Obstacle Detection Based on Dense Stereovision for Urban ACC Systems

Ciprian Pocol[*], Sergiu Nedevschi[*], Marc-Michael Meinecke[**]

[(*)] Technical University of Cluj-Napoca
Computer Science Department, Gh. Baritiu Street 26, Romania
{ciprian.pocol, sergiu.nedevschi}@cs.utcluj.ro

[(**)] Volkswagen AG, Electronic Research, P.O.Box 1776, 38436 Wolfsburg, Germany
marc-michael.meinecke@volkswagen.de

***Abstract*–The dense (all pixels in image) stereo reconstruction, real-time computable nowadays, brings 3D reconstructed points even for less textured image regions, and has a lower percentage of wrong reconstructed points. This article presents novel algorithms for obstacle detection, using dense stereo reconstruction. They analyze, on the top view of the scene, the local density and vicinity of the 3D points, and determine the occupied areas which are then fragmented into obstacles with cuboidal shape: without concavities and only with 90° convexities. The orientation of the obstacles is determined in order to get a very good fitting of the cuboidal model to the obstacles in the scene ahead and, consequently, to minimize the free space which is encompassed by the cuboids. The main abilities of the approach are: generic obstacle detection, determination of obstacles' orientation, confident fitting of the cuboidal model.**

***Index terms*–dense stereo vision, obstacle detection, obstacle modeling, urban traffic, automatic cruise control**

I. INTRODUCTION

Although the nowadays automatic cruise control (ACC) systems are a great achievement, they have limitations even in highway traffic where the scenario is quite simple. In the crowded city traffic solutions are not available yet. Currently the ACC systems detect the obstacles ahead by using either a radar or laser setup.

The stereovision is the most promising technology so far: it is based on a passive sensor, it accurately measures the 3D position of thousands of 3D points in the scene ahead and it allows algorithms working on 3D data and intensity images as well. The stereovision has the disadvantage of requesting high complexity algorithms and processing large amount of data.

The most known stereovision based approaches are: Inverse Perspective Mapping [1]; V-Disparity [2, 3], Warping [4]. They try to avoid the full 3D reconstruction in order to reduce the processing time. Unfortunately these methods have intrinsic limitations due to the methods themselves, due to the abandon of the disambiguation of the obstacles close to each other, and due to the inaccurate modeling of the 3D obstacles.

The strongest point of our approach, based on the full 3D reconstruction of the scene [5], is the possibility to carry out geometrical reasoning for generic obstacle detection regardless the 2D appearance in images. The obstacles are confidently described by using the cuboidal model.

The algorithms presented in this paper work on 3D points provided by a dense stereo reconstruction engine. It provides 3D reconstructed points even for less textured image regions (figure 1.b).

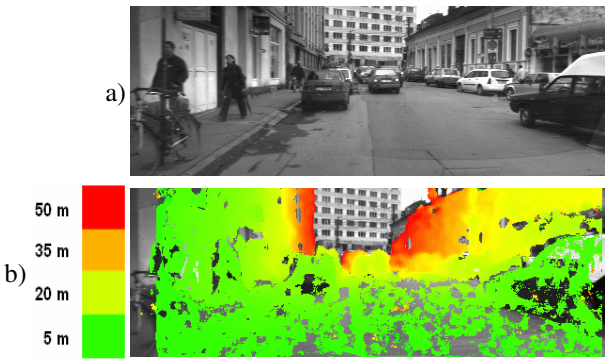


Figure 1. a) Left grayscale image, b) Pixels reconstructed as 3D points. The leftmost part is not reconstructed being not visible in the right image.

The target is to detect the obstacles in terms of 3D position, orientation and size, as boxes (cuboids) circumscribing the obstacles. The confident fitting of cuboids on obstacles is achieved in several steps. By analyzing the vicinity and the density of the 3D points, the occupied areas are localized (section IV). An occupied area

consists of one or more cuboidal obstacles that are close to each other. By applying criteria regarding the shape, the occupied areas may get fragmented into parts that obey the cuboidal shape (section V). The orientation of the obstacles (on the road surface) is of high importance as well (section VI).

The environment is considered as having a planar or a second-degree road surface, with obstacles above it. As guessed, an algorithm for obstacle/road separation is involved.

