

# Global Map Building Based on Occupancy Grids Detected from Dense Stereo in Urban Environments

Florin Oniga<sup>1</sup>, Sergiu Nedevschi<sup>1</sup>, Radu Danescu<sup>1</sup>, Marc-Michael Meinecke<sup>2</sup>

<sup>(1)</sup> *Technical University of Cluj-Napoca, Romania*  
{florin.oniga, sergiu.nedevschi, radu.danescu}@cs.utcluj.ro

<sup>(2)</sup> *Volkswagen AG, Electronic Research, Wolfsburg, Germany*  
marc-michael.meinecke@volkswagen.de

## Abstract

*A method for global map building from occupancy grids is presented in this paper. Occupancy grids provide a low-level representation of the environment, suitable for autonomous navigation tasks, in urban driving scenarios. The occupancy grids used in our approach are computed with a method that outputs an occupancy grid with three distinct cell types: road, traffic isles and obstacles. First, we perform a temporal filtering of the false traffic isles present in the grids. Obstacle cells are separated into static (probably infrastructure) and dynamic. An enhanced occupancy grid is built, containing road, traffic isle, static obstacle and dynamic obstacle cells. The global map is obtained by integrating the enhanced occupancy grid along several successive frames. It can be used in various ways, such as alignment with external maps, or for terrain mapping.*

## 1. Introduction

Most of the global mapping applications are oriented towards the robotics field, with either indoor mapping or outdoor (terrain) mapping. Existing work is oriented toward 3D data alignment and registration for scenes with static content, without dynamic obstacles. Urban traffic scenarios present many dynamic scene items (vehicles, pedestrians, etc).

In [2] the problem of outdoor unstructured terrain mapping and navigation is approached. A robot equipped with a stereo vision system is used for data acquisition. A combination of visual odometry, inertial

ego sensors and GPS is proposed for computing the rotation and translation between the reference systems of successive measurements.

A stereovision guided robot that can plan paths, build maps and explore an indoor environment is presented in [3]. Occupancy grids are used for path computation and map building. Results with the mapping of an indoor room are presented.

In [4] a stereo-based approach for building 3D maps is presented. First, the best local alignment between successive point clouds is computed. Then, a quasi-random updating algorithm minimizes the global inconsistency of the map. The system is tested by performing several indoor mapping experiments.

Digital elevation maps (DEM) can be used to represent and process 3D data from stereovision. They provide a compact representation, with a smaller size and complexity than the coarse set of 3D points.

A complex method for building the digital elevation map of a terrain (for a planetary rover) is proposed in [5]: local planar surfaces are used to filter the height of each DEM cell, and the stereo correlation confidence for each 3D point is included in the filtering process. In [6] the elevation map is built straightforward from the disparity map. The authors avoid using a 3D representation of the reconstructed points by projecting a vertical 3D line for each DEM cell onto the left, disparity, and right image. Based on these projections, the disparity of the point associated with the cell is selected and possible occlusions are detected. The obtained DEM is used for a global path-planning algorithm.

The road, traffic isle and obstacle detection

algorithm presented in [1] uses digital elevation maps (DEM) to represent 3D dense stereo data. The DEM is enhanced based on the depth uncertainty and resolution models of the stereo sensor. A RANSAC-approach, combined with region growing, is used for the detection of the optimal road surface. Obstacles and traffic isles are detected by using the road surface and the density of 3D points. This algorithm outputs the road surface parameters and an occupancy grid (Fig. 1) with three distinct cell types: road, traffic isles and obstacles. The fact that parts of the 3D space ahead are reconstructed from multiple stereo pairs (in successive frames) is not taken into account for results enhancement.

In this paper we perform temporal integration of the occupancy grids computed with the algorithm from [1]. The result is a global map of the ego vehicle's environment. This map provides occupancy information with an increased trust degree compared to the occupancy grids computed from individual frames. In contrast with existing approaches, urban driving scenarios present an additional complexity issue: dynamic obstacles. Therefore, obstacle cells are first separated into static / dynamic obstacles. In section 4 we present how the enhanced occupancy grid is built, containing road, traffic isle, static obstacle and dynamic obstacle cells. In section 5 we present how a global occupancy grid is built (integration of the occupancy grid along several successive frames).

In the next section (2) we present the motion model that we use for computing the ego car's translation and rotation between successive frames.

