Real Time Vehicle Pose Using On-Board Stereo Vision System*

Angel D. Sappa, David Gerónimo, Fadi Dornaika, and Antonio López

Computer Vision Center
Edifici O Campus UAB
08193 Bellaterra, Barcelona, Spain
{asappa, dgeronimo, dornaika, antonio}@cvc.uab.es

Abstract. This paper presents a robust technique for a real time estimation of both camera's position and orientation—referred as pose. A commercial stereo vision system is used. Unlike previous approaches, it can be used either for urban or highway scenarios. The proposed technique consists of two stages. Initially, a compact 2D representation of the original 3D data points is computed. Then, a RANSAC based least squares approach is used for fitting a plane to the road. At the same time, relative camera's position and orientation are computed. The proposed technique is intended to be used on a driving assistance scheme for applications such as obstacle or pedestrian detection. Experimental results on urban environments with different road geometries are presented.

1 Introduction

In recent years, several vision based techniques were proposed for driving assistance. According to the targeted scenario, they can be broadly classified into two different categories: highways and urban. Most of the techniques proposed for highways environments are focused on lane and car detection, looking for an efficient driving assistance system. On the other hand, in general, techniques for urban environments are focused on collision avoidance or pedestrian detection. Although in both domains a similar objective is pursued, it is not possible to develop a general purpose solution able to cope with both problems.

The prior knowledge of the environment is a source of information generally involved in the proposed solutions. For instance, highway driving assistance systems are based on assumptions such as:

- The vehicle is driven along two parallel lane markings, these two lane markings are projected to the left and to the right of an image, respectively [1].
- Lane markings, or the road itself, have a constant width [2].
- A perfectly flat road, or in other words, the camera's position and pitch angle are constant values [3].

* This work was partially supported by the Government of Spain under the CICYT project TRA2004-06702/AUT. The first and third authors were supported by The Ramón y Cajal Program. The second author was supported by Spanish Ministry of Education and Science grant BES-2005-8864.

A. Campilho and M. Kamel (Eds.): ICIAR 2006, LNCS 4142, pp. 205–216, 2006.

Similarly, vision-based urban driving assistance systems, also propose to use the prior knowledge of the environment to simplify the problem. Some of the aforementioned assumptions (e.g., flat road [4]) are also used on urban environment, together with additional assumptions related to urban scenes:

- A number of pedestrians appears simultaneously in the image but they do not occlude each other [5].
- Thermal characteristic of body's part (e.g., the head is often warmer than the body [6], stereo night vision [7]).
- An area of interest is defined in a central region of the image [8], where pedestrians are more likely to be found and the detection is more useful in order to apply avoiding strategies.
- A pedestrian has only vertical edges [9].

In summary, scene's prior knowledge has been extensively used to tackle the driving assistance problem. However, making assumptions not always can help to solve problems; some times, it may provide erroneous results. For instance, constant camera's position and orientation, which is a generally used assumption on highways, is not so valid in an urban scenario. In the latter, camera's position and orientation are continuously modified by factors such as: road imperfections or artifacts (e.g., rough road, speed bumpers), car's accelerations, uphill/downhill driving, among others. Facing up to this problem [2] introduces a technique for estimating vehicle's yaw, pitch and roll. It is based on the assumption that some parts of the road have a constant width (e.g., lane markings). Similarly, [1] proposes to estimate camera's orientation by assuming that the vehicle is driven along two parallel lane markings. Unfortunately, none of these two approaches can be used for an urban scenario, since lanes are not as well defined as those of highways.

Having in mind the mentioned problem, [10] proposes a feature-based approach to compute 3D information from a stereo vision system. Features such as zebra crossings, lane markings or traffic signs painted on the road (e.g., arrows, forbidden zones) are used. Differently to classical approaches, this scheme classifies each pixel according to the grey values of its neighbors. Then, points lying on the road are used to estimate camera's position and orientation. Although presented results are quite promising, the traffic-based-feature detection constraint is one of the main disadvantages of this technique. Moreover, not every urban road contains features such as the aforementioned.

A different approach was presented in [11]. The authors propose an efficient technique able to cope with uphill/downhill driving, as well as dynamic pitching of the vehicle. It is based on a v-disparity representation and Hough transform. The authors propose to model not only a single plane road—a straight line—but also a non-flat road geometry—a piecewise linear curve. This method is also limited since a longitudinal profile of the road should be extracted for computing the v-disparity representation.

