Proc. of the 3rd European Conference on Mobile Robots (ECMR) 2007, Freiburg, pp. 156-161,
Albert-Ludwigs-Universität Freiburg - Universitätsverlag 2007

A Hybrid Kalman Filter Based Algorithm for Real-time Visual Obstacle Detection

Erik Einhorn Christof Schröter Hans-Joachim Böhme Horst-Michael Gross
Neuroinformatics and Cognitive Robotics Lab.
Ilmenau Technical University
Erik.Einhorn@tu-ilmenau.de

***Abstract*— In this paper we present a real-time scene reconstruction algorithm for mobile robots which is applicable for visual collision avoidance and online map building. Our method processes a sequence of images which are taken by a single camera mounted on a mobile robot. In contrast to similar monocular shape-from-motion algorithms, we combine two different approaches: A classic motion stereo approach and an algorithm for scene reconstruction based on extended Kalman filters. We show that the disadvantages of the classic stereo approach are compensated by the Kalman filter and vice versa. Our special method of initializing the Kalman filter leads to a faster convergence compared to other Kalman based approaches that use different methods for initialization. Moreover, we present a feature matching algortihm which is faster and more reliable than the widely-used KLT-Tracker in the domain of scene reconstruction.**

***Index Terms*— monocular vision, shape from motion, extended kalman filter, feature tracking**

I. Introduction and Related Work

Obstacle detection and collision avoidance are very important capabilities of mobile robots. Vision-based approaches provide a large field of view and supply a large amount of information about the structure of the local surroundings. In [12] and [14] appearance-based monocular obstacle detection methods are presented in which each image pixel is classified as belonging either to an obstacle or to the ground based on its color appearance. The underlying assumption in those methods is that the entire ground in each image has a unique and constant color or structure - which is, however, not valid for many environments. In [6] the optical flow is used to detect obstacles. Unfortunately, computing the optical flow for the entire image is very time expensive. To circumvent this problem, the computation in [6] is done for a few pixels only. Other monocular approaches try to recover a three-dimensional model of the scene to gain the exact position of the obstacles. Those approaches use a sequence of images which are captured during the robot's locomotion. Thus different two-dimensional views of a scene are obtained and can be used for the scene reconstruction. This classical problem of computer vision is known as "shape-from-motion".

We apply a sparse feature-based "shape-from-motion" approach that performs a scene reconstruction for distinctive image points (image features). These image features are extracted using the "FAST" high-speed corner detector [11].

This work is partially supported by TAB-Grant #2006-FE0154 to H.M. Gross

As we intend to use the reconstructed scene for obstacle detection and collision avoidance, our camera is mounted in front of the mobile robot and tilted towards the ground. This results in two major problems we have to deal with:

1. The camera is moving along its optical axis: In a sensitivity analysis Matthies and Kanade [9] proved that when using forward motion, shape-from-motion leads to higher uncertainties in the depth estimates. Compared to the ideal lateral camera translation parallel to the image plane - which is used in standard binocular approaches - the estimation must be applied over a long base distance in order to achieve the same accuracy.

2. Many objects are visible during a few frames of the captured image sequence only while the robot is approaching these obstacles. Hence, most image features cannot be tracked over a large number of frames and the scene reconstruction algorithm must be able to provide a reliable estimate by using a few image measurements only.

To overcome the first problem, Matthies et al. [9] suggest scene reconstruction using Kalman filters, since they can integrate the depth and scene information over a long base distance. Consequently, many shape-from-motion solutions that have been researched and published in recent years are based on Kalman filtering [16, 5, 1, 9]. Since these algorithms operate in an incremental way, they have several advantages compared to batch approaches such as bundle adjustment [13]. Furthermore, Kalman filters can be computed in a very efficient way allowing the reconstruction algorithm to operate in real-time. This is essential for online map building and obstacle detection.

Since Kalman filter based methods solve the reconstruction problem in an iterative manner, the speed of convergence depends on the choice of the initial estimate which is used for the initialization of the Kalman filter. If unfavourable initial estimates are used, Kalman filter based approaches tend to suffer from a low speed of convergence, i.e. several iterations must be processed to get a reliable estimation of the obstacle positions. Unfortunately, as stated above, most image features cannot be tracked over many frames and it is not possible to compute enough iterations.

To prevent this problem, we have developed a new hybrid approach for scene reconstruction. In contrast to the shape-from-motion methods mentioned above, our hybrid algorithm combines two completely different approaches: scene reconstruction based on extended Kalman filters (EKF) and a

"classical" correlation-based depth estimation approach.

