Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007

Stereo based Obstacle Detection with Uncertainty in Rough Terrain

Wannes van der Mark, Johan C. van den Heuvel, Frans C.A. Groen

Abstract— **Autonomous robot vehicles that operate in off-road terrain should avoid obstacle hazards. In this paper we present a stereo vision based method that is able to cluster reconstructed terrain points into obstacles by evaluating their relative angles and distances. In our approach, constraints are enforced on these geometric properties by a set of pixel threshold values. Because these values are all computed during an initialisation step, only simple pixel threshold operations remain to be performed during the real-time obstacle detection. An advantage of this novel approach is that the distance uncertainties can be incorporated into the thresholds.**

Detected obstacle points are clustered into objects on the basis of their pixel connectivity. Objects with insufficient pixels, elevation and slope are rejected. Remaining non-obstacle pixels are regarded as ground surface points. They are used to update the orientation of the stereo camera relative to the ground surface. This prevents orientation errors during stereo reconstruction and the subsequent obstacle detection steps.

Our results show the drawbacks of ignoring the uncertainties in the stereo distance estimates for obstacle detection. It leads to over-segmentation and increases the number of falsely detected obstacles. Because our method incorporates these uncertainties, it can detect more of the obstacle surface pixels at larger distances. This leads to significantly less false obstacle detections.

I. Introduction

An advantage of unmanned robot vehicles is that they can transport payloads or perform useful tasks on different locations without the need that a human is nearby. They are therefore able to execute missions in areas that are either too dangerous or too inhospitable for humans.

The unmanned robot vehicle called "EyeRobot" (Fig. 1) was developed by TNO and partners in a project for the Dutch armed forces. It was built to demonstrate robotic reconnaissance and surveillance in large urban or natural environments. For performing these tasks, it is equipped with several different sensor systems. Among them are two pairs of stereo cameras. One is mounted on a pan-and-tilt unit and is used for remote observation tasks. The other pair faces forward in order to find drivable terrain areas for the robot.

When operating fully autonomously, obstacle detection becomes an important issue for robot vehicles such as the EyeRobot. In off-road terrain there are two types of obstacle hazards that the vehicle should avoid. The first type are obstacles located above the ground surface, such as trees, rocks or steep sand dunes. The other type are negative obstacles, such as ditches and holes.

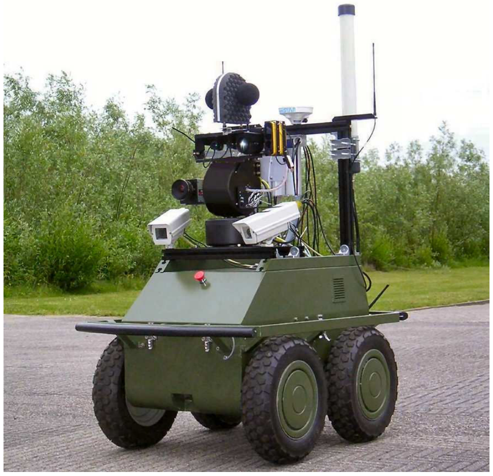

Fig. 1. An image of the EyeRobot robot vehicle demonstrator. The two white cameras are a stereo pair that can be used for obstacle detection. Between these cameras is a pan-and-tilt unit which carries several cameras, microphones and a range finder for remote surveillance tasks.

Computer vision techniques can be applied to distinguish obstacles from drivable surfaces. In this paper we will discuss our approach to stereo based obstacle detection in off-road terrain. The scope of the research is limited to the detection of positive obstacles. We will present an efficient approach that detects and clusters obstacle pixels into objects. It includes a novel method for dealing with the reduced range accuracy of stereo vision at larger distances.

Several approaches to stereo based obstacle detection can be found in the literature. Firstly, there are approaches that use estimated disparities from stereo vision directly. The well-known approach by Labayrade et al. [6], uses histogram analysis in order to find the disparities that belong to the ground surface. Disparities that are located above this surface are regarded as obstacles. An adaptation of this approach for obstacle detection in rough terrain can be found in Broggi et al. [1]. It was used successfully on the TerraMax robot that competed in the DARPA Grand Challenge of 2005.

Approaches that use three-dimensional (3-D) reconstructed points instead, use either digital elevation maps (DEM) [14], [13] or are geometry based. A DEM is a two dimensional grid of which each cell corresponds with certain part of the terrain. The terrain elevation or height in each cell can be derived from the reconstructed stereo data. Cells that are part of positive obstacles will be conspicuous, because their height value is larger than those that are part of the

W. van der Mark and J.C. van den Heuvel are with the Electro-Optics Group at TNO Defence, Security and Safety, Oude Waalsdorperweg 63, P.O. Box 96864, 2509 JG The Hague, The Netherlands wannes.vandermark@tno.nl

F.C.A. Groen is with the Intelligent Systems Laboratorium Amsterdam (ISLA) at the Informatics Institute, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands frans@science.uva.nl

1-4244-1068-1/07/$25.00 ©2007 IEEE.

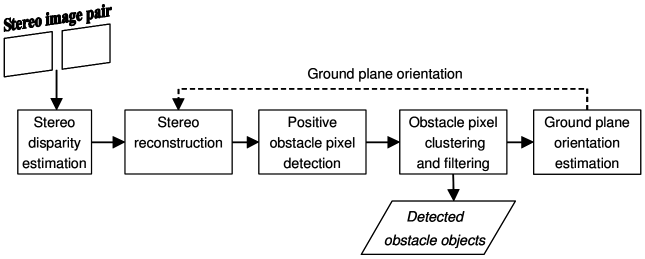
Stereo image pair
Ground plane orientation
Stereo disparity estimation
Stereo reconstruction
Positive obstacle pixel detection
Obstacle pixel clustering and filtering
Ground plane orientation estimation
Detected obstacle objects

Fig. 2. Diagram of the main components of our approach. As input it takes a stereo image pair of unstructured terrain. The result is a list with the detected obstacle objects.

ground surface.

