Monocular Scene Reconstruction for Reliable Obstacle Detection and Robot Navigation

Erik Einhorn Christof Schröter Horst-Michael Gross
Neuroinformatics and Cognitive Robotics Lab
Ilmenau University of Technology
Germany

***Abstract*— In this paper, we present a feature based approach for monocular scene reconstruction based on extended Kalman filters (EKF). Our method processes a sequence of images taken by a single camera mounted frontal on a mobile robot. Using different techniques, we are able to produce a precise reconstruction that is free from outliers and therefore can be used for reliable obstacle detection. In real-world field-tests we show that the presented approach is able to detect obstacles that are not seen by other sensors, such as laser-range-finders. Furthermore, we show that visual obstacle detection combined with a laser-range-finder can increase the detection rate of obstacles considerably allowing the autonomous use of mobile robots in complex public environments.**

***Index Terms*— shape-from-motion, visual obstacle detection, monocular vision, EKF**

I. Introduction

For nearly then years we have been involved in the development of an interactive mobile shopping assistant for everyday use in public environments, such as shopping centers or home improvement stores. Such a shopping companion autonomously contacts potential customers, intuitively interacts with them, and adequately offers its services, including autonomously guiding customers to the locations of desired goods [1]. As part of long-term field trials 9 shopping robots have been in daily use in three different home improvement stores in Germany since March 2008.

For obstacle detection the robots are equipped with an array of 24 sonar sensors at the bottom and a laser-range-finder SICK S300 mounted in front direction at a height of 0,35 meters as shown in Figure 1. Using these sensors, most of the obstacles can be reliably detected. However, during the field trials it became apparent that many obstacles are very difficult to recognize. The main extent of a shopping cart for example is mainly located above the plane that is covered by the laser-range-finder. Also small obstacles like flat pallets are difficult to detect since they lie below the laser-range-finder and can hardly be seen by the sonar sensors due to their diffuse characteristics and low precision. Therefore, it turned out to be necessary to use additional methods for robust and reliable obstacle detection. Vision-based approaches are suitable for this purpose since they provide a large field of view and supply a large amount of information about the structure of the local surroundings.

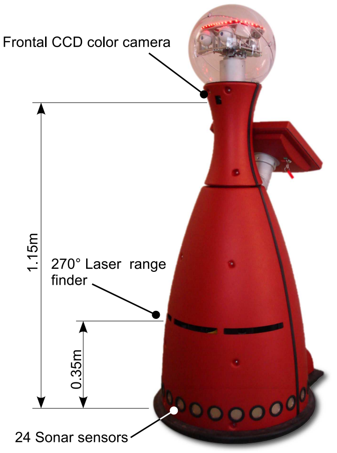


Fig. 1. The robot platform SCITOS A5, a joint development of MetraLabs GmbH[2]and the NICR lab, that is used for our experiments and field trials is equipped with sonar sensors, a laser-range-finder and a frontal CCD camera that is tilted towards the ground.

Recently, time-of-flight cameras have been used successfully for obstacle detection [2]. Similar to laser-range-finders, these cameras emit short light pulses and measure the time taken until the reflected light reaches the camera again. Due to their high costs these cameras may be suitable for robot prototypes but are no option for a series product that we are planning to develop. Another alternative is to use stereo vision for obstacle detection as described in [3] and many others. However, a stereo camera is less compact than a single camera. Furthermore, a monocular approach that uses one camera only is more interesting from a scientific point of view.

In [4] a monocular approach for depth estimation and obstacle detection is presented. Information about scene depth is drawn from the scaling factor of image regions, which is determined using region tracking. While this approach may work well in outdoor scenes where the objects near the focus of expansion are separated from the background by large depth discontinuities, it will fail in cluttered indoor environments like home improvement stores. In [5] we propose an early version of a feature-based approach for monocular scene reconstruction. This shape-from-motion approach uses Extended Kalman Filters (EKF) to reconstruct the 3D position of the image features in real-time in order to identify potential obstacles in the reconstructed scene. Davison et al. [6, 7] use a similar approach and have done a lot of research in this area. They propose a full covariance SLAM algorithm for recovering the 3D trajectory of a monocular camera. Both, the camera position and the 3D positions of tracked image features or landmarks are estimated by a single EKF. Another visual SLAM approach was developed by Eade and

The research leading to these results has received funding from the State Thuringia (TAB-Grant #2006-FE-0154) and the AiF/BMWI (ALRob-Project Grant #KF0555101DF).