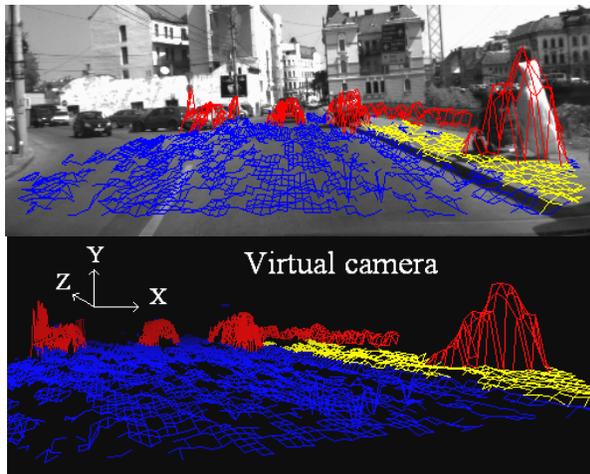


Fig.1. The occupancy grid detected by the algorithm, displayed in 3D. Blue for road, red for obstacles and yellow for traffic isles.

## 2. Ego motion compensation between successive frames

Most of the enhancements we propose in this paper are based on temporal integration. Next, we describe the simple model used for estimating the translation and rotation of coordinates system (ego car related) in time.

Coordinates from the current reference frame  $O(t)$  can be transformed straightforward into the previous reference frame  $O(t-1)$ , assuming the translation  $d$  and rotation angle  $\alpha$  are known (Fig. 2).

The ego car's standard speed and yaw-rate sensors can be used to estimate these parameters. The following motion model was used: the ego has a circular trajectory between successive frames, and the arc length and radius are computed based on the ego car speed  $v$ , yaw-rate value  $\gamma$  and frame relative timestamp  $\Delta t$ . The yaw-rate sensor provides the rotation angle, and the translation is computed geometrically:

$$\alpha = \gamma \cdot \Delta t , \quad (1)$$

The arc length is computed based on the ego speed and frame relative timestamp:

$$L = v \cdot \Delta t \quad (2)$$

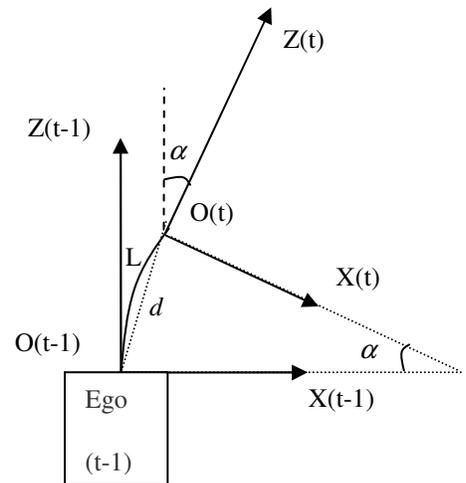


Fig. 2. Ego motion is expressed as the Euclidian distance between origins and relative angle between axes.

The radius of the circular trajectory is:

$$R = \frac{L}{\alpha} \quad (3)$$

Finally, the length of the chord associated with the arc is:

$$d = 2 \cdot R \cdot \sin\left(\frac{\alpha}{2}\right) = 2 \cdot \frac{v \cdot \Delta t}{\alpha} \cdot \sin\left(\frac{\alpha}{2}\right) \quad (4)$$

Beside the yaw angle, the pitch and roll angles might also influence the temporal integration of the occupancy grids. The roll angle might have a small influence for several consecutive frames but for large distances it will always integrate to zero (roads are normally horizontal on the lateral direction). The pitch angle, on the other hand, has larger influences over far distances because it describes the local road geometry. For uphill or down hills it cannot be ignored.

For pitch computation we used the ego motion detection algorithm presented in [7], although a more accurate way would be the use of a gyro sensor. The rotation matrix  $R_t$  used for aligning the reference systems of successive frames ( $O(t)$  and  $O(t-1)$ ) is built based on the relative yaw (1) and the relative pitch angle. The translation  $T_t$  is estimated with (2) and then its components on each axis are computed.

Along a sequence of frames (starting with frame 0), the global rotation matrix  $R_w$  and the global translation vector  $T_w$  must be computed for each new frame. A 3D point  $P_t$  from the current frame  $O(t)$  is represented in the first frame  $O(0)$  of the sequence by  $P_0$ .

$$P_0 = R_{wt} P_t + T_{wt}, \quad (5)$$

where  $R_{wt}$  and  $T_{wt}$  are the global rotation matrix and translation vector between the current reference frame  $O(t)$  and the global (first frame) reference frame  $O(0)$ .