II. Dense Stereo Reconstruction

A calibration process estimates the camera's intrinsic parameters (which are related to its internal optical and geometrical characteristics) and extrinsic ones (which are related to the 3D position and orientation of the camera relative to a global world coordinate system) [6].

The most computational expensive task, the stereo correlation, is performed by hardware, a specialized PCI board (www.tyzx.com). A software module rectifies the left and the right images, the hardware computes the disparities, and then, the 3D points are computed, by software. Both rectification and 3D points computing use the cameras' parameters obtained through the calibration.

Textureless image regions, regions with repetitive texture, regions of the faraway scene and regions that are not visible in both left and right images cannot be reconstructed (figure 1).

The obstacle detection algorithms work on the 3D reconstructed pixels and on the left image space (figure 1).

A 3D coordinate system is defined. The origin of the coordinates system is the left camera. The Z axis is oriented ahead, representing the depth, the X-axis represents the lateral displacement and the Y-axis is used to express the elevation.

III. 3. Obstacle/Road Separation

The obstacle/road separation algorithm ([5], [7]), is able to detect the surface of the road, by modeling it as a second-degree surface.

The only 3D points used by the obstacle detection algorithms are those situated above the road and below the height of the ego car (figures 2, and 3.b).

In figure 3.b one can observe some road points marked as above the road, due to the reconstruction errors.

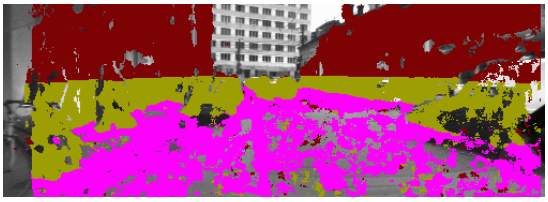


Figure 2. 3D points classified as: road, above road and too high points

IV. Localization Of Occupied Areas

Ideally, the obstacle detection algorithms should neither divide a real obstacle into smaller detected obstacles nor merge more real obstacles into one detected obstacle; two opposite requirements.

Previous experience has shown that these requirements are hard to be fulfilled due to reconstruction errors and limitations. The presented approach favors the merging of more real obstacles into one detected obstacle, named **occupied area**, and then applies fragmentation criteria.

It is supposed that the obstacles do not overlap each other on the vertical direction. In other words, on a top view (figure 3.c) the obstacles are disjoint. Consequently, in what follows, the Y coordinate (the elevation) of the 3D points will be ignored and **all the processing is done using only the X and Z coordinates (top view)**.

Due to the perspective effect of the camera, further obstacles appear smaller in our images, providing fewer pixels, and therefore, less, sparser 3D reconstructed points in the 3D space. On the other hand, the error of the depth reconstruction increases with the distance too, which, contributes to the 3D points sparseness as well.

The obstacle detection algorithms, processing the 3D points, would work easier if they would receive constant density of the points, regardless the distance. One idea would be to artificially introduce 3D points in between the reconstructed ones in order to keep a constant density, but this approach neither enriches the real 3D data nor reduces the depth error, and it slows down the processing.

To counteract the problem of the points' density, a schema to divide the Cartesian top view space into tiles with constant density is proposed (figure 3). The horizontal field of view of the camera is divided into polar slices with constant aperture, trying to keep a constant density on the X-axis. The

depth range is divided into intervals, the length of each interval being bigger and bigger as the distance grows, trying to keep a constant density on the Z-axis.