[2] MetraLabs GmbH, Germany, www.metralabs.com

Drummond [8]. Their graph-based algorithm partitions the landmark observations into nodes of a graph to minimize statistical inconsistency in the filter estimates [9].

However, Eade and Drummond's "Visual SLAM" as well as Davison's "MonoSLAM" are both mainly focusing on the estimation of the camera motion, while a precise reconstruction of the scenery is less important. As we want to use the reconstructed scene for obstacle detection and local map building, our priorities are vice versa. We are primarily interested in a precise and dense reconstruction of the scene and do not focus on the correct camera movement, since the distance of the objects relative to the camera and the robot respectively is sufficient for obstacle avoidance. Actually, we are using the robot's odometry to obtain information on the camera movement. In contrast to Eade and Davison who generally move their camera sidewards in their examples, our camera is mounted in front of the mobile robot and, therefore, moves along its optical axis (see Figure 1). Compared to lateral motion, this forward motion leads to higher uncertainties in the depth estimates due to a smaller parallax. This fact was also proven by Matthies and Kanade [10] in a sensitivity analysis.

The main contribution of this paper is a monocular feature-based approach for scene reconstruction that combines a number of different techniques that are known from research areas like Visual SLAM or stereo vision to achieve a robust algorithm for reliable obstacle detection that must fulfill the following requirements:

1) A dense reconstruction to reduce the risk of missing or ignoring an obstacle
2) The positions of obstacles that appear in the field of view should be correctly estimated as early as possible to allow a fast reaction in motion control
3) Outliers must be suppressed to avoid false positive detections that result in inadequate path planning or unnecessary avoidance movements

The presented algorithm is based on our previous work [5] and was improved by several extensions. In the next sections, we describe our approach in detail and show how it can be used for visual obstacle detection. Finally, we present experimental results and conclude with an outlook for future work.

II. Monocular Scene Reconstruction

As stated before, we use a single calibrated camera that is mounted in front of the robot (see Figure 1). During the robot's locomotion, the camera is capturing a sequence of images that are rectified immediately according to the intrinsic camera parameters. Thus, different two-dimensional views of a scene are obtained and can be used for the scene reconstruction.

A. Feature Selection

In these images distinctive image points (image features) are detected. For performance reasons we use the "FAST" high-speed corner detector [11], since SIFT or SURF features still require too much computation time. The selected features are then tracked in subsequent frames while recovering their 3D positions.

B. State Representation

Davison et al. [6, 7] use a single EKF for full covariance SLAM, i.e. for recovering the camera pose as well as the 3D positions of the tracked image features simultaneously. In this algorithm the inversion of the innovation covariance matrix while computing the EKF update will dominate the overall runtime resulting in a complexity of $O(n^3)$ for large feature feature counts n. Currently, such an approach only is able to handle up to 100 features in real-time.

In [12] the computation of pose and structure is split into two steps. In the first step, a single EKF is applied to recover the camera position using a fixed number of reconstructed features. During the second step n EKFs are used to recover the 3D positions of n features, where one EKF is used per feature. Both steps are repeated in an interleaved way. Obviously, this is a coarse approximation of the full covariance SLAM since correlations between the different features are not taken into account. However, this approximation results in a heavy reduction of the computational complexity to $O(m^3) + O(n)$. Here $O(m^3)$ - the complexity of the pose estimation during the first step - is constant, since the number m of features that are used in the first step remains constant, too. Thus, the overall complexity for large feature counts n is $O(n)$.

Since we require a dense reconstruction of the scene for obstacle detection, we have to cope with a large number of features which cannot be handled by a full covariance SLAM approach in real-time. Therefore, we also use one EKF per feature to recover the structure of the scene similar to [12]. Each feature i is associated with a state vector $\mathbf{y}_i$ that represents the 3D position of the feature and a corresponding covariance matrix $\mathbf{\Sigma}_i$.