Geometry based approaches try to find points that belong to obstacle surface by analysing the geometric relations between points. In Matthies et al. [9], a vertical range profile of the terrain is extracted from the points inside the image columns. Obstacle points are identified by searching for steep slopes in the range profile.

A more sophisticated approach, developed by the same research group, can be found in Manduchi et al. [7]. It uses two criteria in order to check if two points belong to the same obstacle. The first enforces a minimum y_{min} and maximum y_{max} bound on the vertical distance between the points. The second checks if the line connecting both points has a sufficiently high angle θ with the ground plane. By checking multiple point pairs, detected points can be clustered into separate obstacles.

In the following sections we will present our approach to the detection of positive obstacles in off-road terrain. A diagram of the main components in our approach is shown in Fig. 2. Firstly, disparity estimation is performed on the image pair captured by the vehicle mounted stereo camera. The resulting disparity image is converted into a 3-D point set by applying the stereo reconstruction method discussed in section II. Subsequently, points are searched that belong to positive obstacle surfaces with the approach explained in section III. The next step, described in section IV, clusters detected points into separate obstacle instances. Finally, the ground plane orientation used in the 3-D reconstruction step is updated with the method explained in section V. Quantitative obstacle detection results with real stereo image data will be presented in section VI. The conclusions can be found in section VII.

II. Stereo reconstruction and uncertainty

Estimating distance with stereo vision involves finding the disparity between corresponding image points from the left and right images. In previous work, we developed a framework of several real-time dense disparity estimation algorithms [8]. For the obstacle detection in this study we use the single matching window variant with the left-right consistency check for error and occlusion detection. This variant has a good trade-off between performance and processing speed. Interpolation is used to obtain sub-pixel accurate estimates for the disparity values.

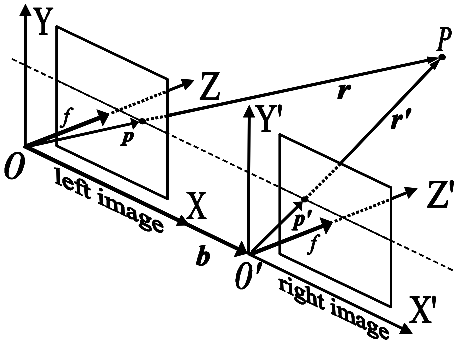

Fig. 3. The geometry of a parallel (rectified) stereo rig.

A prerequisite for the real-time disparity estimation is that corresponding stereo image points lie on the same scan-lines. Because it is difficult to align stereo cameras manually, a rectification technique [3] is used instead. Now suppose that both rectified cameras observe the same 3-D point P, as is shown in Fig. 3. In the left camera reference frame it is indicated by vector $\boldsymbol{r}$, while it is also indicated by vector $\boldsymbol{r}$*'* in the right camera reference frame. If the normalised pin-hole camera model is assumed for image projection, there is also a left $\boldsymbol{p}$ and a right $\boldsymbol{p}$*'* image vector. Both vectors are related by the disparity d in the following way:

$$\boldsymbol{p}' = \begin{pmatrix} x' \\ y' \\ f \end{pmatrix} = \begin{pmatrix} x-d \\ y \\ f \end{pmatrix} \quad \text{and} \quad \boldsymbol{p} = \begin{pmatrix} x \\ y \\ f \end{pmatrix} . \qquad (1)$$

Here, the scalars x and y are the horizontal and vertical image positions. Because the pin-hole model is normalised the focal length f of each camera is equal to 1.0.

In order to reconstruct $\boldsymbol{r}$ or $\boldsymbol{r}$*'*, the scalar difference λ has to be estimated from the disparity d and the camera baseline distance b:

$$\boldsymbol{r} = \lambda \boldsymbol{p} \quad \text{and} \quad \boldsymbol{r}' = \lambda \boldsymbol{p}' \quad \text{with} \quad \lambda = \frac{b}{d} . \qquad (2)$$

A problem in stereo vision is that the disparity d cannot be estimated precisely. Errors in the estimate for d will have consequences for the precision of λ and consequently for $\boldsymbol{r}$. It is assumed that the distance errors in λ have a normal distribution with variance $V[\lambda]$. The propagation of the disparity error into the distance variance can be calculated with a method developed by Kanatani [5]:

$$V[\lambda] = \frac{2\varepsilon^2 \lambda^4}{b^2} = 2\varepsilon^2 \frac{b^2}{d^4} . \qquad (3)$$

Here, ε indicates the noise level in the pixel coordinates of $\boldsymbol{p}$ and $\boldsymbol{p}$*'*. In our disparity estimation algorithms [8], a quadratic method is used to obtain a sub-pixel disparity estimate. It is mentioned in the work by Hirschmüller et al. [4] that applying this method leads to an interpolation factor of approximately $1/8$ disparity. The value of ε is therefore set to $1/8$ of a pixel.

Due to the distance uncertainty, the position of stereo reconstructed points can only be determined up to a certain degree. Because we only consider errors in scalar distance, the uncertainty is limited to a line emanating from camera, with the same direction as vector $\boldsymbol{p}$. The probability that the point is located on the line piece between the vectors $\boldsymbol{r}_{\min}$ and $\boldsymbol{r}_{\max}$ (Eq. 2), depends on the choice for parameter σ:

$$\lambda_{\min} = \lambda - \sigma\sqrt{V[\lambda]} \text{ and } \lambda_{\max} = \lambda + \sigma\sqrt{V[\lambda]} \ . \qquad (4)$$

In our approach, we use $\sigma = 3.0$ so that it encompasses about 99% of the distance uncertainty.