The depth estimation algorithm is used to compute a reliable initial estimate for the EKF, which then will refine the estimate and recover the three-dimensional model. We show that this novel kind of initialization leads to a significantly better convergence of the filter.

On the other hand, it is well known that a correlation-based algorithm for depth estimation sometimes produces outliers if false matches are found during the search for corresponding image points. In a sole correlation-based approach these outliers would lead to virtual obstacles in the reconstructed model which do not exist in the real scene. In contrast, the EKF of our hybrid approach is able to robustly correct and reduce these outliers.

These considerations show that by combining the mentioned algorithms, the disadvantages of the traditional stereo approach are compensated by the extended Kalman filter and vice versa.

II. Scene-Reconstruction

Since depth estimation and scene reconstruction using Kalman filtering are common techniques in computer vision, they will not be described in detail here. Further information can be found in [3, 16, 5, 1, 15].

Fig. 1 illustrates the complete architecture of our approach.

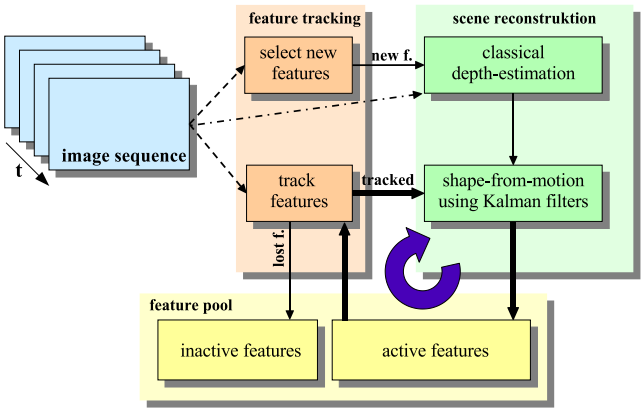

Fig. 1. Architecture of the hybrid approach

Our motion stereo approach is inspired by the work of Bunschoten and Kröse [3], where a multi-baseline depth estimation algorithm for panoramic image data is presented. Based on their work, we have developed a similar correlation-based algorithm for projective cameras. To obtain the image data, we work with a single projective camera mounted on our mobile robot to capture not only a sequence of images $I_0, \ldots, I_n$ - each taken from a different pose (i.e. position and orientation) during the robot's locomotion - but also the corresponding odometry data measured by the robot drive. Hence, for each image of the sequence the approximate position of the camera is known, including uncertainty in odometry measurements from systematic and non-systematic errors.

To correct these errors we use correspondences the feature tracker has found over the frames of the image sequence to estimate the pose of the camera. Since the translation vector of the camera movement can be computed up to a scale only, we are content with estimating the angle of roll and the pitch of the camera, since inaccuracies in the orientation of the camera cause the largest error in the scene reconstruction. Starting with values provided by the robot's odometry, both angles are varied using Gauss-Newton iteration in order to minimize the Sampson error, which is defined by the used image point correspondences and the fundamental matrix.

Let $\mathbf{P}_i$ be the camera projection matrix corresponding to the image I_i which can be computed from the robots odometry using the corrected roll and pitch angle, then the 3D scene point $\mathbf{X}$ is projected to the image point $\tilde{\mathbf{x}}_i = \mathbf{P}_i\tilde{\mathbf{X}}$ of image I_i [1]. Let $\mathbf{x}_0$ be an image point of image I_0, then the corresponding 3D scene point X must be located on the back-projected ray $\tilde{\mathbf{X}}(\lambda) = \mathbf{P}_0^+\tilde{\mathbf{x}}_0 + \lambda\tilde{\mathbf{c}}_0$, where $\mathbf{P_0}^+$ is the pseudoinverse of the camera projection matrix and c_0 is the camera center of the camera in image I_0. If the 3D scene point $\mathbf{X}$ is visible in a different image I_i from a different position, it must be located on the projection of the ray on the image plane of image I_i:

$$\tilde{\mathbf{x}}'_i(\lambda) = \mathbf{P}_i\mathbf{P}_0^+\tilde{\mathbf{x}}'_0 + \lambda\mathbf{P}_i\tilde{\mathbf{c}}_0 \quad (1)$$

