HABILITATION THESIS

Solutions for modeling and perception of dynamic 3D environments

Assoc. Prof. Radu Gabriel DANESCU, PhD

Faculty of Automation and Computer Science
Technical University of Cluj-Napoca

2014

Table of contents

HABILITATION THESIS
"Solutions for modeling and perception of dynamic 3D environments"

a) ABSTRACT

This thesis presents the scientific activity and achievements of the candidate after defending his PhD thesis at the Technical University of Cluj-Napoca on 12.12.2009, and receiving the PhD title confirmation by the Ministry of Education and Research's Order No. 3492, dated 23.03.2010. Before the PhD defense, the candidate's research activity was focused on model-based object tracking for driving assistance applications, using the stereovision as measurement source. After finishing the PhD, the candidate remained active in the field of stereovision based perception, focusing on two major directions: designing solutions for modeling and tracking freeform dynamic 3D environments, and solutions for space surveillance based on large baseline stereovision. For both these directions, the results obtained were significant enough to warrant publication in several ISI journal articles and conference papers.

1. Solutions for modeling and tracking freeform dynamic 3D environments: while many environments can be broken down into discrete pieces that can be modeled by geometrical entities such as boxes or parametrical curves, and the parameters of these geometrical entities can be tracked, sometimes the environment gets too complex, or the level of detail needed by the perceiving actor exceeds the assumed simplifications. For these reasons, alternative, freeform models such as occupancy grids and elevation maps can be employed. The candidate's objectives were to create new freeform world models, based on the occupancy grid and elevation map paradigm, and use them for world tracking.

The occupancy grid solutions found in the literature were mostly dedicated to static environments, the dynamic solutions being few and, from the candidate's point of view, cumbersome and limited. The candidate's proposed dynamic occupancy grid solution is based on dynamic particles that are the building blocks of the world, having position and speed, and which can migrate from one grid cell to another. In this way, a multi-modal probability density of the cell's state, which includes occupancy and speed, is naturally represented. The particles are created and destroyed based on stereovision-derived measurement information, using a computationally efficient resampling algorithm. The particle population allows, at cell level, the estimation of occupancy and speed. The dynamic properties of the cells can be then used, if needed, for extraction of geometric objects, which will already have speed and orientation, even without model based tracking.

The elevation map solutions available in the literature were exclusively dedicated to static environments. The solution proposed by the candidate, the particle-based dynamic elevation map, is able to successfully model and track complex dynamic environments, estimate the heights of the map cells, and the cell's speed. The main challenges faced in designing the system were related to incorporating the 3D uncertainty of the stereovision into the particle weighting process as a multi-modal measurement model, and designing a particle motion mechanism that can handle quickly enough the dynamic elements of the scene. The novel world model, the particle-based dynamic elevation map, was extended even further with gray level information, so that the tracking process can benefit from the image aspect of the scene, besides the 3D information, and also for a more detailed description of the perceived environment.

2. *Large baseline stereovision for space surveillance:* the Earth is surrounded by a huge number of orbiting objects, at various distances, travelling through space at various speeds. Some of these objects are useful, and some are simply leftovers from old space missions or from previously operating satellites. As the space is getting crowded, the

importance of keeping an eye on the objects orbiting the Earth grows. Multiple techniques for space surveillance exist, some relying on active ranging such as RADAR, but mostly based on optical devices, which merely receive the light reflected by the targets, and thus require significant less power to operate. The existing techniques for optical based space surveillance rely on image sequences produced by a single image source (a single telescope), and use the orbital constraints for determining the target's range. The candidate saw an opportunity in the space surveillance field, the use of stereovision for automatic detection and ranging of the Earth orbiting objects. As the distance of these objects is in the range of thousands of kilometers, the baseline of the Earth-based stereoscope had to be in the order of kilometers or tens of kilometers. In order to achieve a functioning system, the following challenges had to be overcome: synchronization of the two observation stations, without the possibility of using a common trigger signal, as normal stereovision systems employ; intrinsic calibration of the optical systems; continuous calibration of the rotation matrices using stars as reference points, as the systems track the sky and therefore its orientation has to be continuously updated; detection of the candidate satellite features from the images, sometimes in a low contrast condition and in the presence of significant sensor noise (caused by long exposure and high gain, needed to increase the sensitivity); the challenge of stereo matching and 3D parameters computation. An experimental system capable of detecting and ranging satellites in the LEO, MEO, GEO and HEO orbits has been successfully set up.

The main achievements and results detailed in Chapter (b-i): *Scientific, professional and academic achievements.*

The candidate's near future research activity will be focused on the two main directions that have produced the results described in this thesis: modeling and tracking complex, dynamic 3D environments, and stereovision-based space surveillance.

The main challenges that will be tackled in the context of dynamic environment modeling and tracking are:

- Developing a dynamic world model and tracking solution that will integrate the stereovision information in the measurement process without first transforming it into a raw elevation map. The raw stereo information will include disparity and grayscale values for each pixel in the image, and the measurement model will relate these values and their uncertainty directly to the tracking mechanism. In this way the error of the measurement can be estimated with much better precision, which will improve the tracking results.
- Transforming a world model and tracking method into a sensor fusion technique, by integrating multiple measurement sources in the measurement process. As an intermediate representation, either the dynamic occupancy grid or the dynamic elevation maps are suitable for this attempt.

The main challenges that are still open in the field of stereovision based space surveillance are:

- Improving the quality of the range estimation, by an in depth analysis of the sources of uncertainty in the measurement process and devising solutions to remove these uncertainties.
- Designing a tracking algorithm based on estimating the state of the target, state which is in fact made out of the orbital parameters. A method for determining the orbit parameters from the stereo results, and a prediction system capable of propagating these parameters, have to be designed.

For the long term, the candidate estimates that his research will be focused on perception systems for robotics, driving assistance applications and space surveillance, but also on generic computer vision topics.

A more detailed description of each topic can be found in Chapter (b-ii): *Scientific, professional and academic future development plans.*

b) ACHIEVEMENTS AND DEVELOPMENT PLANS

(b-i) Scientific, professional and academic achievements

Articles constituting the habilitation thesis

1. **R. Danescu**, F. Oniga, S. Nedevschi, "Modeling and Tracking the Driving Environment with a Particle Based Occupancy Grid", *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, No. 4, December 2011, pp. 1331-1342.
2. **R. Danescu**, C. Pantilie, F. Oniga, S. Nedevschi, "Particle Grid Tracking System for Stereovision Based Obstacle Perception in Driving Environments", *IEEE Intelligent Transportation Systems Magazine*, vol. 4, No. 1, March 2012, pp. 6-20.
3. **R. Danescu**, F. Oniga, S. Nedevschi, "Particle Grid Tracking System for Stereovision Based Environment Perception", in Proc. of the *IEEE Intelligent Vehicles Symposium (IEEE-IV 2010)*, June 2010, San Diego, USA, pp. 987-992.
4. **R. Danescu**, "Obstacle Detection Using Dynamic Particle-Based Occupancy Grids", *International Conference on Digital Image Computing: Techniques and Applications 2011 (DICTA 2011)*, pp. 585-590.
5. **R. Danescu**, S. Nedevschi, "A Particle-Based Solution for Modeling and Tracking Dynamic Digital Elevation Maps", *IEEE Transactions on Intelligent Transportation Systems*, in print, DOI 10.1109/TITS.2013.2291447.
6. **R. Danescu**, S. Nedevschi, "A Flexible Solution for Modeling and Tracking Generic Dynamic 3D Environments", in Proc. of the *IEEE Intelligent Transportation Systems Conference 2013 (IEEE-ITSC 2013)*, October 2013, The Hague, The Netherlands, pp. 1686-1692.
7. **R. Danescu**, F. Oniga, V. Turcu, O. Cristea, "Long Baseline Stereovision for Automatic Detection and Ranging of Moving Objects in the Night Sky", *Sensors*, vol. 12, No. 10, October 2012, pp. 12940-12963.
8. **R. Danescu**, A. Ciurte, V. Turcu, "A Low Cost Automatic Detection and Ranging System for Space Surveillance in the Medium Earth Orbit Region and Beyond", *Sensors*, vol. 14, No. 2, February 2014, pp. 2703-2731.
9. O. Cristea, P. Dolea, V. Turcu, **R. Danescu**, "Long baseline stereoscopic imager for close to Earth objects range measurements", *Acta Astronautica*, vol. 90, No. 1, September 2013, pp. 41–48.
10. F. Oniga, M. Miron, **R. Danescu**, S. Nedevschi, "Automatic Recognition of Low Earth Orbit Objects From Image Sequences", *International Conference on Intelligent Computer Communication and Processing*, Cluj-Napoca, 2011, pp. 335–338.

Scientific, professional and academic experience of the candidate

Research projects (selection)

1. **Project manager** of PNII-PCCA project "Automatic Medium and High Earth Orbit Observation System Based on Stereovision", 2012-2016, total value 650000 euro, http://cv.utcluj.ro/amheos/ .
2. **Project manager** of CNCSIS-TD project "Road and lane detection in urban traffic scenarios", 2006-2007, total value 4500 euro.
3. Team member of the Eureka European project CoMoSef (2012-2015), "Co-operative Mobility Services of the Future".
4. Team member of the European FP7 project INTERSAFE-2 (2008-2011), "Cooperative Intersection Safety".
5. Team member of the European FP7 project PAN-ROBOTS (2012-2015), "Plug and Navigate ROBOTS for smart factories".
6. Team member of the project financed by Volkswagen AG "SCABOR - Stereo Camera-Based Object Recognition", 2001-2004.
7. Team member of the project financed by Volkswagen AG "DESBOR - Dense Stereo Camera-Based Object Recognition", 2005-2007.
8. Team member of the PNII-Partnership project LEOSCOP (2008-2011), "Experimental Low Earth Orbit Surveillance Stereoscope".
9. Team member of the PNII-Idei project PERSENS (2008-2011), "Sensorial perception, modelling and representation of the world model for driving assistance systems".
10. Team member of the PNII-PCCA project SMARTCODRIVE (2012-2015), "Cooperative Advanced Driving Assistance System Based on Smart Mobile Platforms and Road Side Units".
11. Team member of the PNII-PCE project MULTISENS (2012-2015), "Multi-scale multi-modal perception of dynamic 3D environments based on the fusion of dense stereo, dense optical flow and visual odometry information".

Scientific impact and recognition

- 643 citations, h=13, according to Google Scholar Citations
- 287 citations, h=9, according to Scopus
- 190 citations, h=6, according to ISI Web of Knowledge

Reviewer of ISI journals

- IEEE Transactions on Intelligent Transportation Systems (IEEE)
- IEEE Transactions on Vehicular Technology (IEEE)
- Remote Sensing (MDPI)
- Measurement (Elsevier)

Professional membership

- Member of IEEE
- Member of IEEE Intelligent Transportation Systems Society

Awards

Four ISI articles have been awarded by UEFISCDI under the "Awarding the research results" program:

1. http://uefiscdi.gov.ro/userfiles/file/PREMIERE_ARTICOLE/REZULTATE_CUMULATE_ETAPA_II%20-8_IUNIE%281%29.pdf ;
2. http://uefiscdi.gov.ro/userfiles/file/PREMIERE_ARTICOLE/articole%202011/evaluare/REZULTATE%20IANUARIE%20ACTUALIZAT.pdf;
3, 4: http://uefiscdi.gov.ro/userfiles/file/PREMIERE_ARTICOLE/ARTICOLE%202013/LISTA%202%20REACTUALIZATA%2018%20DECEMBRIE%20CAROLINA.pdf

Teaching activity

12 years of teaching activity, covering the following courses: Image Processing (year 3), Design with Microprocessors (year 3), and laboratory works at the following disciplines: Image Processing, Design with Microprocessors, Computer Architecture, Pattern Recognition Systems, Computer Vision (master).

The candidate has supervised more than 20 diploma works, and about 10 master theses, since 2009.

Additional information can be found on the candidate's web page, http://users.utcluj.ro/~rdanescu .

1. Solutions for modeling and tracking freeform dynamic 3D environments

1.1. The particle based occupancy grid

1.1.1. Introduction

The tasks of modeling and perceiving the dynamic 3D environments face continuous challenges, because there are multiple types of scenarios, of different degrees of order and complexity. Some environments are well-regulated, and the types of static and dynamic objects are easily modeled and tracked using geometrical models and their parameters. The obstacles can be modeled as cuboids having position, size and speed, and the driving surface delimiters can be modeled as parametrical curves. In the driving assistance field, the highway and most of the urban and rural sections of road are usually suitable for geometrical modeling and tracking.

The conditions change when the environment to be tracked is an intersection, a busy urban center, or an off-road scenario. Even if parts of this environment can be tracked by estimating the parameters of a geometrical model, many essential parts of the environment will not fulfill the constraints of the models. Also, sometimes it is better to have static and dynamic information about the environment before a model can be instantiated and tracked, or it may use this additional information in model fitting and model-based tracking. For these reasons, solutions for intermediate level representation and tracking are devised. These intermediate representation and tracking solutions can be based on occupancy grids.

Maybe one of the first uses of occupancy grids, under the name of probabilistic local maps, is presented by Elfes in [1], in the context of sonar based robot navigation. Another paper by the same author [2] names the occupancy maps occupancy grids, and describes the probability inference mechanism for handling the uncertainty of a range sensor in computing the probability of each cell's occupancy state. In the same reference we find a definition of the occupancy grid: "the occupancy grid is a multi-dimensional random field that maintains stochastic estimates of the cells in a spatial lattice".

The initial occupancy grids, such as those presented in [1] and [2], are simple 2D maps of the environment, each cell describing the probability of it being occupied or free. However, for many tracking applications, especially in the driving assistance field, there is a need for estimating the dynamic parameters of the environment, namely the speed of each grid cell. By adding the speed factor in the environment estimation, the complexity increases significantly, as the cells are now strongly interconnected. The work of Coué et al, presented in [3], uses a 4D occupancy grid, where each cell has a position and two speed components along each axis. By estimating the occupancy of each cell in the 4D grid, the speeds for the classical cells in the 2D grid can be computed.

Another solution for the representation of speeds is presented by Chen et al, in [4]. Instead of having a 4D grid, this solution comes back to 2D, but uses for each cell a distribution of speeds, in the form of a histogram. The Bayesian inference mechanism relies on sensor data and antecedent cells, the list of antecedents being decided by the speed hypotheses.

A simpler, but limited way of handling the dynamic aspects of the environment is presented in [5]. Instead of estimating the speed of each cell, this solution relies on "occupancy trails", which are specific patterns, similar to the motion blur of the camera, which can be used to derive the trajectory and therefore the speed of the moving objects. A

more sophisticated method is presented in [6], where the inconsistencies in the static grid are detected as soon as they appear, and a multi-model Kalman filter tracker is initialized to track the dynamic object.

We can attempt a first classification of the dynamic occupancy grid solutions (not the grids themselves) into fully dynamic, as those presented in [3], [4] and [7], and static-dynamic hybrids, as those presented in [5] and [6].

One of the most important features of an occupancy grid tracking solution is the way the sensor model is used for grid update. The most time efficient way of updating a grid is to rely on the inverse sensor model, which derives the probability of a cell being occupied directly from sensor readout, assuming the occupancy of each cell is independent of its neighbors. This solution is maybe still the most popular, mainly in static grids [6]. However, the work of Thrun [8] proved that forward sensor probability models are preferable even in the case of static grids, even if this significantly increases the complexity of computation.

The occupancy grids can have multiple spatial representations, and in [9] we are shown a comparison between three types of grids, the Cartesian (classic), the polar (distance and angle) and the column/disparity grids. All these grids have advantages and drawbacks. A Cartesian grid is closer to the real world representation, and can handle velocities easier, while the other types of grids are more "sensor-friendly", making the computation of the sensor uncertainties easier.

1.1.2. Solution overview

This chapter presents a dynamic occupancy grid solution based on an original world model, made out of moving particles. Based on the surveyed literature, the occupancy grid tracking solution presented in this paper can be classified as having a Cartesian representation, using a forward sensor probability model, and producing a fully dynamic grid. The proposed method is most closely related to the works presented in [4] and [7], which use a speed probability distribution for each cell in the grid, instead of modeling the dynamic grid as a high dimensional space, as in [3].

The proposed solution comes as an improvement over these techniques, because due to the use of moving particles the representation of the speed probability distribution and the estimation of this distribution are no longer a concern. The velocity distribution is not approximated as a histogram [4] or as a mixture of Gaussians [7], there is no assumption that one cell belongs to only one object with only one velocity, and the estimation of speed results naturally from the survival or elimination of the particles. The particles in a cell can have different speeds, and therefore they can handle the situation of overlapping objects, or the most likely situation when the objects are too close and the uncertainty of one overlaps over the uncertainty of the other. The complexity of the algorithm is linear with the number of cells in the grid and with the maximum number of particles in a cell, a tradeoff between accuracy and response time being always available as a simple parameter. Also, integrating other motion parameters, such as acceleration, does not increase the complexity of the tracking algorithm, because it only alters the way the position of the particles in time is computed.

The first step of the algorithm is the prediction, which is applied to each particle in the set. The positions of the particles are altered according to their speed, and to the motion parameters of the ego vehicle. Also, a random amount is added to the position and speed of each particle, for the effect of stochastic diffusion. The second step is the processing of measurement information. This step is based on the raw occupancy cells provided by dense stereo processing, and provides the measurement model for each cell. The measurement

model information is used to weight the particles, and resample them in the same step. By weighting and resampling, the particles in a cell can be multiplied or reduced. The final step is to estimate the occupancy and speeds for each cell, and to group the cells into 3D oriented objects, for result evaluation.

1.1.3. The world model

The world is represented by a 2D grid, mapping the bird-eye view 3D space into discrete 20 cm x 20 cm cells. The size of the grid is 250 rows x 120 columns (this corresponds to a scene size of 50x24 meters). The aim of the tracking algorithm is to estimate the occupancy probability of each grid cell, and the speed components on each axis. The tracking goals are achieved by the use of a particle-based filtering mechanism.

Considering a coordinate system where the z axis points towards the direction of the ego-vehicle, and the x axis points to the right, the obstacles in the world model are represented by a set of particles $S = \{p_i \mid p_i = (c_i, r_i, vc_i, vr_i, a_i), i = 1...N_S\}$, each particle i having a position in the grid, described by the row r_i (a discrete value of the distance in the 3D world z) and the column c_i (discrete value of the lateral position x), and a speed, described by the speed components vc_i and vr_i. An additional parameter, a_i, describes the age of the particle, since its creation. The purpose of this parameter is to facilitate the validation process, which will be described in a subsequent section. The total number of particles in the scene N_S is not fixed. This number depends on the occupancy degree of the scene, that is, the number of obstacle cells. Having the population of particles in place, the occupancy probability of a cell C is estimated as the ratio between the number of particles whose position coincides with the position of the cell C and the total number of particles allowed for a single cell, N_C.

$$P_O(C) = \frac{|\{p_i \in S \mid r_i = r_c, c_i = c_c\}|}{N_C} \tag{1.1.1}$$

The number of allowed particles per cell N_C is a constant of the system. In setting its value, a tradeoff between accuracy and time performance should be considered. A large number means that on a single cell multiple speed hypotheses can be maintained, and therefore the tracker can have a better speed estimation, and can handle fast moving objects better. However, the total number of particles in the scene will be directly proportional with N_C, and therefore the time consumption will increase.

