Erik Einhorn, Markus Filzhuth, Christof Schröter,
Horst-Michael Gross:

Monocular Detection and Estimation of Moving Obstacles for Robot Navigation

URN: urn:nbn:de:gbv:ilm1-2011200640

Zuerst erschienen in:
Proceedings of the 5th European Conference on Mobile Robots, ECMR 2011, September 7-9, 2011, Örebro, Sweden. - Örebro : Centre for Applied Autonomous Sensor Systems (AASS), 2011. - S. 121-126.
[http://aass.oru.se/Agora/ECMR2011/proceedings/papers/ECMR2011_0059.pdf ; 21.02.2012]

Monocular Detection and Estimation of Moving Obstacles for Robot Navigation

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

***Abstract*— The detection of motion and moving objects or persons with stationary monocular cameras has been extensively studied. However, those techniques fail if the camera is moving itself. In this paper, we present a method for detecting and estimating the position of moving objects using a monocular camera that is mounted in front of a mobile robot platform. The position estimates are used for obstacle avoidance and robot navigation. We apply image warping to compensate the ego-motion of the camera. This allows us to use standard techniques for motion detection. The final position and velocity estimates are obtained using Extended Kalman Filters. Combined with a monocular scene reconstruction our approach allows the robust detection and avoidance of both static and moving obstacles by using a single monocular camera as the only sensor.**

***Index Terms*— visual obstacle detection, motion detection, monocular scene reconstruction**

I. Introduction

With steadily increasing processing power and newly evolving hardware, approaches for visual navigation gain more and more importance in mobile robotics. For obstacle detection vision based sensors can obtain a larger amount of information about the structure of the local surroundings compared to traditional range-measuring sensors like laser range finders or sonar sensors that operate in a single plane only.

There is a large variety of vision sensors that are suitable for obstacle detection. Time-of-flight cameras are able to measure the distance between camera and obstacles directly by emitting short light pulses. Due to their still high costs, these cameras may be suitable for robot prototypes but are no option for our purposes. Microsoft's Kinect depth camera uses structured light patterns to obtain the depth of objects in the scene. Due to the low costs and the precise depth measurements, this technique will surely have a huge impact in mobile robotics and robot navigation. However, in several experiments we came across a few disadvantages of that camera. Even for smaller robots, its field of view is too narrow to cover the whole area in front of the robot. Moreover, the camera can be used indoors only. In outdoor environments, the emitted infrared light pattern is outshone by the sunlight and cannot be detected by the camera.

Beside these depth cameras and stereo cameras, monocular approaches are an adequate alternative for obstacle detection. The majority of such approaches use feature-based techniques that reconstruct the depth or the entire 3D position of each feature. In our previous works, we have developed such an approach for visual obstacle detection that utilizes images captured by a single monocular camera mounted in front of a mobile robot [7]. Our Structure-from-Motion approach employs 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. We have shown that this method can significantly improve the reliability of obstacle detection when combined with laser range finders [6].

Similar to other related approaches [2, 5] our previous approach assumes that the scene is static. Moving objects could not be estimated and were ignored. However, since our mobile robots operate at home and in public environments [8] that are populated by walking people and other dynamic objects, a proper handling of those obstacles is required. In this paper, we present an extension of our previous approach that is now able to detect moving objects and allows the estimation of their position and velocity. The presented method is the first approach for monocular obstacle detection that is able to detect both static and dynamic obstacles.

II. Related Work

For detecting moving objects or persons with stationary cameras a large number of approaches exists that use difference images, motion history images [3] or background subtraction.

The detection of moving objects using images of a monocular camera mounted in front of a mobile robot is a more difficult problem since the ego-motion of the robot induces an inherent optical flow in the images that must be distinguished from the flow of the moving objects. In [11] a feature-based method is presented that uses image correspondences over two and three frames. Different constraints are applied to differentiate between features located on static objects and those located on moving objects that violate the constraints. In a comprehensive analysis Klappstein et al. [11] show, that objects moving in parallel to the camera with a lower velocity can hardly be detected. Moreover, detecting objects moving anti-parallelly towards the camera is only possible if an additional heuristic is used [10].