This equation is the parametric description of a line, which is well known as "epipolar line". In order to estimate the depth of the image point $\mathbf{x}_0$, we subsequently move along the epipolar line by varying the parameter λ. For each λ the above equation gives the image coordinate $\mathbf{x}_i$ on the epipolar line corresponding to a scene point at depth $\frac{1}{\lambda}$. For each $\mathbf{x}_i$ the similarity with pixel $\mathbf{x}_0$ is evaluated. As measure of correlation we use the sum of absolute differences (SAD) between windows centered at $\mathbf{x}_i$ and $\mathbf{x}_0$. Using a correlation window with 16×16 pixels, the SAD can be computed very efficiently using MMX/SSE assembler instructions of today's CPUs. As in [3] and [10] we accumulate the SAD values obtained for different images $I_1 \ldots I_n$ for the same $\mathbf{x}_0$ and λ (see Fig. 2). Eventually, the most likely depth value for the given pixel $\mathbf{x}_0$ can be determined immediately by means of the parameter λ which yields the minimal SAD. Therefore, explicit triangulation is not necessary [3].

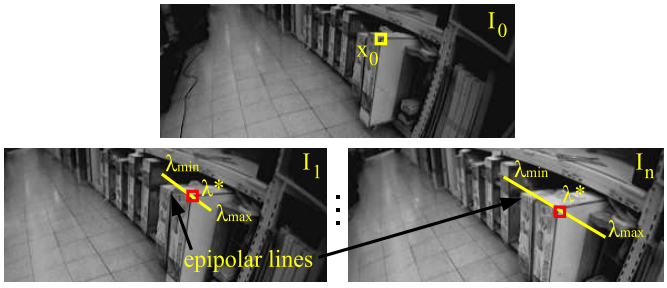

Fig. 2. For the same $\mathbf{x}_0$ in the reference image I_0 the SAD is computed along the epipolar line and accumulated over several images $I_1 \ldots I_n$.

As shown in Fig. 1 the depth estimation is performed for newly selected features only. For these new features no correspondences are known from feature tracking since they

[1] In contrast to [7] we notate homogenous vectors as $\tilde{\mathbf{x}}$ and Euclidean vectors as $\mathbf{x}$.

were not selected in previous frames because the points were not visible or too far away from the camera and did not appear as features. In order to determine the depth of these new features the described multi-baseline depth estimation algorithm uses the current image and previously recorded ones.

The estimated depth is then used to compute the approximate 3D position of the feature in the real scene. This position is used as a reliable initial estimate for the Kalman filtering, which then will refine the estimate and recover the three-dimensional model. In contrast to [5, 1] where one single EKF with a large state vector is used to recover the 3D positions of all features (model points), we use a separate EKF for each feature point. According to [16] this leads to a linear space and time complexity in terms of the number of features while the loss in accuracy is small. Similar to [16] we choose the 3D position of the feature point as state vector $\mathbf{x} \in \mathbb{R}^3$ which is to be estimated. Since $\mathbf{x}$ is the absolute position of the point in relation to the world coordinate frame, it is independent of the robot's movements $\mathbf{u}$ and the state transition function simplifies to:

$$f(\mathbf{X}, \mathbf{u}, \mathbf{w}) = \mathbf{X} + \mathbf{w}\,, \tag{2}$$

where $\mathbf{w}$ denotes a random variable with a normal probability distribution.

The measurement of the 3D scene point $\mathbf{X}$ is performed by projecting it onto the image surface of the camera. To obtain the Euclidean vector of the homogenous image point, we divide by the homogenous coordinate and get the non-linear measurement function:

$$h(\mathbf{X}, \mathbf{v}) = \left(\frac{\mathbf{P}_{[1]}\tilde{\mathbf{X}}}{\mathbf{P}_{[3]}\tilde{\mathbf{X}}}, \frac{\mathbf{P}_{[2]}\tilde{\mathbf{X}}}{\mathbf{P}_{[3]}\tilde{\mathbf{X}}} \right)^T + \mathbf{v} \tag{3}$$

In this equation $\mathbf{P}_{[i]}$ denotes the i-th row vector of the projection matrix $\mathbf{P}$.

As quoted in [15], the Kalman filter algorithm consists of two steps: the "prediction" step which computes an a priori estimate and the "correction" (or measurement update) step where the a priori estimate is improved and an a posteriori estimate is obtained by taking the observed measurement into account. The observed measurement is the position of the real image point in the current image which is provided by a feature tracker that tracks each image point over consecutive frames. With each new frame the tracked features will pass through this Kalman filter cycle and their 3D positions will be estimated more precisely in each iteration.