Reconstructed vectors from stereo are defined in the left or right camera reference frame. For obstacle detection, it is more convenient to use a reference frame that is associated with the terrain. In our approach, we choose a reference frame where the XZ-plane is parallel to the ground surface. The Z-axis points away from the vehicle, while the X-axis is orientated to the right side. Converting reconstructed vector $\boldsymbol{r}$ to vector $\boldsymbol{s}$, in this reference frame, involves a rotation defined by matrix $\boldsymbol{R}$:

$$\boldsymbol{s} = \boldsymbol{R}\boldsymbol{r} \ . \qquad (5)$$

The same rotation can also be applied to convert both vectors $\boldsymbol{r}_{\min}$ and $\boldsymbol{r}_{\max}$ to $\boldsymbol{s}_{\min}$ and $\boldsymbol{s}_{\max}$.

For our experiments, we obtained an initial estimate for rotation matrix $\boldsymbol{R}$ while the vehicle is parked on a flat and paved surface. A large plate with a checkered calibration pattern was placed on the ground in front of the vehicle for this purpose. We then used the camera calibration technique developed by Zhang [15] to estimate the orientation difference between the plate and the camera reference frame. It should be noted that the resulting matrix $\boldsymbol{R}$ only has to be measured once. In section V a method will be explained for updating the orientation without a calibration object.

III. Positive obstacle pixel detection

By applying stereo reconstruction, a disparity image can be converted into a large set of 3-D points. The problem is now to identify the points among this set that belong to obstacle surfaces. The method for detecting obstacle points is explained in this section.

Our approach is based on the method developed by Manduchi et al. [7], because it enables the clustering of obstacle points into objects. This has the advantage that false positive detections can be rejected based on the geometric properties of the clustered object points. As discussed earlier, the original approach uses two criteria to check whether or not two points belong to the same obstacle surface. We have developed an approach where this check can be computed more efficiently. Furthermore, our approach also enables the inclusion of uncertainty in the stereo distance estimates.

A. Threshold based detection with uncertainty

The two criteria that decide whether two points belong to the same obstacle surface can be understood as a bounded space that limits the possible point locations. For a single point $\boldsymbol{s}_1$, this space resembles a truncated up-side-down cone. Only points that are located within the space satisfy the two criteria. Fig. 4 shows the space on the right side and its image projection on the left side. Projected to the image domain, the space is reduced to a trapezoid shaped area. Its dimensions depend on where point $\boldsymbol{s}_1$ is located relative to the cameras in the ground plane reference frame.

Instead of determining the trapezoid dimensions for every point or possible distance, our approach only uses a fixed set of differently sized trapezoids. There are n number of trapezoids in the set and the corresponding truncated cones are spaced at regular distances of z_{step} apart. If $z_{\min}$ is the nearest and $z_{\max}$ is the furthest reconstructed distance, then the distance z_i of the i-th truncated cone is equal to:

$$z_i = z_{\min} + i z_{\text{step}} \quad \text{with} \quad z_{\text{step}} = \tfrac{1}{n}\left(z_{\max} - z_{\min}\right) \ . \quad (6)$$

Each pixel j within the trapezoid boundaries has its own distance threshold value Δz_j. As can be seen in Fig. 4, this value corresponds to the distance between the cone surface and the gray coloured plane at distance z_i that splits the truncated cone in two halves. Therefore, a pixel with distance value z falls within the truncated cone space if the absolute difference with z_i is smaller than or equal to Δz_j:

$$|z - z_i| \leq \Delta z_j \ . \qquad (7)$$

Because we use a fixed set of trapezoids, the Δz_j value of each trapezoid pixel only has to be calculated once. This can be done during initialisation so that the required computation during processing, is reduced to a threshold operation for each point pair.

A problem with the Δz_j threshold value is that it does not consider inaccuracies in stereo reconstructed distances. Because of the projective nature of imaging, these inaccuracies increase with distance. Therefore, pairs of points that lay far away might not be clustered to the same obstacle surface due to large distance estimation errors. In order to reduce this problem we propose to include a measure for this inaccuracy in the obstacle point check. As discussed in the stereo reconstruction section, Eq. 4 can provide a line piece that corresponds to the distance uncertainty of a reconstructed point. This line piece is between vectors $\boldsymbol{s}_{\min}$ and $\boldsymbol{s}_{\max}$ in the ground surface reference frame. Vector $\Delta\boldsymbol{s}$ is the difference of both vectors:

$$|\boldsymbol{s}_{\max} - \boldsymbol{s}_{\min}| = \Delta\boldsymbol{s} = \left(\Delta s_x \ \Delta s_y \ \Delta s_z\ \right)^{\top} \ . \qquad (8)$$

The Δs_z value indicates the uncertainty in z_i of a trapezoid pixel. Note that the absolute difference with z_i is used in Eq. 7 to check whether or not a pixel is an obstacle point. Therefore, the Δs_z value can simply be added to Δz_j to include the distance uncertainty. This leads to the following criterion to identify obstacle points:

$$|z - z_i| \begin{cases} \leq \Delta z_j + \Delta s_z & \text{is an obstacle point} \\ > \Delta z_j + \Delta s_z & \text{is not an obstacle point} \end{cases} \qquad (9)$$

B. Precalculation of the threshold trapezoids pixels

Our approach relies on performing an initialisation step where all the Δz_j values for all trapezoid pixels are calculated. The required geometry is explained in this section. Suppose that vector $\boldsymbol{s}_1$ indicates a point at distance z_i in the ground plane orientated reference frame. Fig. 4 shows this

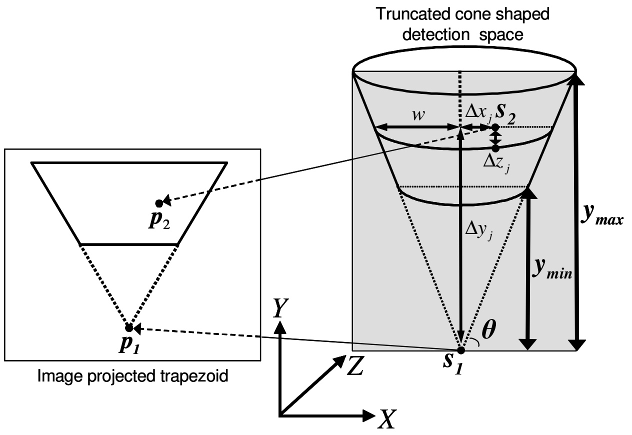

Fig. 4. Geometry used for computing the Δz_j values.