The point  $P_t$  transformed in the previous reference frame  $O(t-1)$  is:

$$P_{t-1} = R_t P_t + T_t \quad (6)$$

The following relationship exists between  $P_{t-1}$  and  $P_0$ :

$$P_0 = R_{wt-1} P_{t-1} + T_{wt-1} \quad (7)$$

By replacing  $P_{t-1}$  from (6) into (7):

$$P_0 = R_{wt-1} R_t P_t + R_{wt-1} T_t + T_{wt-1} \quad (8)$$

Since (5) and (8) must be equivalent, the global rotation and translation for the current frame  $O(t)$  can be computed from the global rotation and translation of the previous frame  $O(t-1)$ :

$$R_{wt} = R_{wt-1} R_t, \quad T_{wt} = R_{wt-1} T_t + T_{wt-1} \quad (9)$$

For the first frame,  $t=0$ , the global rotation is the identity matrix and the global translation is zero.

$$R_{w0} = I_3, \quad T_{w0} = 0_{3,1} \quad (10)$$

A 3D point  $P$  is represented in the occupancy grid/DEM space as (row, column, height), therefore these formulas will be applied straightforward in the occupancy grid space used for 3D data representation.

### 3. Traffic isles filtering based on temporal persistence

The algorithm presented in [1] detects traffic isles and less than 5% of the detected traffic isles are false. However, they can appear in front of the ego car and cause false collision situations. In [8] a curb detection algorithm was presented, based on a multi-frame persistence map. We adapted this concept of multi-frame persistence for filtering traffic isles.

Similar to false curbs, false traffic isles have an important feature: they persist only for a limited number of consecutive frames (mostly for two frames). Furthermore, traffic isles are static scene items. Based on these features, we propose a fast and efficient approach to filter false traffic isles: a multi-frame persistence map is built (the ego motion between frames is taken into consideration) and only traffic isle cells that persist for several frames are validated.

Traffic isles are static related to the road surface: if the ego motion between successive frames is compensated, then traffic isle cells from the current frame should overlap traffic isle cells from the previous frame.

First, let us review the concept of multi-frame persistence map (PM). The PM is a rectangular map of the same size as the DEM. A cell  $(i,j)$  ( $i$  for the Z direction and  $j$  for X) of PM shows the lifetime (in

consecutive frames) of the DEM cell (i,j): for how many consecutive frames was the cell detected as traffic isle, in a global reference frame (the same 3D location relative to the road, along the sequence of frames).

The persistence map  $PM_T$  for the current frame is built from the  $PM_{T-1}$  of the previous frame and the set of traffic isle cells detected on the  $DEM_T$  of the current frame.

For each location (i, j) of  $PM_T$ :

- If (i,j) is not an edge point then  $PM_T(i,j)=0$ ;
- Otherwise, if (i,j) is edge then:
  1. Compute the coordinates of current frame point (i, j) in the previous frame, as real numbers ( $i'$ ,  $j'$ ).
  2.  $PM_T(i,j)=\text{MAXIMUM}(W) + 1$ , where the set  $W$  contains the persistence values of the previous frame  $PM_{T-1}$  for the 4 closest neighbors of ( $i'$ ,  $j'$ ),

$$W = \left\{ \begin{array}{l} PM_{T-1}(\lfloor i' \rfloor, \lfloor j' \rfloor), PM_{T-1}(\lfloor i' \rfloor, \lceil j' \rceil), \\ PM_{T-1}(\lceil i' \rceil, \lfloor j' \rfloor), PM_{T-1}(\lceil i' \rceil, \lceil j' \rceil) \end{array} \right\}.$$

Using the maximum persistence of the four cells closest (instead of a single point) to the real coordinates from the previous frame is required to compensate some sources of errors:

- Possible lack of accuracy from the ego sensors,
- Most of the time, integer coordinates (i,j) from  $DEM_T$  do not have a correspondent  $DEM_{T-1}$  location with integer coordinates ( $i'$ ,  $j'$ ),
- Pitch angle variation might occur between frames (due to road bumps etc), causing small depth shifts of the coordinates between successive frames.



Fig. 3. The first frame of the sequence used.

Once the persistence map  $PM$  is computed, traffic isle cells should be selected as cells having the persistency higher than a threshold. A threshold too low will cause many false traffic isle cells to occur, while a value too high will cause unjustified delay for detecting true traffic isle cells.