III. Feature Tracking

In order to track the image features over several frames, we apply a feature matching algorithm. First we select the image features independently in each frame using the FAST corner detector [11]. Similar to the IPAN feature tracker [4] corresponding features are matched in subsequent frames then. While the IPAN tracker solves a pure motion correspondence problem by using three consecutive frames and solely kinematic constraints, we only use two frames. To eliminate the resulting ambiguities, we additionally take the image similarity into account.

Let I_{t-1} and I_t be two consecutive frames of the image sequence. In order to find the correspondences $\mathbf{x}_{t-1}^{(i)} \leftrightarrow \mathbf{x}_t^{(i)}$ between the previously selected image features of both frames, possible hypotheses of matching points are chosen first. Each hypothesis $h = (\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t^{(j)})$ consists of a pair of two potentially matching points $\mathbf{x}_{t-1}^{(i)}$ and $\mathbf{x}_t^{(j)}$ of the frames I_{t-1} and I_t (see Fig. 3).

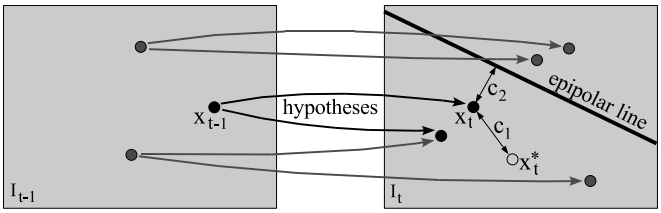

Fig. 3. Possible matching hypotheses for features of the two images I_{t-1} and I_t. For one feature $\mathbf{x}_t$ in image I_t the distance c_1 from the epipolar line and the distance c_2 from the predicted feature position $\mathbf{x}_t^*$ are indicated. Both distances are used to compute the hypothesis cost function.

To reduce the number of hypotheses we use a maximum speed constraint, i.e. we only choose pairs of image points that satisfy

$$\left\| \mathbf{x}_{t-1}^{(i)} - \mathbf{x}_t^{(j)} \right\|_2 \leq r_{max} \tag{4}$$

where r_{max} defines the area around the last feature position where corresponding features are searched for. Hence, it defines a maximum speed at which a feature can cross the frame within the image sequence. Features that are moving faster can not be tracked. Therefore, an appropriate value for r_{max} depends on the amount of the optical flow within the image sequence.

For each hypothesis $(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t^{(j)})$ we define a cost function that will be derived in the following section. As proposed in [8] we combine both, Guided-Tracking and Guided-Matching. For each image point $\mathbf{x}_{t-1}$ of the last frame we use its reconstructed 3D position $\tilde{\mathbf{X}}^*$ in order to predict its location $\mathbf{x}_t^*$ in the current frame I_t:

$$\tilde{\mathbf{x}}_t^{*(i)} = \mathbf{P}_t \tilde{\mathbf{X}}^{*(i)}\,. \tag{5}$$

Since we perform an initial depth estimation as described in the previous section, an estimate of the 3D position is already available for newly selected features. For features that have been tracked over several frames more accurate estimations of the 3D positions were computed by the Kalman filters and their location in the current frame can be predicted more precisely.

If the 3D position of the feature is estimated correctly, feature points $\mathbf{x}_t^{(j)}$ close to the predicted location $\mathbf{x}_t^{*(i)}$ are possible candidate matches for $\mathbf{x}_{t-1}^{(i)}$. As metric we use the squared Euclidean distance:

$$c_1 = \left\| \mathbf{x}_t^{*(i)} - \mathbf{x}_t^{(j)} \right\|_2^2\,. \tag{6}$$

Additionally, corresponding image points must satisfy the epipolar constraint, hence an image point $\mathbf{x}_t^{(j)}$ that corresponds to $x_{t-1}^{(i)}$ is located on or near the epipolar line that is induced by $x_{t-1}^{(i)}$. The distance of the image point $\mathbf{x}_t^{(j)}$ from that epipolar

line can be computed as follows:

$$c_2 = \frac{\left|\tilde{\mathbf{x}}_t^{(j)\top}\mathbf{F}\tilde{\mathbf{x}}_{t-1}^{(i)}\right|}{\sqrt{(\mathbf{F}\tilde{\mathbf{x}}_{t-1}^{(i)})_1^2 + (\mathbf{F}\tilde{\mathbf{x}}_{t-1}^{(i)})_2^2}}, \tag{7}$$

where $\mathbf{F}$ is the corresponding fundamental matrix which again is computed using the robot's odometry. Alternatively, the Sampson distance [7] could be used, which, however, is computationally more complex.