The speed estimation of a grid cell can be estimated as the average speed of its associated particles, if one assumes that only one obstacle is present in that cell. Of course, the particle population can handle the situation when multiple obstacles, having different speeds, share the same cell, and in this case the speed estimate of the cell must be computed by clustering.

$$(vc_C, vr_C) = \frac{\sum_{p_i \in S, x_i = x_c, z_i = z_c} (vc_i, vr_i)}{|\{p_i \in S \mid r_i = r_c, c_i = c_c\}|} \tag{1.1.2}$$

Thus, the population of particles is sufficiently representative for the probability density of occupancy and speed for the whole grid. Multiple speed hypotheses can be maintained simultaneously for a single cell, and the occupancy uncertainty is represented by the varying number of particles associated to the cell. The goal of the tracking algorithm can

now be stated: using the measurement information to create, update and destroy particles such that they accurately represent the real world.

1.1.4. Prediction

This step will derive the present particle distribution from the past information, preparing the particle set for measurement. The prediction equations will use odometry and motion model information.

The basic odometry information available through the CAN bus of a modern car is the speed v and the yaw rate $\dot{\psi}$. Together with the time interval Δt elapsed between measurements, these parameters can be used to compensate for the ego-motion, and separate it from the independent motion of the objects in the scene. Between measurements, the ego-vehicle rotates with an angle ψ, and travels a distance d.

$$\psi = \dot{\psi}\Delta t \tag{1.1.3}$$

$$d = \frac{2v\Delta t \sin\frac{\psi}{2}}{\psi} \tag{1.1.4}$$

The origin of the grid representation is displaced along the two coordinate axes by d_c and d_r.

$$d_c = d \sin\frac{\psi}{2} / DX \tag{1.1.5}$$

$$d_r = d \cos\frac{\psi}{2} / DZ \tag{1.1.6}$$

We denote by DX and DZ the cell size of the grid (in the current implementation, 0.2 m). A point in the grid, at row r and column c, is displaced by an amount computed using the following equation:

$$\begin{bmatrix} c_n \\ r_n \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} c \\ r \end{bmatrix} - \begin{bmatrix} d_c \\ d_r \end{bmatrix} \tag{1.1.7}$$

The prediction is achieved using equation 1.1.8, which combines the deterministic drift caused by the ego-motion compensation and the particle's own speed, with the stochastic diffusion caused by the uncertainties in the motion model. The quantities δc, δr, δvc and δvr are randomly drawn from a Gaussian distribution of zero mean and a covariance matrix $\mathbf{Q}$ equivalent to the state transition covariance matrix of a Kalman filter. The covariance matrix is diagonal, with the standard deviations for the speed components corresponding to a real-world amount of 1 m/s, and the standard deviations for the position corresponding to a real-world value of 0.1 m. These values will ensure that the system is able to cope with fast-moving objects even at a 10 fps frame rate.

$$\begin{bmatrix} c \\ r \\ v_c \\ v_r \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_n \\ r_n \\ v_c \\ v_r \end{bmatrix} + \begin{bmatrix} \delta c \\ \delta r \\ \delta vc \\ \delta vr \end{bmatrix} \tag{1.1.8}$$

From the grid model point of view, the prediction has the effect of moving particles from one cell to another, as seen in figure 1.1.1. The occupancy probability is thus dynamically adjusted using the particle's motion model and the vehicle odometry.

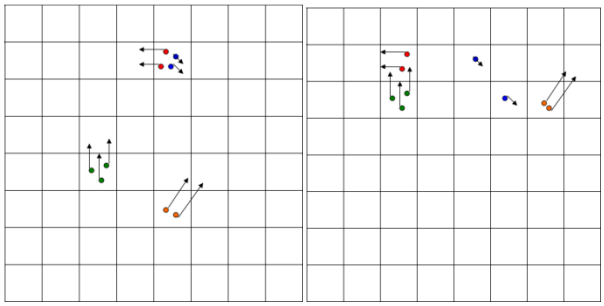


Fig. 1.1.1. Particles in the grid, before and after prediction.

1.1.5. The measurement model

The measurement model relates the measurement data, which is a binary occupied/free condition derived from the stereovision-generated elevation map [10], to the conditional probabilities *p(measurement | occupied)* and *p(measurement | free)*, which will weigh the particles. In order to compute these probability values, several steps must be performed.

First, the uncertainty of the stereo reconstruction process is computed. The uncertainty of the distance reconstruction, in the case of a rectified system, is given by:

$$\sigma_z = \frac{z^2 \sigma_d}{bf} \tag{1.1.9}$$

In the above equation, z denotes the distance (in the real world coordinates), b is the baseline of the stereo system, f is the focal distance in pixels, and σ_d is the error in disparity computation (usually about 0.25 pixels, for a good stereo reconstruction engine).

The error in lateral positioning (usually much smaller than the error in z), can be derived from the distance error. This error depends on the lateral position x (in the real world coordinates) and the distance z.

$$\sigma_x = \frac{x \sigma_z}{z} \tag{1.1.10}$$

The 3D errors are mapped into grid cell errors, by dividing them with the grid cell size on x and z.

$$\sigma_{row} = \frac{\sigma_z}{DZ}$$
$$\sigma_{column} = \frac{\sigma_x}{DX} \qquad (1.1.11)$$

The values of σ_{row} and σ_{column} are computed offline, at the initialization phase, for each cell in the grid.

In order to compute the conditional probability of the measurement cell, under the occupied or free assumption one has to take into account a reality that is specific to stereovision sensors. The stereo sensor does not perform a scan of the scene, and therefore it does not output a single bird-eye view point for a real-world obstacle cell. We'll take as example a pillar, which has almost no width, and no depth spread. The representation of a pillar in the occupancy grid should be a single cell. If the pillar were observed by a scanner-type sensor, this sensor will output a cell, displaced from the true position by an amount specific to the sensor error. For the stereo sensor, things are different, because the camera observes the whole height of the pillar, and therefore each pillar pixel will get a distance and a lateral position. This means that once the pillar information is "collapsed" in the 2D grid representation, each part of the pillar may fall in a different cell, and the pillar will generate a spread of cells. The size of the spread area is controlled by the grid uncertainties on the *c* and *r* axes (real world *x* and *z*).

This property leads to a good cue, which will contribute to the conditional probabilities of the measurement cells under the occupied/free assumption. The obstacle cells in the measurement grid around the current cell position, in an area of σ_{row} height and σ_{column} width, are counted, and the number of found obstacle cells is then divided by the total number of cells in the uncertainty area. This ratio is denoted as $p_{density}(m(r,c) \mid occupied)$.

$$p_{density}(m(r,c) \mid occupied) = \frac{\sum_{row=r-\sigma_{row}}^{row=r+\sigma_{row}} \sum_{col=c-\sigma_{column}}^{col=c+\sigma_{column}} O(row,col)}{(2\sigma_{row}+1)(2\sigma_{column}+1)} \qquad (1.1.12)$$

O(row, col) denotes the "occupied" value of the measurement grid, at position *row* and *col*. This value is 1 when an obstacle cell is present and 0 when not.

The density cue for the "free" assumption is:

$$p_{density}(m(r,c) \mid free) = 1 - p_{density}(m(r,c) \mid occupied) \qquad (1.1.13)$$

A graphic comparison between the raw measurement data and the density cue (conditional probability) of the measurement under the "occupied" assumption is given in the following figure.

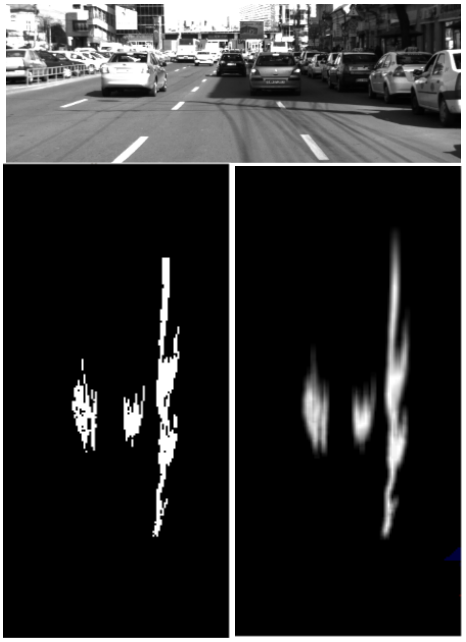


Fig. 1.1.2. From the raw occupancy grid to the raw measurement density cues. Bottom-left: raw occupancy grid, bottom-right: density cue for the occupied cell hypothesis.

Not all cells in the grid can be observed directly, and this fact must be taken into consideration by the tracking algorithm. Due to the limitations of the primary source of information, the stereovision-based raw occupancy grid, some of the cells are never observed. The raw occupancy grid only covers a longitudinal distance from 0 to 40 meters, a lateral span of 13 meters. Also, the field of view of the camera (angular span) limits the areas that are visible at close distance. The cells that are excluded by the field of view and distance limitations are marked as obstructed (unobservable) by default.

Another way for a cell to become unobservable is if it is obstructed by an obstacle cell that is located between it and the observation origin (camera position). In order to decide if a cell is in such a situation, polar coordinates have to be used. Each cell is mapped to a polar grid. Then, for each angle, the cells are scanned in the order of their distance. Once a raw occupied cell is found, an obstruction counter is incremented for every cell that is behind the first occupied one. Then, the obstruction values are re-mapped into the Cartesian grid.

Once each cell has an obstruction value, the final analysis is performed. Each cell that has an obstruction value higher than 10 is considered obstructed and considered as such in the particle weighting and resampling phase. However, this is not the only way the obstruction property is used. If a raw measurement cell is marked as “occupied”, but from the obstruction analysis it is found to be obstructed, the occupied cell is removed. This will make the raw occupancy map look more like a scanner-derived map. This reduction of measurement information must be performed before the computation of the other particle weighing cue, which relies on the distance from measurement.

The obstruction-related processing steps are illustrated in figure 1.1.3. The left panel shows the raw measurement data, the middle panel shows the obstruction value for each cell (the lighter, the more obstructed), and the right panel shows the measurement data that remains after the obstructed cells are removed. This data set is used for the next cue computation.

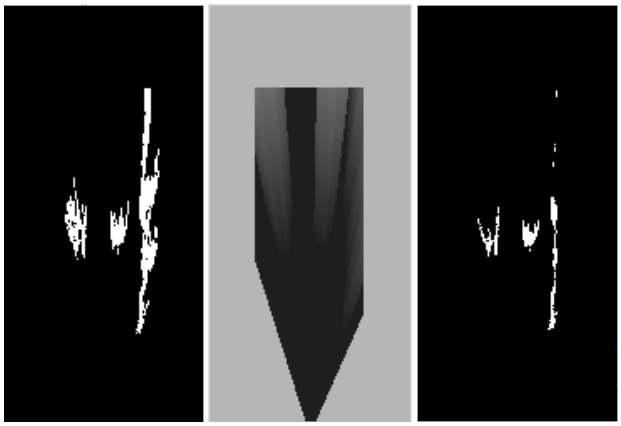


Fig. 1.1.3. Handling the occlusions. Left – original measurement information, middle – obstruction value for each cell, right – unobstructed measurement.

The distance from measurement cue:

For each cell in the grid, the distance to the nearest occupied cell in the measurement grid must be computed. For that, a modified version of the distance transform algorithm presented in [11] is used. The main issue is that not only the distance to the nearest measurement point must be known, but also the distance components on the two coordinate axes, row and column. The reason for this requirement is that the standard deviations for the positioning errors are different on the row and on the column, and therefore one cannot be substituted for another.

The proposed distance transform algorithm performs like the classical two-pass L1 norm one, but instead of updating only the cell distance to the nearest measurement, the position of the nearest measurement is updated along. The following algorithm updates a distance matrix $D(r,c)$, initialized with zero for measurement occupancy cells, and with 255 for the free cells, and two position matrices M_r and M_c that hold the row and the column of the nearest occupied measurement cell. The values of M_r and M_c are initialized to the current row and column of each cell.

```
Algorithm DistanceTransform
For r=1 to max_r
For c=1 to max_c
        Update (r, c, -1, 0)
        Update (r, c, 0, -1)
End For
End For
For r = max_r to 1
        For c = max_c to 1
                Update (r, c, 1, 0)
                Update (r, c, 0, 1)
        End For
End For

Function Update(r, c, n, k)
If D(r, c) > D(r+n, c+k) + 1
        D(r, c) = D(r+n, c+k) + 1
        Mr(r, c) = Mr(r+n, c+k)
        Mc(r, c) = Mc(r+n, c+k)
End If
```

After the distance transform algorithm is applied, the distance-to-measurement-occupied on rows and on columns, for each cell can be found by:

$$d_{row}^{occupied}(r,c) = | r - M_r(r,c) |$$
$$d_{column}^{occupied}(r,c) = | c - M_c(r,c) | \quad (1.1.14)$$

The distance to measurement-free-cell is computed as the difference between the double of the distance standard deviation and the distance-to-occupied, saturated to zero.

$$d_{row}^{free}(r,c) = \max(2\sigma_{row}(r,c) - d_{row}^{occupied}(r,c), 0)$$
$$d_{column}^{free}(r,c) = \max(2\sigma_{column}(r,c) - d_{column}^{occupied}(r,c), 0) \quad (1.1.15)$$

These distances are converted to a probability density value using the multivariate Gaussian equation (equation 1.1.16). The same equation is applied for both *free* and *occupied* distances, and therefore the condition *status* is a placeholder for both situations.

$$p_{\text{distance}}(m \mid status) = \frac{1}{2\pi\sigma_{row}\sigma_{column}} e^{-\frac{1}{2}\left(\left(\frac{d_{row}^{status}}{\sigma_{row}^{status}}\right)^2 + \left(\frac{d_{column}^{status}}{\sigma_{column}^{status}}\right)^2\right)} \quad (1.1.16)$$

At the end of this step, the values $p_{distance}(m(r,c) \mid occupied)$ and $p_{distance}(m(r,c) \mid free)$ are available for each cell.

1.1.6. Particle weighting and resampling

The classical steps of a particle filter based tracker are resampling, drift, diffusion, and measurement (weighting). This behavior replaces a population of a fixed number of particles with an equal number of particles, which approximates an updated probability density function over a space of parameters. However, this approach works when the particles are hypotheses of the state of a system, not when the particles are the system itself (we can see our tracked world as physically composed of particles).

The proposed algorithm tries to use the particles in a dual form – as hypotheses, and as building blocks of the world that we track. Their role as building blocks has been already explained. However, if the reasoning is restricted to a single cell in the grid world, we can see *that the particle is also a hypothesis*. A particle in a grid cell is a hypothesis that this cell is occupied, and that the cell has the speed equal to the speed of the particle. More particles in the cell mean that the hypothesis of occupancy is strongly supported. Less particles in the cell means that the hypothesis of the cell being free is supported. One can regard the difference between the number of particles in a cell and the total number of particles allowed in a cell as the number of particles having the occupancy hypothesis zero.

Weighting the particles

If the number of particles in a cell is considered to be constant, some of the particles having the occupancy value “true” while some having it “false”, the mechanism of weighting and resampling can be applied.

Assuming that the measurement data does not contain speed information, the weight of the particle depends only on the “occupied” hypothesis. Also, this means that all the particles having the same occupied hypothesis will have the same weight.

For each cell at position *r*, *c* in the grid, the weights for the free and for the occupied hypotheses is obtained by fusing the cues computed from the measurement data using the methods previously described.

$$w_{occupied}(r,c) = p_{density}(m(r,c) \mid occupied).p_{\text{distance}}(m(r,c) \mid occupied) \quad (1.1.17)$$

$$w_{free}(r,c) = p_{density}(m(r,c) \mid free).p_{\text{distance}}(m(r,c) \mid free) \quad (1.1.18)$$

The equations 1.1.17 and 1.1.18 hold if the cell in the grid is not marked as obstructed, as described in the previous section. If the cell is obstructed, the weights of the occupied and free hypotheses will be equal, $w_{occupied}(r,c) = w_{free}(r,c) = 0.5$.

The number of particles having the "occupied" hypothesis true is the number of "real" particles in the cell.

$$N_{OC}(r,c) = |\{p_i \in S \mid r_i = r, c_i = c\}| \quad (1.1.19)$$

The number of particles (hypotheses) having the "occupied" value false is the complement of N_{OC}. N_C is the maximum number of particles allowed in a cell, and this number is a constant of the algorithm.

$$N_{FC}(r,c) = N_C - N_{OC}(r,c) \quad (1.1.20)$$

The total posterior probability of a cell being occupied and of a cell being free can be computed from the number of free/occupied hypotheses, and their corresponding weights. In the following equations the row and column parameters are not shown, but they are implied.

$$P_{OC} = \frac{w_{occupied} N_{OC}}{w_{occupied} N_{OC} + w_{free}(N_C - N_{OC})} \quad (1.1.21)$$

$$P_{FC} = \frac{w_{free}(N_C - N_{OC})}{w_{occupied} N_{OC} + w_{free}(N_C - N_{OC})} \quad (1.1.22)$$

The aggregate particle weights P_{OC} and P_{FC} are used for particle resampling. The resampling of the particle population is done at the end of the measurement step, so that the next cycle can start again with an updated population of particles without concerning about their weight.

Resampling

A classical resampling algorithm would make N_C random draws from the previous particle population of a cell, while the weight of each particle controls its chances of being selected. Because the "cell free" hypothesis particles are not relevant, the proposed resampling method will instead decide for each real particle (particle having the occupied hypothesis true) whether it is destroyed or multiplied (and, if multiplied, how many copies of it are created).

The following algorithm describes the process of resampling, which is materialized as duplication or removal of particles from the particle set. The key solution for a real-time operation is that all the heavy computing tasks are executed at cell level, mostly by the use of LUT's, while the particle level processing is kept very light.

Algorithm Resample
For each cell C
Compute N_{OC} and P_{OC}
Compute resampled number of particles N_{RC}
$N_{RC}=P_{OC}N_C$
Compute ratio between actual number of particles and the number of resampled particles

$$f_C = \frac{N_{RC}}{N_{OC}}$$

End For
For each particle p_i
Find corresponding cell C
If (f_C>1) – *number of particles will increase*
F_n = Int(f_C) Integer part
$F_f = f_C$ -Int(f_C) Fractional part
For k=1 **to** F_{n-1}
S.Add(p_i.MakeCopy)
End For
r = random value between 0 and 1
If ($r<F_f$)
S.Add(p_i.MakeCopy)
End if
End if

If (f_C <1) – *number of particles will decrease*
r = random value between 0 and 1
If ($r> f_C$)
S.Remove(p_i)
End if
End if
End For

The system will compute the number of particles that each cell should have after the process of resampling has been completed. The ratio f_C between this number and the existing number of particles in the cell will tell us if the particles have to be duplicated or removed. If f_C is higher than 1, the number of particles has to be increased. The integer part of the difference between f_C and 1 tells us the number of certain duplications a particle must undergo (for instance, if f_C is 2, each particle will be doubled). The fractional part of the difference is used for chance duplication: each particle will have a probability of being duplicated equal to the fractional part of this difference.

If f is lower than 1, the number of particles has to be decreased, by removing some of the particles. Each particle has 1- f_C chance of being eliminated.

At this point the cycle is complete, and the tracking algorithm can process a new frame. Secondary estimations for occupancy, speed, or clustering the cells into objects can be performed at the end of this step.

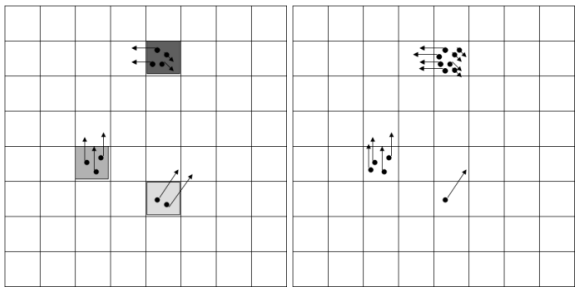


Fig. 1.1.4. Weighting and resampling. The weight of the occupied hypothesis is encoded in the darkness of the cell of the left grid.

1.1.7. Particle system initialization

Although the measurement step takes care of particle creation and deletion, this step only works if there are particles to be duplicated or deleted. For the prediction-measurement cycle to work, the particle population has to be initialized.