Different parametrizations for the 3D positions of the features have been proposed in literature. The most compact representation is the XYZ-representation where the position of each feature is parameterized by its Euclidean coordinates in 3-space. Davison et al. [7] have shown that this representation has several disadvantages since the position uncertainties for distant features are not well represented by a Gaussian distribution. Instead, they propose an inverse depth representation, where the 3D position of each feature i can be described by the following vector:

$$\mathbf{y_i} = (\mathbf{c}_i\,,\,\theta_i\,,\,\varphi_i\,,\,\lambda_i)^\top, \tag{1}$$

where $\mathbf{c}_i \in \mathbb{R}^3$ is the optical center of the camera from which the feature i was first observed, and θ_i, ϕ_i is the azimuth and elevation of the unit ray that points from $\mathbf{c}_i$ to the 3D point of the feature. This ray is given by its direction vector:

$$\mathbf{m}\left(\theta_i, \phi_i\right) = \left(\cos\theta_i \cos\phi_i\,,\,\cos\theta_i \sin\phi_i\,,\,-\sin\theta_i\right)^\top \tag{2}$$

The last element λ_i of the state vector in equation(1) denotes the inverse of the features depth $d_i = \lambda_i^{-1}$ along the ray.

Parsley et. al [13] have shown that this inverse depth λ_i might become negative during the EKF update and proposed an alternative negative logarithmic parameterization where the inverse depth λ_i is replaced by $l_i = -\log(d)$. In our experiments, this parametrization resulted in a inferior convergence of the EKFs. Therefore, we are going to use the inverse parametrization here.

C. Feature Tracking

While the robot is moving, the image features are tracked in subsequent frames. In [5] we used a feature matching approach that finds correspondences between homologue features in subsequent frames based on a bipartite graph matching. While that approach is suitable for SIFT or SURF features, it has some shortcomings with less complex feature descriptors like image patches.

Here, we use a guided active search for tracking the features through the image sequence. As descriptor we utilize a 16×16 pixel image patch around each feature.

First, the image position $\mathbf{x}_i^-$ of each feature is predicted by projecting the current estimate of its 3D position $\mathbf{y}_i$ back onto the image plane using $\tilde{\mathbf{x}}_i^- = h\left(\mathbf{y}_i, \mathbf{P}\right)$ with the measurement function[3]:

$$h\left(\mathbf{y}_i, \mathbf{P}\right) = \mathbf{P}\left(\lambda \tilde{\mathbf{c}}_{\mathbf{i}} + \begin{pmatrix} \mathbf{m}\left(\theta_i, \phi_i\right) \\ 0 \end{pmatrix}\right). \qquad (3)$$

Here $\mathbf{P} = \mathbf{KR}\left[\mathbf{I} \mid -\mathbf{c}\right]$ is the projection matrix containing the current orientation $\mathbf{R}$, the current position $\mathbf{c}$, and intrinsic calibration matrix $\mathbf{K}$ of the camera the current image was captured with (see [14] for details). The current camera pose is obtained from the robot's odometry data. We will get back to this with some more information in a later subsection.

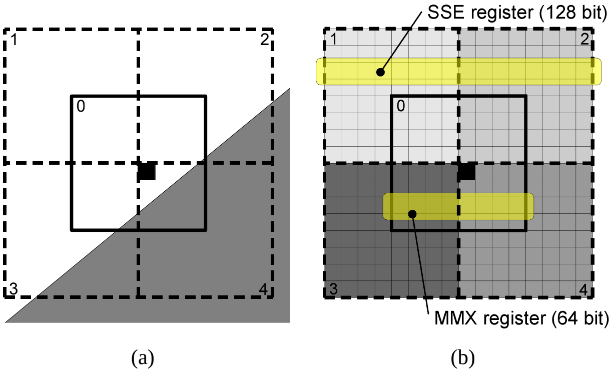
For each feature i, the corresponding image point is searched in the current image around the predicted image position $\mathbf{x}_i^-$ by computing the sum of absolute differences (SAD) with the image patch that is stored as descriptor of the feature. The image point that yields the lowest SAD is chosen. To achieve sub-pixel precision, we fit a 2D parabola into the computed SAD error surface around the chosen image point and use the coordinate of the apex as position of the corresponding image point. The search is restricted to an elliptical region that is defined by projecting the error covariance $\mathbf{\Sigma}_i$ of the feature's 3D position estimate to the image plane. The covariance matrix of this elliptical region is computed within the EKF and is known as innovation covariance:

$$\mathbf{S}_i = \mathbf{H}_i \mathbf{\Sigma}_i \mathbf{H}_i^\top + \mathbf{R}_i, \qquad (4)$$

where $\mathbf{H}_i$ denotes the Jacobian of the measurement function in equation (3) and $\mathbf{R}_i$ is the 2×2 measurement covariance matrix that is set to $\mathbf{R}_i = 5\,\mathbf{I}$ in our experiments.

