The noise vector $\vec{n}$ in MonoSLAM equals:

$$\vec{n} = \begin{pmatrix} \vec{V}^W \\ \vec{\Omega}^R \end{pmatrix} = \begin{pmatrix} \vec{a}^W \Delta t \\ \vec{\alpha}^R \Delta t \end{pmatrix}, \tag{1}$$

and the unknown accelerations are assumed to be zero mean Gaussian processes.

However, the high-speed camera can readout images several times faster (Δt is equal e.g. 5ms) than a standard camera (Δt is 33ms). Due to this high frame rate we assume that the velocities ($\vec{v}^W$ and $\vec{\omega}^R$) can *change* in average, and the accelerations ($\vec{a}^W$ and $\vec{\alpha}^R$) are expected to be *constant* in average.

The *constant linear and angular velocity* motion model can be extended to a second order model. The second order model includes linear $(\vec{a}^W)$ and angular $(\vec{\alpha}^R)$ accelerations in the camera state vector $\vec{x_v}$. We expect that this model can handle more erratic and jitter motion as depicted schematically in Fig. 4.

The camera vector state using the *constant linear and angular velocity* model has 13 states and comprises:

$$\vec{x}_v = \begin{pmatrix} \vec{r}^W & \vec{q}^{WR} & \vec{v}^W & \vec{\omega}^R \end{pmatrix}, \tag{2}$$

where the metric 3D position vector $\vec{r}^W$ and linear velocity $\vec{v}^W$ are estimated relative to a fixed world frame W. Quaternion $\vec{q}^{WR}$ represents the orientation between the robot frame R carried by the camera and the world frame. The angular velocity $\vec{\omega}^R$ is estimated in the robot frame R.

The extended camera state vector $\vec{x_v}$ in the *constant linear and angular acceleration* model has 6 new states and includes:

$$\vec{x}_v = \begin{pmatrix} \vec{r}^W & \vec{q}^{WR} & \vec{v}^W & \vec{\omega}^R & \vec{a}^W & \vec{\alpha}^R \end{pmatrix}, \tag{3}$$

where the zero order ($\vec{r}^W$ and $\vec{q}^{WR}$) and the first order ($\vec{v}^W$ and $\vec{\omega}^R$) states are inherited from the first order motion model. The new states comprises the linear $(\vec{a}^W)$ and angular $(\vec{\alpha}^R)$ accelerations.

We assume that in each time step when using the *constant linear and angular acceleration* model, an unknown linear jitter $\vec{j}^W$ and unknown angular jitter $\vec{\eta}^R$ cause an impulse of

Fig. 7. The setup for the well-controlled experiments consists of the gigabit ethernet camera mounted onto a 7DOF robotic arm.

linear $(\vec{A}^W)$ and angular $(\vec{\Psi}^R)$ accelerations. These accelerations are again zero mean Gaussian processes, and the noise vector is equal:

$$\vec{n} = \begin{pmatrix} \vec{A}^W \\ \vec{\Psi}^R \end{pmatrix} = \begin{pmatrix} \vec{j}^W \Delta t \\ \vec{\eta}^R \Delta t \end{pmatrix}. \tag{4}$$

The camera state update is computed as follows:

$$\begin{aligned} \vec{f}_v &= \begin{pmatrix} \vec{r}^W_{new} & \vec{q}^{WR}_{new} & \vec{v}^W_{new} & \vec{\omega}^R_{new} & \vec{a}^W_{new} & \vec{\alpha}^R_{new} \end{pmatrix}^T \\ &= \begin{pmatrix} \vec{r}^W + \vec{v}^W \Delta t + \frac{1}{2}(\vec{a}^W + \vec{A}^W)\Delta t^2 \\ \vec{q}^{WR} \otimes \vec{q}(\vec{\omega}^R \Delta t + \frac{1}{2}(\vec{\alpha}^R + \vec{\Psi}^R)\Delta t^2) \\ \vec{v}^W + (\vec{a}^W + \vec{A}^W)\Delta t \\ \vec{\omega}^R + (\vec{\alpha}^R + \vec{\Psi}^R)\Delta t \\ \vec{a}^W + \vec{A}^W \\ \vec{\alpha}^R + \vec{\Psi}^R \end{pmatrix}, \end{aligned} \tag{5}$$