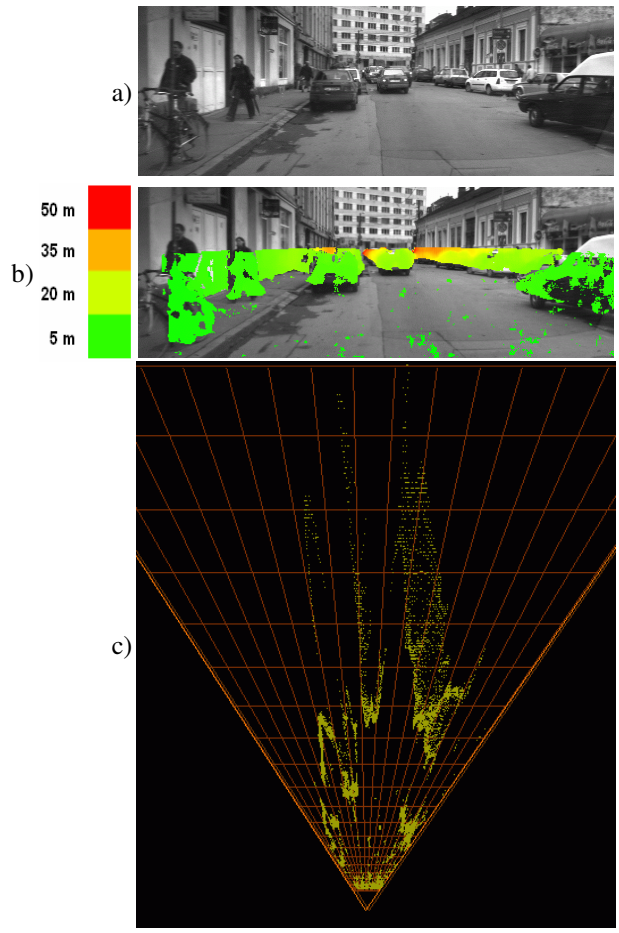


Figure 3. Division into tiles. a) Gray scale image, b) 3D points – perspective view, c) 3D points – top view; the tiles are here considerably larger for visibility purpose; wrong reconstructed points can be seen in random places; reconstruction error is visible as well.

A specially **compressed space** is created, as a matrix (figure 4.a). The cells in the compressed space correspond to the trapezoidal tiles of the Cartesian space. The compressed space is, in fact, a bi-dimensional histogram, each cell counting the number of 3D points found in the corresponding trapezoidal tile. For each 3D point, its cell in the compressed space is computed C(*Row*, *Column*), and the cell value is incremented.

The column formula is:

$Column = ImageColumn/c$

where *ImageColumn* is the left image column of the 3D point and c is the number of adjacent image columns grouped into a polar slice as shown in figure 3.c ($c = 6$).

The row of the compressed space, for a 3D point, has a formula obtained through a longer way:

The Cartesian interval corresponding to the first row of the compressed spaces is:

$[Z_0 \dots Z_0 + IntervalLength(Z_0)] = [Z_0 \dots Z_1]$

where $Z_0=Zmin$, a minimum detection depth and $IntervalLength(Z) = k*Z$ is the length of the interval beginning at a certain Z (further intervals are longer, k is empirically chosen). Thus

$Z_0 + IntervalLength(Z_0) = Z_0+k*Z_0 = Z_0*(1+k) = Z_1$.

The Cartesian interval corresponding to the second row of the compressed spaces is:

$[Z_1 \dots Z_1 + IntervalLength(Z_1)] = [Z_1 \dots Z_2]$

where $Z_1 = Z_0 + IntervalLength(Z_0)$, in other words the second interval begins where the first one ends:

$Z_1=Z_0*(1+k)$ and

$Z_1 + IntervalLength(Z_1) = Z_1+k*Z_1 = Z_1*(1+k) = Z_0*(1+k)\ (1+k) = Z_0*(1+k)^2 = Z_2$.

The ending of the second interval is the beginning of the third one:

$Z_2 = Z_0*(1+k)^2$

The Cartesian interval corresponding to the n^{th} row of the compressed spaces is:

$[Z_n \dots Z_n + IntervalLength(Z_n)]$, where

$Z_n=Z_0*(1+k)^n$ (provable by mathematical induction)

For a certain 3D point, having depth Z, the i^{th} interval it belongs to is $[Z_i \dots Z_i + IntervalLength(Z_i)] = [Z_i \dots Z_{i+1}]$. From the formula

$Z=Z_0*(1+k)^i$,

the i, as an integer number, is

$$Row = i = [\log_{1+k} \frac{Z}{Z_0}]$$

As a bottom line, each 3D point is transformed, from the Cartesian space to the compressed space, into the cell C(*Row, Column*).

There is one more remark for the building process of the compressed space: due to the fact that the used points are situated in a 3D band with a constant 3D height (along the Y-axis), as described in section III, that band, on the image space, has a decreasing height as the depth increases (see figure 3.b), having less 3D points respectively; to counteract this, the density of the cells is proportionally amplified with the depth.