More recently, [12] has introduced a specific image stabilization technique for pitch angle compensation. It is based on the study of a row-wise histogram computed from the edges of the current image. Histograms from consecutive

frames are used to compute their corresponding vertical offset. This approach, although very efficient in terms of computing time, has two important drawbacks. First of all, the image should contain several horizontal features, since the whole process relies on the accumulation of horizontal edges. Secondly, current camera orientation is related to the previous frame, therefore, since relative errors can not be removed, the global error value increases with the time—the drift problem.

In this paper, a new approach based on 3D data computed by a commercial stereo vision system is presented. It aims to compute camera's position and orientation, avoiding most of the assumptions mentioned above. The proposed technique consists of two stages. Initially, the original 3D data points are mapped onto a 2D space. Then, a RANSAC based least squares fitting is used to estimate the parameters of a plane fitting to the road; at the same time camera's position and orientation are directly computed, referred to that plane. Independently of the road geometry, the provided results could be understood as a piecewise planar approximation, due to the fact that road and camera parameters are continuously computed and updated. The proposed technique could be indistinctly used for urban or highway environments, since it is not based on a specific visual traffic feature extraction but on raw 3D data points.

The remainder of this paper is organized as follow. Section 2 briefly describes the stereovision system. Section 3 presents the proposed technique. Experimental results on urban scenes are presented in Section 4. Finally, conclusions and further improvements are given in Section 5.

2 Stereovision System

A commercial stereo vision system (Bumblebee from Point Grey[1]) was used. It consists of two Sony ICX084 color CCDs with 6mm focal length lenses. Bumblebee is a pre-calibrated system that does not require in-field calibration. The baseline of the stereo head is 12cm and it is connected to the computer by a IEEE-1394 connector. Right and left color images were captured at a resolution of 640x480 pixels and a frame rate near to 30 fps. After capturing these right and left images, 3D data were computed by using the provided 3D reconstruction software. Fig. 1 shows an illustration of the on board stereo vision system.

3 Proposed Technique

Let $S(r, c)$ be a stereo image with R rows and C columns, where each array element $(r, c)(r \in [0, (R-1)]$ and $c \in [0, (C-1)])$ is a scalar that represents a surface point of coordinates (x, y, z), referred to the sensor coordinate system. Fig. 1 depicts the sensor coordinate system attached to the vehicle's windshield. Notice that vertical variations between consecutive frames—due to road imperfections, car accelerations, changes in the road slope: *flat/uphill/downhill*

[1] `[www.ptgrey.com]`

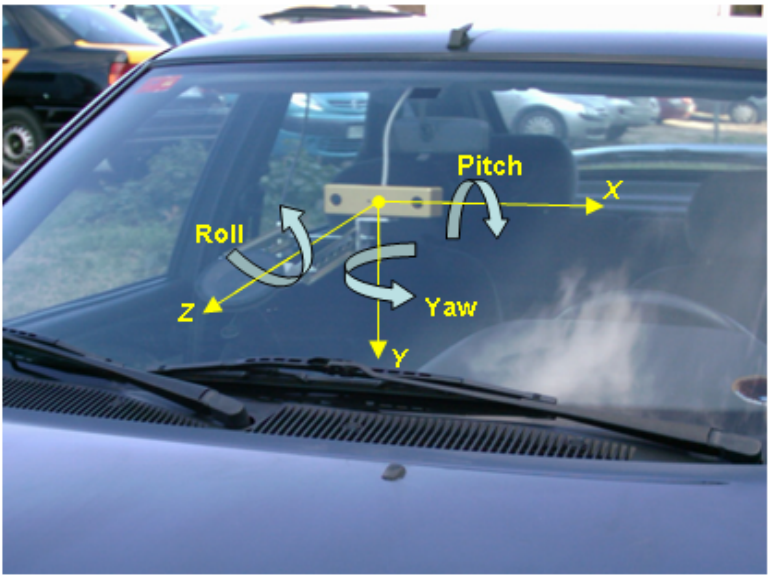

Fig. 1. On board stereo vision sensor with its corresponding coordinate system

driving, etc—will mainly produce changes on camera's position and pitch angle. In other words, yaw and roll angles are not so affected by those variations. Therefore, without loss of generality, their changes are not considered.