From a strictly probabilistic point of view, each cell's state is unknown at startup, which means that the cell has equal probability of being occupied or free. In the proposed tracking system, this would mean that each cell should be assigned a number of particles equal to half the total number of particles allowable in a cell. However, this approach would significantly reduce the speed of the system, and would require permanent re-initialization.

The solution is to use the measurement occupancy grid to create particles. If a measurement cell is of type obstacle, its $p(m(r,c) \mid occupied)$ is high, and there are no particles in the corresponding tracked grid cell, a small number of particles will be created. The initial speed components *vr* and *vc* of the created particles will be sampled randomly from an initial range of possible values, and the initial position is confined to the creation cell. In this way, the initialization is a continuous process.

Particles are automatically removed when they go outside the grid area, in the prediction phase. Another case of "administrative" removal (removal not caused by the probability mechanism) is when, due to particle drifting, the number of particles in a cell exceeds the allowed value.

1.1.8. Individual object detection based on the particle occupancy grid

After each grid cell receives an occupancy probability and a speed estimation, the next step is to use these results for extracting the individual objects in the scene. For this purpose a labelling algorithm able to take advantage of the dynamic properties of the grid is used.

```
Algorithm Labelling
Input: Cell – grid of cells, with occupancy value and speed
Output: Label – grid of labels
Uses Queue – queue of pairs of row, column coordinates
While ( (r,c) = FindUnlabeledOccupiedCell() != NULL)
CurrentLabel = MakeNewLabel()
Label (r, c) = CurrentLabel
Queue.Insert((r,c))
Area (CurrentLabel)=1
MinRow(CurrentLabel)=r;
MaxRow(CurrentLabel)=r;
MinColumn(CurrentLabel)=c;
MaxColumn(CurrentLanel)=c;
While Not Queue.IsEmpty()
        (rr, cc) = Queue.Remove()
        Update MinRow(CurrentLabel)
        Update MaxRow(CurrentLabel)
        Update MinColumn(CurrentLabel)
        Update MaxColumn(CurrentLabel)
        Increment Area(CurrentLabel)
        If Not AreaRatioCheckOk()
                Queue.ForceEmpty()
                Break()
        End if
        For Each (rri, cci) in Neighborhood(rr, cc)
           If UnlabeledOccupiedCell(rri, cci)
```

```
                    If Compatible (Cell(rr, cc), Cell(rri, cci))
                        Label(rri, cci)= CurrentLabel
                        Queue.Insert (Point(rri, cci))
                    End if
                End if
            End For
        End While
        End While
```

The labelling algorithm is based on the generic solution of breadth first exploration of graphs, as the occupied cells can be regarded as nodes in the graph and the neighbourhood relationship between them as edges. In the remaining of this section the main elements of the cell grouping algorithm are described.

The input to the labelling algorithm is the occupancy grid, an array of *Cell*-s, indexed by the row and column. Each cell has the following properties: *occupancy*, ranging from 0 to 1, expressing the probability that the cell is occupied, *speed*, the absolute value of the estimate speed for the cell, and *orientation*, which describes the orientation of the estimated speed vector.

The output of the algorithm will be the 2D array *Label*, having a unique identifier for each connected component, which hopefully will correspond to a real-world object. Initially, this array is initialized with zero.

The algorithm starts by finding occupied cells that have no label assigned to their position yet. A cell is considered to be occupied if the *occupancy* value is above a fixed threshold of 0.5. A new label is generated, and assigned to this initial point. A set of parameters are initialized at this step: the *area*, which is the number of cells assigned to a connected component, and the extreme coordinates of this component, minimum row, minimum column, maximum row and maximum column. These parameters will be updated as the labelling algorithm will gradually cover the whole object.

The main labelling loop will execute as long as the queue is not empty. A coordinate pair (which corresponds to a cell already labelled) is extracted from the queue, and the points in its *Neighbourhood* are analyzed. The size of the neighbourhood is defined by the uncertainty of the stereo measurement, as the neighbourhood is a rectangle of size $2\sigma_{row}$ by $2\sigma_{column}$, centred in the current point.

Each neighbour (rr_i, cc_i) of the current position is tested for compatibility before it is added to the connected component. An ordinary labelling algorithm takes into account only the vicinity relations, but this can sometimes lead to false connection between different, but closely positioned objects, as shown in figure 1.1.5.

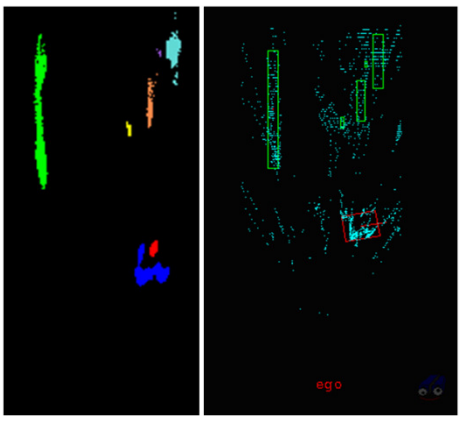


Fig. 1.1.5. Vicinity-only labelling, without taking into account the dynamic cell information. Left – label grid, right – resulted 3D objects.

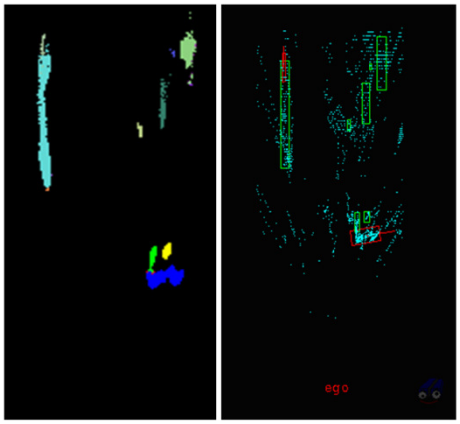


Fig. 1.1.6. Labelling using the speed compatibility criteria.

The compatibility test is used exactly for the purpose of preventing such false associations. The neighbours of the current cell will receive the current label only if the speed characteristics of these cells are similar. Two cells are said to be compatible if:

- The difference in the orientation of the speed vectors in the two cells is less than 30 degrees.
- The difference in speed vector magnitudes is less than 30% of the value of the largest magnitude of the two cells.

Another problem of the classical labelling approach for object extraction is that the algorithm will connect anything as long as the compatibility is valid, and sometimes the resulted objects have a considerable size. Also, most of the huge objects are static, and therefore their orientation is difficult to estimate (extracting the orientation from shape is not very robust). The most problematic case is the one described in figure 1.1.7, when a non-oriented cuboid will be completely unsuited for the large structure.

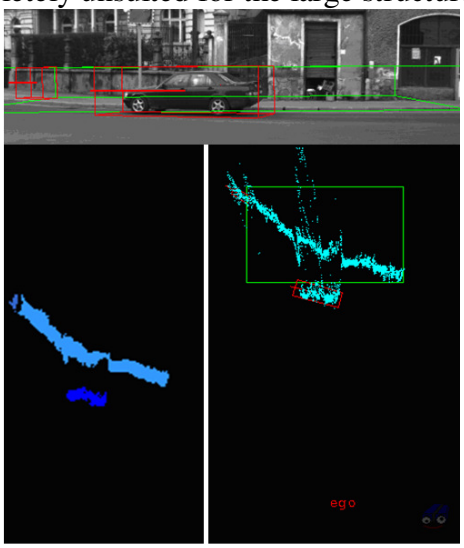


Fig. 1.1.7. Massive static objects.

In order to prevent the generation of such deformed objects, we have designed the *AreaRatioCheck* test. The following coefficient is computed any time a cell position is extracted from the queue, and the area and the coordinate limits are updated:

$$\rho = \frac{A}{(MaxRow - MinRow)(MaxColumn - MinColumn)} \tag{1.1.23}$$

When the object's length or width is above four meters, the area ratio is tested against a 0.5 threshold. This condition means that the labelled object should fill its non-oriented box at least 50 %. If the test fails, the labelling process is interrupted by forced emptying of the queue, and then resumed with a new label. The effect is shown in figure 1.1.8, where the large static object is broken into smaller pieces which depict the real world geometry more accurately.

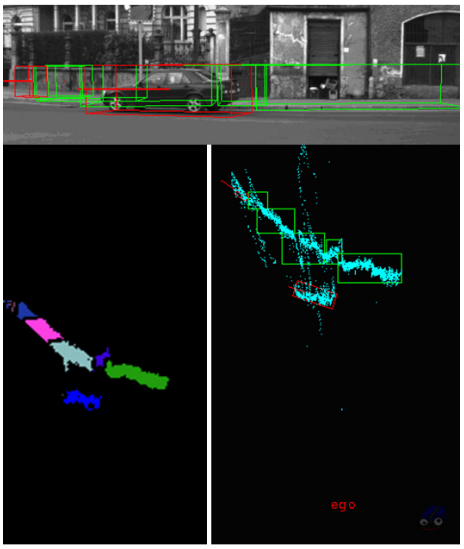


Fig. 1.1.8. Large objects are split into smaller pieces.

After the labelling process is finished, each object is identified by its label *L*. The first property that is computed is the speed vector $\vec{v}_L$. In order to estimate the speed of an object, a weighted average of the speeds of the object's cells is performed, with the occupancy probability fulfilling the role of weight.

$$\vec{v}_L = \frac{\sum_{r,c} P_O(r,c)\vec{v}(r,c)(Label(r,c)=L)}{\sum_{r,c} P_O(r,c)(Label(r,c)=L)} \tag{1.1.24}$$

The magnitude and the orientation of the speed vector are computed, and a static versus dynamic discrimination of the objects is performed (all objects having the speed vector magnitude above 1.5 m/s are dynamic). The orientation of the static objects is not computed, and their extremities are given by the maximum and minimum row and column numbers.

The orientation of the dynamic objects is given by the orientation of their speed vectors. The width and the length of the dynamic objects are computed by computing the distances of the object's labelled grid positions from the axis of orientation, already known.

1.1.9. Experimental results

Qualitative assessment

Qualitative tests, which allow us to monitor the general behavior of the system in complex situations, are performed on video sequences recorded in real urban traffic. These tests show how the occupancy grid is computed, how the speed vector for each cell is

estimated, and how the grid results are grouped into cuboidal objects having position, size, orientation and oriented speed vector. The speed of the cells is displayed in color, using Hue for orientation and Saturation for magnitude. Due to the need for compact representation of the grid results, we have also encoded the occupancy probability as the color's Intensity, making full use of the whole HSI color space.

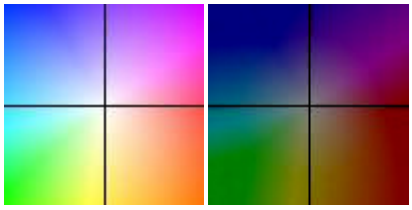


Fig. 1.1.9. Color coding for speed vectors (full and half occupancy).

Video files, describing results in different traffic situations, can be downloaded from this page: http://users.utcluj.ro/~rdanescu/gridtrackingtests.htm . The main qualitative test is the sequence http://users.utcluj.ro/~rdanescu/long_sequence.avi , which shows the results over a significant distance through Cluj-Napoca. Some highlights of this sequence are presented in figure 1.1.10:

a) Crossing pedestrian, mixed with lateral traffic and static distant objects.

b) Incoming vehicle, static lateral scenery.

c) Two incoming vehicles, the most distant one only visible for a couple of frames.

d) Moving vehicle against static wall, ego vehicle performing a sharp turn left.

e) Distant object, accurately tracked.

f) Moving object against static background. The protrusion from the static background near the moving object is actually an occluded stationary car. The ego vehicle is performing a sharp right turn, which causes the instability in the estimation of the static nature of the background in the top right corner. Also, that area was previously occluded by the moving vehicle, which means that the static nature of the cells has not yet been detected, due to the short observation time.

g) Distant crossing vehicle going through stationary vehicles. The ego vehicle is turning right.

h) Tracking a moving target through a narrow corridor of stationary vehicles.

The behavior of the system in the case of occlusions is highlighted by the sequence http://users.utcluj.ro/~rdanescu/cluj-occlusion.avi . Key points from the sequence are presented in figure 1.1.10. While the ego vehicle is performing a sharp left turn, a vehicle comes from our right, and is occluded by a vehicle coming from our left. The occluded vehicle is also maneuvering, changing its heading to its left. While occluded, its particle distribution becomes diffuse, accounting for possible exit trajectories, and the correct heading is quickly identified as the object becomes observable again.

An extensive sequence, recorded while observing an intersection with the ego vehicle standing still, produced the results that are available in the file http://users.utcluj.ro/~rdanescu/wob-occlusion.avi . A highlight of this sequence is shown in figure 1.1.11. A vehicle comes from our right, then turns left and proceeds to exit the scene.

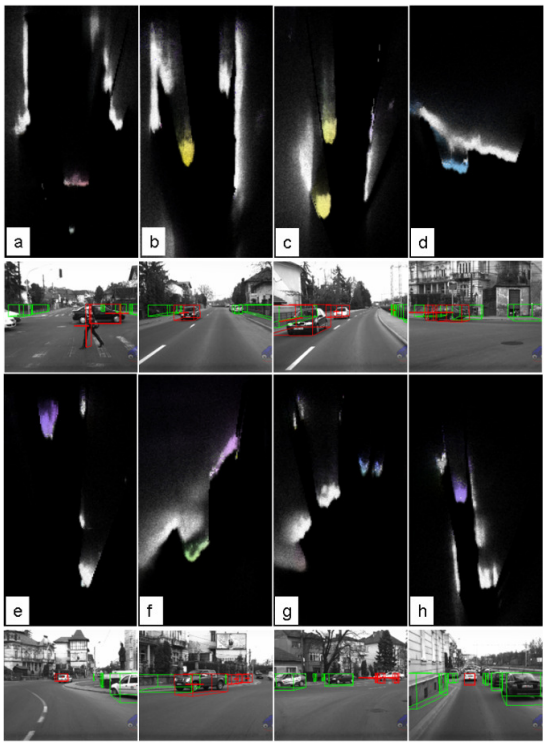


Fig. 1.1.10. Extended sequence in urban traffic – highlights.

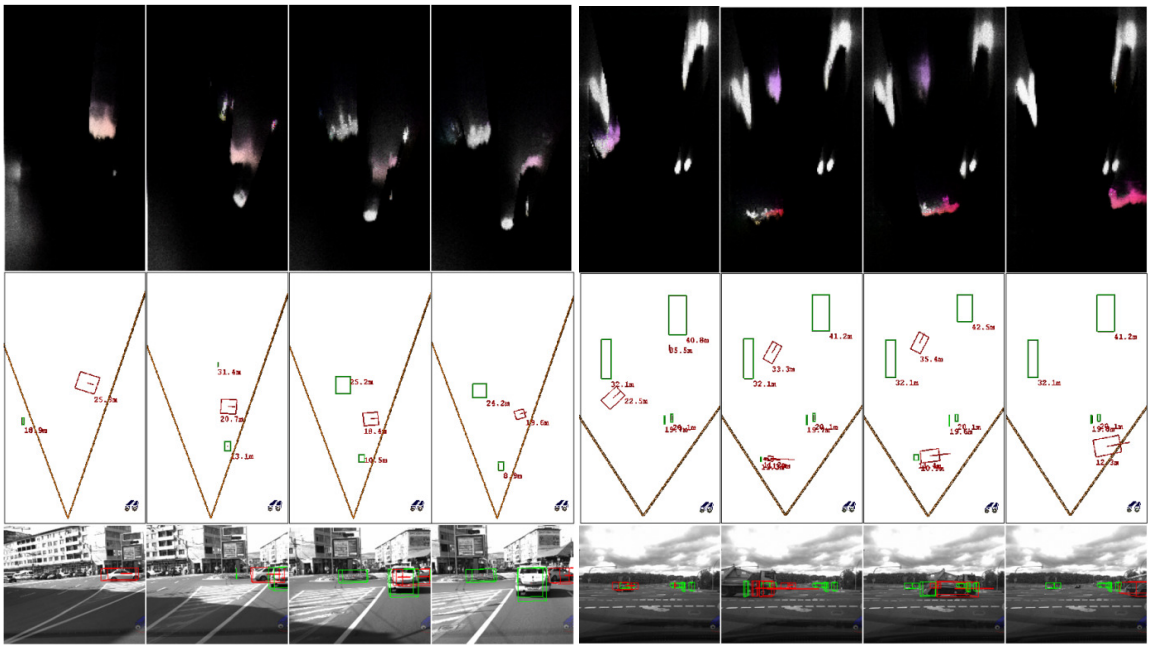


Fig. 1.1.11. Left – handling occlusions, right – handling close objects of different speeds.

During this maneuver it occludes the static object near its left side, but does not become joined with this structure due to the speed-sensitive nature of the cell clustering algorithm. One can see how the occupancy becomes diffused as the object is occluded by a large truck, which then again occludes the static objects on the right.

Numerical evaluation in controlled environment

The numerical evaluation was performed on sequences acquired in controlled scenarios, with known target speed and orientation. We have performed four tests, with the same orientation, -45 degrees, but different speeds, 30 km/h, 40 km/h, 50 km/h, 60 km/h. The

results that were evaluated are the estimated speed and orientation of the 3D cuboid resulted from clustering the occupied grid cells. These results are compared to the ground truth, and they are also compared to the results of another means of intermediate extraction of 3D dynamic information, the optical flow combined with stereovision. The results of optical flow that are taken into consideration are the speed and orientation of the 3D cuboid obtained from grouping the points having 3D and speed information [12]. The controlled test sequence is highly favorable to the optical flow approach, as the vehicle is clearly visible, has plenty of features that can be matched from one frame to another, a situation which provides plenty of good speed vectors to be averaged into an accurate vector of the cuboid.

The results of speed and orientation estimation are displayed in the graphs shown in figures 1.1.13 to 1.1.16. The grid tracking results are shown with the red dotted line. One can see that both methods quickly converge towards the ground truth, but the grid tracking results are more stable (lower error standard deviation) and more accurate (lower mean absolute error). This fact is confirmed by the tables 1.1.1 and 1.1.2.

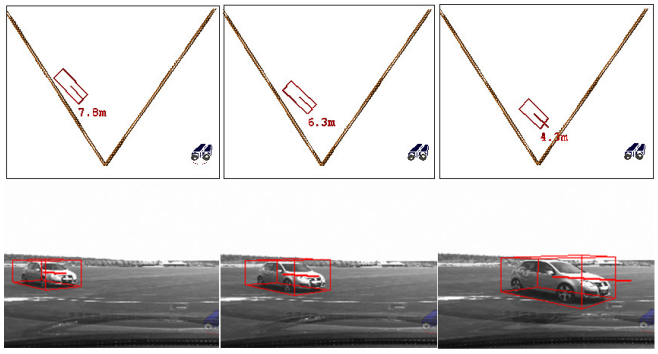


Fig. 1.1.12. Controlled test sequence.

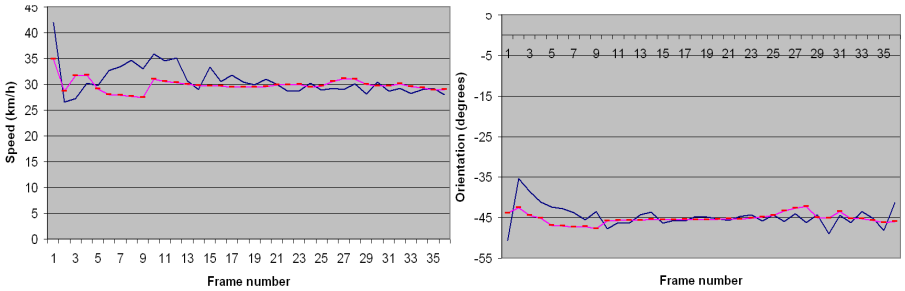


Fig. 1.1.13. Speed and orientation estimation, 30 km/h test.