where the quaternion product $\otimes$ is defined in [19]. The notation:

$$\vec{q}(\vec{\omega}^R \Delta t + \frac{1}{2}(\vec{\alpha}^R + \vec{\Psi}^R)\Delta t^2)$$

represents the quaternion trivially defined by the addition of two angle-axis rotation vectors:

$$\vec{\omega}^R \Delta t \text{ and } \frac{1}{2}(\vec{\alpha}^R + \vec{\Psi}^R)\Delta t^2.$$

The EKF process noise covariance Q_v is equal to:

$$Q_v = \frac{\partial \vec{f}_v}{\partial \vec{n}} P_n \frac{\partial \vec{f}_v}{\partial \vec{n}}^T, \tag{6}$$

where $\frac{\partial \vec{f}_v}{\partial \vec{n}}$ is computed as follows:

$$\frac{\partial \vec{f}_v}{\partial \vec{n}} = \begin{pmatrix} \frac{\partial \vec{r}}{\partial \vec{A}} & \frac{\partial \vec{r}}{\partial \vec{\Psi}} \\ \frac{\partial \vec{q}}{\partial \vec{A}} & \frac{\partial \vec{q}}{\partial \vec{\Psi}} \\ \frac{\partial \vec{v}}{\partial \vec{A}} & \frac{\partial \vec{v}}{\partial \vec{\Psi}} \\ \frac{\partial \vec{\omega}}{\partial \vec{A}} & \frac{\partial \vec{\omega}}{\partial \vec{\Psi}} \\ \frac{\partial \vec{a}}{\partial \vec{A}} & \frac{\partial \vec{a}}{\partial \vec{\Psi}} \\ \frac{\partial \vec{\alpha}}{\partial \vec{A}} & \frac{\partial \vec{\alpha}}{\partial \vec{\Psi}} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} I \Delta t^2 & 0 \\ 0 & \frac{\partial \vec{q}}{\partial \vec{\Psi}} \\ I \Delta t & 0 \\ 0 & I \Delta t \\ I & 0 \\ 0 & I \end{pmatrix}. \tag{7}$$

The implementation of the *constant linear and angular acceleration* motion model requires nontrivial Jacobians. Similarly as in [4], the complex differentiation of these Jacobians is tractable, but is out of the scope of this paper.

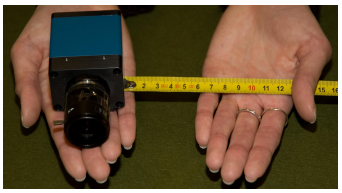

Fig. 8. The gigabit ethernet camera has been thrown and caught for a distance approximately equal 0.1m.

C. Motion Model Parameters

Choosing EKF process noise Q_v is an important part of the filter deployment. The only part of Q_v, which can be parameterized is the covariance matrix P_n of the noise vector $\vec{n}$. This matrix is diagonal, as required when the linear and angular components are uncorrelated.

In MonoSLAM, using the *constant linear and angular velocity* motion model, the covariance noise matrix P_n is equal:

$$P_n = \begin{pmatrix} SD_a^2\ \Delta t^2 & 0 \\ 0 & SD_\alpha^2\ \Delta t^2 \end{pmatrix}, \tag{8}$$

where the standard deviation parameters (SD_a and SD_α) define the smoothness of motion we expect.

If the *constant linear and angular acceleration* motion model is applied, the covariance noise matrix P_n is equal:

$$P_n = \begin{pmatrix} SD_j^2\ \Delta t^2 & 0 \\ 0 & SD_\eta^2\ \Delta t^2 \end{pmatrix}, \tag{9}$$

and standard deviation parameters (SD_j and SD_η) again define the kind of motion we assume.

If the camera with fast readout speed (e.g. Δt is 5ms) is used and the noise covariance matrix P_n contains the same values as when using the standard camera (Δt is 33ms), the motion model expects very smooth motion. To increase the *localization* robustness using the high-speed camera, setting P_n parameters to larger values in both motion models is sufficient. An example is displayed in Fig. 5.