The proposed approach consists of two stages. Initially, 3D data points are mapped onto YZ plane (see Fig. 1). In a second stage, a subset of those mapped points is used for fitting a plane to the road. Finally, camera's position and orientation are directly computed, referred to that plane. Both stages are further detailed below.

3.1 3D Data Point Projection and Noisy Data Filtering

The aim at this stage is to find a compact subset of points, ζ, containing most of the road's points. Additionally, noisy data points should be reduced as much as possible in order to avoid both a very time consuming processing and a wrong plane fitting.

Assuming null yaw and roll angles, original 3D data points, (x_i, y_i, z_i), are mapped onto a 2D discrete representation $P(u, v)$; where $u = (round)(y_i \cdot \sigma)$ and $v = (round)(z_i \cdot \sigma)$. σ represents a scale factor defined as: $\sigma = ((R + C)/2)/((\Delta X + \Delta Y + \Delta Z)/3)$; R, C are the image's rows and columns respectively, and Δs is the working range in every dimension—on average (34x12x50)meters. Every cell of $P(u, v)$ keeps a pointer to the original 3D data point projected onto that position, as well as a counter with the number of mapped points. Fig. 2*(top)* shows a 2D representation obtained after mapping a 3D cloud—every black point represents a cell with at least one mapped point. Notice the large amount of noisy points highlighted on the figure.

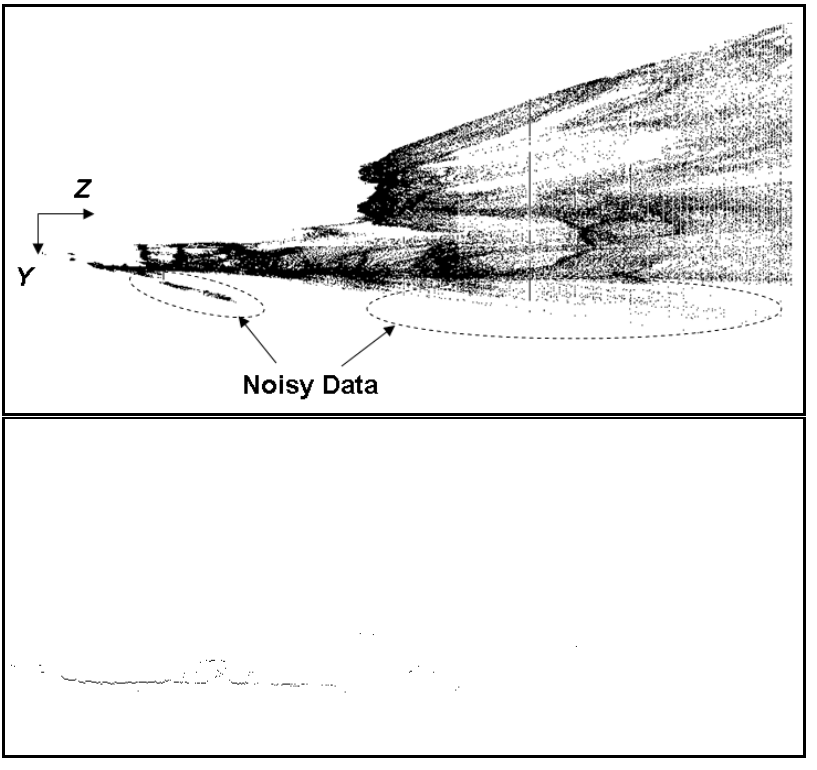

Fig. 2. YZ projection. (*top*) The whole mapped cloud of points. (*bottom*) Selected points to be used by the RANSAC technique.