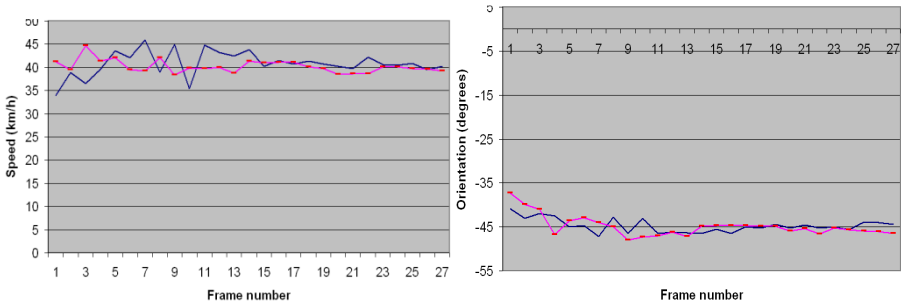


Fig. 1.1.14. Speed and orientation estimation, 40 km/h test.

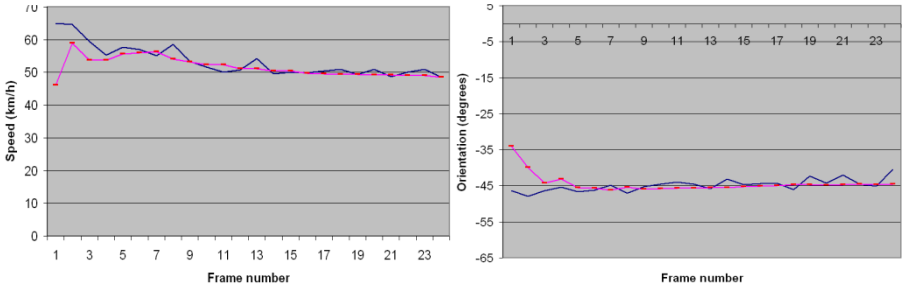


Fig. 1.1.15. Speed and orientation estimation, 50 km/h test.

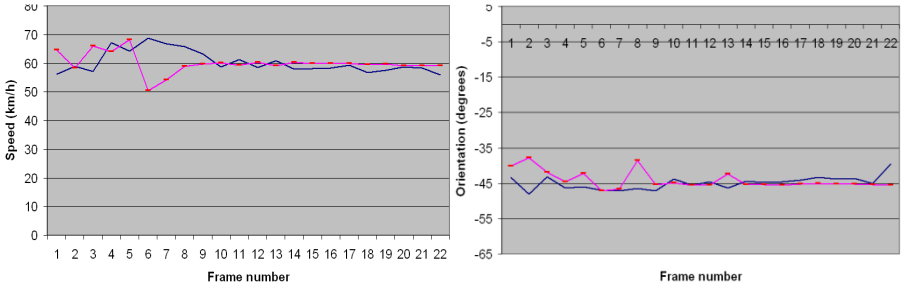


Fig. 1.1.16. Speed and orientation estimation, 60 km/h test.

TABLE 1.1.1
NUMERICAL RESULTS – SPEED ESTIMATION ACCURACY

Speed of target	Particle grid MAE	Particle grid STDEV	Optical flow MAE	Optical flow STDEV
30 km/h	0.9016	0.9731	2.0141	2.3087
40 km/h	1.0184	0.9730	2.1181	1.9017
50 km/h	2.4989	2.3370	3.7329	4.4966
60 km/h	2.1279	1.3858	3.0677	2.2725

TABLE 1.1.2
NUMERICAL RESULTS – ORIENTATION ESTIMATION ACCURACY

Speed of target	Particle grid MAE	Particle grid STDEV	Optical flow MAE	Optical flow STDEV
30 km/h	0.9728	0.8376	1.8219	2.0122
40 km/h	1.0321	0.8616	1.1962	1.0146
50 km/h	0.4695	0.2659	1.2775	1.1095
60 km/h	0.9343	0.6739	1.4554	1.1634

The time performance depends on the obstacle load of the scene, which influences the total number of particles. For a typical urban scene, and a total number of particles in a cell N_C=50, the total running time is about 40 ms per frame, on an Intel Core 2 Duo processor at 2.1 GHz. Due to the fact that the particle tracking system shares the processor with other sensorial processing algorithms such as lane detection, object classification and so on, the total frame rate is about 10 fps.

1.1.10. Conclusion

This chapter described a solution for dynamic environment modeling and tracking, which employs particles in order to estimate the occupancy and speed of the cells of an occupancy grid. This flexible and real-time solution is capable of correctly track dynamic environments even at high relative speeds, without the need of a very high frame rate from the measurement system. The test sequences prove that the method is sensitive enough to detect and estimate the speed of a pedestrian, but also the speed of a fast moving vehicle. The accuracy of the speed and orientation estimation is proven by the tests conducted in controlled situations.

The particle grid tracking solution is an elegant extension of the dynamic occupancy grid solutions that were surveyed. The particle population approach relieves the designer of the choice of a speed probability distribution for each cell, and can handle multiple divergent speed hypotheses. Also, the speed distribution does not have to be estimated, and the measurement data only controls the creation or deletion of particles. The proposed technique is a new view of the occupancy grid problem, a view oriented towards practical implementation, and a view that can open the door to interesting extensions.

1.2. The particle based dynamic elevation map

1.2.1. Introduction

While the dynamic occupancy grids are a reasonable representation for a dynamic 3D environment if the interested entity (robot, vehicle) moves on a planar surface, and only cares for the environment elements from the obstacle / not obstacle point of view and, if obstacle, the dynamic characteristics of it may be of interest, sometimes a more detailed freeform description of the environment is necessary.

Digital elevation maps (DEM) are a simple yet powerful way of modeling complex 3D environments. The environment is represented as a 2D grid, each cell in the grid being described by its height. The digital elevation maps can be large data structures, used for terrain mapping [13], a function which makes them extremely useful for planetary exploration tasks [14], but they can also be local structures, used for robotic navigation [15], environment representation for driving assistance systems [10], or even indoor pedestrian tracking [16]. The digital elevation maps can be built in real time, using multiple types of 3D sensors, the most popular being of the laser [17] and of the stereo vision [10] [16] family. The cells of the elevation map can be then analyzed and classified into traversable, obstacles and others [10] [15].

The elevation map representation of the environment is sometimes described as having 2.5 dimensions [18], because the description is not complete – bridges and tunnels, for example, cannot be fully represented. For this reason, researchers have proposed several extensions. One of the problems of elevation maps, described in [18], is that when they are built using the average (or maximum) height of the sensorial points in each grid cell, structures such as bridges and tunnels will appear as non-traversable. Assuming the overhanging structure is of no concern, the same paper presents an optimized map building algorithm which looks for gaps in the vertical structures and generates the map of the drivable surface below. In [19], we find a further extension of the elevation map, called Multi Level Surface Map, which can successfully model the overhanging structures. The environment is organized as a 2D map, but instead of storing occupancy or height, each cell stores a set of surface patches, which are defined as Gaussian distributions of height and depth. This way, the surface under a bridge will be one surface patch, and the bridge itself another. The heights are defined by their mean and standard deviation, which are updated in a probabilistic fashion. An even more general extension, presented in [20], is the multi-volume occupancy grid, which is a probabilistic representation of height volumes for each map cell, each volume having a starting position from the ground and a height, the crucial difference being that the volume can be either occupied or empty, and the occupancy being a probability value. This way, free and occupied volumes can be modeled, and uncertainty can be associated. Another solution that combines elevation and occupancy, using the uncertainty element (called “credibility”), can be found in [21].

All these elevation map solutions available in the literature are, unfortunately, static. The purpose of my work was to devise a workable solution for an elevation map capable of modeling dynamic environments, and to devise a tracking algorithm for updating this map. Inspired from the previous results regarding the particle-based occupancy grid, a solution for dynamic elevation map modeling and tracking based on particles is proposed.

The unified modeling and tracking solution is based on particles that are not simply state hypotheses, as in classical particle-based tracking solutions, but are the building blocks of the 3D world. The particles can move from one cell to another, providing an elegant and intuitive mechanism for prediction, and can be created and destroyed based on their agreement with the measurement data (the state update). In the previous chapter, this particle mechanism was used for modeling and tracking dynamic occupancy grids. The moving particle, however, can carry many items of information: speed, position, and height. A population of particles having speeds and heights becomes a probabilistic model for a fully dynamic elevation map.

1.2.2. Proposed world model: the particle-based dynamic elevation map

An elevation map representation of a 3D scene in the coordinate system *XYZ* (consider a coordinate system with the origin on the ground in front of the vehicle, the *X* axis pointing forward, the *Y* axis pointing to the left, and the *Z* axis pointing up) is a function $Z(X,Y)$, which assigns to every point (X,Y) of the horizontal plane *XOY* a height coordinate *Z*. This continuous mapping is further approximated by dividing a finite region of the *XOY* plane into cells, each cell i being identified by a position in a finite matrix, described by a row r_i and a column c_i. A height value h_i is assigned to each cell, and thus the elevation map approximation becomes an array of height values.

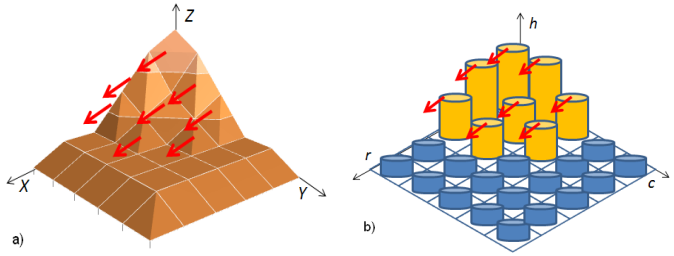


Fig. 1.2.1. a) The dynamic elevation map, a 3D surface with attached speed vectors; b) the continuous surface is approximated by a grid of fixed size cells, with heights and speeds for each cell.

If the 3D scene is dynamic, each cell of the discrete elevation map can have an assigned speed. If the application is limited to the driving scenario, it can be assumed that the objects in the scene move mainly in the horizontal plane, and the speed vector has only two components, v^x and v^y. Thus, in the continuous case we have two functions, $v^x(X,Y)$ and $v^y(X,Y)$, and in the discrete case we can speak of v^r_i and v^c_i – the row speed and the column speed for each cell i in the map, as shown in Fig. 1.2.1.

Thus, a dynamic digital elevation map can be described by three arrays of values, h_i, v^r_i and v^c_i. In an ideal world, all these values can be measured, and an accurate world description can be generated. In the real world the sensors have limited range, limited precision, limited reliability, and all these problems lead to some cells of the map to be unobservable, or to have a poor measurement of their height. All these limitations cause uncertainties, and these uncertainties have to be represented in the world model. Thus, instead of computing single values for height and speed components, the system must compute probability densities. A cell i in the dynamic elevation map is associated to a random variable

$\mathbf{X}_i = (h_i, v_i^r, v_i^c)^T$, which has three dimensions (height, row speed, and column speed). The objective of the tracking algorithm is to compute the probability density of $\mathbf{X}_i$, for each cell *i* in the map, based on a sequence of measurements $\mathbf{Z}(0)\ldots\mathbf{Z}(t)$. The measurement $\mathbf{Z}$ includes all available sensorial data for the time instant *t*, not limited to the current cell.

$$p(\mathbf{X}_i(t) \mid \mathbf{Z}(0),\mathbf{Z}(1),...,\mathbf{Z}(t)) \propto p(\mathbf{Z}(t) \mid \mathbf{X}_i(t))p(\mathbf{X}_i(t) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1)) \quad (1.2.1)$$

The tracking problem is formulated as a Bayesian recursive estimation of probability densities, as described by (1.2.1). The past state density $p(\mathbf{X}_i(t-1) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1))$ and the state transition model $p(\mathbf{X}_i(t) \mid \mathbf{X}(t-1))$ are combined to form the predicted state density $p(\mathbf{X}_i(t) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1))$, and the sensorial information at time *t* is used to update the state through the observation model $p(\mathbf{Z}(t) \mid \mathbf{X}_i(t))$.

Assuming that only the immediate past matters (the first order Markov model assumption), the prediction for a cell *i* can be computed from the past estimated states of all the cells *j* in the grid, $p(\mathbf{X}_j(t-1) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1))$, and the dynamic model $p(\mathbf{X}_i(t) \mid \mathbf{X}_j(t-1))$.

$$p(\mathbf{X}_i(t) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1)) = \sum_j p(\mathbf{X}_i(t) \mid \mathbf{X}_j(t-1))p(\mathbf{X}_j(t-1) \mid \mathbf{Z}(0),...,\mathbf{Z}(t-1)) \quad (1.2.2)$$

The most commonly used techniques for approximating probability densities in tracking applications are the Gaussian function (mostly used in Kalman filtering) and the particle (sample) set of values. A description of the most popular solutions for representing and tracking probability density functions (PDF) can be found in [22]. The particle-based solutions are preferred when the PDF is multi-modal or its shape is not known a priori. Another reason why this work is based on particles is that a mechanism for moving them from one cell to another can be intuitively defined.

The dynamic elevation map will be described, at time *t*, by a set *S*(*t*)of particles, each particle *k* being described by a state vector $\mathbf{x}_k(t)$:

$$S(t) = \{\mathbf{x}_1(t), \mathbf{x}_2(t), ..., \mathbf{x}_{N_S(t)}(t)\}, \text{where}$$
$$\mathbf{x}_k(t) = ({}_p c_k(t), {}_p r_k(t), {}_p h_k(t), {}_p v_k^c(t), {}_p v_k^r(t))^T \quad (1.2.3)$$

Each particle *k* is located in the grid cell identified by the row ${}_p r_k$, and the column ${}_p c_k$. The grid is a map of 250 rows x 120 columns, and each cell in the grid is a rectangle of 20 cm x 20 cm. Thus, the grid spans a surface of 50x24 meters in the horizontal (*XOY*) plane. Each particle represents a hypothesis of the state of the cell: a possible height ${}_p h_k(t)$, a possible forward speed ${}_p v^r{}_k(t)$ and a possible lateral speed ${}_p v^c{}_k(t)$, as depicted in Fig. 1.2.2. The row, column and speed of a particle are expressed as multiples of the cell size D_X and D_Y (currently 20 cm), and the height is expressed as a multiple of the height element of size D_H (currently D_H=1 cm).

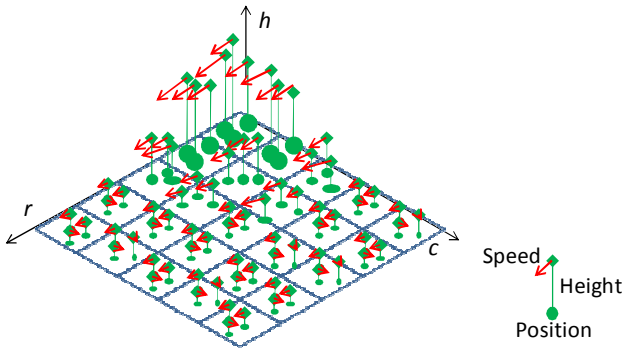


Fig. 1.2.2. The particle dynamic elevation map. Each cell has a population of particles, each particle having height and speed. The particle population can approximate a multi-modal probability distribution of heights and speeds for each cell in the map.

Based on the particle set $S(t)$, the probability densities involved in the tracking process can be approximated. The multi-modal probability density of the state of a cell i is derived from the particles whose position ${}_pc_k$ and ${}_pr_k$ coincides with the row and column of the cell i, r_i and c_i.

The dynamic model is described by (1.2.4). Assuming that the past state of a cell j is described by the particle value $\mathbf{x}_k(t-1)$, the current state can be described by a sample drawn from a normal distribution centered in $\mathbf{f}(\mathbf{x}_k(t-1))$ and having a covariance matrix $\mathbf{Q}_i(t)$. The function $\mathbf{f}$ encodes the uniform motion model of a particle and the translation and rotation motion of the observation platform, while the uncertainty matrix encodes the possible differences between the assumed models and the real world.

$$p(\mathbf{X}_i(t) \mid \mathbf{X}_j(t-1) = \mathbf{x}_k(t-1)) \approx N(\mathbf{f}(\mathbf{x}_k(t-1)), \mathbf{Q}_i(t)) \tag{1.2.4}$$

The prediction described by (1.2.2), based on the past state and the dynamic model, will take the form of altering the position and velocity of all particles, by applying the motion model equation $\mathbf{f}$ (a process called particle drift) and adding random quantities controlled by the matrix $\mathbf{Q}_i(t)$, a process called particle diffusion.

The measurement model $p(\mathbf{Z}(t) \mid \mathbf{X}_i(t) = \mathbf{x}_k(t))$ is defined for each cell i, and depicts the probability density of the measurement $\mathbf{Z}(t)$ under the assumption that the state of the cell is described by the particle k. This density is assumed to be a normal distribution centered in $(r_i, c_i, {}_ph_k(t))$ and having an error covariance matrix $\mathbf{\Sigma}_i(t)$, which describes the uncertainty of the sensor:

$$p(\mathbf{Z}(t) \mid \mathbf{X}_i(t)) = \mathbf{x}_k(t)) \approx N((r_i, c_i, {}_ph_k(t))^T, \mathbf{\Sigma}_i(t)) \tag{1.2.5}$$

The measurement model based update, described by (1.2.1), will take the form of assigning to each particle a weight proportional to the agreement between the particle's state and the measurement. This weight is not included in the proposed world model, because as soon as the particles are weighted they are re-sampled, a process which generates a new particle population from the weighted one, the particle's weight controlling the chances of it being replicated [23]. Thus, the new, weight-free particle population encodes the updated probability density.

1.2.3. Overview of the tracking algorithm

The purpose of the dynamic elevation map tracking algorithm is to continuously estimate the probability density for the height and speed of each cell in the grid. As the probability densities are represented by particles, the purpose of the tracking algorithm is to continuously update the particle population of the scene, a process driven by the measurement data. A flowchart of the tracking algorithm is presented in Fig. 1.2.3. The main tracking steps follow the drift-weight-resample mechanism of the particle filter variant called CONDENSATION, described in [23].

The first step of the tracking cycle is the *Particle Drift*, a process that takes the particle population resulted at the end of the previous cycle and applies the motion equations of the host vehicle and of the particles themselves in order to predict their present positions. The particles are moved from one cell to another due to the motion of the host vehicle relative to the observed scene, and due to the speed values of the particles themselves.

After drift, the particles are subjected to the process of *Diffusion*. The states of the particles (position, height, and speed) are altered by small random amounts, which reflect the uncertainties that affect the evolution of the scene in time (or the difference between how a real scene alters its state as the time passes, and the way we predict the evolution of states). *Drift* and *Diffusion* form the prediction, preparing the particle population to meet the measurement and be altered by it.

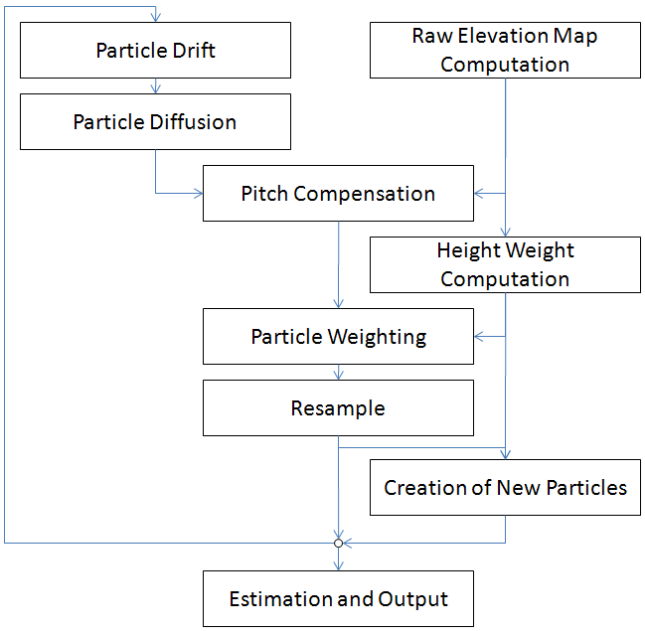


Fig. 1.2.3. Dynamic elevation map tracking algorithm.