D. Skipping Vision Information

The standard camera (Δt is 33ms) is typically providing a fairly stable vision input, and missing frames are unlikely to occur. However, the asynchronous high-speed camera (e.g. Δt is 5ms) is not so reliable. A skipped or missing frame causes the Δt to increase, and this will effect on enlarged searching regions, as depicted in Fig. 6.

To handle erratic motion using the high-speed camera, the values in the noise covariance matrix P_n (for both motion models) should be adjusted to large values. However, it is suitable to keep a balance between the motion model robustness and the real-time demand. The larger P_n values require to search for feature matches in larger regions, and that is a more time consuming operation.

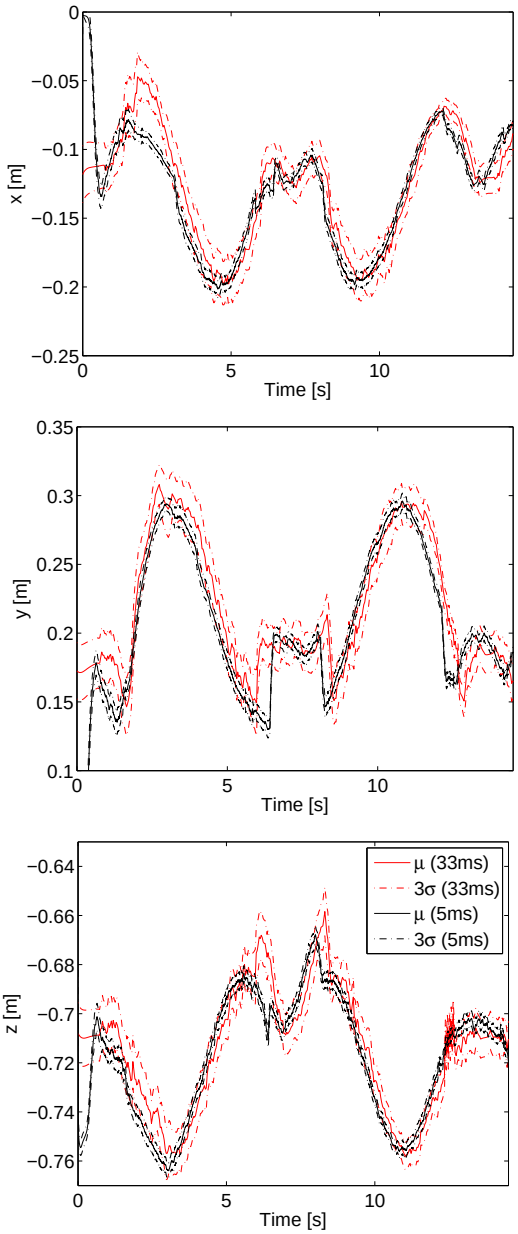

Fig. 10. The *localization* uncertainties using two different vision readout speeds (Δt): 33ms and 5ms. The camera performed the same movements repeatedly as depicted in Fig. 9, but only the camera world frame positions of the first 15s are displayed here.

IV. Experimental Results

To present the improved robustness in MonoSLAM in practice, the performed experiments are explained in three parts. Firstly, the setup and the performed precise robotic arm and high acceleration handheld movements are briefly introduced. Secondly, the accuracy of ground truth is summarized. Thirdly, the *localization* and *mapping* evaluations are compared with the calculated ground truth.