Although c_1 and c_2 are both related to the epipolar geometry, c_2 does not rely on the estimated 3D position of the feature and yields different costs than c_1 if the estimated 3D position of the feature is not determined exactly yet.

As stated above, we also use a similarity constraint to eliminate ambiguous matchings. For each pair of potentially matching points $\mathbf{x}_{t-1}^{(i)}$ and $\mathbf{x}_t^{(j)}$, we compute the similarity of their neighborhood patterns. Again we use the SAD as measure of correlation:

$$c_3 = \mathrm{SAD}_W(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t^{(j)}) \tag{8}$$

Putting things together, we can specify our cost function as weighted sum of the three functions defined above:

$$\mathrm{cost}(\mathbf{x}_{t-1}^{(i)}, \mathbf{x}_t^{(j)}) = w_1 c_1 + w_2 c_2 + w_3 c_3 \tag{9}$$

The weights have been chosen empirically. Using the synthetic image data that is described in section IV, we performed different test series where we varied one weight while leaving the others constant and measured the tracking error and the number of tracked features. We finally chose the weights that yield the smallest tracking error: $w_1 = 1$, $w_2 = 3$ and $w_3 = 20$.

From all hypotheses those with minimal matching costs are chosen by a greedy algorithm. Hypotheses whose costs are larger than a certain threshold are rejected. An appropriate threshold depends on the image data.

Finally, all chosen hypotheses represent the corresponding image points.

IV. Results

In order to make a quantitative analysis and to be able to compare our hybrid approach with others, we have rendered a sequence of a synthetic scene consisting of 250 frames and including the ground truth depth images using the raytracer POV-Ray[2]. The ground truth depth images are used to measure the tracking error and the error of the reconstructed 3D model. We used realistic textures and added some Gaussian image noise. To simulate odometry errors and the sway of the camera, we added Gaussian noise to the camera position and orientation while rendering the images. Fig. 5 shows a top view of the synthetic scene and a rendered image. In the top view the camera trajectory is plotted and its position at certain frames is marked.

[2] `http://www.povray.org/`

A. Feature tracking

In Fig. 6 the KLT tracker and the guided feature matching algorithm proposed in this paper are compared. Here we have used Birchfield's implementation of the KLT feature tracker [2]. In the diagram, the tracking error is plotted for each frame of the synthetic sequence. Due to guided matching, our feature matching algorithm is able to track the features more precisely. The KLT tracker produces larger tracking errors especially during frames 150-171 and 180-200 when the camera is moving around the two corners and during frames 80-145 where near obstacles induce a large optical flow.

In the left diagram of Fig. 4, the mean tracking error of both feature trackers is shown for different feature counts. If more features are tracked per frame, the tracking error increases for both trackers but the error of the guided feature matching approach remains smaller.

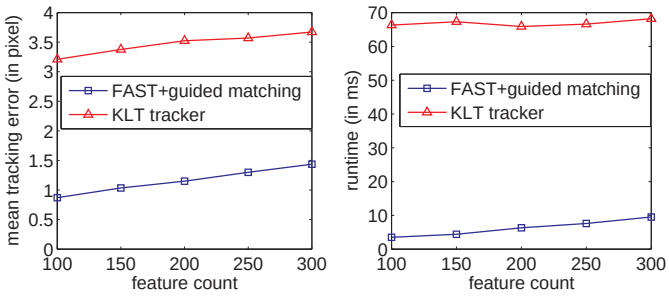

Fig. 4. **Left:** Mean tracking error averaged over all frames of the synthetic sequence for different feature counts. **Right:** Runtime that is needed for feature selection and feature tracking depending on the number of features that are selected in each frame. The time was measured on a Pentium 4 with 3.4 GHz.

Although the tracking error of the KLT tracker can also be reduced if guided tracking is used and the tracker is provided with the predicted feature locations as described in the previous section, the runtime of the KLT tracker remains a problem for realtime applications.

Using the proposed feature matching algorithm, we were able to reduce the runtime that is needed for feature tracking dramatically as shown on the right plot of Fig. 4.