The measurement comes in the form of a *Raw Elevation Map*, a static height map of the same size as the one that is tracked, derived directly from dense stereo information processing. This raw map is affected by the sensor-specific errors, which need to be taken into consideration.

The first collision between the particle population and the measurement data is in the process of *Pitch Compensation*. The pitch angle can change quite abruptly, in an unpredictable way, due to imperfections in the road surface. Changes of this angle affect the height of the cells in the map significantly, and for this reason the system must estimate a pitch difference between frames, and adjust the particle heights to the new pitch.

After the pitch-based height adjustment, the particles are subjected to *Particle Weighting*. This process assigns to each particle a weight which reflects the quality of the match between the height of the particle in a specific cell and the measurement height of the raw elevation map. This process must take into account the specific uncertainties of stereovision (the observation model). For computation speedup, the probabilistic observation model is built as a height weight look-up table for each grid cell, a process called *Height Weight Computation*. Then the particle weighting process becomes a simple assignment of a value from a Look-Up Table (LUT).

After each particle receives a weight based on their agreement with the measurement data, a new population of particles is generated for each cell, in the process of *Resampling*. The weight of a particle influences the chances of this particle to be selected. This way, the particles having the height closer to the height of the measurement survive and multiply, while the others are destroyed.

If a cell in the grid has very few particles (or none), the measurement data, already pre-processed in the form of a height weight LUT, will be used to *create new particles*, which will have random speeds, and a height distribution consistent with the height weight LUT.

With the updated particle population, the system is ready for the next tracking cycle. While the result of the tracking process is the particle population itself, the useful result is a dynamic elevation map, containing a height value and a speed vector for each cell. The *Estimation and Output* step will compute these values, and will generate a scene description using a popular 3D modeling language, for analysis and visualization.

1.2.4. Algorithm description

Particle Drift and Diffusion

The state transition probability model is implemented by the deterministic drift and the stochastic diffusion. The deterministic drift changes the state of the particles by taking into account two factors: the movement of the observing vehicle, which causes a relative movement of the whole scene in the vehicle's coordinate systems, and the movement of the mobile particles, according to their speed. The observing vehicle's movement in the horizontal, *XOY* plane, can be computed from its speed *v*, and its yaw rate $\dot{\psi}$, which are read from the CAN bus, and which are integrated over the time interval between two measurement frames, Δt.

The speed vector of the particle, consisting of the two components ${}_pv^c{}_k(t\text{-}1)$ and ${}_pv^r{}_k(t\text{-}1)$, is also related to the observing vehicle's coordinate system. For this reason, when the observing vehicle rotates, the speed vector of the particle must rotate in the opposite direction, so that its direction in the scene remains unchanged.

After the observation platform motion corrected particle positions and speeds are computed, the drift process is completed by adding the particles' motion caused by their own speed. After drift, the particles are subjected to diffusion. The state of each particle is altered by random quantities $\delta_p c(t)$, $\delta_p r(t)$, $\delta_p h(t)$, $\delta_p v^c(t)$ and $\delta_p v^r(t)$, drawn from a normal distribution of zero mean and experimentally adjusted covariance matrix $\mathbf{Q}_i(t)$, depicting the state transition uncertainty.

After the drift and diffusion are applied for each particle in the scene, a final step is to ensure that each cell in the grid has a number of particles less or equal to N_C, the maximum allowed number of particles in a grid cell (a constant of the system, currently 200). For this reason, if prediction assigns to a cell more particles than the maximum allowed number, excess particles are destroyed. The destruction process is random, having no preference for existing particles in the cell or for newcomers.

The Sensorial Information: the Raw Elevation Map

The main source of sensorial data for elevation map tracking is a dense stereovision system [24], which is able to extract 3D information for the (mildly) textured areas in the stereo image pair. The 3D points are subsequently assigned to corresponding cells in the *XOY* grid, the height of a grid cell *i* being the *Z* coordinate of the highest point assigned to the cell. The density of 3D points per cell is also computed, and used for basic validation, under the assumption that road cells will have a lower point density than obstacle cells. This validation allows the elimination of erroneously high elevation values, which are mostly caused by stereovision mismatches. A detailed description of the raw elevation map computation technique can be found in [10]. For the elevation map tracking algorithm, the following items of information from the raw elevation map are used:

- Measurement height of each cell i, denoted by z_i. For convenience, the heights are organized as a 2D array of values that can be accessed by specifying the row and the column coordinate, thus z_i is also written as $z(r_i, c_i)$.
- Data availability for each cell, denoted by d_i. d_i=1 means that height for this cell is available, and d_i=0 means that no measurement data is available for cell i. For convenience, the d_i values are organized as a 2D array that can be accessed by specifying the row and the column coordinate, thus d_i is also written as $d(r_i, c_i)$.

Due to the fact that not all pixels in the image will obtain 3D coordinates from the stereovision engine, and not all areas in the grid are visible, due to occlusions and limited field of view, not all cells in the raw map have a measured height. The tracking algorithm is made aware of this situation by d_i. Fig. 1.2.4. shows an example of raw elevation map.

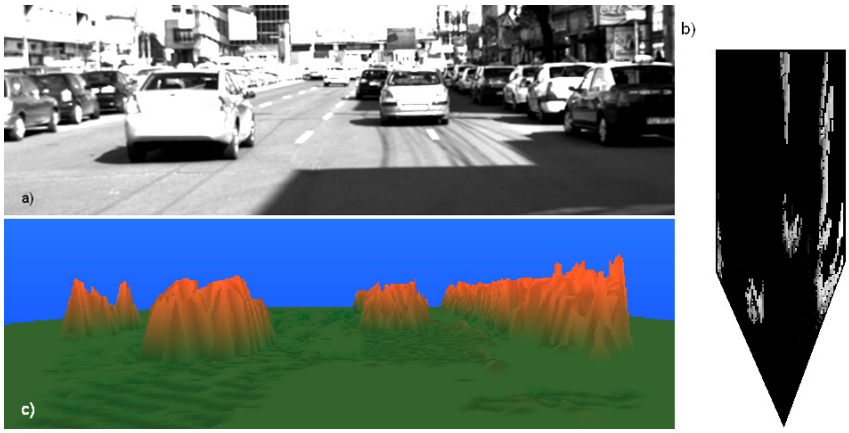


Fig. 1.2.4. The raw elevation map, extracted from dense stereovision: a) original grayscale image; b) top view of the grid, with heights encoded as grayscale values. The sensorial information covers only a part of the world map, and the areas that are not sensed are depicted in light gray; c) 3D representation of the raw map. The non-textured cells represent missing height data in the raw map, due to field of view limitations or errors of stereo reconstruction, such as those caused by dark shadows.

Pitch Angle Compensation

According to the world model described in section 1.2.2, a particle k is located in the elevation grid at the row coordinate ${}_pr_k$ and column coordinate ${}_pc_k$, and has a height ${}_ph_k$. Due to the continuous nature of the observation process, the heights of the map are not expected to change abruptly from one frame to another, except when the observation vehicle performs a pitching motion. The effect of a pitch angle variation between two frames, $\Delta\alpha$, translates into

a difference between the height of a particle and the measured height for the cell containing the particle:

$$\Delta\alpha_k = \tan^{-1}\left(\frac{({}_p h_k - z({}_p r_k, {}_p c_k))D_H}{D_X \, {}_p r_k}\right) \tag{1.2.6}$$

Naturally, (1.2.6) is valid only when the particle is part of a static structure, and its height is close to the true height of the structure in the scene. These conditions are not valid in the general case, but we can make several assumptions:

- The vast majority of the particles in the scene belong to static structures. These structures include the road surface.
- The average height in each cell is close to the real one, even if the individual particles have significant deviations from this height.
- The changes in height due to motion in the scene are minor compared to the abrupt changes due to pitching.

Based on these assumptions, equation (1.2.6) can be averaged for all particles located in cells that have valid data, and a reasonable estimate of the pitch variation between frames can be computed.

$$\Delta\alpha = \frac{\sum_{k=1}^{N_S} \tan^{-1}\left(\frac{({}_p h_k - z({}_p r_k, {}_p c_k))D_H}{D_X \, {}_p r_k}\right) d({}_p r_k, {}_p c_k)}{\sum_{k=1}^{N_S} d({}_p r_k, {}_p c_k)} \tag{1.2.7}$$

After the pitch angle variation is estimated, the height ${}_p h_k$ of each particle k can be corrected:

$${}_p h_k = \frac{({}_p h_k D_H - \Delta\alpha \, {}_p r_k D_X)}{D_H} \tag{1.2.8}$$

Height Weight Computation and Particle Weighting

The process of particle weighting is the particle filter instantiation of the measurement (observation) model $p(\mathbf{Z}(t) \mid \mathbf{X}_i(t) = \mathbf{x}_k(t))$. This model describes the conditional probability density of the measurement $\mathbf{Z}(t)$ given a possible cell state value $\mathbf{X}_i(t) = \mathbf{x}_k(t)$. A particle is a state hypothesis, and the probability of the measurement given this state will be encoded as a particle weight, which will describe how well the particle hypothesis matches the measurement data.

The sensorial information, the raw elevation map, is just a conveniently modified version of the stereovision-derived 3D point data, therefore the probabilistic observation model will be derived from the observation model of stereovision, a three-dimensional normal distribution centered in the real world 3D coordinate, and having a covariance matrix defined by the distance error standard deviation σ_X, the lateral error standard deviation σ_Y and the vertical error standard deviation σ_z.

The expected error standard deviations of the three coordinates computed by a stereo reconstruction process depend on the system's baseline (distance between cameras) b, the

focal distance in pixels f, and disparity computation uncertainty (matching uncertainty) σ_d. The stereo reconstruction process is seen as a non-linear transformation of the vector (u,v,d), containing a point's position (u,v) in the image space and the disparity d, into the vector of 3D coordinates (X,Y,Z). The error is thus computed by propagating the covariance matrix of (u,v,d) through the Jacobian linearization of the 3D reconstruction transformation [25].

After the measurement uncertainties for each cell are computed, they can be transformed into weights that will be assigned to the particles. The measurement model is depicted as a Gaussian distribution centered in the true row, column and height, having the covariance matrix formed by the three standard deviations, σ_r, σ_c and σ_h.

The classical approach for particle weighting, in this situation, is to compute the distance from the particle's position and height $({}_pr_k, {}_pc_k, {}_ph_k)$ to the measurements (r, c, z) inside an acceptable search zone, and transform this distance into a probability value using the Gaussian equation. This approach is not computationally efficient. Instead, the process of *Height Weight Computation* creates, for each cell i in the map, a weight LUT for all possible heights (which are represented as integers, multiples of 1 cm, thus a LUT size of 300 positions will accommodate all relevant heights found in the scene).

The creation of the weight LUT is a data-driven approach, which will approximate the computation of weight by distance to the measurement in a more computationally efficient way. For each cell i, an influence region of $4\sigma_{r,i}$ x $4\sigma_{c,i}$ around the central position (r_i, c_i) is analyzed. Each measured height z inside the search region will receive as weight the value of a Gaussian function G_i centered in (r_i, c_i), having the standard deviations $\sigma_{r,i}$ and $\sigma_{c,i}$. This is a straightforward application of the Gaussian observation model, in the horizontal plane. Formally, the histogram value for a height candidate h, at coordinates (r_i, c_i), for the cell i, is computed as:

$$H_i(h) = \sum_{\tau=r_i-2\sigma_{r,i}}^{r_i+2\sigma_{r,i}} \sum_{\kappa=c_i-2\sigma_{c,i}}^{c_i+2\sigma_{c,i}} d(\tau,\kappa) G_i(\tau - r_i, \kappa - c_i)\delta(z(\tau,\kappa) - h) \qquad (1.2.9)$$

In (1.2.9), d is the data availability map, and δ is the Kronecker delta function. The multiplication terms $d(\tau,\kappa)$ and $\delta(z(\tau,\kappa) - h)$ indicate that only the valid cells in the raw map, having the height z equal to the height candidate h will be taken into consideration.

The process of height LUT creation based on the available height data and the associated Gaussian weights is depicted in figure 1.2.5.

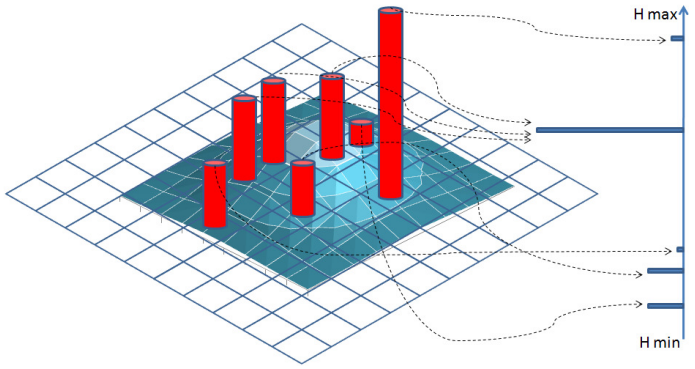


Fig. 1.2.5. Building the height histogram for a cell. A candidate height, located inside the stereo uncertainty region, will have a vote weighted by the value of the 2D Gaussian kernel centered in the current cell.

According to the measurement model, the particle's weight will depend on its distance to the measurement along all three coordinate axes. Equation (1.2.9) only accounts for the displacement in the horizontal plane. The distance between the particle's height and one of the heights in the histogram H_i can be transformed into a probability value by using a Gaussian function. The same result can be achieved more efficiently by convolving the height histogram H_i with a 1D Gaussian kernel K_i, of standard deviation $\sigma_{h,i}$. Equation (1.2.10) transforms the sparse height histogram H_i into a continuous weight LUT, W_i:

$$W_i = K_i * H_i \tag{1.2.10}$$

A particle k, in a cell i, will get the following weight:

$$w_k = W_i(h_k) \tag{1.2.11}$$

Resampling

The resampling process creates a new population of particles, using the current population and their weights. This is the process that makes the particle population reflect the posterior probability of the state of the scene that is tracked, a probability which is the result of the prediction (drift and diffusion) and of the measurement (weighting).

Resampling is applied for each cell i, at each time instance t, after the particles are weighted. The total number of allowed particles for a cell is N_C, a constant which is currently set at 200. The real number of particles in the cell, $N_{R,i}$, resulted by drift and diffusion, may be lower than N_C. For re-sampling, it is assumed that the cell holds a higher number of particles, $N_A = 1.25\ N_C$. The difference between the real number of particles in the cell, $N_{R,i}$, and the augmented maximum number of particles N_A is the number of "empty" particles, particles which are in fact empty places. The re-sampling mechanism will perform the following steps:

1. Weight the empty particles with a default low weight, which we chose to be the average value of the height weight LUT, W_i.
2. Normalize the weights of all N_A particles so that their sum becomes 1.
3. Perform N_C random extractions from the total particle population, real and empty. The weight of the particle controls its chances of being selected – high weight particles will be selected multiple times, lower weight ones may not be selected at all. If empty particles are selected, the final number of particles in the cell will be lower than N_C.

The resampling mechanism completely replaces the particle set at each measurement time t (each frame). Therefore, there is no need for particle deletion, as the particles with low weight (the unfit particles) will not be selected, and thus they will be automatically removed.

This resampling algorithm differs slightly from the classical solution [23] due to the presence of empty particles. The effects of having empty particles are the following:

1. If most of the real particles of a cell have a low weight due to their lack of fitness to the measurement data, the particle population in that cell decreases.
2. The difference between N_A and N_C ensures that even if a very good fit between the particles and the measurement data occurs, there is always a chance to make some room for new particles.

The main reason for designing the mechanism for reducing the number of particles in a cell, a mechanism that acts in an accelerated manner when the data does not fit the prediction, is the need for a mechanism to clear the way for the particles belonging to moving objects. If a vehicle moves across a road surface, the particles that have a low height must be cleared so that the particles of the moving object have a place to go. Without an accelerated

elimination of the unfit particles, the convergence of the particle population to the new height of the obstacle will be slow, and the system will not react quickly enough to a dynamic scene.

Creation of New Particles

If the number of particles in a cell, $N_{R,i}$, is lower than $N_C/2$, and the cell i has a valid height $z(r_i,c_i)$ in the raw measurement map, the algorithm will create a number of $N_C/2$-$N_{R,i}$ new particles. The speeds of the new particles will be sampled from a normal distribution centered in 0, and the heights will be sampled from the multi-modal distribution of height measurement influencing cell i, represented by the weight LUT W_i. The process of creating new particles is applied after each resampling step.

Estimation

If the particle population in a cell i is higher than a threshold that we set at $2N_C/3$, the height and speed of the cell can be estimated by averaging the values of all particles k that are located inside the cell, using equations (1.2.12) and (1.2.13). The operator $|S|$ denotes the cardinality of a set S. The particles involved in the estimation are those generated by resampling, therefore the particle weights are no longer needed.

$$h_i = \frac{\sum_{\mathbf{x}_k \in S,\, {}_p c_k = c_i,\, {}_p r_k = r_i} {}_p h_k}{|\{\mathbf{x}_k \in S \mid {}_p r_k = r_i,\, {}_p c_k = c_i\}|} \tag{1.2.12}$$

$$(v_i^c, v_i^r) = \frac{\sum_{\mathbf{x}_k \in S,\, {}_p c_k = c_i,\, {}_p r_k = r_i} ({}_p v_k^c,\, {}_p v_k^r)}{|\{\mathbf{x}_k \in S \mid {}_p r_k = r_i,\, {}_p c_k = c_i\}|} \tag{1.2.13}$$

The estimated dynamic elevation map is then transformed into a Virtual Reality Modeling Language (VRML) scene [26], for visual inspection, as seen in figure 1.2.6.a and 1.2.6..b. From the tracked elevation map one can also derive an occupancy grid, the occupancy probability for each cell being computed as the ratio between the number of particles having a height higher than a threshold T and the total number of particles in the cell. Figure 1.2.6.d shows the occupancy grid derived from the dynamic elevation map, for a sample threshold T=50 cm.

$$P_{Occ}(i) = \frac{|\{\mathbf{x}_k \in S \mid {}_p r_k = r_i,\, {}_p c_k = c_i,\, {}_p h_k > T\}|}{|\{\mathbf{x}_k \in S \mid {}_p r_k = r_i,\, {}_p c_k = c_i\}|} \tag{1.2.14}$$

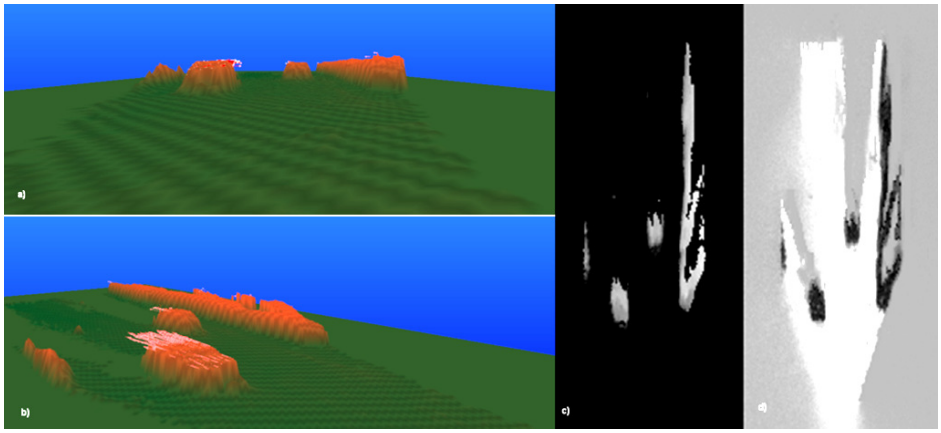


Fig. 1.2.6. Estimation results for the tracked scene: a) 3D rendering of the tracked elevation map, with speed vectors, perspective view; b) 3D rendering of the tracked elevation map, with speed vectors, lateral view; c) top view of the tracked map, with heights encoded as grayscale values; d) occupancy grid estimation from the tracked map, darker values encoding a higher occupancy probability.