We performed the same experiment as in [8], but for traffic isles instead of curbs, for selecting the optimal threshold: a sequence of 200 frames was acquired while driving through an empty parking lot, without any objects or traffic isles in the analyzed ROI (Fig. 3). The road surface had normal texture, with common features such as braking traces, different color patches etc.

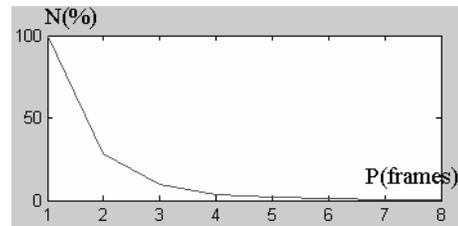


Fig. 4. The percentage  $N$  of cells as a function of the persistence value  $P$ .

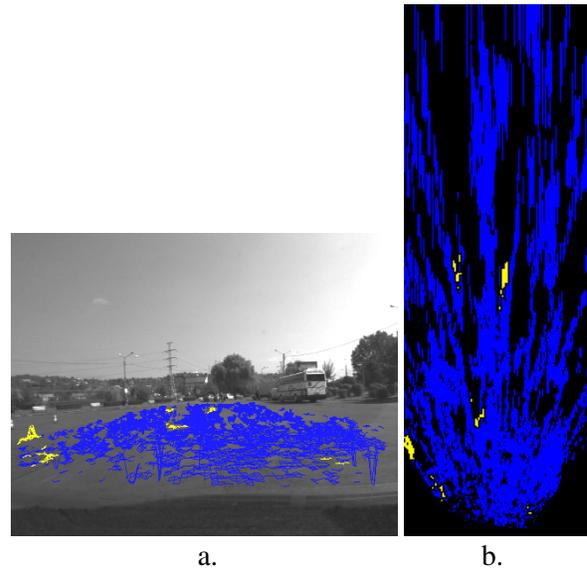


Fig. 5. False traffic isles (yellow) appearing on the training sequence: left image projection (a) and the classified grid (b, yellow - traffic isles and blue - road).

All of the detected traffic isles cells are false, due to the poor accuracy of the 3D reconstructed road. A total number of about 40000 traffic isle cells were detected (an average of 200 cells per frame). The total number of traffic isle cells having the same persistence value

was evaluated (Fig. 4) relative to the total number of traffic isle cells. A distribution similar to the one in [10] was obtained (this is natural since curbs are the borders of traffic isles – they present similar features).

A value  $Th$  of 3 or 4 frames is acceptable for the threshold, greatly reducing the number of traffic isle cells. The only downside is that a true traffic isle cell is validated only after  $Th$  frames since it entered the analyzed ROI. However, this has minor influence upon detection (no traffic isles are missed). Even for a speed of 50km/h, it will take about 30-40 frames for the ego car to reach a traffic isle cell placed at the maximum depth.

#### 4. Static / dynamic obstacles separation

The confidence of the obstacles detected by [1] can be increased by tracking each obstacle over time (along consecutive frames). This would help to filter false obstacles and also to extract dynamic obstacle features. Urban scenes are often crowded with static obstacles (parked vehicles, poles etc.). This can make the task of tracking multiple obstacles quite difficult.

To provide a smaller set of candidates for tracking, we propose the detection of static obstacles in a manner similar to the temporal filtering of traffic isles.

If the ego motion is compensated between successive frames then obstacle blobs that overlap precisely along several frames are probably static. Two approaches can be used to find static obstacles.

One way to check how well obstacle blobs overlap along consecutive frames is to use the computed density map proposed in [1].

Considering an obstacle blob from the current frame, we compute its location in the previous frame (assuming it is static). Next, the blob of densities from the current frame, associated with the obstacle, is matched with the density map of the previous frame. Matching (a SAD function) is performed at multiple locations around the estimated obstacle location in the previous frame. If a local maximum of the matching function is detected at the estimated location (or close-to) then we consider the obstacle has zero or negligible speed, which cannot be sensed from stereo data (few km/h). Such obstacles with no speed are considered static.

This approach provides good results but it can be time consuming due to the SAD matching function, applied at multiple locations.