B. Scene reconstruction using synthetic data

In Fig. 7 the mean model error of the reconstruction is plotted against the number of iterations to compare the effect of different initial estimates on the speed of convergence: A common way of initializing the Kalman filters is to use a constant depth of the feature points as it is applied in [16]. This may be valid if the extensions of the object are limited and its approximate distance to the camera is known. However, if the camera is moving through an indoor or outdoor environment, this kind of initialization yields large errors. The mean error of a Kalman filter based algorithm which is initialized using a constant depth is illustrated by the solid graph in Fig. 7. We have chosen a depth of 3 meters, which gives the best results for our synthetic scene.

The graph which is marked with +'s shows the mean error if the filters are initialized randomly using a Gaussian noise with a mean value of 3 meter and a variance of 1.

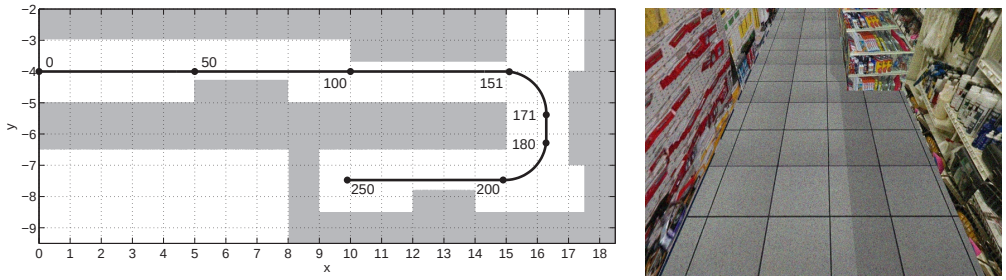

Fig. 5. **Left:** top view of the synthetic scene. The camera position is marked for certain frames. **Right:** rendered image of the scene with additional image noise

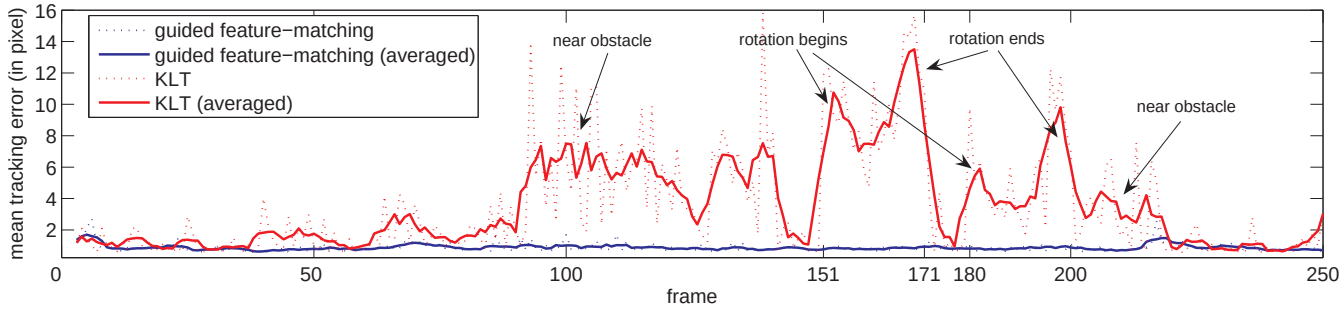

Fig. 6. Mean tracking error for each frame of the synthetic image sequence. Due to the large optical flow while the camera is rotating and approaching near obstacles, the tracking error of the KLT tracker becomes larger while with guided matching it remains small.

Another selection of the initial estimate is shown by the graph which is marked with ○'s. Here, the depth of each detected feature is chosen in a way that the height of the initial feature position is zero, i.e. the features are initialized on the ground plane. This kind of initialization has certain advantages when used for obstacle detection because false positive detections can be reduced. However, it leads to a huge initial model error and therefore a poor speed of convergence.

The thick solid graph in Fig. 7 shows the mean error of our hybrid approach. As expected, it converges significantly faster due to the smaller error of the initial model provided by the depth estimation. Compared to other plain Kalman filter based algorithms our method achieves a much smaller model error especially during the first few iterations while the plain approaches need 5-10 iterations to accomplish the same accuracy that our approach achieves instantly by the initial depth estimation (the dashed line).