Finally, points defining the ζ subset are selected as follow. Firstly, those cells of $P(u, v)$ containing less mapped points than a predefined threshold are filtered, by setting to zero its corresponding counter—points mapped onto those cells are considered as noisy data. The threshold value was experimentally set as a percentage of the maximum amount of points mapped onto a cell. On average, every cell of $P(u, v)$ corresponds to an area of (5.6cmx5.6cm) of the original cloud of points; the maximum amount of points mapped onto a cell is about 250 points. The threshold value was set 6% of that value—i.e., 15 points. After filtering noisy data, a selection process picks one cell per column. It goes bottom-up through every column and picks the first cell with more points than the aforementioned threshold. 3D data points mapped onto selected cells define the sought subset of points, ζ. Fig. 2*(bottom)* depicts cells finally selected. The ζ subset of points gathers all the 3D points mapped onto those cells.

3.2 RANSAC Based Plane Fitting

The outcome of the previous stage is a subset of points, ζ, where most of them belong to the road. In the current stage a RANSAC based technique [13] is

used for fitting a plane to those data[2], $ax + by + cz = 1$. In order to speed up the process, a predefined threshold value for inliers/outliers detection has been defined (a band of ±5cm was enough for taking into account both 3D data point accuracy and road planarity). An automatic threshold could be computed for inliers/outliers detection following robust estimation of standard deviation of residual errors [14].

The proposed plane fitting works as follow.

Random sampling: *Repeat the following three steps K times or up to the current plane parameters are similar to those ones computed from the previous frame (during the first iteration this second condition is not considered)*

1. Draw a random subsample of n different 3D points from ζ (looking for a trade-off between CPU processing time and final result, n has been finally set to 3).
2. For this subsample, indexed by $k(k = 1, ..., K)$, compute the plane parameters (a, b, c).
3. For this solution $(a, b, c)_k$, compute the number of inliers among the entire set of 3D points contained in ζ, as mentioned above using ±5cm as a fixed threshold value.

Solution:

1. Choose the solution that has the highest number of inliers. Let $(a, b, c)_i$ be this solution.
2. Refine $(a, b, c)_i$ considering its corresponding inliers, by using the least squares fitting approach [15], which minimize the square residual error $(1 - ax - by - cz)^2$.
3. In case the number of inliers is smaller than 10% of the total amount of points contained in ζ, those plane parameters are discarded and the ones corresponding to the previous frame are used as the correct ones. In general, this happens when 3D road data are not correctly recovered since occlusion or other external factor appears.

Finally, camera's position and orientation, referred to the fitted plane, are easily computed. Camera's position—height—is estimated from the plane intersection with the Y axis $(1/b)$ and the current plane orientation. Camera's orientation—pitch angle—is computed from the current plane orientation.

4 Experimental Results

The proposed technique has been tested on different urban environments. More than two hours of stereo video sequence were processed on a 3.2 GHz Pentium IV PC with a non-optimized C++ code. The proposed algorithm took, on average, 350 ms per frame. Fig. 3 presents variations in the camera's position

[2] Notice that the general expression $ax + by + cz + d = 0$ has been simplified dividing by $(-d)$, since we already have $(d \neq 0)$.

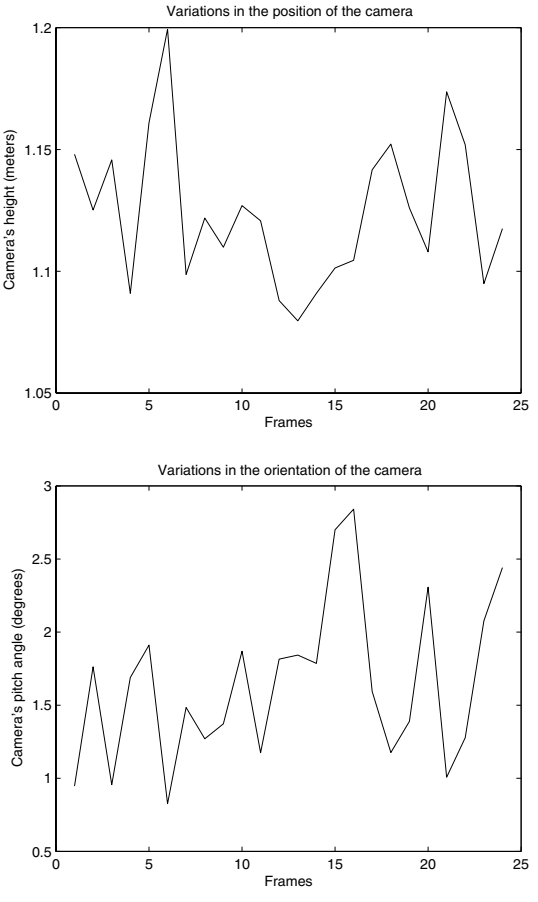

Fig. 3. Variations in the camera's position and orientation, related to the current plane fitting the road. Note that only 2 fps are plotted.