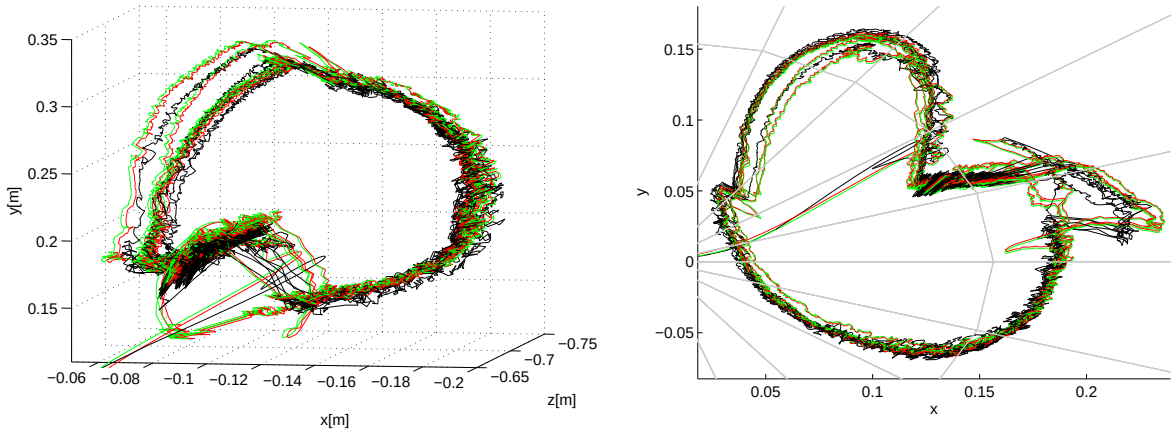

Fig. 9. Three loops of the well-controlled movement, using the high-speed camera (Δt equals 5ms) carried by the robotic arm. Left are the three-dimensional camera world frame positions and right are the camera rotations. The rotations in MonoSLAM are represented as normed quaternions, which can be visualized in a unit sphere. The red lines are the estimated camera poses using the *constant linear and angular velocity* motion model, and the green lines are the computed ground truth. The black lines are the estimated camera poses using the *constant linear and angular acceleration* motion model.

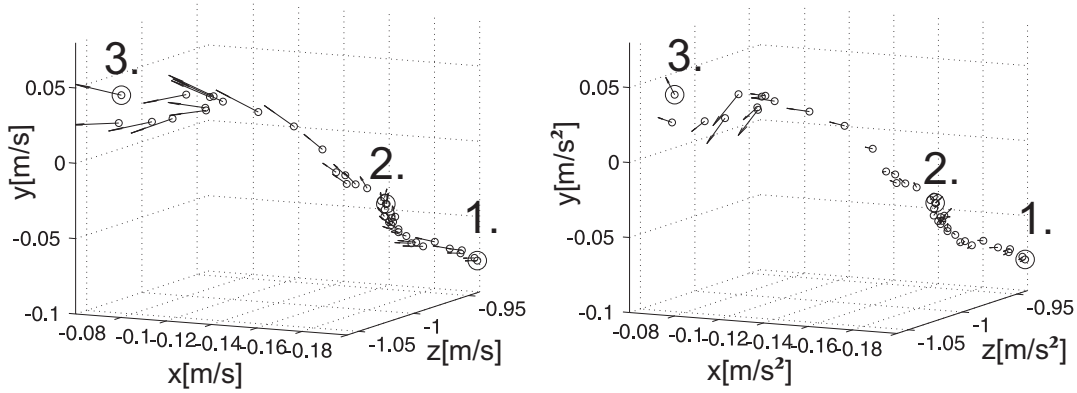

Fig. 11. The distance between the initial (1) and final (3) camera positions is equal 0.175m. The high-speed camera has been thrown (2) and caught (3) over a distance of approximately 0.1m. In MonoSLAM, the *constant linear and angular acceleration* motion model has been used. The velocity (left) and acceleration vectors (right) are scaled.

A. Experimental Setup

The setup for the well-controlled experiments consists of: a commercial Amtec[1] 7DOF robotic arm (see Fig. 7), a robotics software package[2] and the high-speed camera. The arm has been used to perform an accurate, pre-defined and smooth movement, which was repeated with different camera readout speeds.

For the two types of handheld experiments the high-speed camera has been needed. Firstly, the camera has been shortly thrown and caught (see Fig. 8). Secondly, a more erratic movement has been performed to compare the motion models.

In this work, we have been using a very fast GE680C Prosilica[3] gigabit ethernet camera (see Fig. 7 and Fig. 8), which - thanks to the 1/3" CCD sensor - offers a very good image quality.