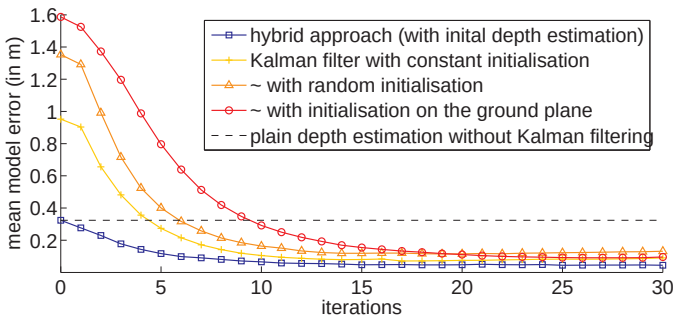

Fig. 7. The hybrid approach presented in this paper (thick solid line) converges faster than a plain Kalman filter which uses simple heuristics for choosing the initial estimates.

C. Scene reconstruction using real image data

To obtain real image and odometry data, the new robot platform SCITOS[3] with an on-board PC based on a 1.6 GHz Intel Centrino processor is employed. For image acquisition a 1/4" CCD firewire camera (FireFly, Point Grey Research, Inc.) with a wide angle lense (f=2.8 mm, FOV: $\approx 65°$) is installed. The camera is mounted at a height of 1.15 m and tilted by $36°$ towards the ground, and the image sequence is captured using a frame rate of 15 frames/s.

Fig. 8 shows a map which was created while the robot was moving through an indoor environment with a velocity of 0.6-0.8 m/s. The estimated positions of the features are visualized using red and orange dots. The estimated z-coordinate is used only to determine if a point belongs to an obstacle or if it lies on the floor, i.e. if the z-coordinate of a point is smaller than a threshold of 0.2 m, it is regarded as belonging to the ground plane and not included in the map. The gray map in the background was built using a laser range finder. The light gray areas of the laser map were visible to the laser range finder only but *not* to the front camera due to its narrower field of view. However, that also means these obstacles were never observed in front of the robot, making them insignificant for collision avoidance. The dark gray areas of the map were visible to the camera and can be used as reference. As to be seen from Fig. 8 the accuracy of the map which was built using our approach is similar to the laser-built reference map. Moreover, our visual method is able to detect some obstacles which are not "visible" to the laser because they are too small and lie beneath the laser range finder. Those obstacles were labeled manually and are highlighted by the red color in Fig. 8.

[3] `http://www.robots-for-research.com/`

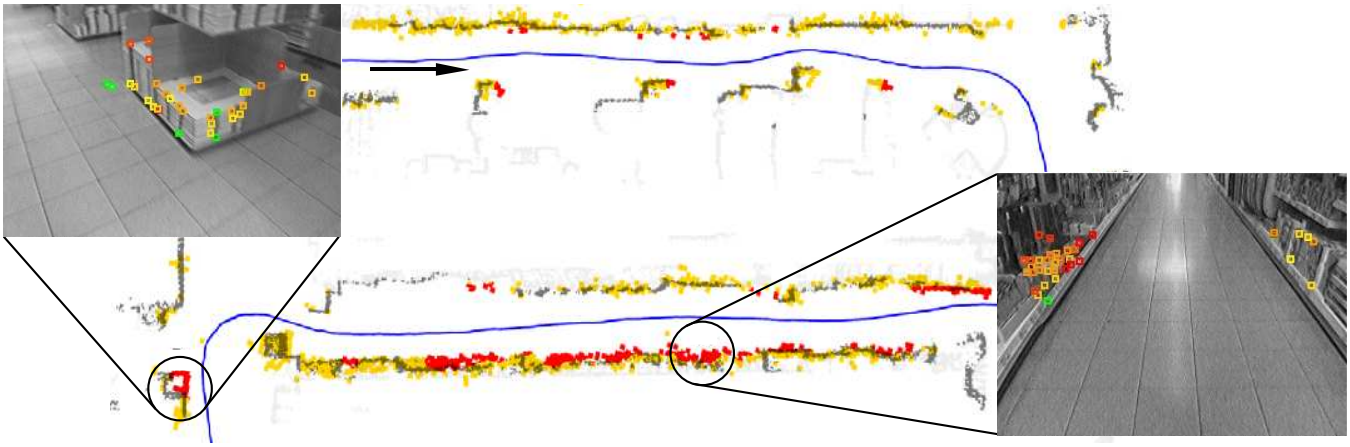

Fig. 8. **Middle:** A map which was built using our algorithm while the robot was moving through an indoor environment. The gray map in the background was built using a laser range finder (the light gray areas of the map were visible to the laser range finder only but *not* to the front camera due to its narrower field of view). The red parts of the map show obstacles that were solely detected by the visual approach but not by the laser range finder. **Left and Right:** Images of two obstacles which were not visible to the laser range finder but that have been detected by our visual approach. The features are marked with squares. The color indicates the height of the feature above the ground plane. (green: feature belongs to the ground plane, yellow: height $\geq$ 0.2 m, red: height $\geq$ 0.5 m)