A problem that is related to the problem of detecting moving objects in a monocular image sequence is motion segmentation, i.e. the segmentation of the image into regions of pixels moving coherently across the image sequence. In [13] a feature-based approach for real-time motion segmentation is presented that uses an expectation-maximization algorithm to group neighboring features with similar trajectories. However, the problem of detecting objects moving parallel to the camera remains. Those objects would most likely be assigned to the segments of background motion.

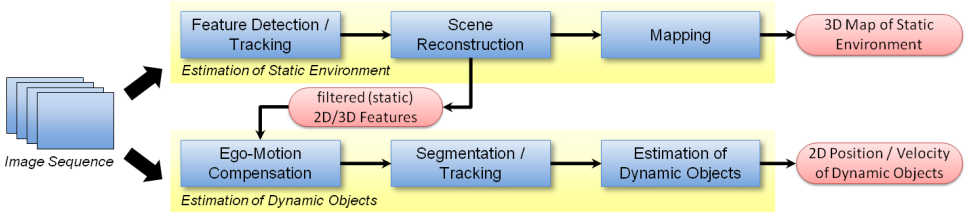


Fig. 1. The architecture of our approach. The upper part shows the scene reconstruction for static parts of the environment. In the captured image sequence, features are tracked, and their 3D positions in the scene are estimated using EKFs. The resulting features with their 3D positions are used for creating a volumetric 3D occupancy map of the static scene. The tracked features and their estimated 3D position are also used for the detection and estimation of dynamic objects in the scene. As shown in the lower part, the features are used for ego-motion compensation in the input image sequence. Afterwards, the dynamic objects are detected, segmented, and tracked to provide information that is used to recover their positions and velocities.

Beside the detection of dynamic objects, the estimation of their 3D positions and trajectories is even more difficult. In addition to an overall scale ambiguity, further difficulties arise due to the ego-motion of the camera making it impossible to estimate the object positions and trajectories if no additional assumption about the object movements are made. There are approaches for Nonrigid Structure-from-Motion [15, 12, 4, 1, 16] that tackle these difficulties by using shape priors [4] or shape bases [16] to constrain the problem. Other authors [12, 1] apply an orthogonal approach and try to solve the problem in the trajectory space instead of the shape space. In those feature-based methods the position and movement of each point is represented by a linear combination of basis trajectories, e.g. the discrete cosine transform basis. The reconstruction is then performed by estimating the coefficients of such a combination of basis trajectories. The approach presented in [12] is most related to our problem. However, the authors analyze the reconstructibility of the point trajectories and come to the conclusion that the reconstruction of the points and their trajectories is poor if the trajectories are correlated to the camera's movement and if the camera is moving slowly and smoothly. Since the movement of our camera mounted in front of the robot performs this kind of movement this method is not applicable in our case.

III. Overview

We use a single calibrated camera that is mounted in front of the robot. During the robot's locomotion, the camera is capturing a sequence of images $(\ldots, I_{t-1}, I_t, I_{t+1}, \ldots)$ that are rectified immediately according to the intrinsic camera parameters in order to correct the lens distortions. Using the robot's odometry data, the corresponding camera position, expressed by its projection matrix $\mathbf{P}_t = \mathbf{K}\mathbf{R}_t\,[\mathbf{I}\,|\,-\mathbf{c}_t]$, can be computed for each image I_t, containing the orientation $\mathbf{R}_t$, the position $\mathbf{c}_t$, and the intrinsic calibration matrix $\mathbf{K}$ of the camera (see [9] for details). Both the camera's position and its orientation are expressed with respect to a global reference coordinate frame.

The complete architecture of the software system that processes this input data is shown in Figure 1. The monocular Structure-from-Motion approach [7] for reconstructing the 3D shape of the static scene is shown in the upper part of that figure. In each captured and preprocessed image (frame), distinctive image features are selected using the Shi-Tomasi corner detector [14]. These features are tracked over the acquired image sequence while their 3D positions are estimated using EKFs. Similar to the camera's pose, these 3D positions of the features are computed with respect to the same global reference coordinate frame [7].