This gigabit ethernet camera can provide VGA resolution vision output at 200Hz. In MonoSLAM, 320x240px images are more suitable, because VGA resolution would require more computationally intensive vision processing. This camera can do the binning on the chip, and this allows to address smaller images directly, faster, and at *requested* frame rates.

[1] www.amtec-robotics.com
[2] www.amrose.dk
[3] www.prosilica.com

	True			Reconstructed		
	X[m]	Y[m]	Z[m]	X[m]	Y[m]	Z[m]
1	0.105	0.07425	0.0	0.10497	0.07428	-0.00010
2	-0.105	0.07425	0.0	-0.10499	0.07429	-0.00013
3	0.105	-0.07425	0.0	0.10502	-0.07428	0.00008
4	-0.105	-0.07425	0.0	-0.10502	-0.07426	-0.00005

TABLE I
THREE-DIMENSIONAL FEATURES' TRUE VS. RECONSTRUCTED POSITIONS. THE ROOT MEAN SQUARE ERROR (RMSE) OF THE RECONSTRUCTED POSITIONS IS SMALLER THAN 0.1MM.

B. Ground Truth Accuracy

To present the ground truth accuracy, an example of a comparison of four reconstructed features' positions and their known values is given in Tab. I. This shows that the estimation using the procedure described in Section II-B is accurate to

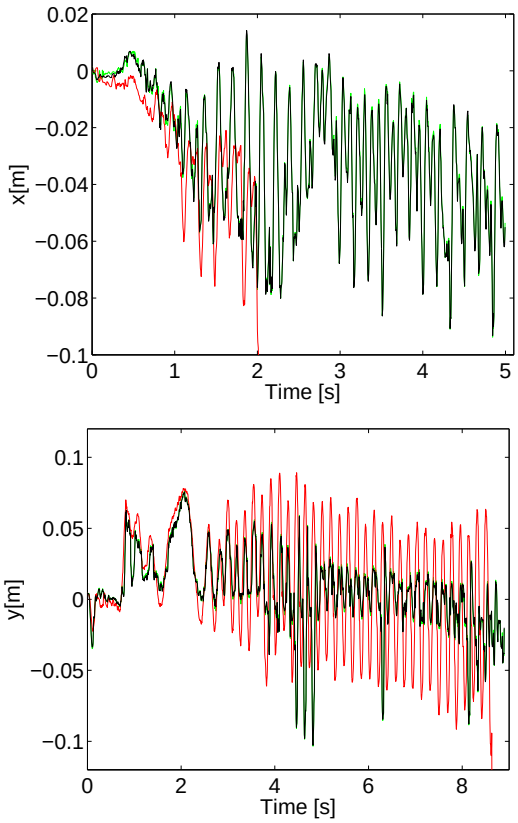

Fig. 12. In the upper row a handheld jitter motion has been performed in world x direction, and the camera world frame x positions are depicted. In the bottom row a jitter movement in world y direction is displayed as camera y positions. The *constant linear and angular velocity* model (red lines) lost the camera pose during both motions (time equals 2s in upper row and short after 8s in bottom row). The *constant linear and angular acceleration* model (black lines) proved to be more robust to these movements. The green lines are the computed ground truth.

	Constant Velocity Motion Model	
Δt[ms]	Translation [m/s^2]	Rotation [rad/s^2]
33	4	6
5	40	60
	Constant Acceleration Motion Model	
Δt[ms]	Translation [m/s^3]	Rotation [rad/s^3]
33	40	60
5	4000	6000

TABLE II

STANDARD DEVIATION PARAMETERS FOR BOTH MOTION MODELS.

within 0.1 percent of the true values, which refers to an estimate of better than 0.1mm.

C. Localization Evaluations

Prior to the evaluation of MonoSLAM *localization* results against the ground truth we explain the used criterions, and present the motion models' parameters.