and orientation during a sequence of about one minute long—only variations in the camera height position and pitch angle, both related to the current fitted plane, are depicted. Notice that this short video sequence was recorded on a quite flat road; hence a maximum height variation of about 10 cm is produced after starting the motion, due to vehicle's acceleration. On the other hand, an angle pitch's variation of less than 2 degrees is produced during the vehicle trajectory.

Fig. 4 presents another sequence (4.8 minutes long) of an urban environment containing a gentle downhill, vehicle's accelerations and a speed bumper. This illustration shows that variations in the camera's position and orientation cannot be neglected.

Since ground truth is not known beforehand, several frames were randomly chosen and used to check the obtained results. In these cases, their corresponding

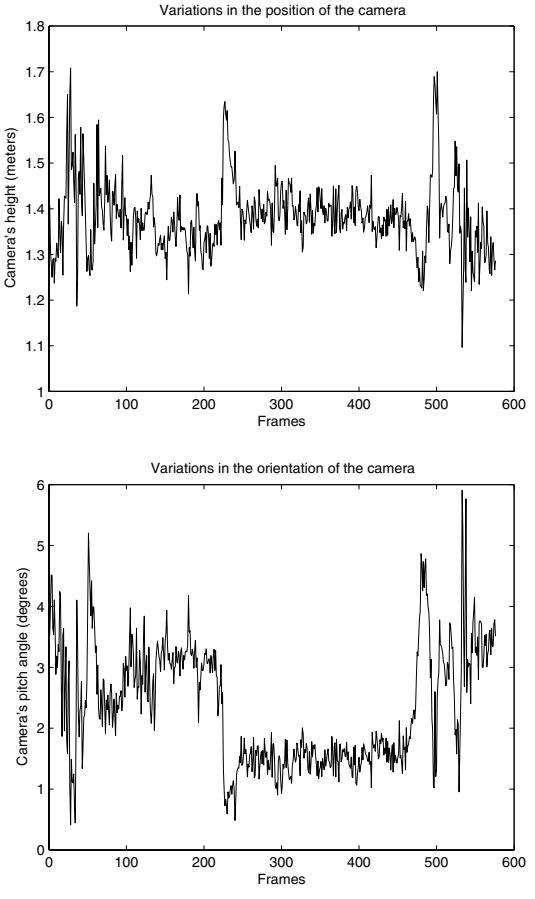

Fig. 4. Variations in the camera's position and orientation, related to the current plane fitting the road. Note that only 2 fps are plotted.

vanishing lines, also referred to as horizon line, were manually computed[3] and used as ground truth to compare with the ones automatically obtained, by using the information provided by the proposed technique. A vanishing line is automatically computed by placing a point over the road, far away from the current camera reference system (e.g., 3000m). By using the camera's position and orientation, together with the corresponding focal length (Section 2), that point in the 3D space is back-projected to the 2D image space, so that its corresponding row position can be used as the vanishing line.

Fig. 5 shows three frames where different scenarios can be tested—uphill, flat and downhill driving. Left column corresponds to horizon lines manually

[3] By drawing two parallel lines in the 3D space and getting their intersection in the image plane.

Fig. 5. Illustrations of vanishing lines manually estimated (*a*) and automatically computed by the proposed technique (*b*). (*top*) Uphill driving illustration. (*middle*) Flat road. (*bottom*) Downhill driving scene.

computed, while right column presents the ones automatically computed. In both cases similar positions were obtained—in the worst case a maximum difference of 6 pixels was obtained with the flat road scene. Notice that assuming a constant vanishing line position (i.e., based on a flat road and constant camera's position and orientation) is not a right choice, since as it is shown in these examples, it can move in a range of about 100 pixels. Moreover, it could drive to wrong results as it will be presented below.