The cells having no points represent free space. The cells having just a few points (less than 20 – empiric value) are also considered free, most probably that those points were wrong reconstructed. The other cells, having many points, reveal the existence of obstacles.

a)

b)

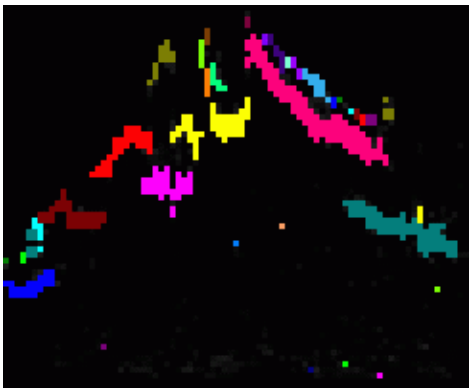


Figure 4. The compressed space (for scene in figure 3) – a bi-dimensional histogram counting 3D points (a). Groups of cells obtained by labeling (b)

The compressed space is used for occupied area detection. The detection is based on a labeling algorithm applied only on the cells having many points. The small groups are rejected. Most probably that the rejected groups come from 3D reconstruction errors generated by repetitive patterns or less textured image regions.

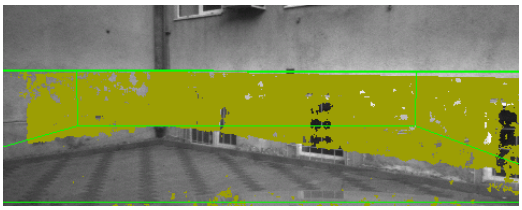
V. Fragmentation Of Occupied Areas Into Cuboidal Obstacles

The occupied areas may contain more obstacles and by consequence may have miscellaneous shapes (figure 4.b). The obstacle tracking algorithm [8] as well as the ADAS applications need the individual cuboidal obstacles. Therefore the fragmentation of the occupied areas into the individual cuboidal obstacles is required. Two fragmentation criteria are proposed:

A. CONCAVITY FREE SHAPES

The shape of a cuboidal obstacle has no concavities. The shape of the occupied area (figure 5.b) of a concave corner of a building (figure 5.a) is a relevant example (although it is particular, having right angle) and must be fragmented into two parts, one for each wall.

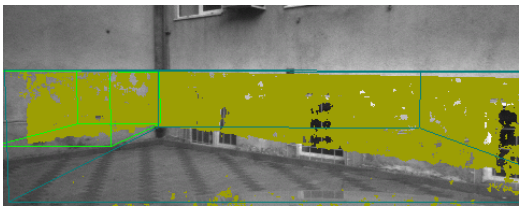
a)



b)

c)

d)



e)

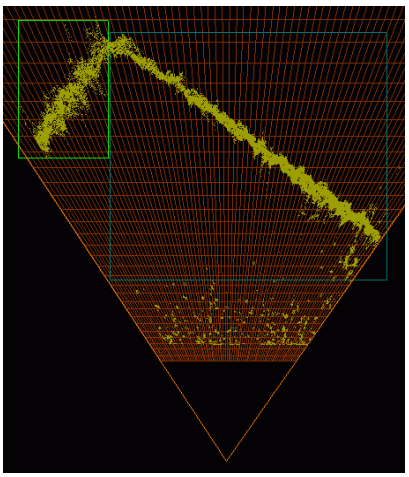


Figure 5. Fragmentation of a concave occupied area: a) free space included in the circumscribed cuboid, b) labeling in the compresses space, c) visible sides of the envelope of occupied area and the two subparts – compressed space, d) and e) the two circumscribed cuboids – perspective view and top view respectively

The solution is to determine the envelope of the cells of each occupied area, and then for each visible side (towards the camera) of the envelope (figure 5.c), the concavity between the side and the occupied cells is determined. If the concavity is large and deep, its deepest point gives the column (painted in red in figure 5.c) where the fragmentation will be done. The two subparts are subject to be fragmented again and again as long as they have significant concavities.

By reconsidering the 3D coordinates (including Y) of the points that have filled the cells of an occupied area, the limits of the circumscribing cuboid are determined (figure 5.d – perspective view, and figure 5.e – top view). At this point, the cuboids are parallel with coordinate axes and the occupied areas may be named "obstacles" even though the criterion *B* might fragment them again.