One major problem of patch-based approaches for feature matching are occlusions near object edges, where the patch covers two different objects with large depth discontinuities (see Figure 2a). During the matching, this leads to a decision conflict since the part of the patch that belongs to the background object moves in a different way than the foreground object. As a result, the reconstructed 3D points along object borders are blurred in different depths. For stereo matching, various adaptive window approaches have been proposed to tackle this problem.

Here, we apply a variation of the multiple window approach presented in [15] and [16]. Instead of using a single 16×16 pixel correlation window, the window is split into five sub-windows as shown in Figure 2. The SADs are computed for each sub-window C_i. The final correlation value C is formed by adding the correlation value C_o of the central sub-window and the values of the two best surrounding correlation windows C_b and C_s:



(a)

(b)

Fig. 2. (a) The correlation window is split into 5 sub-windows and allows better tracking along object boundaries. (b) For computing the SAD of each sub-window the data can be stored efficiently in SSE/MMX registers.

$$C = C_0 + C_b + C_s,\ b = \underset{i>0}{\operatorname{argmin}}\, C_i,\ s = \underset{i>0, i \neq b}{\operatorname{argmin}}\, C_i. \qquad (5)$$

This measure of similarity performs better near object boundaries since at least two sub-windows are located on a single object in most cases. Depending on the dominant image structure, the correspondence is either attached to the foreground or the background object, reducing the blur along the reconstructed object borders. Using the SSE2 processor instruction `PSADBW`, the correlation values can be computed efficiently. This instruction simultaneously computes the SAD for 16 consecutive pixels while it sums the SADs for the first 8 pixels separately from the back most pixels within one 128 bit SSE register. Therefore, splitting the window into 5 sub-windows results in very little computational overhead compared to a single correlation window. This performance improvement is a major reason for choosing the SAD as similarity measure. Besides the correlation value, we compute an occlusion score by adding the correlation values of the two worst matching surrounding sub-windows:

$$C_{occ} = C_b + C_s,\ b = \underset{i>0}{\operatorname{argmax}}\, C_i,\ s = \underset{i>0, i \neq b}{\operatorname{argmax}}\, C_i. \qquad (6)$$

Both the correlation value C and the occlusion score C_{occ} are normalized by the number of pixels in the used sub-windows.

D. Descriptor Update

Davison et al. [7] also use the image patch around the feature as descriptor. While they capture this descriptor only once when the feature is first observed, we used a contrary philosophy in [5], where we update the descriptor every time the feature is tracked in a new image. Both variants have pros and cons. If the descriptor is never adapted, the feature cannot be tracked over long distances since the appearance changes too much due to affine and perspective deformations, especially when using a forward moving camera or robot. If, on the other hand, the descriptor is updated with every frame, tracking errors might be accumulated over several frames,

[3] For better differentiation we notate homogeneous vectors as $\tilde{\mathbf{x}}$ and Euclidean vectors as $\mathbf{x}$, where $\tilde{\mathbf{x}} = \left(\mathbf{x}, 1\right)^\top \cdot s$, $s \in \mathbb{R}$

and the descriptor might move along the edges of object boundaries and does not represent a single fixed feature. This usually occurs near occlusions and leads to incorrect estimates.

Therefore, we use the aforementioned occlusion score C_{occ} to determine whether updating the descriptor is reasonable or not. If the normalized occlusion score C_{occ} exceeds a certain threshold the descriptor remains unachanged, otherwise it is replaced by the corresponding patch in the current image. Using this technique, most features can be tracked over long distances while the projective deformations are compensated by permanent descriptor updates. Feature descriptors near occlusions are not updated to allow stable tracking along object boundaries.