For static obstacle detection, we perform this monocular scene reconstruction for 200-300 salient features of the scene simultaneously. Before the reconstructed features are used to build a map of the environment, they have to undergo some post-processing where unreliable estimates with large covariance and uncertainty are removed.

The lower part of Fig. 1 shows the data flow for the detection of dynamic obstacles, which is the main scope of this paper. The tracked features processed by the above static scene reconstruction are used to perform an ego-motion compensation: the input images are warped in order to eliminate the effect of the robot's movement in the images. After this step, the images can be treated as if they were captured by a static camera. Hence, standard operations such as the computation of difference images can be applied to detect and segment the moving obstacles within the images. Such objects are then tracked and their positions in the images are used to estimate their positions and velocities in the scene. These processing steps are described in more detail in the following two sections.

IV. Ego-Motion Compensation

One major problem for detecting moving objects is the inherent optical flow that is induced by the movement of the robot or camera. We try to eliminate this effect in the images by virtually correcting the viewpoint of the images. We will see, that the ego-motion compensation finally allows us to apply methods known from motion detection with static cameras. The ego-motion compensation is done by image warping. We will show that this image warping can be easily computed using the image features that are tracked by the approach for static scene reconstruction. However, the ego-motion compensation is successful only if features on static scene objects are used. For simplicity we call these features *static features*, while features on moving objects are called *dynamic features* in the following. The usage of dynamic features for the ego-motion compensation leads to inferior results and would disallow the detection of moving objects in later processing steps of our approach.

A. Feature Filtering

In the next paragraphs, we will describe several constraints and criteria that allow us to classify the features that are

extracted and tracked during the static scene reconstruction into static features and dynamic ones.

As one constraint one could use the *epipolar constraint*. For a static feature at position x_t in frame I_t its corresponding image point $x_{t'}$ in frame $I_{t'}$ will be located along the epipolar line that can be described by $l = \mathbf{F}\tilde{\mathbf{x}}_t$, where $\tilde{\mathbf{x}}_t$ denotes the homogeneous coordinate of x_t and $\mathbf{F}$ is the fundamental matrix that describes the geometry between the two involved camera poses (see [9] for details). A dynamic feature in comparison will have a larger distance from that epipolar line.

Though, instead of the epipolar constraint we use a different constraint for *long-term stability* that implicitly contains the epipolar constraint but uses more than two frames to classify the feature. As stated in section III, the 3D positions of the tracked features are estimated iteratively using EKFs. Beside the 3D positions, the EKFs additionally compute the covariance of the position estimates. During the scene reconstruction the algorithm assumes that the scene is static. Hence, dynamic features are estimated with large covariances since they violate the assumptions of the estimation algorithm. This is where the epipolar constraint is implicitly taken into account. The dynamic features can then be filtered out by ignoring all features whose covariance is above a certain threshold. Moreover, the reconstruction algorithm uses the feature estimates to guide the feature tracking process by predicting the position of the corresponding image points in the next frame. Dynamic features that move quickly and perpendicular to their epipolar line, will be too far away from the predicted image location, and the approach for static scene reconstruction will lose track of them. Hence, those dynamic features are filtered out at a very early stage already during the static scene reconstruction.

Unfortunately, there are still dynamic features that pass through the stage of static scene reconstruction and that have 3D position estimates with a low variance. This happens for features located on objects that move in parallel to the camera. Those features primarily do not violate the assumption of a static scene. Instead the estimation algorithm is able to find static and stable 3D position estimates that correspond to the observed 2D feature locations in the image sequence. Of course, those position estimates are not correct for dynamic features. The error depends on the movement of the object.

Dynamic features on objects moving faster than the camera and hence away from it are reconstructed behind the camera with a negative depth. For features on objects that are moving in the same direction as the camera but with a lower speed the estimated depth is too large. Since our camera is tilted towards the ground, those features are reconstructed below the ground plane. Features that move with the same speed as the camera are reconstructed at infinity and hence also below the ground plane. We apply simple plausibility tests and classify those features with a negative depth and a negative height as dynamic features in order to filter them out. It should be noted, that the detection of dynamic features is done only to exclude them from the ego-motion compensation. They are not used to detect moving obstacles at this point.