1.2.5. Experimental results

Evaluation of Scene Reconstruction Accuracy

The most important goal of an elevation map algorithm is to accurately depict the scene being observed. For this reason, we have to compare the computed map with a ground truth map, a map generated by using a sensor that is both very precise and delivers a data density compared to that of stereo. Luckily, the Karlsruhe Institute of Technology compiled the KTTI Vision Benchmark Suite [27], a large set of data consisting of grayscale and color image pairs, Velodyne laser points, and GPS data, all synchronized and calibrated. For the evaluation process, the following steps were performed:

1. Stereo reconstruction, using our algorithms, on the rectified image pairs of the KITTI dataset.
2. Raw elevation map computation, using the reconstructed stereo points.
3. Elevation map tracking, using the raw elevation map as measurement.
4. Raw elevation map computation, using the Velodyne points. This map is regarded as ground truth.
5. Computation of the differences between the ground truth map and the raw stereo map, and of the differences between the ground truth map and the tracked map. The differences are computed only when both the ground truth and the evaluated map have valid heights.

The error analysis presented in this section was done on the sequence 2011_09_26_drive_0009, part of the raw data sequences available for download at [28]. Fig. 1.2.7 shows a comparison between the stereo reconstructed 3D points and the laser 3D points, for a frame in this sequence.

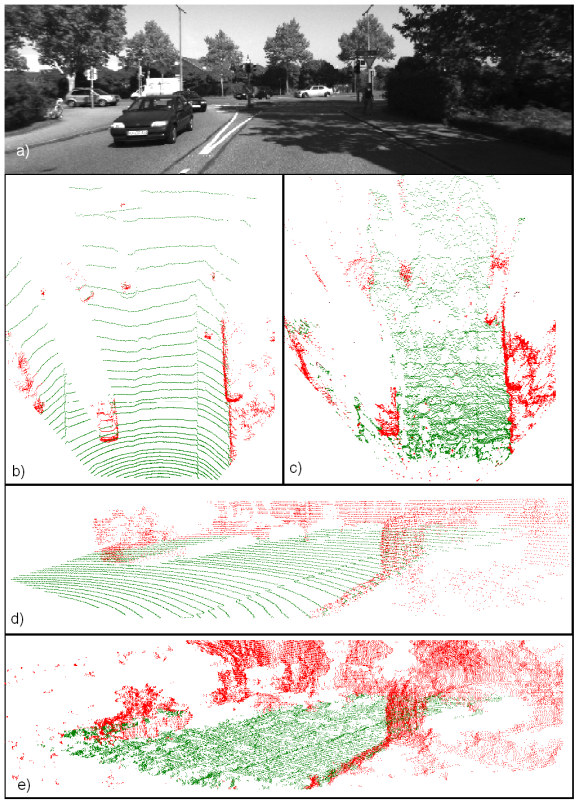


Fig. 1.2.7. A snapshot of the evaluation sequence. a) grayscale left image; b) laser 3D points, top view; c) stereo 3D points, top view; d) laser 3D points, lateral view; e) stereo 3D points, lateral view.

Based on the differences between the stereo-derived maps (raw and tracked) and the laser ground truth map, three types of errors are computed:

1. The *Badly Computed Heights* (BCH) percent, which is the ratio between the estimated heights that have the absolute difference from the ground truth higher than a threshold T (currently 0.15 m), and the total number of heights that can be compared to the ground truth.
2. The *Root Mean Square Error* (RMSE), an average error of height estimation compared to the ground truth.
3. The *Density* percent, the ratio between the number of estimated cell heights and the total number of cells that can have a height in the raw map (the theoretically observable cells).

The first two error indicators are inspired from [29], a paper which defines common accepted measures for evaluation of dense stereo algorithms, measures that we consider to be also suited for evaluation of dense elevation maps.

TABLE 1.2.1.
PERFORMANCE COMPARISON FOR HEIGHT ESTIMATION

Elevation Map	% Density	% BCH	RMSE (m)
Raw Map	41.12	27.99	0.19
Tracked Map	60.96	24.19	0.17

The results are presented in Table 1.2.1. What is clearly visible is that tracking the elevation map provides a reduction in BCH, and in RMSE, while achieving a nearly 50% increase in density. This means that the tracked map is a much more complete description of

the scene, increasing the number of cells that have a valid height, and this increase in data density comes with no precision cost, but with a slight precision gain.

The graphs in Fig. 1.2.8 show the BCH situation for different distances and heights. It is of no surprise that the errors increase with the distance, as this is an intrinsic property of stereovision. Also, the errors increase with the height being estimated, a phenomenon which we suggest is related in fact to distance uncertainties, a high structure being estimated at the wrong distance having a very large error with respect to the nearby ground.

The graphs in Fig. 1.2.9 show the RMSE situation for different distances and heights. As expected, the errors go up with the distance, and with the height.

From figures 1.2.8 and 1.2.9 it is apparent that the tracked elevation map improves the raw map results consistently, on almost all distances and heights.

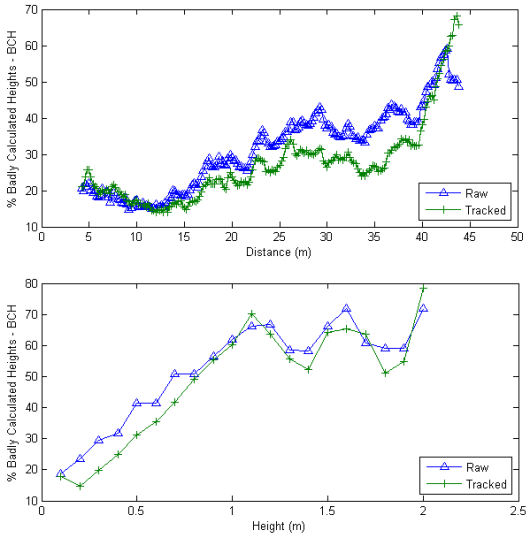


Fig. 1.2.8. Badly calculated heights (BCH) % comparison. Blue – raw map, green – tracked map. Top: BCH comparison per distance from the observing vehicle. Bottom: BCH comparison per height.

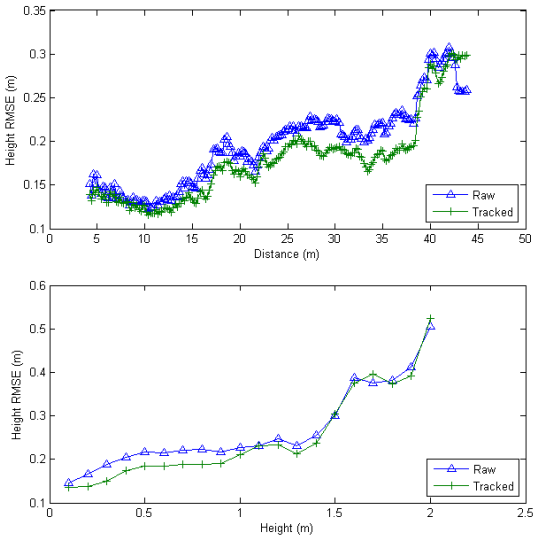


Fig. 1.2.9. Root Mean Square Error (RMSE) comparison. Blue – raw map, green – tracked map. Top: RMSE comparison per distance from the observing vehicle. Bottom: RMSE comparison per height.

Evaluation of Speed Estimation Accuracy

In addition to a denser and more accurate estimation of the heights of the perceived scene, the tracked elevation map adds dynamic information to this description. For each cell in the map the speed vector magnitude and orientation can be estimated using the speed vectors of its associated particles. In order to evaluate the quality of the speed estimation, sequences recorded in controlled environments, with known speed of the tracked obstacles, are used. In what follows, an analysis of eight sequences is presented: four with the obstacle coming from the front at 45 degrees orientation (Fig. 1.2.10.), and four with the vehicle coming from behind, at the same orientation (Fig. 1.2.11.). Each scenario was repeated with four different speeds, 30, 40, 50 and 60 km/h. The target vehicle's stable speed is reached outside of the observer's field of view.

Each test sequence is 2 to 5 seconds long (40 to 100 frames), and, depending on its speed, the vehicle is observed for 50 to 70% of the sequence's duration. As the scene consists only of the test vehicle and the road, only the speed of the particles that have a height higher than 50 cm is analyzed.

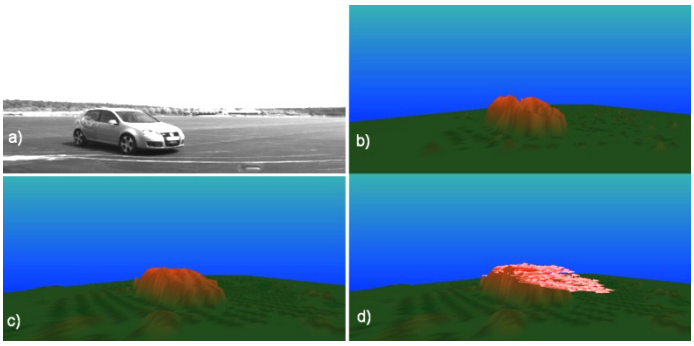


Fig. 1.2.10. Speed estimation test, with the target vehicle approaching at 45 degrees. a) left grayscale image; b) raw elevation map; c) tracked elevation map; d) tracked elevation map with speed vectors.

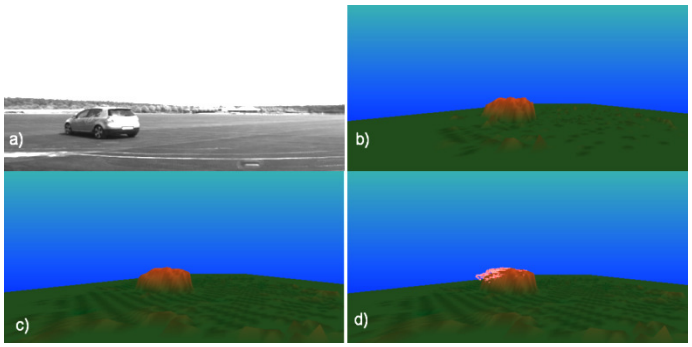


Fig. 1.2.11. Speed estimation test, with the target vehicle receding at 45 degrees. a) left grayscale image; b) raw elevation map; c) tracked elevation map; d) tracked elevation map with speed vectors.

First, a speed magnitude histogram is constructed for each frame, counting the number of particles for each speed value candidate, from 0 to 100 km/h. Figures 1.2.12 and 1.2.13 show time sequences of these histograms, plotted as 3D surfaces, for the ground truth speeds of 30 km/h and 60 km/h (ground truth shown on the plots as the narrow vertical spike), on the two vehicle orientation scenarios. The speed value clustering behavior can be observed from these plots: as the vehicle enters the scene, the histograms become narrower

around the ground truth value, and then they become diffuse again as the vehicle goes out of the field of view (but not completely out of the map).

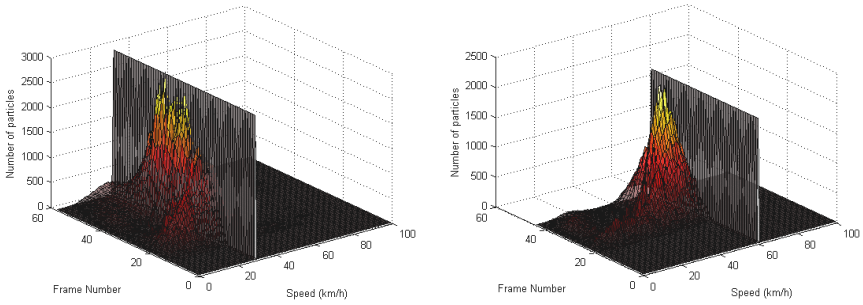


Fig. 1.2.12. Particle speed histograms, for the incoming vehicle scenarios. Top: time evolution of the histograms for the 30 km/h test speed; Bottom: time evolution for the 60 km/h test speed.

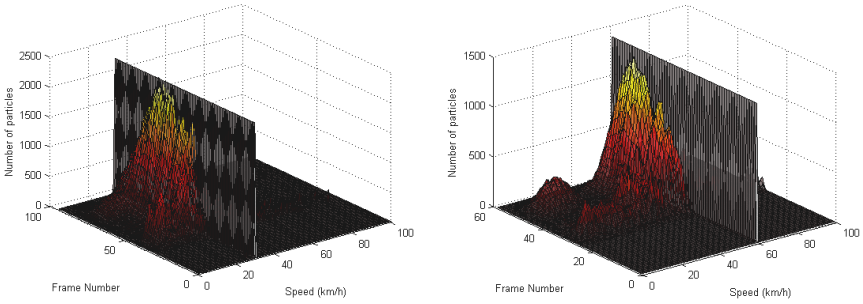


Fig. 1.2.13. Particle speed histograms, for the receding vehicle scenarios. Left: time evolution of the histograms for the 30 km/h test speed; Right: time evolution for the 60 km/h test speed.

From the speed histograms, an estimation of the perceived speed of the moving obstacle can be extracted. For the two orientation scenarios, the estimated speeds are plotted against the ground truth, as shown in Fig. 1.2.14 and Fig. 1.2.15.

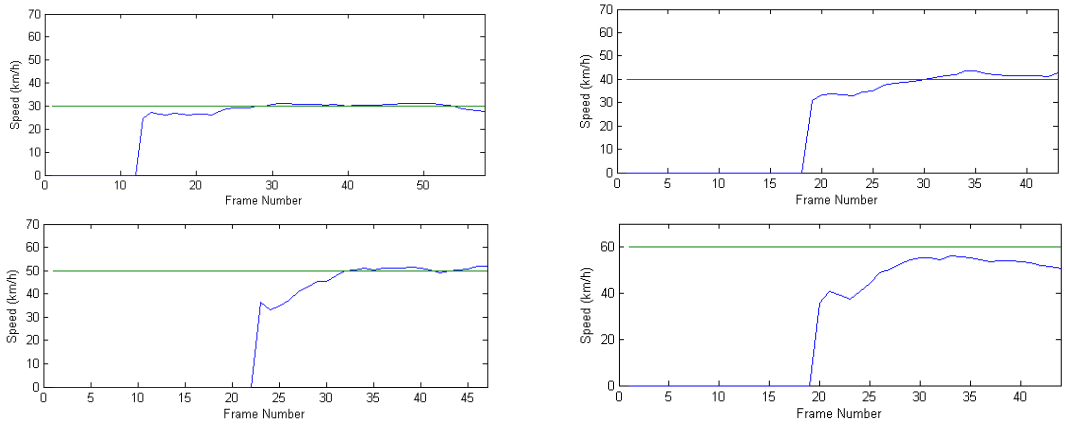


Fig. 1.2.14. Estimated object speed, for 30, 40, 50 and 60 km/h ground truth speeds, for the incoming vehicle scenario.

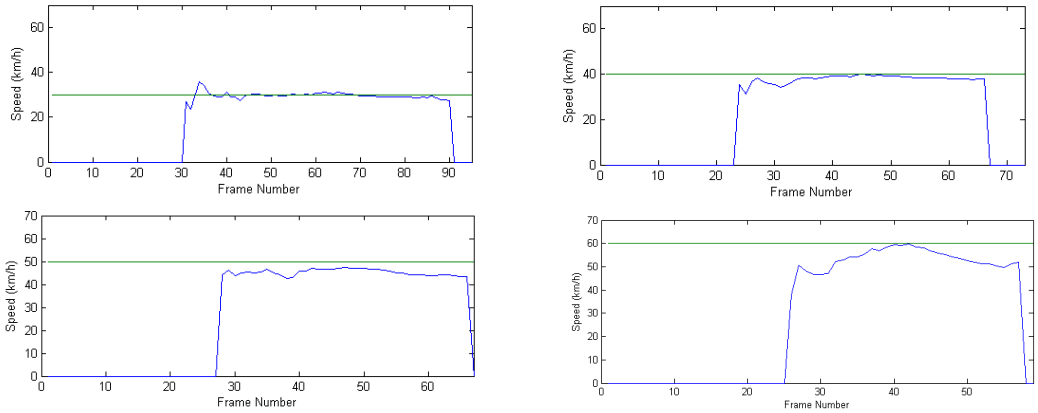


Fig. 1.2.15. Estimated object speed, for 30, 40, 50 and 60 km/h ground truth speeds, for the receding vehicle scenario.

An error analysis of the speed estimation was performed per sequence, taking into consideration only those frames where the object was actually visible. The estimated mean speeds, and the root mean square errors against the ground truth, are presented in tables 1.2.2 and 1.2.3.

TABLE 1.2.2
SPEED MEASUREMENT EVALUATION - INCOMING VEHICLE SCENARIO

Ground truth speed (km/h)	Mean estimated speed (km/h)	RMSE (km/h)
30	29.2650	1.9720
40	38.9354	3.9316
50	46.9964	6.5184
60	50.0729	11.7318

TABLE 1.2.3
SPEED MEASUREMENT EVALUATION – RECEDING VEHICLE SCENARIO

Ground truth speed (km/h)	Mean estimated speed (km/h)	RMSE (km/h)
30	29.6231	1.6149
40	37.8779	2.6842
50	45.2310	4.9515
60	53.1327	8.2875

The graphs in figures 1.2.14 and 1.2.15, and tables 1.2.2 and 1.2.3, show that the elevation map tracking algorithm can reliably estimate the speeds of moving objects. The error of speed estimation increases with the speed of the target object, a behavior which is to be expected, as a higher speed of the object means a higher deviation from the initial average zero speed of the cell (when a cell is first populated with particles, they receive random values from a probability distribution of zero mean). The process of drift and resampling will gradually remove the wrong speeds, and multiply the correct ones, but this is not an instantaneous process. Another reason for higher errors at higher speeds is that the faster moving object is observed for a smaller period of time, meaning that the speed stable time in the graph is lower with respect to the transitional times.

Qualitative Evaluation

The elevation map tracking system has been tested on multiple sequences of real traffic scenes, acquired in Cluj-Napoca, Romania. The resulted 3D models of the perceived environment were compared to the acquired images, as we looked for: likeness of the resulted model to the real scene, speed vectors orientation and rough magnitude, describing the general motion characteristics of the objects in the scene, density of estimation, obvious errors. The system proved to be able to handle the complex traffic scenes in a robust manner, increasing the quality of the height perception of the raw stereo-derived elevation map. In figures 1.2.16, 1.2.17 and 1.2.18 some of the observed scenes are illustrated.

A high resolution video file, showing the system's behavior for a two minutes driving sequence can be accessed at [30].

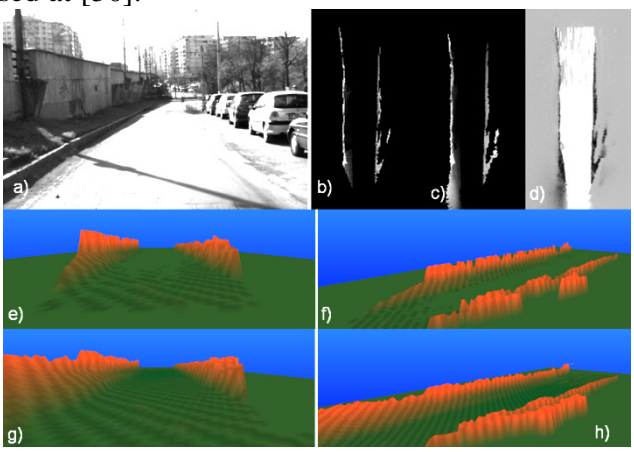


Fig. 1.2.16. Large, continuous static structures. a) left grayscale image; b) raw map, intensity encoding for height; c) tracked map, intensity encoded for height; d) occupancy grid estimation; e) raw map, 3D visualization, perspective view; f) raw map, 3D lateral view; g) tracked map, 3D perspective view; h) tracked map, 3D lateral view.