E. Odometry Correction

As stated before, we use the robot's odometry to retrieve the position of the camera for each image. However, due to latencies in transmitting the image data from the camera to the memory the time when the images were captured cannot be determined precisely. Depending upon the current CPU usage and load on the main bus, the delay may vary between 30 and 50 ms. Therefore, the odometry data cannot be assigned exactly to an image. These inaccuracies constitute a negative impact, especially if the angular velocity of the robot is changing rapidly. Additional errors are caused by the joggle of the camera when driving over a bumpy floor.

To correct these inaccuracies, we tried to estimate the camera's position using different methods. We used another EKF additional to the 3D positions of the features in an interleaved way similar to [12], and we applied a Gauss-Newton method to estimate the orientation of the camera by minimizing the back-projection error. Both, the EKF and the Gauss-Newton method were able to recover the camera pose or orientation respectively but did not achieve a higher precision than SCITOS's odometry, which already has a good accuracy. As an alternative we tried to use a particle filter (PF) for estimating the camera pose. First the particles are updated using a motion model. Then we choose a constant number of $m = 15$ features that were tracked in the current image. Thereby, features whose 3D positions are estimated with sufficient precision are chosen in a way to cover the image uniformly. The importance weight of each particle k is then computed by adding the squared Mahalanobis distances between the projected 3D positions $\mathbf{x}_i^{-(k)} = h(\mathbf{y}_i, \mathbf{P}_k)$ of the selected features and their tracked image position $\mathbf{x}_i$ with respect to the innovation covariance $\mathbf{S}_i$ from equation (4) :

$$w_k = -\log\left(\sum_{i=0}^{m}\left(\mathbf{x}_i^{-(k)} - \mathbf{x}_i\right)^\top \mathbf{S}_i^{-1}\left(\mathbf{x}_i^{-(k)} - \mathbf{x}_i\right)\right), \quad (7)$$

where $\mathbf{P}_k$ is the projection matrix computed from the camera pose that is estimated by the k-th particle.

This PF achieves better results than the Gauss-Newton method and the EKF for pose estimation. We assume that this is a result of the shape of the error function that is minimized by both methods. Although the error function is smooth in large scale, it is bumpy near the minimum due to slightly erroneous image measurements making it difficult to find the proper minimum for an iterative method. However, this needs to be further investigated.

F. Measurement Update

After the features are tracked and the camera pose is refined, the 3D positions of the features will be updated using the usual EKF update equations leading to a more precise reconstruction of the scenery.

G. Feature State Initialization

Lost features that left the camera's field of view or that cannot be tracked in the previous step are replaced be new features. Different methods for initializing the state of new features have been proposed in related literature. Some researchers initialize the features at a constant depth while others use a delayed initialization, where the position of a new features is estimated using a PF e.g. before it is inserted into the EKF cycle. However, since we want to use the approach for obstacle detection, we have to obtain a reliable estimate as early as possible in the estimation process. In [5] we have shown how to use a multi-baseline stereo approach for initializing new features. The approach uses the images that were captured *before* the feature was first detected and searches along the epipolar line for corresponding image regions by computing the SAD. By accumulating the SAD error over multiple images, a reliable initial inverse depth estimate is obtained. Additionally, we treat the SAD error along the epipolar line as probability distribution and fit a Gaussian distribution near the minimum in order to obtain a variance of the initial estimate that is used for initializing the error covariance matrix $\mathbf{\Sigma}_i$.

III. Obstacle Detection

For obstacle detection, we perform the described monocular scene reconstruction for 200-300 salient features of the scene simultaneously. Afterwards, the reconstructed features have to undergo some post-processing where outliers and unreliable estimates are removed. From all reconstructed features, we only use those that meet the following criteria:

- the estimated height must be above 0.1m; obstacles below this threshold cannot be detected safely
- the variance of the estimated inverse depth taken from the error covariance matrix $\mathbf{\Sigma}_i$ must lie below a threshold of 0.005
- the distance to the camera must have been smaller than 3m when the feature was observed for the last time

The last criterion mainly removes virtual features that arise where the boundaries of foreground and background objects intersect in the image. These features do not correspond to a single 3D point in the scene and cannot be estimated properly.