The above criteria are able to reveal a large number of features on moving objects. However, features on objects that are moving towards the camera cannot be detected as dynamic features. They are reconstructed as static obstacle closer to the camera compared to their real distance. In fact, they are reconstructed near the position where a potential collision with the moving object would occur. Hence for obstacle detection and avoidance, adding those features into the static obstacle map results in an acceptable behavior of the robot. Nevertheless, our processing step for detecting moving objects in the warped images is also able to detect most of these moving objects. This will be described in section V.

B. Image Warping

The static features that were classified by the above criteria can now be used for the actual ego-motion compensation which is done by image warping. Image warping allows us to morph any image I_t taken at camera position $\mathbf{P}_t$ to the "perspective" $\mathbf{P}_{t'}$ of any another frame $I_{t'}$. The warp $\mathcal{W}$ depends on the two positions of the camera and hence - in our case of a single continuously moving camera - on the two time stamps t and t':

$$I_{t\to t'} = \mathcal{W}_{t\to t'}(I_t) \tag{1}$$

The warped image $I_{t\to t'}$ corresponds to I_t being taken from the same camera position where $I_{t'}$ was captured from. Moreover, if the warp was perfect and the scene was static, the warped image $I_{t\to t'}$ would be equal or similar to the image $I_{t'}$:

$$I_{t'} \approx I_{t\to t'} \tag{2}$$

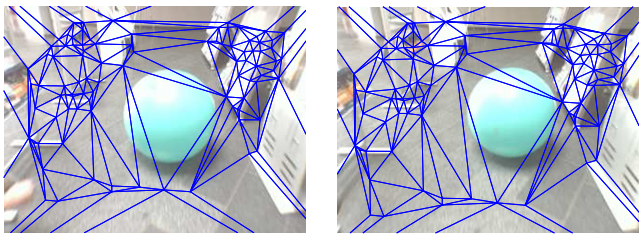
The warp is approximated as piecewise affine transform of a triangle mesh. As vertexes of the mesh we use the static features $F = \left\{f^{(1)}, \ldots, f^{(n)}\right\}$ that are tracked between the two frames I_t and $I_{t'}$. Let $x_t^{(i)}$ denote the 2D image position of the feature $f^{(i)}$ within image I_t, while $x_{t'}^{(i)}$ denotes its image position within image $I_{t'}$. The triangle mesh is generated by computing a Delaunay triangulation of the 2D points $x_t^{(1)}, \ldots, x_t^{(n)}$ within the image I_t as shown in Fig. 2. The mesh is then used to warp each image point in I_t. The image features $x_t^{(i)}$ are warped according to their tracked image position in the image $I_{t'}$:

$$x_{t'}^{(i)} = \mathcal{W}_{t\to t'}(x_t^{(i)}).$$

For all the other pixels in between, their position is bi-linearly interpolated within the triangle that is spanned by the surrounding three image features according to the Delaunay triangulation.

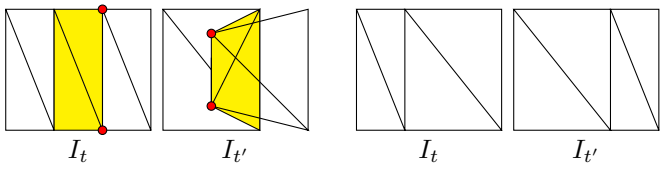
When generating the mesh, one must take into account that the positions of the features and hence the vertices of the mesh move in image $I_{t'}$. This can result in triangles that fold over, i.e. their orientation flips and their back-faces become visible as shown in Fig. 3. Such triangles result in inferior results of the warp. Hence, we identify the vertices that cause the fold-over in order to remove them from the mesh. Afterwards, the mesh is triangulated again.