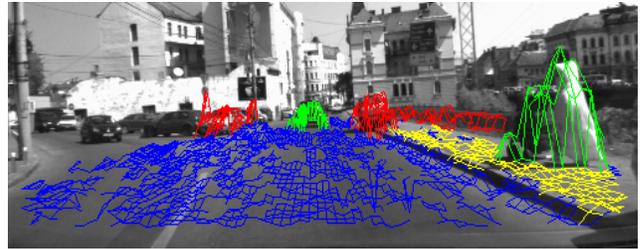


Fig. 6. The enhanced occupancy grid: static obstacles are displayed with red and dynamic obstacles with green.

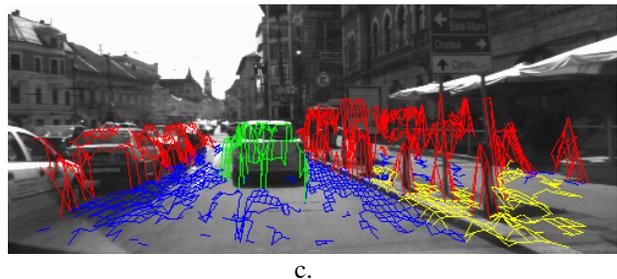
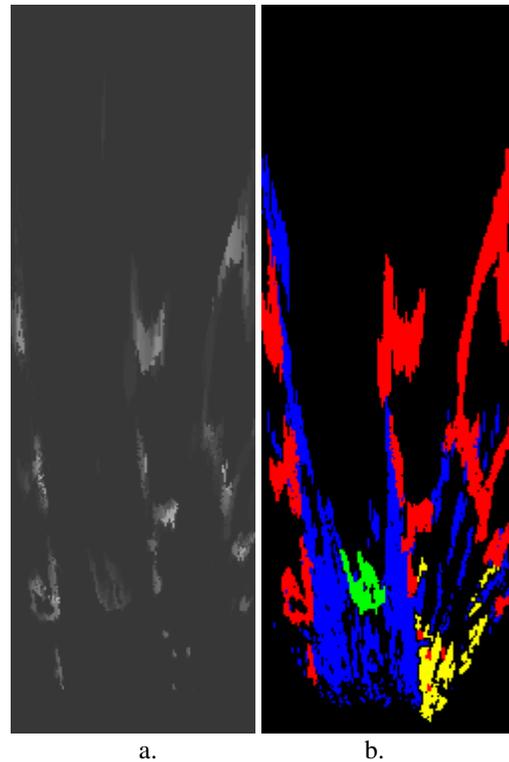


Fig. 7. The persistence map (a) is shown, for the scenario in c. The enhanced grid (with static / dynamic obstacles) is shown in b. The front car is the only moving obstacle; the vehicles on the left are waiting for the green light.

Obstacles that do not overlap along frames (with compensated ego motion) are considered dynamic and

a different class is assigned to their cells (Fig. 6). Kalman or other tracking approach can be employed for dynamic features computation (speed, acceleration, etc).

The faster alternative is to build a persistence map for obstacle cells, in a similar manner as proposed for the filtering of false traffic isles (section IV). The persistence map will show the lifetime for each obstacle cell (for how many consecutive frames it was labeled as an obstacle cell, in a global reference system).

Static obstacle cells will tend to have an increased lifetime while dynamic ones will have a shorter lifetime. However, depending on the obstacle size and speed, some of the obstacle cells might overlap cells from the same obstacle along several frames. This can offer false clues about the nature of individual cells. Therefore, the discrimination will be performed at blob level, in the following manner:

- For each obstacle blob the average persistence value is computed from the persistence values of its cells.
- If the average persistence of the blob is above a threshold (a desired number of frames with no or negligible blob movement), the blob is considered static, otherwise dynamic.

## 5. Global Grid Computation

The occupancy grid computed is covering only a region of the 3D space ahead, limited by the characteristics of the stereo sensor. This area might be enough for navigation, but it might be too small for task such as alignment with local maps, due to the lack of enough infrastructure items (traffic isles, poles, etc).

Based on the ego motion, we propose the merging of the occupancy grids, detected along several frames, into a global occupancy grid (Fig. 8). Low-level integration (at cell level) is performed for the static scene items (road, traffic isles and static obstacles). A simple voting strategy is used to establish the class of each cell from the global grid (multiple frames contribute for each global cell).

Optionally, dynamic obstacles can be integrated into the global grid, but at a higher level of representation (blobs).