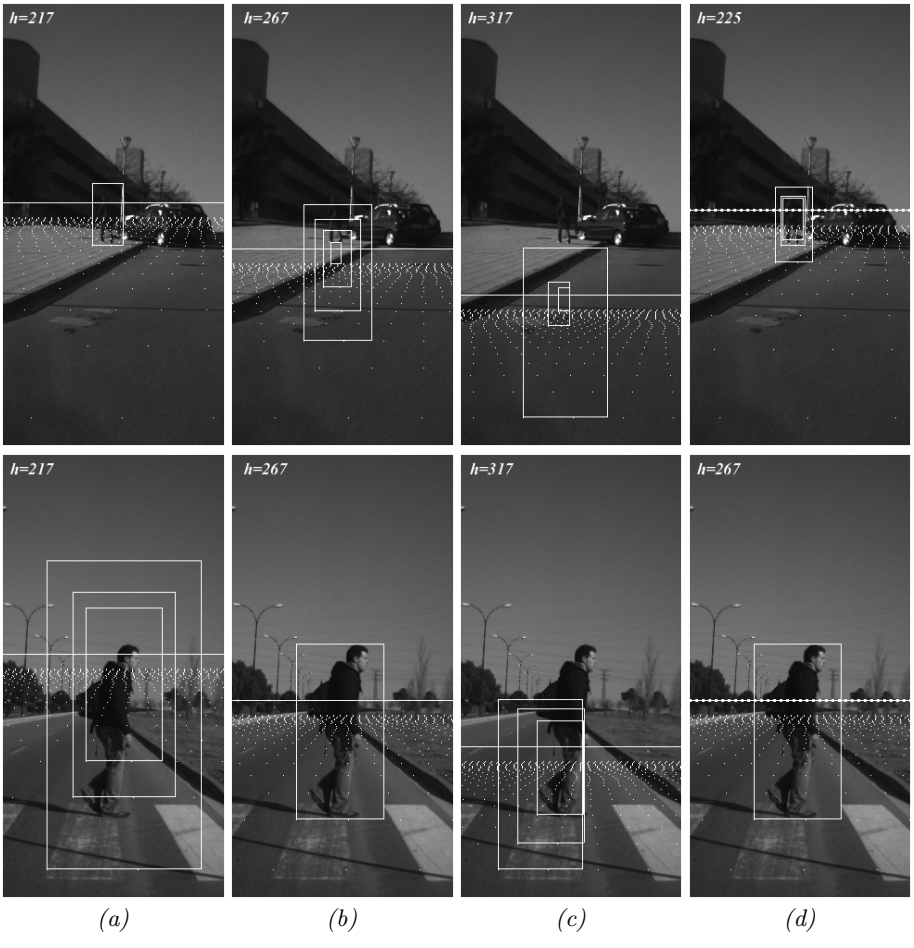

Fig. 6. Searching bounding boxes using fixed and automatically computed vanishing lines. In all the cases only very few bounding boxes are highlighted. (*a*) Fixed vanishing line for an uphill driving scenario. (*b*) Fixed vanishing line assuming a flat road. (*c*) Fixed vanishing line for a downhill driving. (*d*) Automatically computed vanishing line by using the proposed technique. Notice as, only in the latter case, the vanishing line position is correctly placed in both scenarios.

The proposed technique is already being used on a shape-based pedestrian detection algorithm in order to speed up the searching process. Although out of the scope of this paper, Fig. 6 presents illustrations of two different scenarios showing the importance of having the right estimation of camera's position and orientation. In these illustrations, *(a)*, *(b)* and *(c)* columns show results by using three different, but constant, vanishing line positions, while *(d)* column depicts the corresponding results obtained by using a vanishing line position automatically computed by the proposed technique. Following the algorithm presented in [16], a 3D grid, sampling the road plane, is projected on the 2D image. The

projected grid nodes are used as references to define the bottom-left corners of pedestrian sized searching windows. These windows, which have a different size according to their corresponding 3D position, move backward and forward over the assumed plane looking for a pedestrian-like shape. Therefore, a wrong road plane orientation—i.e., vanishing line—drives to a wrong searching space, so that the efficiency of the whole algorithm decreases. A few searching bounding boxes are highlighted on Fig. 6 to show their changes in size according to the distance to the camera.