The described warp can be efficiently computed on contemporary graphics hardware by using the image I_t as a texture for rendering the triangle mesh. The features' positions in image



I_t $I_{t'}$

Fig. 2. Generated triangulation for both images I_t and $I_{t'}$. While the same mesh topology is used in both images, the positions of the vertices are different in image $I_{t'}$ since the features have moved due to the ego-motion of the camera.



I_t $I_{t'}$ I_t $I_{t'}$

Fig. 3. **left:** The two yellow triangles fold over in image $I_{t'}$ since two of the features that are used for the triangulation moved significantly differently. The vertices/features that caused the fold-over are marked with red circles. **right:** The two vertices were removed from the mesh to resolve the fold-over.

I_t are used as texture coordinates and the features' positions in the image $I_{t'}$ are used as model coordinates for each vertex of the mesh.

The upper row in Fig. 4 shows the warping for an image sequence consisting of 5 frames. The warping was computed using the triangle mesh that was built from the features tracked by the approach for static scene reconstruction as described above. Dynamic features where removed by the aforementioned criteria. The two leftmost and the two rightmost images are warped to the perspective of the middle image. While the images were captured, the robot and the blue ball moved forward. The blue ball was slightly faster than the robot. Due to the ego-motion compensation, the static scene remains still in the images although the robot moves forward. Additionally, it is apparent that the blue ball moves forward.

V. Detection and Estimation of Moving Objects

Using the described ego-motion compensation, we can implement the actual detection of moving objects using methods known from motion detection with static cameras. For performance reasons, we use difference images which can be computed efficiently. According to Eq. 2 the difference image of two input images is close to zero, if one image is warped into the perspective of the other image and if the scene is static: $I_{t\to t'} - I_{t'} \approx 0$. Moving objects on the other hand will produce large differences and, therefore, can be detected in the difference images. To reduce the influence of image noise and to increase the signal to noise ratio, we do not only use two images for generating the difference image. Instead, we use 5 consecutive images from the captured image sequence.

Let I_t denote the reference image where the motion should be detected. Then we use two images I_{t-2}, I_{t-1} that were captured before and two images I_{t+1}, I_{t+2} that were captured after the reference image. All images (except the reference image) are warped into the perspective of the reference image I_t. Afterwards, the difference images between the four warped images and the reference image I_t are computed as shown in the second row of Fig. 4. Moreover, the difference images are binarized using a threshold that was chosen experimentally. As shown in Fig. 4 the binarized difference images are then combined in pairs using an AND operation, i.e. in the resulting binary image a pixel is set only if the pixel was set in both binarized difference images. The resulting images are combined using an OR operation. Finally, remaining noise is removed by applying an opening and closing morphological operator. This procedure has yielded the best results in our experiments. It reduces the noise in the difference images and preserves the outline of the moving object even if the size of the object increases or decreases when the object's distance changes. As seen in the above figure, some smaller image regions especially along occlusion boundaries were also classified as moving objects although they belong to the static environment. These regions would also be treated as moving objects in the next processing steps of our approach. However, their position and velocity estimates would finally reveal that those objects are in fact static.

In the final processed binary difference image, bounding boxes are generated around each connected image segment. These bounding boxes are used as features of the extracted moving objects. They provide the approximate position and size of each object within the image and are used for tracking the moving objects within the image sequence. At the moment, we use a single hypothesis tracker, i.e. we assume that there is only one moving object within the field of view of the camera. Currently, the bounding box with the largest area is chosen as hypothesis of the single moving object. The other bounding boxes are ignored.

The position and velocity estimation is done using an EKF. As stated in the first sections it is not possible to reconstruct the position of a moving object from a moving camera without additional constraints. We therefore assume that all moving objects touch the ground and hence the lower border of the extracted bounding box is located on the ground plane in the scene. Consequently, the state of the moving object can be modeled by $\mathbf{y} = (\mathbf{p}, \mathbf{v})^\top$, where $\mathbf{p} \in \mathbb{R}^2$ represents the position of the object on the ground plane in world coordinates and $\mathbf{v} \in \mathbb{R}^2$ denotes the velocity of the object along the ground plane. The resulting state transition function is given as $\mathbf{y_k} = \mathbf{A}\mathbf{y}_{k-1}$, with:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