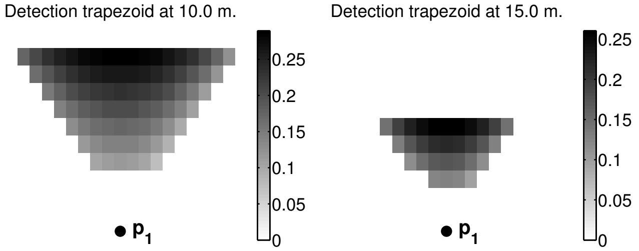

Fig. 5. Two examples of trapezoid detection areas. The one on the left side was calculated with $z_i = 10.0$ m, the other one with $z_i = 15.0$ m.

point and the plane, that lies at distance z_i. It also shows the front part of the cone shaped obstacle point search space with its origin at $\boldsymbol{s}_1$.

The problem is now to calculate Δz for an arbitrary image point indicated by the vector $\boldsymbol{p}_2$. This is possible by first calculating the point $\boldsymbol{s}_2$ where the rotated vector $\boldsymbol{p}_2$, extended by μ, intersects the plane at z_i. The plane can be defined by its support vector $\boldsymbol{s}_1$ and a surface normal vector $\boldsymbol{n}$. Because the plane is perpendicular to Z-axis, vector $\boldsymbol{n}$ is parallel to it. Scalar μ is now equal to the fraction of the vector dot products of $\boldsymbol{n}$ with both $\boldsymbol{s}_1$ and $\boldsymbol{p}_2$:

$$\boldsymbol{s}_2 = \mu \left(\boldsymbol{R}\boldsymbol{p}_2\right), \mu = \frac{\boldsymbol{n} \cdot \boldsymbol{s}_1}{\boldsymbol{n} \cdot \boldsymbol{R}\boldsymbol{p}_2} \quad \text{and} \quad \boldsymbol{n} = \left(0\ 0\ 1\right)^\top . \tag{10}$$

Fig. 4 also shows the intersection point $\boldsymbol{s}_2$. Because it lies in the same plane as the cone origin vector $\boldsymbol{s}_1$, there is only a horizontal Δx_j and vertical offset Δy_j. The radius w of the cone at vertical offset Δy_j can be computed with:

$$w = \Delta y_j \tan\left(\tfrac{1}{2}\pi - \theta\right), \quad |\boldsymbol{s}_2 - \boldsymbol{s}_1| = \begin{pmatrix} \Delta x_j \\ \Delta y_j \\ 0 \end{pmatrix} . \tag{11}$$

The desired depth difference Δz_j between the cone surface and vector $\boldsymbol{s}_2$ can be computed from the radius w and the horizontal Δx offset:

$$\Delta z_j = \tan\left(\varphi\right) \Delta x_j \quad \text{with} \quad \varphi = \arccos\left(\frac{\Delta x_j}{w}\right) . \tag{12}$$

Eqs. 10 until 12 can be used to compute all the Δz_j of pixels within the trapezoid boundaries. The calculations involved, only have to be performed during initialisation. The number of pixels per trapezoid is set to a limit of 50 in order to further reduce the computational complexity of applying the threshold. For trapezoid regions calculated at large distances this has no consequences because their pixel size is smaller than the limit. However, for nearby distances, the calculated threshold values are sorted on their Euclidian pixel distance with point $\boldsymbol{p}_1$. This insures that only the nearest pixels are inspected. Two examples of pre-calculated trapezoids are shown in Fig. 5.

IV. Obstacle point clustering and filtering

Points found by the positive obstacle pixel detection may belong to different obstacles. However, there can also be wrongly detected points on non-obstacle surfaces. Further geometric checks are needed to remove these false detections. For this step, neighbouring obstacle pixels are first clustered. Each clustered set is regarded as an object surface. Several geometric properties can be checked to distinguish real obstacle surfaces from false ones.

We use a morphological approach to cluster the detected obstacle points into object clusters. Because each reconstructed point corresponds to a pixel, it has a maximum of eight neighbouring points. By computing the absolute difference of their z-distances it can be determined whether a point and its neighbour belong to the same obstacle surface. The threshold of this difference is equal to $z_{\text{step}} + \Delta s_z$ to include the distance uncertainty. Here, the distance uncertainly Δs_z is that of the centre pixel.

By applying the eight connected pixel clustering, the obstacle points can be merged into one set per object. In most cases, this results into a list of several obstacle points sets. Our approach uses three properties to determine if a point set is a real obstacle. The number of points per set is the first property. If there are less than 10, the set is regarded as noise and is removed. Another property is the height of the bounding box that encloses a point set. A set will be removed if its height is less than y_{min}. Finally, the median slope of a point set is checked. This is used to find obstacles that are too "flat" to be a real danger to the vehicle. For its calculation, image columns are searched that intersect with the obstacle pixels of a set. The top and bottom obstacle pixel are determined in each column found. These pixels correspond to two space points on a line. The angle between this line and the Y-axis is regarded as a local approximation of the slope in the column. The median slope is obtained by collecting the slope for all columns. When an obstacle is too flat its median slope will also be small. In our approach, obstacles with a median slope angle smaller than $5.0°$ are rejected as false obstacles.