B. NO EMPTY CORNERS OF THE CUBOIDS

A real obstacle is confidently detected whether the space between the sides of its oriented cuboid (its determination is presented in the section VI) and the visible shape of the cloud of its 3D points (on the top view) has a small area. In other words the shape must have only quasi-90^o convexities. The visible free area in each corner of the cuboid is calculated in square meters, and, in order to be relevant (the relevance of the metric measurements decreases with the distance), it is transformed into the number of cells of the compressed top view space. When one of the corners has a free space of more than k=10 cells (an empirically chosen threshold), there must be performed a fragmentation into two or more sub-obstacles that have confident cuboids.

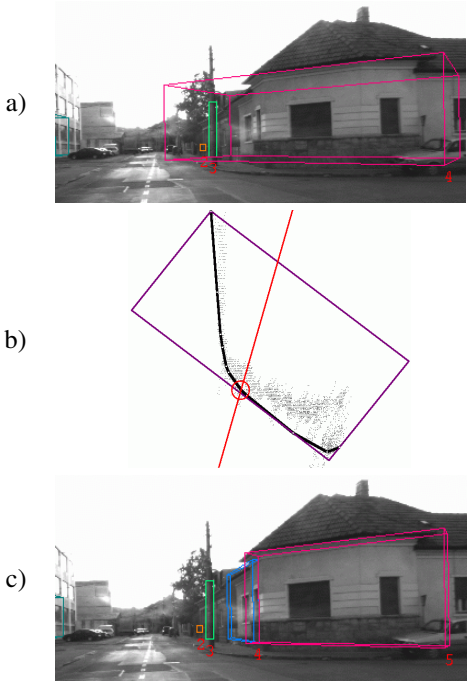


Figure 6. Fragmentation of an obstacle that doesn't have a cuboidal shape: a) initial cuboid, b) top view of the initial cuboid and the optical ray of the fragmentation, c) the two sub-obstacles obtained by fragmentation

An example is shown in figure 6.a and b, where no oriented cuboid would be confident. The fragmentation is done by using the optical ray that passes through one of the vertices of the free corner. The chosen vertex is the one that is the most interior in the obstacle. The procedure is recursively applied on the newly obtained (sub)obstacles as needed. The fragmentation of the large cuboid in figure 6.a is shown in figure 6.c.

VI. DETERMINING THE ORIENTATION OF OBSTACLES

In fig 7.a, it can be observed that, even though the real obstacle is oblique oriented, the cuboid encompasses some free space because the cuboid is parallel with the coordinate system. The shape of the cloud of 3D points is modeled by their envelope

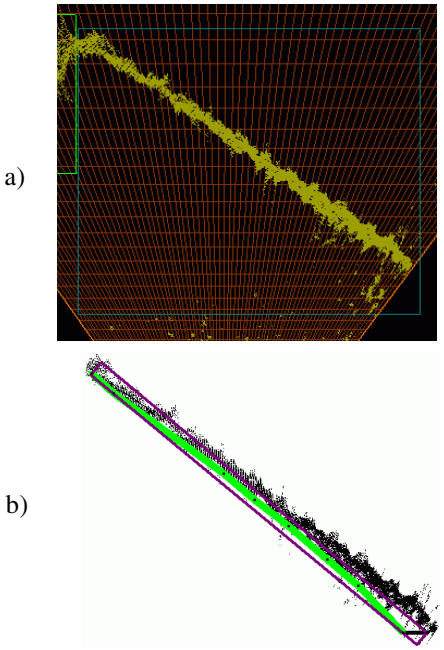


Figure 7. Obstacle orientation (top view): a) un-oriented box, b) the longest chain of visible envelope sides (green) and its surrounding rectangle

and an analysis of its visible sides (towards the camera) can determine the orientation of the obstacle. If the analysis cannot determine a preponderant orientation of these sides, the box remains parallel with the axes of the coordinates system; it is un-oriented.

In figure 7.b such an envelope is shown. The algorithm searches chains of consecutive sides having a low standard deviation of the slope of its components. If the length of the longest chain (the thick green one) is at least 70% from the length of the visible envelope, the obstacle orientation is the weighted average of the slope of the chain's sides. As weights we use here the lengths of the sides.

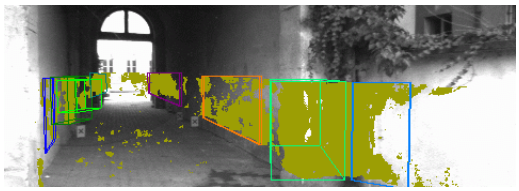
A rectangle with the found orientation is circumscribed on the visible sides and gives the base (top view – figure 7.b) of the oriented cuboid of the obstacle. The perspective result is shown in figure 8.b. Note: the envelope is not quite convex here, due to some approximations for speed up.