As measurement for the EKF update, the midpoint of the bounding box's bottom edge in the image is used. During the update the image position of the midpoint is projected onto the ground plane. This projection can be described by a homography $\mathbf{H} = \mathbf{P}_t\mathbf{G}$, where $\mathbf{P}_t$ is again the projection matrix of the camera. $\mathbf{G} = \left(\begin{array}{ccc} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{c} \\ \hline 0 & 0 & 1 \end{array}\right)$ describes the orientation and location of the ground plane that is spanned by the two vectors $\mathbf{e}_1$, $\mathbf{e}_2$ and goes through the point $\mathbf{c}$.

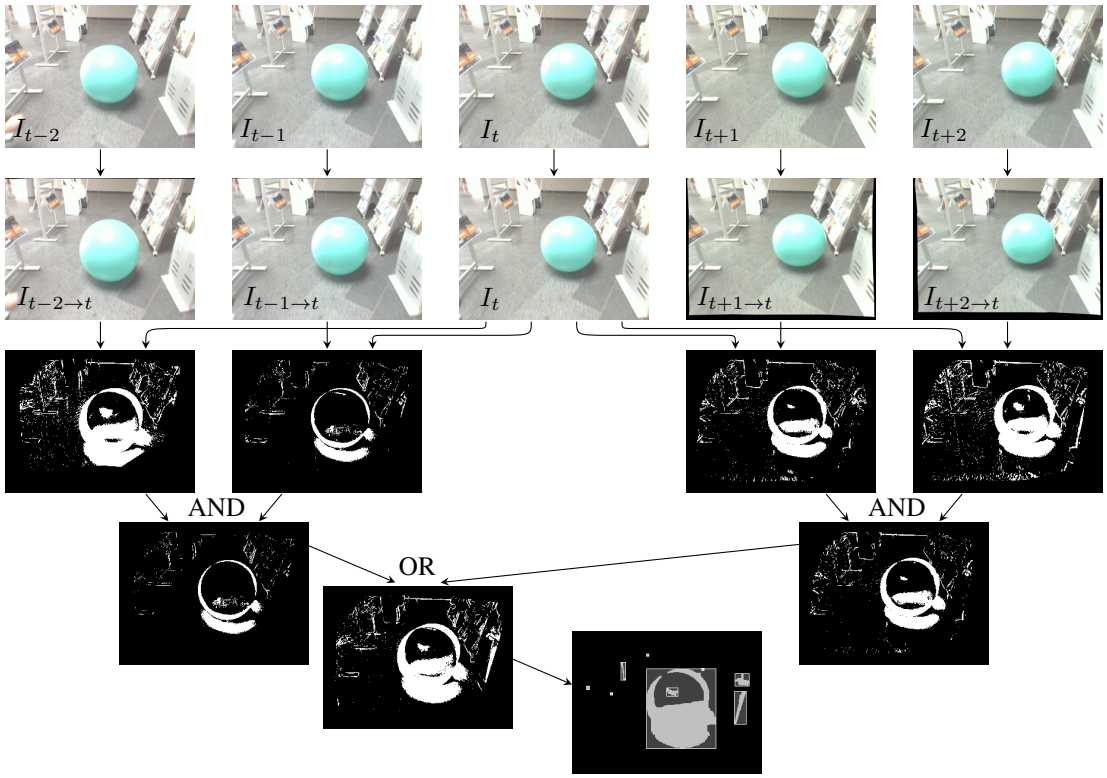


Fig. 4. Processing chain for the detection of moving objects. **first row:** Input sequence consisting of 5 frames. **second row:** Two images before and two images after the reference frame I_t are warped into the perspective of the reference frame. **third row:** Binarized difference images between the warped images and the reference image. **fourth row:** Difference images are combined in pairs using an AND operation. **fifth row:** Resulting images are combined using an OR operation. **sixth row:** Final detection of moving objects after opening and closing operations with extracted bounding boxes.

Using the above stated transition and projection functions, the usual EKF prediction and update steps are iteratively processed for each new image and hence for each extracted bounding box to continuously estimate the position and velocity of the moving object. The final position estimate is used for navigation and obstacle avoidance in addition to the reconstructed features of the static scene.