To evaluate the improved robustness of MonoSLAM using the high-speed camera, these experiments are introduced:

- repeated, well-controlled and precise robotic arm movements have been executed,
- the camera has been thrown and caught and

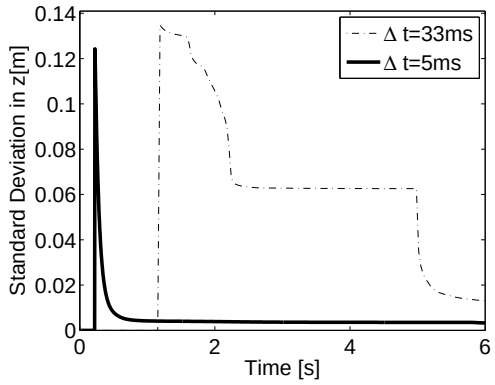

Fig. 13. In MonoSLAM, the uncertainty of a feature location is decreasing as presented in [1]. If the fast readout speed is used, then the feature three-dimensional position uncertainty is converging several times faster.

- handheld jitter motions in different world frame directions have been performed.

The criterions to compare the true and estimated camera poses are:

- the root mean square error (RMSE) to calculate the deviations of *positions* and
- the RMSE to compute the discrepancies of *rotations*.

For the comparison purposes the camera rotation representation is transformed from quaternions to Roll-Pitch-Yaw (RPY) radians.

An overview of both MonoSLAM motion models' parameters is given in Tab. II. These parameters have been adjusted according to the sections III-C and III-D.

1) Robotic Arm Movements: When performing smooth robotic arm movements using 5ms camera readout speed (Δt), both motion models were providing robust results as depicted in Fig. 9. The positions' RMSE was 0.227m and the rotations' RMSE was 0.2922rad, when using the *constant linear and angular velocity* motion model. However, the *constant linear and angular acceleration* motion model provided less accurate results: the positions' RMSE was 1.461m and the rotations' RMSE was 1.9595rad.

An important *localization* advantage when using the fast vision readout speed is that the uncertainties of the camera poses are obviously smaller as depicted in Fig. 10.

2) Thrown Camera: The setup for camera throwing and catching is displayed in Fig. 8, where the distance between two hands was approximately 0.1m. This experiment has been performed repeatedly, and the result is that MonoSLAM can operate in real-time capturing both the free as well as high accelerations in the throwing and catching motions.

As an example, Fig. 11 displays the individual track points and for each point the estimated motion using the *constant linear and angular acceleration* model. The throwing of the camera does not contain significant jitter, so the accuracy differences between the two motion models are similar to the robotic arm movements. Again the initialization used is a black rectangular object (see 1-b) in the first field of view.

3) High Acceleration Handheld Movements: Finally we want to demonstrate the behavior for fast and shaky motion. Fig. 12 depicts the motion following rapid up and down movements of the camera in hand. Maximum frequency of the up/down motion is about 5Hz. The graphs show the MonoSLAM reconstruction using the two motion models. As the Figure indicates, the *constant linear and angular acceleration* model (Δt equals 5ms) successfully tracks such a motion while the *constant linear and angular velocity* model first shows considerably high inaccuracy and then looses track.

D. Mapping Evaluations

MonoSLAM using dense visual information can faster converge features' location uncertainties as depicted in Fig. 13.

The known MonoSLAM problem is that the processing time associated with the EKF update is $O(n^2)$, where n is the number of features in the map. Due to the real-time demand all experiments in this paper have been performed in a small scene with approximately 20 features. MonoSLAM has been able to operate in such an environment in real-time using the high frame rate vision input (Δt equals 5ms).

V. Conclusion

The main contribution of this paper is the introduction of a high-speed camera to monocular SLAM. In addition we proposed to use a combination of this fast readout speed with a second order motion model to allow MonoSLAM to operate under motions with significant jitter. Such a very fast MonoSLAM algorithm running at a frame rate of 200Hz was investigated. We performed repeated and well-controlled experiments with a robot arm and various handheld movements. The comparison of the camera poses is made against accurate ground truth from off-line vision. As expected the constant velocity model is better suited for rather smooth motions. In handheld motions with an up and down motion at a frequency of about 5Hz, the acceleration model is superior and successfully tracks throughout the sequence. Additionally, the high frame rate operation of MonoSLAM makes it possible to throw the camera from one hand to the other.