The features that pass the above filters may still contain a few outliers. Therefore, we examine the neighborhood of each feature. Features that contain less than 4 neighbors within a surrounding sphere with a radius of 0.3m are regarded as outliers and will be rejected. The remaining features are inserted into an occupancy map by projecting them on the xy-plane. This occupancy map is merged with a laser map by

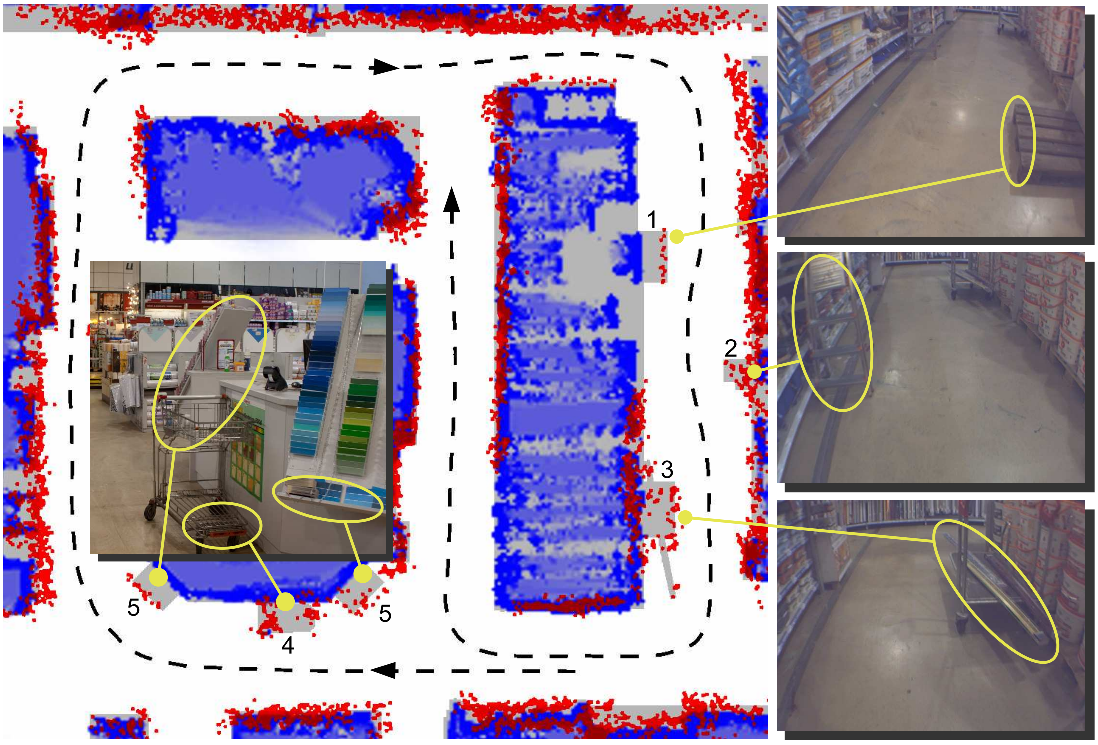


Figure 3. Map created by combining visual information (red dots) and laser-range-finder (blue). The robot's trajectory and moving direction is denoted by the dashed line. The ground truth is highlighted in gray. The visual map consists of about 8200 reconstructed points. Obstacles detected using vision only are labeled using numbers. The images on the right show the obstacles as seen by the front camera. The image on the left was taken using a handheld 8 megapixel camera.

choosing the highest occupancy for each cell in both maps. Finally, the merged map is used for both local path planning and obstacle avoidance.

IV. Results

Figure 3 shows such a map where laser and visual information is merged. For image acquisition a 1/4" CCD fire-wire camera is installed on the robot that is mounted at a height of 1.15m and tilted by 35° towards the ground (see Figure 1). The occupancy map that is created using the laser-range-finder is colored in blue where the different shades of blue correspond to the probability that a cell is occupied. The position of the features that were reconstructed using visual information and the approach presented in this paper are colored in red. In the map, a total number of about 8,200 visual features is shown. While creating the map, a total number of 15,400 points was reconstructed, where 6,000 features where filtered due to a bad variance, 1,000 features were classified as belonging to the ground and 100 were detected as outliers.

For better evaluation and for visualization purposes a ground truth map was created and is highlighted in gray in the background of Figure 3. For building the ground truth, we took images of the scene using a hand held Canon EOS 350D 8.0 megapixel camera and used a bundle adjustment tool[4] for creating a precise reconstruction of the scene which finally was edited and labeled manually.