V. Ground plane orientation update

Terrain elevation is an important feature for detecting obstacle pixels. An assumption in our approach is that it can be measured along the Y-axis of the reference system associated with the ground plane. A problem with driving over rough terrain is that the vehicle can undergo roll and pitch motions. It changes the ground plane orientation relative to the vehicle mounted stereo camera. Uncorrected, there will be a misalignment of terrain height with the coordinate system Y-axis. This can lead to problems with the obstacle detection. The paper by Manduchi et al. [7] briefly mentions the use of inclination sensors or a vision routine to correct for these orientation changes. Here, we propose our own vision based solution.

The change in ground plane orientation is modelled by a small rotation $\boldsymbol{R}_u$. It is applied to the initial orientation $\boldsymbol{R}$ obtained from the ground plane calibration:

$$\hat{\boldsymbol{R}} = \boldsymbol{R}_u \boldsymbol{R} \; . \tag{13}$$

The new orientation is defined by $\hat{\boldsymbol{R}}$. In order to measure $\boldsymbol{R}_u$, the ground surface pixels from the previous frames are used. An interesting aspect here is that the remaining set of points, after removal of obstacle points, must belong to the ground plane surfaces. Therefore, only the remaining pixels are used for orientation estimation.

Suppose that there is a set of N vectors $\boldsymbol{s}_{i=1\ldots N}$ that belong to the ground surface. Rotation $\boldsymbol{R}_u$ can be estimated using the following steps. First, compute the centroid of the set and subtract it from each vector. Then compose the matrix $\boldsymbol{M}$ by concatenating the resulting vectors $\bar{\boldsymbol{s}}_i$:

$$\boldsymbol{M} = [\bar{\boldsymbol{s}}_1 \; \bar{\boldsymbol{s}}_2 \cdots \; \bar{\boldsymbol{s}}_N]^\top \quad \text{with} \quad \bar{\boldsymbol{s}}_i = \boldsymbol{s}_i - \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{s}_j \; . \tag{14}$$

With the Singular Value Decomposition [11] this matrix can be decomposed in $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^\top$. The smallest singular value in the diagonal matrix $\boldsymbol{D}$ has a corresponding singular column vector $\boldsymbol{n}$ in matrix $\boldsymbol{V}$. This vector is the least squares fit for the normal of the plane.

We seek the rotation that turns $\boldsymbol{n}$ into the vector $\boldsymbol{y}$ that is aligned with the Y-axis. The rotation axis, indicated by normalised vector $\boldsymbol{a}$, is obtained with the cross product:

$$\boldsymbol{a} = \frac{\boldsymbol{n} \times \boldsymbol{y}}{\|\boldsymbol{n} \times \boldsymbol{y}\|} \quad \text{with} \quad \boldsymbol{y} = (0 \; 1 \; 0)^\top \; . \tag{15}$$

The smallest rotation angle between both vectors can be computed by taking $\varphi = \arccos{(\boldsymbol{n} \cdot \boldsymbol{y})}$. Together, the rotation angle φ and axis $\boldsymbol{a}$ can be used to form a quaternion $\boldsymbol{q}$ [10]:

$$\boldsymbol{q} = \left(\cos(\tfrac{1}{2}\varphi) \; \left[\sin(\tfrac{1}{2}\varphi)\boldsymbol{a}^\top\right]\right)^\top \; . \tag{16}$$

By converting the quaternion $\boldsymbol{q}$, the rotation matrix $\boldsymbol{R}_u$ can be obtained.

It should be noted that at least three vectors are needed to estimate $\boldsymbol{R}_u$. These vectors should not indicate points that are collinear. Furthermore, in order to remove remaining outliers, the robust Least Median of Squares estimation technique [12] is applied.

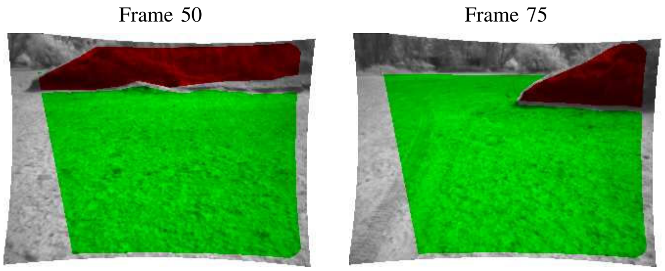

Fig. 6. Example labelled images from the test sequence. The colour red indicates positive obstacles and the colour green indicates the ground surface.

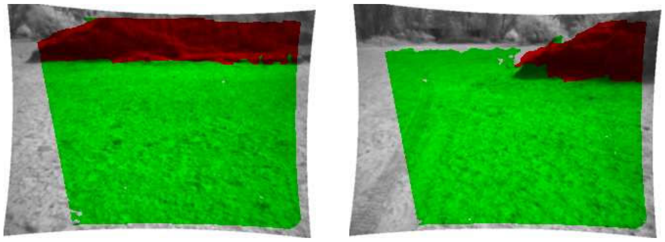

Fig. 7. Obstacle detection output of our approach.

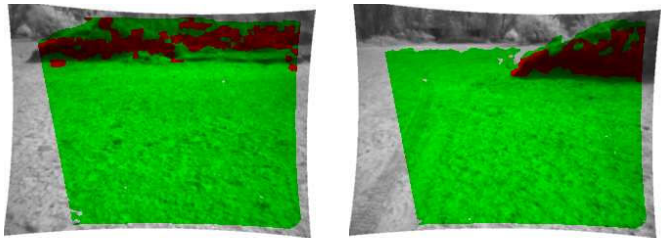

Fig. 8. Obstacle detection output of the original approach.