5 Conclusions and Further Improvements

An efficient technique for a real time pose estimation of on-board camera has been presented. After an initial mapping and filtering process, a compact set of points is chosen for fitting a plane to the road. The proposed technique can fit very well to different road geometries, since plane parameters are continuously computed and updated. A good performance has been shown in several scenarios—uphill, downhill and a flat road. Furthermore, critical situations such as car's accelerations or speed bumpers were also considered. Although it has been tested on urban environments, it could be also useful on highways scenarios.

Further work will be focused on developing new strategies in order to reduce the initially chosen subset of points; for instance by using a non-constant cell size for mapping the 3D world to 2D space (through the optical axis). A reduced set of points will help to reduce the whole CPU time. Furthermore, the use of Kalman filtering techniques and other geometries for fitting road points will be explored.

References

1. Liang, Y., Tyan, H., Liao, H., Chen, S.: Stabilizing image sequences taken by the camcorder mounted on a moving vehicle. In: Procs. IEEE Intl. Conf. on Intelligent Transportation Systems, Shangai, China (2003) 90–95
2. Coulombeau, P., Laurgeau, C.: Vehicle yaw, pitch, roll and 3D lane shape recovery by vision. In: Proc. IEEE Intelligent Vehicles Symposium, Versailles, France. (2002) 619–625
3. Bertozzi, M., Broggi, A.: GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection. IEEE Trans. on Image Processing **7**(1) (1998) 62–81
4. Bertozzi, M., Broggi, A., Chapuis, R., Chausse, F., Fascioli, A., Tibaldi, A.: Shape-based pedestrian detection and localization. In: Procs. IEEE Intl. Conf. on Intelligent Transportation Systems, Shangai, China (2003) 328–333
5. Bertozzi, M., Broggi, A., Fascioli, A., Graf, T., Meinecke, M.: Pedestrian detection for driver assistance using multiresolution infrared vision. IEEE Trans.on Vehicular Technology **53**(6) (2004) 1666–1678
6. Bertozzi, M., Broggi, A., Lasagni, A., Del Rose, M.: Infrared stereo vision-based pedestrian detection. In: Procs. IEEE Intelligent Vehicles Symposium, Las Vegas, USA (2005) 24–29

7. Liu, X., Fujimura, K.: Pedestrian detection using stereo night vision. IEEE Trans. on Vehicular Technology **53**(6) (2004) 1657–1665
8. Broggi, A., Bertozzi, M., Fascioli, A., Sechi, M.: Shape-based pedestrian detection. In: Procs. IEEE Intelligent Vehicles Symposium, Dearborn, USA (2000) 215–220
9. Lefée, D., Mousset, S., Bensrhair, A., Bertozzi, M.: Cooperation of passive vision systems in detection and tracking of pedestrians. In: Proc. IEEE Intelligent Vehicles Symposium, Parma, Italy. (2004) 768–773
10. Franke, U., Gavrila, D., Görzig, S., Lindner, F., Paetzold, F., Wöhler, C.: Autonomous driving goes downtown. IEEE Intelligent Systems and Their Applications (1998) 40–48
11. Labayrade, R., Aubert, D., Tarel, J.: Real time obstacle detection in stereovision on non flat road geometry through "V-disparity" representation. In: Proc. IEEE Intelligent Vehicles Symposium, Versailles, France. (2002) 646–651
12. Bertozzi, M., Broggi, A., Carletti, M., Fascioli, A., Graf, T., Grisleri, P., Meinecke, M.: IR pedestrian detection for advaned driver assistance systems. In: Procs. 25th. Pattern Recognition Symposium, Magdeburg, Germany (2003) 582–590
13. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Graphics and Image Processing **24**(6) (1981) 381–395
14. Rousseeuw, P., Leroy, A.: Robust Regression and Outlier Detection. John Wiley & Sons, New York (1987)
15. Wang, C., Tanahashi, H., Hirayu, H., Niwa, Y., Yamamoto, K.: Comparison of local plane fitting methods for range data. In: Proc. IEEE Computer Vision and Pattern Recognition, Hawaii. (2001) 663–669
16. Ponsa, D., López, A., Lumbreras, F., Serrat, J., Graf, T.: 3D vehicle sensor based on monocular vision. In: Procs. IEEE Intl. Conf. on Intelligent Transportation Systems, Vienna, Austria (2005) 1096–1101