The map covers an area of 14m×12m within a home improvement store where our tests were conducted. This test area contains typical obstacles that we identified as problematic during the field test since they cannot be detected by the laser-range-finder due to their reflection properties, their form or too low height. Some of these obstacles are numbered from 1 to 5 in Figure 3. In detail these obstacles are:

1) an empty Euro-pallet with a height of 11cm
2) a ladder
3) a low shopping cart with goods that jut out at both ends
4) a high shopping cart
5) shelves that extend into the scene.

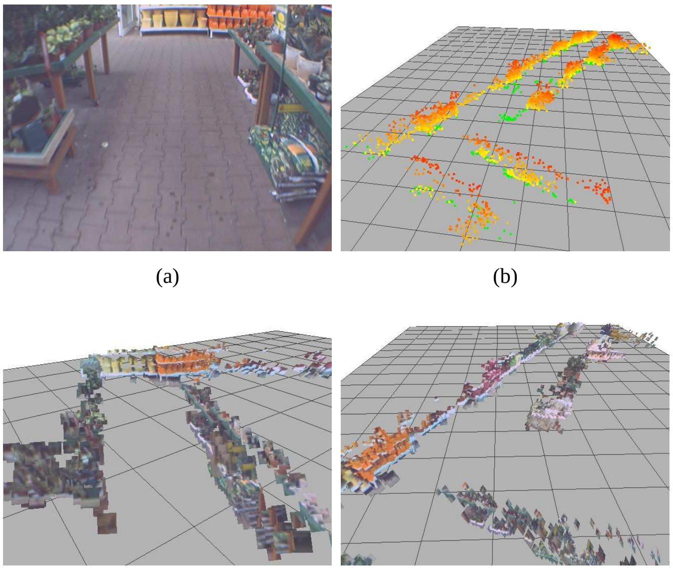
All of these obstacles cannot be seen by the laser-range-finder and, therefore, might result in collisions. However, using our visual approach these obstacle can be detected robustly. In Figure 4 we try to quantify this result. For each obstacle, we have manually labeled those parts of the outline that are relevant for navigation and obstacle avoidance during the above test run using the ground truth map. The statistics in Figure 4 show the percentage of the relevant obstacle boundaries that were detected by our visual approach, the laser-range-finder and a combination of vision and laser. These results show that major parts of the above mentioned obstacles can be detected. Furthermore, it can be seen that the detection rate for all relevant objects in the scene can be increased significantly by 20% compared to obstacle detection using a laser-range-finder only.

Additional tests were carried out in the garden center of the home improvement store where a bumpy stone floor leads to vibrations. Figure 5a shows a front camera image of this area. However, neither the increased shaking of the camera due to the rough ground outdoors resulted in a degradation of the reconstruction nor did the repetitive texture of the floor lead to outliers or false positive obstacle detections. Features detected on the floor were estimated correctly and classified as free and passable (Figure 5b). Figures 5c and 5d show two synthetic views of the reconstructed scene, where the point features were rendered using their image patches. Using this

[4] Bundler: `http://phototour.cs.washington.edu/bundler/`

obstacle	visual	laser	visual+laser
1	63%	-	63%
2	71%	-	71%
3	71%	-	71%
4	68%	10%	68%
5	82%	-	82%
others	85%	78%	96%
total	**83%**	**72%**	**93%**

Fig. 4. Percentage of obstacle boundaries that can be detected using the presented visual approach, a laser-ranger-finder and a combination of both for the 5 labeled obstacles and the rest of the scene shown in Figure 3.



(a)

(b)

(c)

(d)

Fig. 5. (a) front camera image, (b) the reconstructed scene where the height of the reconstructed features is coded by the color (green: < 0.10m, yellow-red: 0.10m-1.15m), (c-d) synthetic views of the obstacles that were generated by rendering the patches of the reconstructed points.

technique, arbitrary views of the scene could be generated and used for appearance-based SLAM for example.

Our approach has been tested on an Intel Core 2 Duo, 2 GHz CPU. In spite of utilizing one core only we are able to process up to 30 frames per second while reconstructing 200-300 features simultaneously. Depending on the robot's driving speed, we only need to process 10-15 frames per second leaving enough CPU resources for other applications like map building, navigational tasks, user tracking and human-machine interaction.