VI. Results

We have tested our approach with numerous data sets of real data that was recorded while the robot was moving through an indoor environment. As moving obstacles we used an untextured ball as shown in Fig. 4 and a person who walked in front of the robot as shown in Fig. 5 and Fig. 6. We tested different kinds of movements relative to the robot. The detection and estimation of obstacles that moved perpendicular to the robot's trajectory was easily managed by our approach (Fig. 5). More difficult is the detection of objects that move parallel to the robot. Nevertheless, those obstacles were successfully detected and estimated correctly by the proposed approach. An example is shown in Fig. 6 where a person is walking in front of the robot into the same direction with a slightly higher velocity. In the left column of Fig. 6 the frames 10, 20 and 30 of the captured images sequence taken by the robots front camera are shown. Image regions where motion was detected using our proposed method are labeled with red color. Beside some smaller artifacts near occlusion boundaries the walking person was correctly detected. The blue circle denotes the estimated position of the person on the ground plane reconstructed using an EKF as described in the previous section. The green circles indicate the reconstructed positions of the person within the next ten frames that are not shown as separate images. The sizes of the circles indicate the uncertainties of the estimates. In the right column a bird's-eye view of the scene is shown, where the reconstructed static features are additionally visualized as black dots. Together with the estimated position of the moving object, such a 2D map can be used for obstacle avoidance and navigation.

As ground truth, the range measurements of a laser range finder are indicated in all images by a thin black line. In the images of the front camera, the dashed line shows a projection of the range measurements at the height where the laser is mounted, while the solid line shows a projection of the range measurements onto the ground plane in order to allow for a better comparison with the position estimates computed by our approach, which are located on the ground plane, too.

Fig. 5. Detection of a moving person that is walking from left to right while the robot is moving forward. Image regions, where motion was detected, are highlighted in red. Our approach correctly detected the motion of the leg that is moving forward, while the leg that stands on the floor is classified as static.

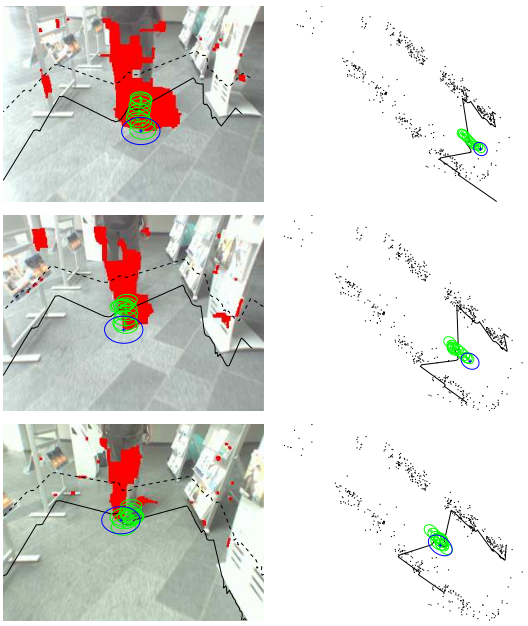


Fig. 6. **left column:** Three frames of the input sequence: image regions where motion was detected are highlighted in red. The position estimate of the moving object is indicated by the blue circle. The green circles show the reconstructed positions for 10 further frames where no image is shown. For comparison, measurements of a laser range finder are shown as black line. **right column:** Bird's-eye view of the scene. The reconstructed features of the static scene are additionally shown as black dots.

Especially in the bird's-eye view it becomes apparent that our approach estimates the position of the moving person correctly. Qualitatively, the precision is comparable with the precision of the laser range finder.

In Fig. 7, the velocity of the person that is additionally estimated by the EKF is plotted for each frame of the image sequence. The person comes into the field of view in the third frame. After a short initialization phase of 2 frames, the velocity is correctly estimated at around 1 m/s. The small oscillation in the graph is caused by the non-uniform leg movement of the person.