Additionally, the corresponding camera images are shown for two of these obstacles. It can easily be seen that one part of the left obstacle is not included in the laser map since it is too small and located below the laser plane. This would have led to a collision if solely laser based navigation had been used. Using our hybrid approach for visual obstacle detection instead, this obstacle can be detected very well.

V. Conclusion and Future Work

In this paper we have presented a hybrid shape-from-motion approach which can be used for obstacle detection. Our algorithm combines traditional multi-baseline stereo depth estimation and Kalman filter based scene reconstruction in order to compensate the disadvantages, which both methods show if they are applied separately from each other.

In contrast to plain Kalman filter based algorithms which often use simple heuristics to choose the initial estimates, we initialize our filters using depth estimates computed by a traditional stereo method. By carrying out a quantitative analysis we were able to show the superiority of this novel kind of initialization which results in better convergence and higher precision of the final feature estimates. This is essential for reliable obstacle detection since most obstacles cannot be tracked over many frames, and a moving robot must be able to react on obstacles which appear in the foreground. Moreover, we have shown that our approach can be used for visual obstacle detection and is able to detect many obstacles that cannot be "seen" by a laser range finder. By combining a laser range finder and our visual approach, a very reliable and robust obstacle detection can be achieved.

Using a resolution of 320x240 pixels, our algorithm can process up to 40-50 images per second. This high performance is reached by using the proposed efficient feature matching algorithm that uses the estimated 3D position of each feature to perform guided tracking.

As in many other monocular approaches we assume that the robot moves in a static scene. During the initial depth estimation dynamic objects are recognized and discarded since no correspondence can be found along the epipolar line. Estimating moving objects remains an open problem for monocular scene reconstruction.

References

[1] A. J. Azarbayejani, T. Galyean, B. Horowitz, and A. Pentland. Recursive estimation for CAD model recovery. In *Proc. of the 2nd CAD-Based Vision Workshop*, pp. 90-97, Champion, PA, 1994.

[2] S. Birchfield. KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker, `http://www.ces.clemson.edu/~stb/klt/`

[3] R. Bunschoten and B. Kröse. Robust scene reconstruction from an omnidirectional vision system. *IEEE Trans. on Robotics and Automation*, 19(2), pp. 351–357, 2003.

[4] D. Chetverikov and J. Veresty. Tracking feature points: A new algorithm. In *Proc. of 14th International Conference on Pattern Recognition*, pp. 1436–1438, Brisbane, Australia, 1998.

[5] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Structure from motion causally integrated over time. In *IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 24, No. 4*, pp. 523–535, 2002.

[6] M. Grünewald and J. Sitte. A resource-efficient approach to obstacle avoidance via optical flow. In *Proc. of the 5th International Heinz Nixdorf Symposiom (AMIRE)*, pp. 205–214, Paderborn, 2001.

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2006.

[8] A. Konouchine, V. Gaganov, and V. Vezhnevets. Combined guided tracking and matching with adaptive track initialization. *GraphiCon*, pp. 311–326, 2005.

[9] L. Matthies, T. Kanade, and R. Szeliski. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3(3), pp. 209–238, 1989.

[10] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(4), pp. 353–363, 1993.

[11] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pp. 430–443, 2006.

[12] T. Taylor, S. Geva, and Boles W.W. Monocular vision as a range sensor. *CIMCA 2004 Proc.*, pp. 566–575, 2004.

[13] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision Algorithms: Theory and Practice*, pp. 298–375, Corfu Greece, 2000.

[14] I. Ulrich and I. Nourbakhsh. Appearance-based obstacle detection with monocular color vision. *In Proc. of AAAI*, pp. 866–871, 2000.

[15] G. Welch and G. Bishop. An introduction to the Kalman Filter. Technical report, Chapel Hill, NC, USA, 1995.

[16] Y. Yu, K. Wong, and M. Chang. A Fast Recursive 3D Model Reconstruction Algorithm for Multimedia Applications. *Proc. of ICPR*, pp. 241–244, 2004.