V. Conclusion and Future Work

In this paper, we have presented an algorithm for monocular scene reconstruction and shape from motion. We have described several improvements to [5] that make the reconstruction more reliable and help to reduce outliers. These techniques allow the approach to be used for robust real-time obstacle detection. In realistic field tests, we have shown that some obstacles that are not visible to distance measuring active sensors, like laser-range-finders, can be safely detected by our vision based approach. Furthermore, we were able to show that visual obstacle detection combined with a laser-range-finder can increase the detection rate of obstacles considerably. During the next months, we will carry out long-term tests to evaluate whether and how much the number of collisions or near-collisions can be decreased during the daily usage of the robots.

Currently, we are developing an active vision approach that selects features in areas where the obstacle situation is unclear and where more detailed scene reconstruction is necessary, instead of selecting the features uniformly over the whole image as it is done so far.

Additionally, we are going to research a method to estimate the position of moving objects. However, since the position of moving objects can be reconstructed up to a scaling factor only, we will focus on obstacles that reach to the ground. At the moment, features along moving objects are rejected during feature tracking and filtered after the reconstruction due to their high variance in the position estimate.

References

[1] H.-M. Gross, H.-J. Böhme, Ch. Schröter, St. Müller, A. König, Ch. Martin, M. Merten, and A. Bley. ShopBot: Progress in Developing an Interactive Mobile Shopping Assistant for Everyday Use. In *Proc. of IEEE Int. Conf. on SMC*, pages 3471–3478, 2008.

[2] T. Schamm, S. Vacek, J. Schröder, J.M. Zöllner, and R. Dillmann. Obstacle detection with a Photonic Mixing Device-camera in autonomous vehicles. *International Journal of Intelligent Systems Technologies and Applications*, 5:315–324, Nov. 2008.

[3] P. Foggia, J.M. Jolion, A. Limongiello, and M. Vento. Stereo Vision for Obstacle Detection: A Grap-Based Approach. *LNCS Graph-Based Representations in Pattern Recognition*, 4538:37–48, 2007.

[4] A. Wedel, U. Franke, J. Klappstein, T. Brox, and D. Cremers. Real-time Depth Estimation and Obstacle Detection from Monocular Video. *DAGM Symposium on Pattern recognition*, 4174:475–484, 2006.

[5] E. Einhorn, Ch. Schröter, H.-J. Böhme, and H.-M. Gross. A Hybrid Kalman Filter Based Algorithm for Real-time Visual Obstacle Detection. In *Proc. of the 3rd ECMR*, pages 156–161, Freiburg, Germany, 2007.

[6] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. on PAMI*, 29(6):1052–1067, 2007.

[7] J. Civera, A.J. Davison, and J. Montiel. Inverse Depth Parametrization for Monocular SLAM. *IEEE Trans. on Robotics*, pages 932–945, 2008.

[8] E. Eade and T. Drummond. Monocular SLAM as a Graph of Coalesced Observations. In *IEEE Int. Conf. on Computer Vision*, pages 1–8, 2007.

[9] E. Eade and T. Drummond. Unified Loop Closing and Recovery for Real Time Monocular SLAM. In *Proc. of the BMVC*, 2008.

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

[11] E. Rosten and T. T. Drummond. Machine learning for high-speed corner detection. In *Proc. of the ECCV*, volume 1, pages 430–443, 2006.

[12] Y. Yu, K. Wong, and M. Chang. A Fast Recursive 3D Model Reconstruction Algorithm for Multimedia Applications. In *Proc. of the Int. Conf. on Pattern Recognition, ICPR*, volume 2, pages 241–244, 2004.

[13] M.P. Parsley and S.J. Julier. Avoiding Negative Depth in Inverse Depth Bearing-Only SLAM. In *Int. Conf. on Intelligent Robots and Systems, IROS*, 2008.

[14] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0-521-54051-8, second edition, 2006.

[15] H. Hirschmüller, P. Innocent, and J. Garibaldi. Real-Time Correlation-Based Stereo Vision with Reduced Border Errors. *International Journal of Computer Vision*, 47:229–246, 2002.

[16] W. van der Mark and D.M. Gavrila. Real-time dense stereo for intelligent vehicles. In *IEEE Trans. on Intelligent Transportation Systems*, 2006.