VI. Results

One of the key differences between our and the original approach by Manduchi et al. [7] is the use of a fixed set of trapezoid shaped threshold windows. The other is that our approach considers the uncertainty in distance values obtained from stereo reconstruction. Therefore, we compare the results of our approach to those of a version that uses the original approach to check if pixels belong to the same obstacle surface. This involves computing the angle and height difference for each inspected point pair. It should be noted that this version does not use the tree-labelling algorithm, to cluster detected points into obstacle clusters, as described in the original paper [7]. In order to limit the differences between the two approaches, the pixel based method of section IV is used instead. Both approaches also use the vision based orientation update routine described in section V. During the evaluation, parameters that are used by both methods are set to; $y_{\min} = 0.1$ m, $y_{\max} = 0.3$ m and $\theta = 45°$. For our method, the trapezoid related parameters are set to; $z_{\min} = 2.0$ m, $z_{\max} = 30.0$ m and $z_{\text{step}} = 60$.

In the literature, the evaluation of new approaches to obstacle detection in off-road terrain is often limited to qualitative examples that demonstrate detection capabilities. Instead, we have devised several quantitative tests in order to investigate obstacle detection performance. The idea behind the tests is that positive obstacles pixels and ground surface pixels belong to two distinct classes. Misclassifying positive obstacle pixels as ground pixels leads to false negative detec-

TABLE I

PIXEL BASED DETECTION PROBABILITIES.

			$P(C\mid j)$	
Method	P_C	$\overline{P}_C$	Positive obstacle	Ground plane
Ours	0.988	0.966	0.942	0.991
Original	0.938	0.798	0.599	0.996

tion, while misclassifying ground surface pixels as positive obstacle pixels can be regarded as false positive detection.

For the evaluation, a sequence of 102 stereo images was used where the vehicle drives towards a sand dune obstacle. At the beginning of the sequence, the obstacle is far away. During the sequence, the vehicle moves closer to the obstacle. Each of the sequence images was hand labelled with polygons to indicate pixels that belong to positive obstacles or to the ground surface. Fig. 6 shows two images from the test sequence. Labelled pixels on a positive obstacle have been indicated with red, while ground surface pixels are coloured green. Detection results from our method are shown in Fig. 7. It can be seen that a large majority of the obstacle pixels are detected correctly. Example output from the original method is shown in Fig. 8. Here, it is clear that less obstacle pixels are detected as compared to our method. Many of the obstacle pixels are misclassified as being ground surface.

In the evaluation, the output of the obstacle detection algorithm is compared to the hand labelled ground truth. The number of times that a pixel with ground truth label j has been classified as k is counted. For each image, the result is collected in confusion matrices $CM_{j,k}$. Castano et al. [2] propose three quantitative performance measures based on confusion matrices. The first measure is the ratio of correctly classified points among the total number of points in the complete test set:

$$P_C = \frac{\sum_j CM_{j,j}}{\sum_j \sum_k CM_{j,k}} \ . \tag{17}$$

An estimate of the correct classification probability for each class j is given by:

$$P(C|\,j) = \frac{CM_{j,j}}{\sum_k CM_{j,k}} \ . \tag{18}$$

A disadvantage of the global measure P_C is that it is biased towards the class that occurs the most in the test set. The unbiased overall probability of correct classification $\overline{P}_C$ is equal to average of $P(C|\,j)$ for all classes.

The plot in Fig. 9 shows classification probability $P(C|\,j)$ for each sequence frame. Table I shows the average probabilities for P_C, $\overline{P}_C$ and $P(C|\,j)$ for the whole sequence. It can be seen that both our and the original approach score high classification probabilities for the ground surface pixels. The methods score an average probability of 0.991 and 0.996 respectively.

However, there is a significant difference in the classification probabilities for positive obstacle pixels. Our method scores an average classification probability of 0.942, while the original method only manages to score a probability of 0.599. The plot of Fig. 9 shows that, at the start of the sequence, the classification score of the original method is lower as compared to our method. Its performance does increase during the sequence. At the end of the sequence the performance of both methods seems to drop dramatically. This is caused by the fact that the object is leaving the stereo camera's field of view at the end of the sequence. Because the number of obstacles pixels is very small in these last few frames, the detection reliability of both approaches is also low.

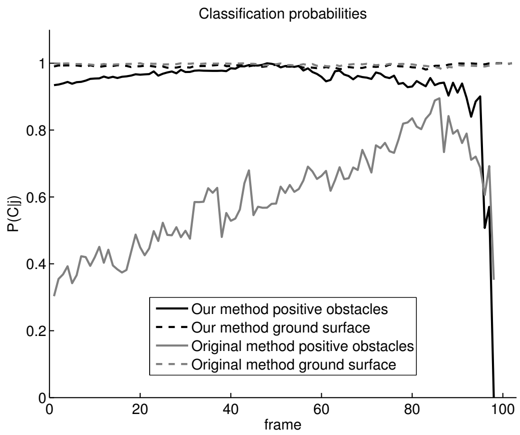

Fig. 9. Detection probabilities for positive obstacle and ground surface pixel classes.

The performance of both methods during the sequence can be explained when the obstacle distance is considered. The original method computes the angle and height differences between point pairs to check if they belong to the same obstacle surface. This degrades the ability of the original method to distinguish between obstacle and other points, because there will also be more errors in the angle and height differences between point pairs. However, distance values will become more accurate during the sequence since the vehicle is driving towards the obstacle. It also improves distinction between obstacle and other points. This explains why the performance of the original method starts out low and improves during the sequence.

Our method does not compute the geometric check used by the original method explicitly. Instead, precalculated threshold values are applied together with a margin to account for the distance uncertainty. The results indicate that this leads to a better distinction between obstacle and other points at further distances.