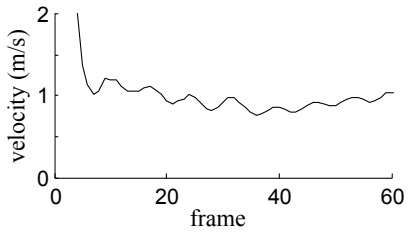


Fig. 7. Estimated velocity of the moving object for each frame of the sequence.

VII. Conclusions and Future Work

In this paper, we presented a method for detecting and estimating the position and velocity of moving objects in monocular image sequences that were captured from a moving robot. Combined with our existing approach for monocular static scene reconstruction, the presented technique allows us to detect both static and dynamic obstacles for robot navigation and obstacle avoidance. Thus, it increases the robustness of our obstacle detection system for real world robot applications in public environments, like tour guides and shopping assistants [8]. Those environments are populated by many persons and other moving obstacles (e.g. shopping carts).

In several experiments, we have tested the robustness of our approach. Even in the difficult case of an object that is moving parallel to the camera, our technique is able to detect the motion and correctly estimates the position of that object. One example for this kind of motion is given in the paper (Fig. 6).

At the moment our algorithm uses a single hypothesis tracker only. Hence, it assumes that there is only one moving object in the camera's field of view. However, it can be easily extended to a multi-hypotheses tracking algorithm that can handle several moving objects.

The presented method for ego-motion compensation using image warping allows the use of difference images for motion detection. In other applications, the same ego-motion compensation technique could be used to allow for the usage of image processing algorithms that were primarily developed for static cameras also for moving cameras.

References

[1] Ijaz Akhter, Yaser Ajmal Sheikh, Sohaib Khan, and Takeo Kanade. Nonrigid Structure from Motion in Trajectory Space. In *Neural Information Processing Systems*, pages 41–48, 2008.

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

[3] J.W. Davis. Hierarchical Motion History Images for Recognizing Human Motion. In *Proceedings IEEE Workshop on Detection and Recognition of Events in Video*, pages 39–46. IEEE Comput. Soc.

[4] A. Del Bue. A Factorization Approach to Structure from Motion with Shape Priors. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

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

[6] E. Einhorn, Ch. Schröter, and H.-M. Gross. Monocular Scene Reconstruction for Reliable Obstacle Detection and Robot Navigation. In *Proc. of the 4th ECMR*, pages 156–161, 2009.

[7] E. Einhorn, Ch. Schröter, and H.-M. Gross. Attention-Driven Monocular Scene Reconstruction for Obstacle Detection, Robot Navigation and Map Building. *Robotics and Autonomous Systems*, 59(5):279–292, 2011.

[8] H.-M. Gross, H.-J. Böhme, Ch. Schröter, St. Müller, A. König, E. Einhorn, Ch. Martin, M. Merten, and A. Bley. TOOMAS: Interactive Shopping Guide Robots in Everyday Use - Final Implementation and Experiences from Long-term Field Trials. In *Proc. of IROS*, pages 2005–2012, 2009.

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

[10] Jens Klappstein. *Optical-Flow Based Detection of Moving Objects in Traffic Scenes*. PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2008.

[11] Jens Klappstein, Fridtjof Stein, and Uwe Franke. Detectability of Moving Objects Using Correspondences over Two and Three Frames. In *Proc of the 29th DAGM conference on Pattern recognition*, pages 112–121, 2007.

[12] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 3D Reconstruction of a Moving Point from a Series of 2D Projections. In *ECCV*, pages 158–171, 2010.

[13] Shrinivas J. Pundlik, Stanley T. Birchfield, and Senior Member. Real-time motion segmentation of sparse feature points at any speed. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 38(3):731 – 742, 2008.

[14] J. Shi and C. Tomasi. Good features to track. In *9th IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[15] J. Taylor, A.D. Jepson, and K.N. Kutulakos. Non-Rigid Structure from Locally-Rigid Motion. In *Proc. Computer Vision and Pattern Recognition Conf.*, pages 2761–2768, 2010.

[16] R. Vidal and D. Abretske. Nonrigid Shape and Motion from Multiple Perspective Views. In *ECCV*, pages 205–218, 2006.