VII. Results

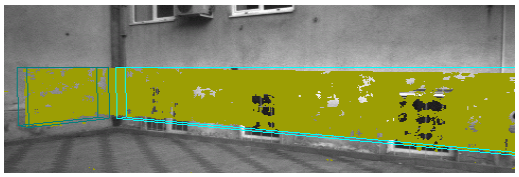
The horizontal field of view of the cameras is about 70° and the resolution is of 512x383 allowing the stereo reconstruction to work fine up to 50 meters. The tests on thousand of images have shown that the approach is a robust one. The real-time performance is of 20 frames/second, for the whole application, when a P4, 2.6 GHz computer is used and the dense stereo correlation is done by hardware.

The figures used in this paper represent only the region of interest consisting of a centered horizontal band.

a)



b)



c)

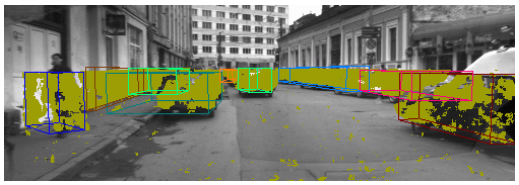


Figure 8. Results. a) the orientation of the second obstacle from the right couldn't be determined, b) an ideal case, c) due to merging of the pedestrian with the gate behind (left), the orientation is strange.

VIII. Conclusions

The proposed obstacle detection approach is original due to the direct use of the 3D reconstructed space and due to the divers geometric reasoning carried out on this space.

The local density and vicinity of the 3D points is analyzed in a special compressed top view space. The compressed space keeps a constant density of the 3D points and neutralizes the error of the reconstructed depth. On the compressed space, a labeling algorithm is applied, on the cells with high density of points, determining the occupied areas.

The occupied areas are fragmented into obstacles that are suitable for the cuboidal model. For this fragmentation, the visible shape (towards the camera) of the 3D points is analyzed and two criteria are used: the shape must have no concavities and the cuboid fitted on the shape must not contain significant free (drivable) space.

The orientation of the obstacles is determined, when possible.

Most of the processing is done on the compressed space which concentrates the useful information of the set of 3D points and leads to a fast computation.

Due to stereo reconstruction errors and limitations, the instantaneous results can differ in successive frames. Thus, a tracking algorithm is needed in order to filter the noise, to reject bad detection and to fill in the detection gaps.

IX. References

[1] M. Bertozzi, A. Broggi, "Real-time lane and obstacle detection on the GOLD system", *Intelligent Vehicles Symposium 1996*, pp. 213-218.

[2] R. Labayrade, D. Aubert, J.-P. Tarel, "V-disparity on Non Flat Roads", *Intelligent Vehicles Symposium 2002*, pp. 646-651.

[3] Z. Hu, K. Uchimura, "U-V-Disparity - An efficient algorithm for Stereovision Based Scene Analysis", *Intelligent Vehicles Symposium 2005*, pp. 48-56.

[4] C. Rossig, R. Hermann, M. Hotter, "Continuous Estimation of the road plane for mesuring obstacles", *Sixth Intern. Conf. on Image Processing and its Applications 1997*, vol. 1, pp. 394-398.

[5] S. Nedevschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, R. Schmidt, T. Graf "High accuracy stereo vision system for far distance obstacle detection", IEEE Intelligent Vehicles Symposium, Parma (IV'04), Italy, June 14-17, 2004, pp. 292-297.

[7] Jean-Yves Bouguet (2003), "Camera Calibration Toolbox for Matlab", MRL – Intel Corp., http://www.vision.caltech.edu/bouguetj/calib_doc/.

[7] F. Oniga, S. Nedevschi, M-M. Meinecke, T-B. To (2007), "Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo", *International IEEE Conference on Intelligent Transportation Systems 2007*, Seattle, USA, Sept. 30 – October 3, 2007, pp. 401-404.

[8] R. Danescu, S. Nedevschi, M.M. Meinecke, T. Graf, "Stereovision Based Vehicle Tracking in Urban Traffic Environments", *International IEEE Conference on Intelligent Transportation Systems 2007*, Seattle, USA, Sept. 30 – October 3, 2007, pp. 271-276.