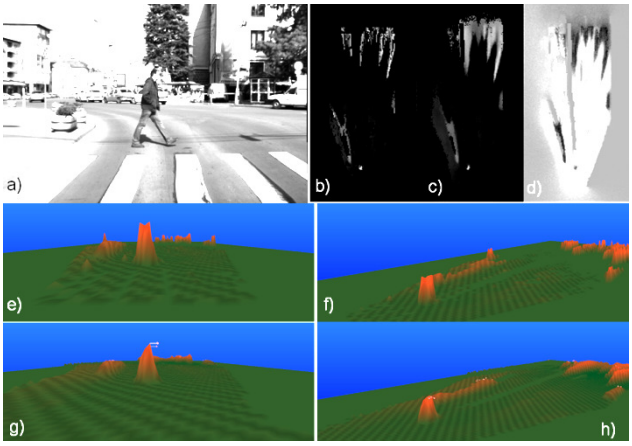


Fig. 1.2.17. Pedestrian tracking, with estimated speed vectors. a) left grayscale image; b) raw map, intensity encoding for height; c) tracked map, intensity encoded for height; d) occupancy grid estimation; e) raw map, 3D visualization, perspective view; f) raw map, 3D lateral view; g) tracked map, 3D perspective view; h) tracked map, 3D lateral view.

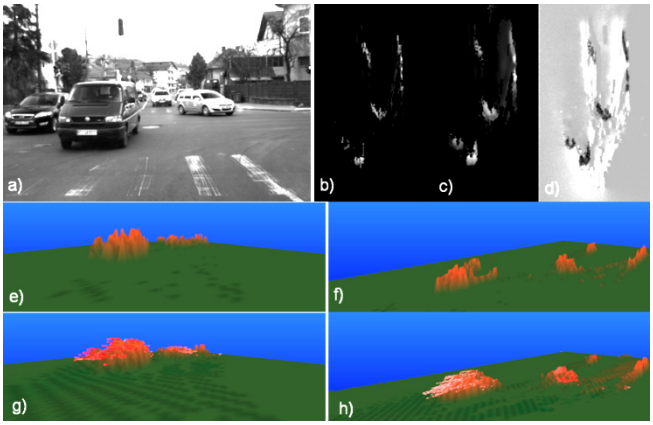


Fig. 1.2.18. Tracking vehicles at an intersection. a) left grayscale image; b) raw map, intensity encoding for height; c) tracked map, intensity encoded for height; d) occupancy grid estimation; e) raw map, 3D visualization, perspective view; f) raw map, 3D lateral view; g) tracked map, 3D perspective view; h) tracked map, 3D lateral view.

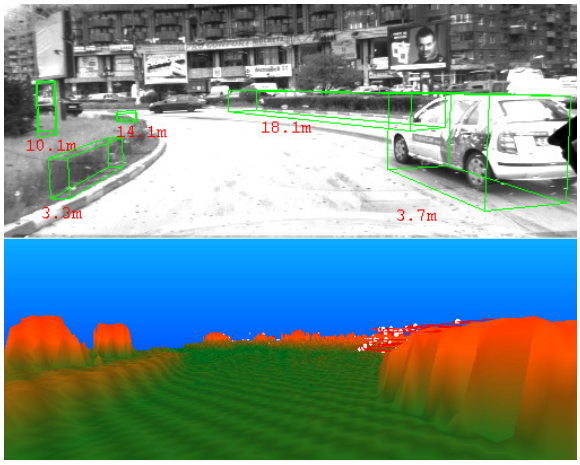


Fig. 1.2.19. Cuboid-based world perception (top) versus dynamic elevation map based perception at a roundabout (bottom).

Fig. 1.2.19 shows a comparison between the classical cuboid-based representation of the environment and the perception based on the proposed particle-based dynamic elevation map, in a situation extremely relevant for driving assistance – navigating a roundabout. One can see that the curved nature and the low height of the roundabout are not faithfully described by the oriented boxes, while the dynamic elevation map describes the environment geometry accurately.

1.2.6. Comparison of World Modeling Techniques

While the classic representation of the obstacles as 3D oriented objects remains the most popular choice for the environment perception systems, especially in the field of driving assistance, various alternatives have been proposed in the recent years, aiming to increase the flexibility of the representation of the environment and the level of perceived detail.

The proposed method is a completely dynamic probabilistic elevation map, having multi-modal probability density of the cell states. Table 1.2.4 presents a non-exhaustive comparison with existing environment description methods, in terms of *flexibility* (the capability of describing free-form obstacle areas as seen from the bird-eye view), *speed estimation capability*, *density* (in the bird-eye view space) and *height estimation capability*. The table shows that the dynamic elevation map combines the advantages of the elevation

map with those of the dynamic occupancy grid, being the most faithful reproduction of the real world.

TABLE 1.2.4
COMPARISON OF THE DYNAMIC ELEVATION MAP WITH OTHER ENVIRONMENT MODELING METHODS

Method	Flexible shape description	Speed estimation	Density	Height estimation
Oriented boxes [31]	No	Yes	Low	Yes
Elevation map [18][19]	Yes	No	High	Yes
Dynamic occupancy grid[17][32]	Yes	Yes	High	No
6D vision [33]	Yes	Yes	Low	Yes
Dynamic stixel [5]	Yes	Yes	Low	Yes
Dynamic elevation map	**Yes**	**Yes**	**High**	**Yes**

1.2.7. Conclusion

This section describes an elegant environment modeling and tracking technique, the particle-based dynamic elevation map. The building block of the proposed model is the dynamic particle, having position, speed, and height, which can populate the grid cells, can migrate between cells, and can be created, multiplied, and destroyed based on the measurement data. Using the particle set, an elegant, intuitive and easily adaptable system was designed to solve the non-parametric PDF Bayesian tracking problem for a dynamic digital elevation map.

The proposed method unifies the dynamic cell-based world tracking specific to occupancy grids, with the power of 3D representation of elevation maps. While other contributions described in the literature combine these two types of representation, the system presented in this paper is able to track dynamic elevation maps directly, in a uniform manner, without discriminating between empty cell, occupied cell, static cell or dynamic cell. The dynamic elevation map is a general dynamic 3D world representation, which can easily be transformed into less general ones, such as the classic elevation map, or the dynamic occupancy grid.

The particle-based dynamic elevation map and the dynamic environment tracking algorithm based on this model form a generic intermediate level perception system, capable of enhancing the performance of the dense stereovision through improving its accuracy and density, and providing speed information for the elements of the scene. A better generic 3D description of the 3D environment can be applied to better extract the road surface for the purpose of lane detection, to better identify the static and the dynamic obstacles for detection, tracking and classification, or to provide a detailed 3D map of the environment.

1.3. The gray level enhanced dynamic elevation map

1.3.1. Introduction

Based on the previous results for a generic 3D dynamic world modeling and tracking, described in sections 1.1. and 1.2, an even more detailed description of the world is proposed, a description that includes, besides the 3D information obtained from stereovision, the image information as well. This new model is a particle-based, gray level enhanced dynamic elevation map. This experimental method for world modeling and tracking is another step towards the greater goal of having a generic world model and perception technique, able to rely on as much sensorial information as possible to infer a highly accurate depiction of the real dynamic 3D scene.

In the following sub-sections, only the relevant changes from the technique described in 1.2 are presented, along with the experimental results.

1.3.2. The graylevel dynamic elevation map world model

Assuming the coordinate system with the origin on the ground in front of the host vehicle, the *X* axis pointing forward, *Y* axis pointing to the right, and *Z* axis pointing up, the horizontal plane *XOY* is divided into cells of 20 cm x 20 cm, each cell *i* being identified by a row coordinate r_i and a column coordinate c_i. Each cell has an associated height value h_i, and a gray level value g_i. Also, since the 3D scene we want to model is dynamic, each cell has an assigned speed vector, which we'll confine to the horizontal plane, and therefore it will have two components, for the *X* and *Y* axes, or, in terms of rows and columns, $v_{r,i}(r_i,c_i)$ and $v_{c,i}(r_i,c_i)$ – the row speed and the column speed for each cell *i* in the map (Fig. 1.3.1).

Each cell *i* in the dynamic gray level elevation map can be described by four values, h_i, g_i, $v_{r,i}$ and $v_{c,i}$. Due to the fact that perfect sensing of a real traffic scene is impossible, one cannot have exact knowledge of these values. These limitations lead to uncertainties, which have to be included in the world model as probability densities.

A cell *i* in the dynamic elevation map is associated to a four-dimensional random variable $\mathbf{X}_i = (h_i, g_i, v_{r,i}, v_{c,i})^T$. The objective of the tracking algorithm is to compute the probability density of $\mathbf{X}_i$, for each cell *i* in the map, based on a sequence of measurements $\mathbf{Z}(0)\ldots\mathbf{Z}(t)$, *t* being the current observation time.

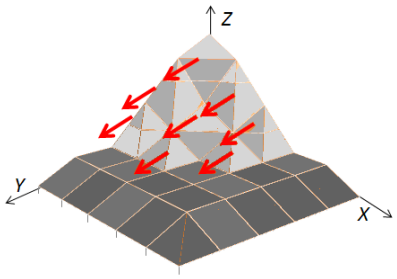


Fig. 1.3.1. The dynamic, gray level elevation map.

The dynamic elevation map will be described, at time *t*, by a set of particles $S(t)=\{\mathbf{x}_k(t) \mid \mathbf{x}_k(t)=(c_k(t),\ r_k(t),\ h_k(t),\ g_k(t),\ v_{c,k}(t),\ v_{r,k}(t))^T,\ k=1\ldots N_S(t)\}$, each particle *k* being located in the 2D grid, in the cell having the row r_k, and the column c_k.

1.3.3. The measurement data

The dense stereovision data is transformed into a raw elevation map. A detailed description of the process can be found in [10]. The raw map is described by three 2D arrays:

- Measured height of each cell, denoted by $z_i(r_i,c_i)$;
- Measured gray level of each cell, denoted by $b_i(r_i,c_i)$. The gray level of a map cell is found by transforming the row, column and height coordinates of the raw height map into *XYZ* coordinates, and projecting them into the left image plane.
- Data availability for each cell, denoted by $d_i(r_i,c_i)$. $d_i(r_i,c_i)=1$ means height for this cell is available, and $d_i(r_i,c_i)=0$ means that no measurement data is available for cell *i*.

A cell may have no valid measured height due to the fact that the stereovision engine may not be able to provide a 3D value for all areas in the image, or due to occlusions and field of view limitations. The tracking algorithm is made aware of this situation by $d_i(r_i,c_i)$. If the cell has no measured height, we'll assume that it has no measured gray level, either.

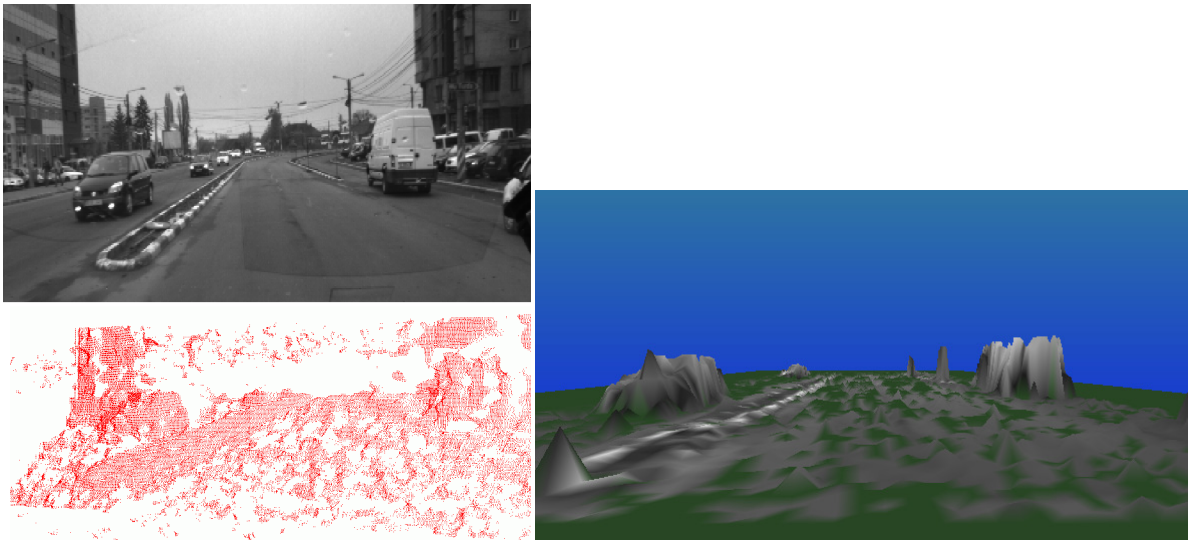


Fig. 1.3.2. Original grayscale left image (top, left), stereo-reconstructed 3D points (bottom, left), and the raw elevation map enhanced with gray level information (right).

1.3.4. Weighting the particles

A method for transforming the raw elevation information into particle weights, taking into account the stereo uncertainties along all three coordinate axes, is described in section 1.2. The same reasoning can be applied for the measured gray values $b(r, c)$, to create a weight histogram G_i, for each candidate gray level g, using equation (1.3.1):

$$G_i(g) = \sum_{\tau=r_i-2\sigma_r}^{r_i+2\sigma_r} \sum_{\kappa=c_i-2\sigma_c}^{c_i+2\sigma_c} d(\tau,\kappa)\Gamma_i(\tau-r_i,\kappa-c_i)\delta(b(\tau,\kappa)-g) \tag{1.3.1}$$

In order to account for the uncertainty in gray level estimation, the gray level histogram G_i will be convolved with a Gaussian kernel $K_{G,i}$, of standard deviation $\sigma_G(i)$, accounting for the uncertainty of gray level estimation (this includes uneven lighting, reflections, etc), obtaining the weight LUT for particle's gray levels, $W_{G,i}$.

$$W_{G,i} = K_{G,i} * G_i \tag{1.3.2}$$

Using the weight LUT's for height, $W_{H,i}$ and for gray level, $W_{G,i}$, the weights of the particles can be easily computed. A particle k, in a cell i, will get the following weight:

$$w_k = W_{H,i}(h_k)W_{G,i}(g_k) \tag{1.3.3}$$

After weighting, the resampling process follows the same steps that are described in section 1.2. After resampling, the speed, height and gray level are estimated for each cell.

1.3.5. Experimental results

In order to assess the world modeling and tracking capabilities of the proposed solution, multiple sequences acquired in the urban traffic of Cluj-Napoca, Romania, were used. In the absence of a definite ground truth, a subjective evaluation of how well the

tracking system was able to provide a virtual 3D representation of the perceived environment is made, in comparison with the raw map used as measurement. Some results are shown in Fig. 1.3.3. The tracking system was able to considerably improve the density of the raw map, to filter out the reconstruction errors, especially in the case of the road surface and to correctly identify the moving elements in the scene and their direction of motion.

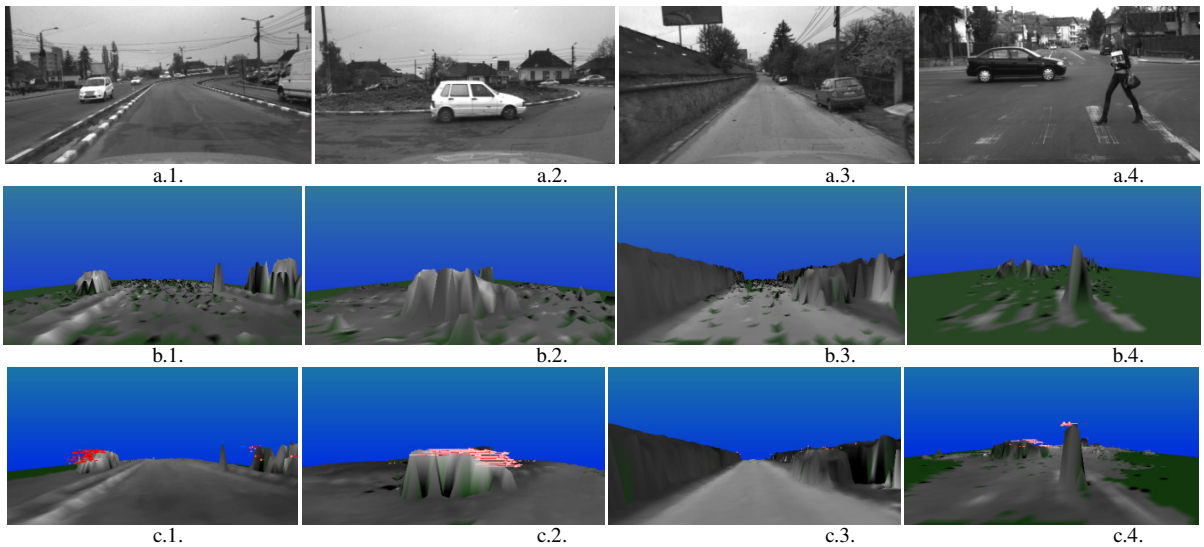


Fig. 1.3.3. Qualitative results in real urban traffic scenarios: a.1.-a.4, original grayscale image; b.1.-b.4, raw elevation map with grayscale information; c.1.-c.4, tracked dynamic elevation map.

1.3.6. Conclusion

This section describes a flexible, experimental method for modeling and tracking complex 3D environments - the particle-based dynamic elevation map enhanced with gray level information. The central element of the solution is the dynamic particle, having position, height, speed and gray value, which can migrate from one map cell to another. A particle population in a cell is a multi-modal probability density approximation for a 4-dimensional state space, an approximation that is updated by controlling the particle population using the cues obtained from measurement. The preliminary results show a system capable of reliably estimating the 3D static and dynamic characteristics of the observed traffic scenes, able to significantly increase the density and accuracy of the raw, stereovision-extracted elevation map, and to add dynamic information for each map cell.

2. Large baseline stereovision for space surveillance

2.1. Large baseline stereovision system for surveillance of the MEO orbits and beyond

2.1.1. Introduction – related work

This chapter presents methods for detecting and ranging Earth orbiting objects based on large baseline stereovision. While optical solutions have long been around for observing space and surveying the Earth's orbit, stereovision has not been, so far, exploited for this task.

The space around the Earth is filled with man-made objects, which orbit the planet at altitudes ranging from hundreds of kilometers to tens of thousands kilometers. Many of these objects are useful: communication, navigation, military, or scientific satellites. Other objects are not so useful, and their increased number becomes a threat to the safety of space operations, space travel, or even for the planet's surface, as many of these objects will, one day, come back to Earth. These objects are also known as Space Debris (SD). Keeping an eye on all objects in Earth's orbit, useful and not useful, operational or not, is known as Space Surveillance, an activity that includes detection, tracking and propagation of orbital parameters of the space objects, followed by cataloguing and analysis. Detailed surveys on the problem of space debris and space surveillance are found in [35] and [36].

According to Inter-Agency Space Debris Coordination Committee (IADC) [37], the space objects are categorized according to the altitude of their orbits in the following three categories: Low Earth Orbit (LEO) for an altitude below 1 500 km, Medium Earth Orbit (MEO) at an altitude of around 20 000 km, and Geostationary Earth Orbit (GEO), also sometimes called the Clarke orbit, at an altitude of 36 000 km. The Geostationary Earth Orbit and the orbits of higher altitude, such as those of the Molniya and Tundra satellites, are also known as High Earth Orbits (HEO). Another way of classifying orbits is this: Low Earth orbit starts just above the top of the atmosphere, while high Earth orbit begins about one tenth of the way to the moon [38]. In [39] the satellites are classified as LEO (altitude less than 2000 km), MEO (altitude between 2000 km and 34000 km), GEO (altitude between 34000 km and 38000 km), and the Remaining Earth Orbit (REO), having the altitude higher than 38000 km.