The classification probabilities only indicate the methods reliability to distinguish obstacle pixels from ground surface pixels. Another important issue is the ability of the methods to cluster obstacle pixels correctly. Monolithic objects should not be segmented in two or more separate objects. To test if over-segmentation occurs, the number of detected objects that occupy the image region of the labelled obstacle was counted. Fig. 10 shows the number of counted objects per labelled obstacle for each frame. For all but one frame,

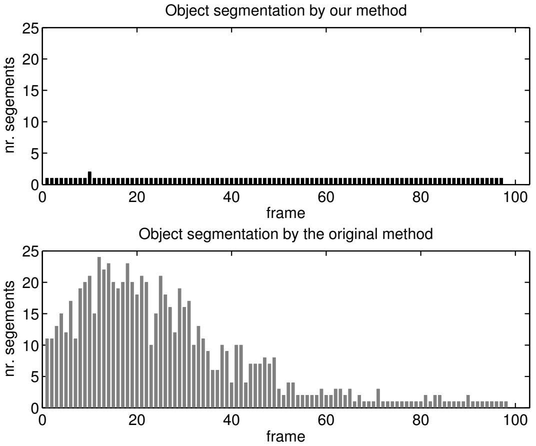

Fig. 10. Number of detected segments in the labelled obstacle area.

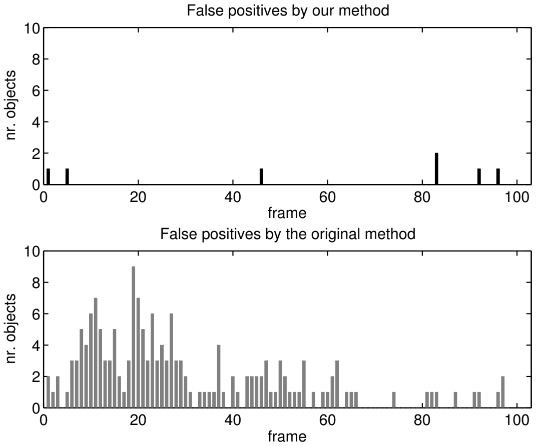

Fig. 11. Number of falsely detected obstacles per frame.

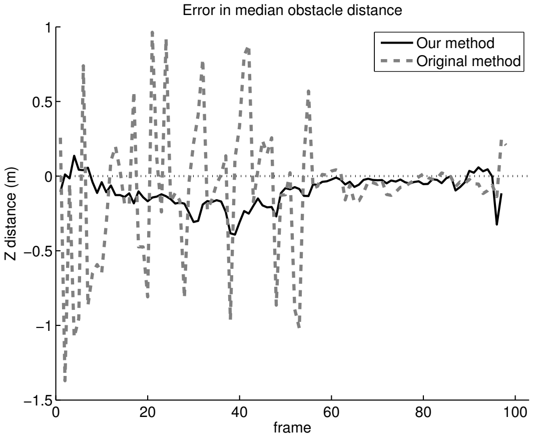

Fig. 12. Error of the median estimated obstacle distance.

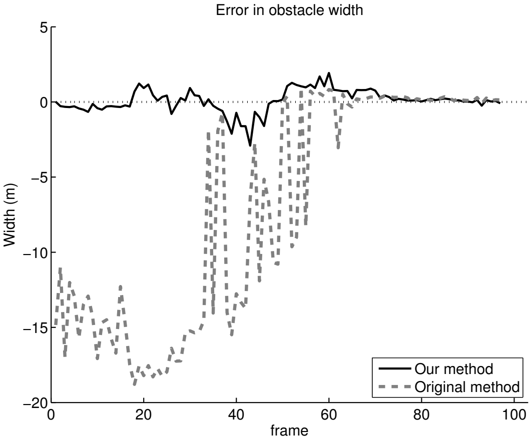

Fig. 13. Error of the measured obstacle width.

our method clusters the labelled object pixels correctly. The original method fails to achieve this for the majority of frames. Particulary when the obstacle is at a large distance, it detects multiple objects instead of one single object. This is a result of the fact that the original method detects fewer obstacle pixels at these distances. Apparently, the remaining pixels do not link up properly to form a single object.

Another property of detection is the amount of false alarms. The number of detected obstacles in the image region that belongs to the ground surface is shown in Fig. 11. Of the 102 frames, there are only 6 frames where our method detects false obstacles. Substantially more false detections are found in the results of the original method. Most of these also occur when the real obstacle is far away.

Obstacle over-segmentation has consequences for the autonomous vehicle guidance. This can be demonstrated by inspecting the measured obstacle geometric properties. Remaining distance and width are properties that are important for obstacle avoidance. If these are measured inaccurately, the vehicle could still hit the obstacle because it can erroneously appear to be smaller or further away.

In this evaluation, the geometric properties of the labelled pixels are used as ground truth. Obstacle distance is computed by taking the median of these pixel range values. Width is obtained by finding a bounding box that fits the obstacle points. The geometric properties of detected obstacles are computed in the same way. Only one detected obstacle by each of the methods is compared with the ground truth. If there are multiple detections due to segmentation, the nearest obstacle to the ground truth is selected.

Fig. 12 shows the difference in median distance of the ground truth points and the detected obstacle points for both methods. It indicates the type of error in the distance estimate. Negative values indicate too small distance estimates and positive values are too large estimates. The plot shows that when the obstacle is nearby, both methods only display small errors. However, the errors of the original method are more severe than those of our method for larger distances. The results for obstacle width are shown in Fig. 13. It reveals that the original method underestimates the width when the obstacle is further away. Our method provides more consistent estimates.