By showing that high frame rate is able to considerably improve the localization robustness of monocular SLAM, new applications become possible. For example, in a future project it will be evaluated to consumer robotics and wearable computing applications.

A. Future Work

The goal of this work is the high-speed *localization* and *mapping* in a larger indoor environment (e.g. office scene). Recently Clemente et al. presented in [20] an interesting and relevant approach, which can build local maps in near real-time. The most exciting about this approach is that it can keep the computational time of the filter bounded.

Our future work includes improving the mapping using an approach based on local coordinate frames, as this can reduce the current computational complexity $O(n^2)$ to $O(n)$ [20].

Acknowledgments

This research was supported by the European Union project XPERO IST-29427 and EPSRC Grant GR/T24685/01.

Authors are very grateful to Paul Smith and other members of Oxford's Active Vision Laboratory for software collaboration.

References

[1] H. F. Durrant-White and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, June 2006.

[2] J. Weingarten and R. Siegwart, "EKF-based 3D SLAM for structured environment reconstruction," in *IEEE International Conference on Intelligent Robots and Systems*, Edmonton, Canada, 2005, pp. 3834–3839.

[3] G. Zunino and H. Christensen, "Simultaneous localization and mapping in domestic environments," in *Multisensor Fusion and Integration for Intelligent Systems*, Baden-Baden, Germany, 2001, pp. 67–72.

[4] A. J. Davison, I. Reid, N. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[5] B. Williams, P. Smith, and I. Reid, "Automatic relocalisation for a single-camera simultaneous localisation and mapping system," in *IEEE International Conference on Robotics and Automation*, vol. 1, April 2007, pp. 2784–2790.

[6] B. Williams, G. Klein, and I. Reid, "Real-time SLAM relocalisation," in *IEEE International Conference on Computer Vision*, 2007.

[7] M. Pupilli and A. Calway, "Real-time visual SLAM with resilience to erratic motion," in *IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2006, pp. 1244–1249.

[8] D. Chekhlov, M. Pupilli, W. Mayol, and A. Calway, "Robust real-time visual SLAM using scale prediction and exemplar based feature description," in *IEEE International Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–7.

[9] E. Eade and T. Drummond, "Scalable monocular SLAM," in *IEEE International Conference on Pattern Recognition*, vol. 1, 2006, pp. 469–476.

[10] Y. Watanabe, T. Komuro, and M. Ishikawa, "955-fps real-time shape measurement of a moving/deforming object using high-speed vision for numerous-point analysis," in *IEEE International Conference on Robotics and Automation*, April 2007, pp. 3192–3197.

[11] T. Komuro and M. Ishikawa, "A moment-based 3D object tracking algorithm for high-speed vision," in *IEEE International Conference on Robotics and Automation*, April 2007, pp. 58–63.

[12] I. Ishii, Y. Nakabo, and M. Ishikawa, "Target tracking algorithm for 1ms visual feedback system using massively parallel processing," in *IEEE International Conference on Robotics and Automation*, April 1996, pp. 2309–2314.

[13] J. M. M. Montiel, J. Civera, and A. J. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, August 2006.

[14] J. Shi and C. Tomasi, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[15] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, vol. 1, May 2006, pp. 430–443.

[16] J. Neira and J. D. Tardós, "Data association in stochastic mapping using the joint compatibility test," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 890–897, December 2001.

[17] G. Schweighofer and A. Pinz, "Robust pose estimation from a planar target," *IEEE Transactions on Pattern Analysis and Maching Intelligence*, vol. 28, no. 12, pp. 2024–2030, 2006.

[18] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment a modern synthesis," 1999.

[19] A. Ude, "Filtering in a unit quaternion space for model-based object tracking," *Robotics and Autonomous Systems*, vol. 28, no. 2-3, pp. 163–172, 1999.

[20] L. Clemente, A. J. Davison, I. Reid, J. Neira, and J. D. Tardós, "Mapping large loops with a single hand-held camera," in *Robotics Science and Systems*, 2007.