## 7. Results

The proposed improvements are effective into obtaining an occupancy grid with fewer false detections

and more information within the grid: more cell types are provided. This new information is valuable for higher level tasks such as navigation. The global occupancy grid, although it is just an experimental approach, provides a more complete representation of the vehicle's surrounding environment. This allows easier alignment of the results with external maps. As a simple test, we computed the global grid (only traffic isles and static obstacles) for a sequence of about 270 frames, along a distance of about 200 meters. The scenario consisted of two consecutive intersections (Fig. 10). The global grid was then scaled and superimposed manually (Fig. 9) to the satellite image of the scenario (available from Google Earth).

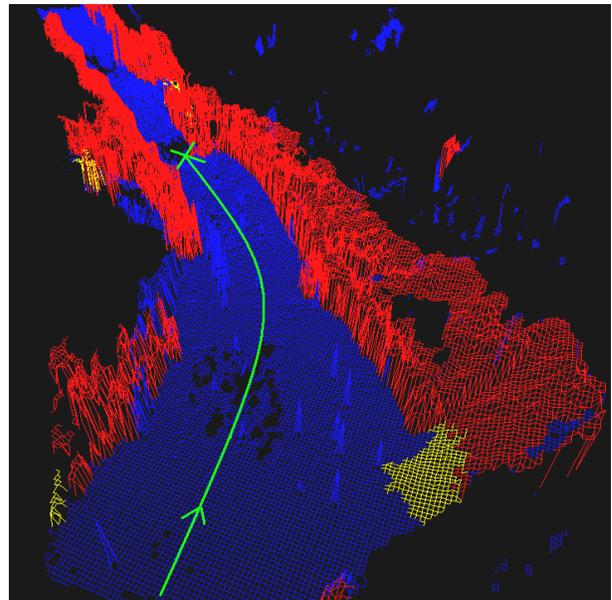


Fig. 8. The global occupancy (top image) grid along 50 frames, while cornering (ego moves along the green curve) in an urban scenario (bottom image).

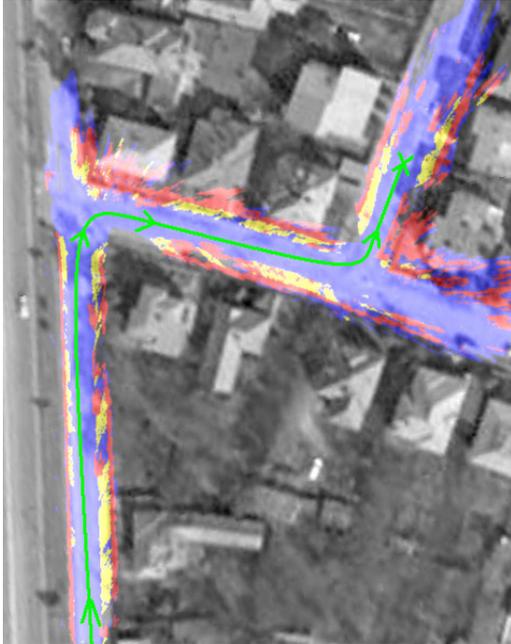


Fig. 9. Global occupancy grid superimposed to the satellite image of the region. The trajectory of the ego car is shown with the green curve.



Fig. 10. From left to right: the first frame of the sequence, a frame with the turn right intersection, a frame with the turn left intersection.

## 5. References

- [1] F. Oniga, S. Nedeveschi, M-M. Meinecke, T-B. To, "Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo", the 10th International IEEE Conference on Intelligent Transportation Systems, Sept. 30 - Oct. 3, 2007, Seattle, Washington, USA.
- [2] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, B. Gerkey, "Outdoor Mapping and Navigation using Stereo Vision", In Proc. of Intl. Symp. on Experimental Robotics (ISER), Rio de Janeiro, Brazil, July 2006
- [3] D. Murray and C. Jennings. "Stereo vision based mapping and navigation for mobile robots", In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'97), pages 1694-1699, New Mexico, April 1997.
- [4] J.M. Saez and F. Escolano: "A Global 3D Map-Building Approach Using Stereo Vision", In Proceedings of IEEE International Conference on Robotics and Automation (ICRA) (2004).
- [5] 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.
- [6] 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.
- [7] S. Bota and S. Nedeveschi, "Camera Motion Estimation Using Monocular and Stereo-Vision", 4th International Conference on Intelligent Computer Communication and Processing, 2008, Cluj-Napoca, Romania.
- [8] F. Oniga, S. Nedeveschi, M-M. Meinecke, "Curb Detection Based on a Multi-Frame Persistence Map for Urban Driving Scenarios", The 11th International IEEE Conference on Intelligent Transportation Systems, 2008, Beijing, China.