The poor performance of the original method is clearly due to the over-segmentation. Comparisons with Fig. 10 show that the large errors occur in the same frames where the object is divided in several segments. There are errors in the

TABLE II

PROCESSING TIMES FOR EACH OF THE VARIOUS ALGORITHM STEPS.

Task	Time
Disparity estimation	0.038 s
Stereo reconstruction	0.002 s
Positive obstacle detection	0.060 s
Positive obstacle clustering	0.001 s
Ground orientation estimation	0.021 s
Total processing time	**0.122 s**

distance and width measurements because each segment only covers a part of the obstacle.

As indicated earlier, a problem for stereo is that the accuracy of 3-D reconstructed points decreases with distance. Because the original method does not consider this, it over-segments real obstacles and detects more false obstacles among points that lie far away. The results of our method show that inclusion of distance uncertainty measures, leads to more reliable detections.

An obstacle detection algorithm is not suitable for autonomous vehicle guidance if its computational demands do not allow for a sufficiently fast processing rate. We therefore measured the time needed to process stereo images by our method. Table II contains the measured processing times of the different components for one stereo pair. The timing tests were performed on an Intel Core 2 computer with a clock speed of 2.66 GHz and 2.0 GB of RAM. It can be seen that the most computational expensive step is the positive obstacle detection. The total time is 0.122 seconds, which allows for a processing rate of 8.197 stereo images per second.

VII. CONCLUSION

We have presented our stereo vision based approach to obstacle detection in off-road terrain for autonomous vehicles. Dense disparity estimation is used for obtaining three-dimensional terrain points from stereo.

The method for detecting obstacles is inspired by the method of Manduchi et al. [7]. The original method used two geometrical constraints in order to check if a pair of points belong to the same obstacle surface. Instead of computing these constraints for each point pair, a fixed set of threshold values utilised in our approach. This has several advantages over the original method. Firstly, it reduces the computational complexity of the obstacle point check to a simple threshold operation. Secondly, the required threshold values are only computed during initialisation and are reused for each stereo pair. Furthermore, it allows for an easy way to deal with the uncertainty of the stereo reconstructed point distances. The uncertainty in reconstructed distance can simply be added to the threshold value.

Detected obstacle pixels are clustered into objects. Resulting objects with insufficient pixels, height and slope are discarded as false obstacles. The remaining ground surface pixels are used to estimate the orientation difference between the stereo camera and the ground surface. This is used to compensate changes in camera orientation while the vehicle is driving over rough terrain.

Quantitative tests show that due to the inclusion of distance uncertainty, our method is able to detect more pixels on obstacle surfaces, especially if they are located further away. This is advantageous for the subsequent clustering of obstacle points. In contrast to the original approach, our method does not segment obstacles into several smaller ones. Analysis of the measured obstacle distance and width have shown that over-segmentation can lead to inconsistent or underestimated values. Underestimation is dangerous because the obstacle appears to be smaller than its real dimensions. In such a case, the vehicle could still hit the obstacle, because steering decisions to avoid the obstacle will be based on incorrect information.

Because our method does not over-segments obstacles, it does not suffer from inconsistent or underestimated obstacle dimensions. It is therefore more suitable for application in autonomous vehicle guidance.

REFERENCES

[1] A. Broggi, C. Caraffi, P. Porta, and P. Zani, “The Single Frame Stereo Vision System for Reliable Obstacle Detection used during the 2005 Darpa Grand Challenge on TerraMax,” in *IEEE Intelligent Transportation System Conference*, Toronto, Canada, September 2006, pp. 745–752.

[2] R. Castano, R. Manduchi, and J. Fox, “Classification experiments on real-world textures,” in *Workshop on Empirical Evaluation in Computer Vision*, December 2001.

[3] R. Hartley, “Theory and practice of projective rectification,” *International Journal of Computer Vision*, vol. 35, no. 2, pp. 115–127, November 1999.

[4] H. Hirschmüller, P. Innocent, and J. Garibaldi, “Real-time correlation-based stereo vision with reduced border errors,” *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 229–246, 2002.

[5] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*, ser. Machine Intelligence and Pattern Recognition 18. Elsevier Science, 1996.

[6] R. Labayrade and J. Tarel, “Real Time Obstacle Detection on Non Flat Road Geometry through ‘V-Disparity’ Representation,” in *IEEE Intelligent Vehicle Symposium*, vol. 2, June 2002, pp. 646–651.

[7] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, “Detection and terrain classification for autonomous off-road navigation,” *Autonomous Robots*, vol. 18, pp. 81–102, 2005.

[8] W. v. d. Mark and D. Gavrila, “Real-time dense stereo for intelligent vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 38–50, March 2006.

[9] L. Matthies and P. Grandjean, “Stochastic performance modeling and evaluation of obstacle detectability with imaging range sensors,” *Transactions on Robotics and Automation*, vol. 10, no. 6, December 1994.

[10] E. Pervin and J. Webb, “Quaternions for computer vision and robotics,” *Computer Vision and Pattern Recognition*, pp. 382–383, 1983.

[11] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, 2nd ed. Cambridge University Press, 1992.

[12] P. Torr and D. Murray, “The development and comparison of robust methods for estimating the fundamental matrix,” *International Journal of Computer Vision*, vol. 24, no. 3, pp. 271–300, 1997.

[13] M. Vergauwen, M. Pollefeys, and L. V. Gool, “A stereo-vision system for support of planetary surface exploration,” *Journal Machine Vision and Applications*, vol. 14, no. 1, pp. 5–14, April 2003.

[14] Z. Zhang, “A stereovision system for a planetary rover: Calibration, correlation, registration, and fusion,” in *IEEE Workshop on Planetary Rover Technology and Systems*, April 1996.

[15] ——, “Flexible camera calibration by viewing a plane from unknown orientations,” in *IEEE International Conference on Computer Vision*, September 1999, pp. 666–673.