The MEO orbit is used mainly by the navigation satellites (GPS, Glonass, Galileo), and by the communication satellites covering the North and South poles. There is no consensus about a MEO protected region [40]. The GEO protected region, defined by the IADC Coordinating Committee, includes the spherical shell centered on geostationary altitude with extent 200 km above and below this altitude and with inclination limits of +15 to – 15 degrees. While operations are usually conducted within about 75 km of geostationary altitude, the GEO protected region is extended in altitude to create a maneuver corridor for relocating spacecraft. Around 390 operational satellites orbit this protected zone, most of them providing telecommunication, broadcasting and meteorological services [41].

Several catalogs were created and continuously updated through time in order to help and improve the monitoring process of the space objects. The entity which maintains the most comprehensive catalog of satellite orbital data is the U.S. Space Surveillance Network (SSN) [42]. Other significant organizations that catalog debris and satellites are NASA, the Russian Space Surveillance System (RSSS) and the Air Forces from few countries (notably

France, Germany, UK and Canada). A future significant contribution is expected in this sense from the ESA surveillance of space system which is actually in an early development phase [9].

The systems that are used nowadays for sky surveillance involve varied systems such as radars, optical telescopes and laser ranging systems [35]. Radar systems are successfully used for Low Earth Orbiting objects (LEO) providing range measurements with high accuracy and, unlike optical systems, they are not sensitive to the meteorological conditions. Still, a great limitation is that they are less accurate for higher orbits. Radar suffers from $1/R^4$ signal power losses between signal transmission, reflection and reception, making detection of distant deep space spacecraft difficult. Another problem is that a radar dish able to detect such long range objects has a very small field of view ($<0.20^o$) and the probability to detect an unknown fast moving object is insignificant.

Optical telescope systems have been employed successfully for many years to detect deep-space satellites, as they detect reflected sunlight, and therefore no active illumination is necessary for the telescope to see its target. Adding to their high performance, the optical systems are cheap and easy to set up. Moreover, the acquired data can be extensively analyzed with image processing and pattern recognition algorithms, in order to detect and track the objects of interest in real time. The main limitation is that optical systems can only operate at night, and need clear skies to acquire accurate observations. Also, the closer to Earth the orbits are (the case of LEO and MEO objects), the more difficult they are to detect, since they appear as fast moving objects and their background changes considerably between frames. The higher orbits (GEO, Molniya) have a significantly lower speed, being visible for a longer period of time in the same restricted area of the sky.

The most known ground-based optical Space Surveillance systems are the following:

Ground Based-Electro-Optical Deep Space Surveillance - GEODSS [44] (for GEO and HEO objects) is a system of telescopes that is able to track objects as small as a basketball, orbiting the Earth at distances beyond the geosynchronous orbit. This system produces 70% of all geosynchronous tracks and 50% of all deep space tracks, tracking over 200 objects not tracked by any other sensor.

Space Surveillance Telescope [45], [46] (GEO, HEO, other deep space objects) is a ground-based system using the latest in optical technology to increase the Space Situation Awareness (SSA) capability. The SST program is a DARPA technology demonstration based on a 3.5 meter f/1.0 telescope, which was started in 2002, and successfully demonstrated in 2012.

The ESA Space Debris Telescope [47], designed mostly for GEO objects, operates a Ritchey-Chrétien telescope of 1 m aperture and 0.7 degrees field of view, located in Tenerife, to detect and follow objects of at least 15 cm in diameter at GEO altitudes (assuming an object albedo of 0.1). The ESA Space Debris Telescope (ESA SDT) covers a sector of 120 degrees of the GEO ring. Further plans by ESA include a space-based orbital debris surveillance system, able to detect much smaller objects, whose specifications are presented in [48].

ROSACE [49] is a GEO and near GEO object tracking system based on a Newton type telescope of 50 cm aperture, able to observe objects up to 19 units of visual magnitude.

The Passive Imaging Metric Sensor (PIMS) Telescopes [50] is an optical system for the surveillance of the GEO orbits and the deep space region, operated by the United Kingdom Ministry of Defense. PIMS telescopes are located in UK, Gibraltar, and Cyprus. The three sensors cover 165 degrees of the GEO ring. The PIMS system can detect GEO objects down to 1 m diameter, with position accuracy better than 10 μrad.

The Zimmerwald Telescope [50] is a system for tracking objects in the GEO ring, operated by the Astronomical University of Berne (AIUB), based on a Cassegrain telescope

with an aperture of 1 m. The sensor can cover a sector of 100 degrees of the GEO ring. Images are taken with a CCD chip of 2048x2048 pixels, with sensitivity up to 20 mag. The Zimmerwald telescope was used as a test site for validating procedures and processing algorithms of the ESA Space Debris Telescope.

While most existing systems are focused on increasing the sensitivity by employing powerful telescopes, with narrow field of views, Space Insight's Starbrook system [51] covers a very large field of view (100 x 60 degrees) by robotic scanning, in the pursuit of MEO objects. This system employs a fast optical system with only 10 cm aperture and a 10 megapixel CCD, and relies on automatic processing for identifying potential targets. Starbrook is actually hosted at a UK facility in Cyprus and services several UK governmental programs.

The large sky area that needs to be covered in the process of Space Surveillance and the large number of resident space objects that need to be detected, tracked and identified requires automatic processing of the acquired images. A comparative study of four different methods for detection of GEO objects from CCD images is presented by Yanagisawa et al. in [52].

The first method is a PC based stacking method that needs around forty CCD images to detect an object. Sub-images from consecutive frames that contain the presumed space object in exactly the same sub-image coordinates are used to create a median image. Doing so, the stars from the background are eliminated in the median image, due to the object velocity, while the space object will be emphasized. This method was able to detect several new asteroids, proving thus its effectiveness. Still, the main weakness of this method is the computational time, since it is required to process a high number of frames, and for each presumed object, there are several likely paths that must be taken into account and checked. This method is effective for known (catalogued) objects, when object's path can be predicted.

The second method is also based on stacking, sped up by using an FPGA implementation and by reducing the image complexity through a binarization pre-processing step. The computational time is therefore reduced to about one thousandth of the time required by the PC based stacking method.

The third method is a line identifying technique, which identifies the moving objects under the assumption that the pixels from consecutive frames, belonging to the same object, should fit a straight line. This method also requires multiple CCD frames for object detection.

The multi-pass multi-period (MPMP) is the fourth method, which uses the average instead of median, reducing thus the analysis time. The authors conclude that each method has its strength and weaknesses, and propose a hybrid approach for an efficient surveillance network, relying on FPGA stacking or MPMP for detection of low speed objects, on line identification for detecting fast moving objects, and on PC stacking for following objects of known trajectory, where the search area is very small.

Levesque et al. present in [53], [54] an approach for satellite detection which relies on the property that a moving satellite will be recorded as a linear streak in a long exposure image. The method assumes that the orbital parameters of the satellite are approximately known. A matched filter is used for streak detection, requiring a detailed image analysis and several image preprocessing steps, including the correction of sensor artifacts such as dead pixels or image noise and the removal of the star bleeding or other signal degradation. Then, all the non-streak objects in the image (e.g., stars) are removed based on their morphological characteristics, thus reducing the number of possible candidates. The satellite streak detection is finally performed based on matched filters. Since the orientation and an estimative position of the satellite are known, the filter will emphasize the streak intensity and establish the real position of the satellite.

An improvement of this method is further proposed by Lévesque and Lelièvre in [55]. This new approach consists in extracting every possible object candidate and then eliminating the false positives through a set of rules based on object features. The rules were made based on a quantification of the object features such as the signal-to-noise ratio and the moments of inertia. Simulated images, generated with controlled parameters were used to establish the best thresholds for satellite streaks detection. A highly precise detection was also confirmed on real images, even in the case of very faint satellite streaks.

Stoeveken and Schildknecht discuss in [56] several methods of space debris identification in CCD images, also taking into consideration the differences between observation strategies (star tracking, target tracking or fixed orientation). The authors highlight the importance of star removal in the image based on coordinates extracted from a star catalogue, the possibility of using the median image for background subtraction, the identification of the relevant object based on specific properties (e.g., linear streak), the need for reduction of false positives caused by cosmic rays or sensor pixel errors, and the speedup in processing possible by predicting the trajectories of the objects of interest in consecutive frames.

After the orbiting object is detected in the acquired images, its orbital parameters can be computed. The classical approach for orbit computation is the method of Gauss, which requires three observations of the same object, at different moments in time. Milani et al. propose in [57] a new algorithm for orbit determination, based on the first integrals of the Kepler problem, which requires only two detections at different passes of the target object. The authors emphasize the need for accurate correlation between different observations of the same object, and propose sophisticated solutions for solving this problem in the case of high observation datasets.

2.1.2. Overview of the contributions in stereovision-based space surveillance

This work presents original, stereovision based solutions for space surveillance, able to detect and range Earth orbiting objects in the LEO, MEO, GEO and HEO regions. From two known locations, set up 37 km apart, synchronized images are acquired using optics adapted to the orbit type (wide angle lenses for LEO objects, and 750 mm focal length telescopes for MEO and beyond). The intrinsic camera parameters are calibrated offline, and so are the translation vectors, which are computed from highly accurate GPS coordinates of the observation locations. The rotation matrices of the two cameras are calibrated for each frame, due to the fact that the Earth's rotation forces us to use an equatorial mount tracker that keeps the stars position fixed in the image. The orbiting objects are detected in two ways – either by their characteristic motion which causes a signature streak, if the object is not too far away, or by the parallax effect which causes their position to differ in the stereo image pair. Stereo correspondence is aided by the epipolar geometry, and is followed by triangulation for 3D position reconstruction.

2.1.3. The sensorial systems

For LEO orbit surveillance, the large baseline stereovision system is composed of two identical observation stations, each station having the following components:

- "Wide-angle" objective with a focal distance of 20mm, sigma EX 20mm F1.8 DG RF aspherical type [58].
- DSLR Camera, type Canon EOS 50D, with a CMOS APS-C chip (22.3mm x 14.9mm, 4572 x 3168 pixels, in color) [59];

- Equatorial tracking mount, type Celestron CG5 [60].
- Laptop computer equipped with a custom USB to TTL interface for camera triggering. The triggering is done using the cable remote interface of the camera.
- GPS Time Receiver for time synchronization and measurement of the observer's location.

For the MEO orbits and beyond, the wide angle lens is replaced by a C6-NGT Telescope (Newtonian Reflector, D=150 mm, F=750 mm) [60]. The DSLR camera is connected to the telescope, which becomes the camera's objective.

Fig. 2.1.1. Detail of the individual sensor, LEO observation mode. The equatorial mount supports the telescope, the camera and the lens. For MEO and beyond, the camera is attached to the telescope.

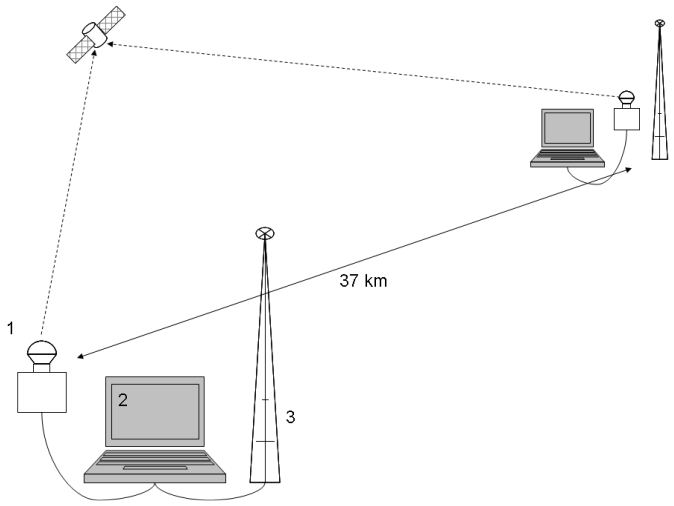


Fig. 2.1.2. Architecture of the stereoscope. The computer 2 controls through the custom USB to TTL interface the triggering sequence for the camera 1. GPS Time Receiver 3 delivered time signals for synchronization.

2.1.4. System synchronization

Stereovision requires synchronized image acquisition from the two cameras. In a classical stereo system the synchronization is achieved by triggering the cameras by a common signal. However, this solution cannot be applied when the cameras are 37 kilometers apart. Thus, two independent units that can generate triggering signals at the same time are needed, without a physical connection between them. Each unit has a schedule list of triggering times, and if the units have a common schedule, the triggering signals will be simultaneous. Unfortunately, working with a schedule means assuming that the two

synchronization units have a precise internal time, which is an assumption that for a common laptop PC is not true. Even if the internal clocks are precisely synchronized, they will quickly fall out of sync.

In the absence of a precise time base, some external signal to keep the two units in sync is required. This signal is the GPS radio signal, which is received simultaneously, once a second, all over the world. Each unit receives this signal with the help of a GPS receiver, connected to the PC. The following algorithm shows the operation of a unit, based on GPS reception.

```
While (NotEmpty ScheduleList)
TriggerTime = ScheduleList.CurrentTime()
GPSTime = GPSReceiver.GetTimeNonBlocking()
If (TriggerTime-GPSTime < 2 s)
        While (GPSTime < TriggerTime)
                GPSTime=GPSReceiver.GetTimeBlocking()
        End While
        Camera.Trigger()
ScheduleList.NextTime()
End If
End While
```

The unit reads the current triggering time from the *ScheduleList*. The current global time is polled through the method *GPSReceiver*.GetTimeNonBlocking(), which will deliver the most recent timestamp of a GPS reception, without blocking the execution of the program. If the global time is close to the triggering time, the program goes in the blocking mode. The GPS receiver is queried by *GPSReceiver*.GetTimeBlocking(), a function that will exit only when a fresh signal is received. In this way, there will be a minimum delay between the reception of the signal and the triggering of the camera, if the received time is the one matching the schedule list.

2.1.5. The coordinate system and the camera parameters

In order to represent the position of the detected objects, a coordinate system whose origin is located in the centre of the Earth, and its coordinate axes are fixed in relation to our planet was chosen. Such a coordinate system is ECEF (Earth Centered, Earth Fixed), whose *OZ* axis is the Earth's axis of rotation, pointing to the North Pole, the *OX* axis joins the centre of the Earth with the point on the surface that has zero latitude and longitude, and *OY* is perpendicular to the *XOZ* plane. The equatorial plane is identical to plane *XOY*, and the zero meridian plane is *XOZ*.

In stereovision terms, the ECEF coordinate system will be referred to as the World Coordinate System, W_C. The two cameras that will observe the sky will have their own coordinate systems, the left camera system centered in C_L and having the axes $X_LY_LZ_L$, and the right camera system centered in C_R, and having the axes $X_RY_RZ_R$. The left image plane will be parallel to the plane $X_LC_LY_L$, and the right image plane will be parallel to $X_RC_RY_R$.

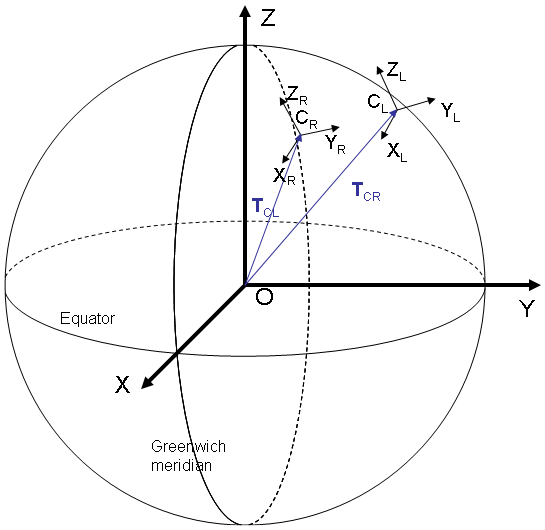


Fig. 2.1.3. The world and camera coordinate systems.

The stereovision process relies on accurate triangulation of corresponding features from the left and right camera images. For this reason, the correspondence between the pixel position and the 3D world must be accurately described by the cameras' parameters. There are two types of parameters, intrinsic and extrinsic.

The extrinsic parameters describe the relation between the coordinate systems of the cameras and the world coordinate system. For each camera, there is a *translation vector*, which describes the position of the camera's optical center in the world coordinate system. These vectors will be denoted by $\mathbf{T_{CL}}$ and $\mathbf{T_{CR}}$. Conversely, one can express the position of the world coordinate system's origin in the cameras' coordinate systems. These translation vectors are denoted $\mathbf{T_{WL}}$ for the left camera, and $\mathbf{T_{WR}}$ for the right camera.

The orientations of the camera coordinate systems with respect to the world coordinate system are described by the *rotation matrices* of the two cameras, $\mathbf{R_{CL}}$ and $\mathbf{R_{CR}}$.

Using the translation vectors and the rotation matrices, one can perform coordinate transformations between the world and camera coordinate systems, for any 3D point.

The specific characteristics of a camera plus lens optical system are described by the intrinsic camera parameters (in what follows, the word "camera" will denote the whole camera-lens assembly). For each camera, there are the following intrinsic parameters:

- *Focal length*, measured in pixels, which is the distance from the optical centre to the image plane. Each camera has its own focal length: f_L for the left camera, and f_R for the right camera.
- *Position of the principal point*, measured in pixels, represents the intersection of the optical axis of the camera and the image plane. For the left camera, the principal point is $C'_L = (x_{CL}, y_{CL})$, and for the right camera $C'_R = (x_{CR}, y_{CR})$.
- The *distortion coefficients* are also included in the intrinsic parameter set. The image distortions are caused by the difference between the ideal pinhole camera model and the real camera and lens assembly. These distortions can be usually modeled by a radial component and a tangential component.

2.1.6. Calibration of the intrinsic camera parameters

For the LEO observation setup, the camera lens is a common wide field of view lens, suitable for calibration with the well known toolboxes. The intrinsic calibration is performed using the Bouguet method, available through the Caltech Camera Calibration Toolbox [14].

The intrinsic calibration process provides the position of the principal point, the focal length, and the distortion coefficients for each camera. For a wide FOV lens, the distortion coefficients are significant, and cannot be neglected. Therefore, a routine for compensating this distortion is executed for each captured frame.

For the MEO, GEO and HEO observation setup, the lens is replaced by a narrow field of view telescope. Due to the fact that the telescope's field of view is extremely narrow compared to regular photographic lenses (less than 2 degrees), the radial distortions are expected to be negligible. Another assumption is that the position of the principal point is not crucial in calibration, as any deviation from the true position of this point will be equivalent to a rotation of the scene with respect to the camera [28], and therefore can be compensated by the extrinsic calibration of the rotation matrix. The validity of both assumptions is proven by experimental results.

Thus, the only intrinsic parameters that must be calibrated are the focal distances: f_L for the left camera, and f_R for the right camera.

Given a pair of stars i and j, of known celestial coordinates and known position in the image plane, the focal distance of the camera, expressed in pixels, is:

$$f(i,j) = \frac{l(i,j)}{\tan\left[\theta(i,j)\right]} \cong \frac{l(i,j)}{\theta(i,j)} \tag{2.1.1}$$

The term $l(i, j)$ denotes the distance in pixels between the image position of the two stars, (x_i, y_i) and (x_j, y_j):

$$l(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.1.2}$$

The term $\theta(i,j)$, expressed in radians, is the angular distance between the two stars. As the stars are catalogued based on their equatorial coordinates Right Ascension (RA) and declination (DEC), which are angular coordinates in a spherical coordinate system centered on Earth, the angular distance between stars i and j can be computed in spherical triangle by the cosine formula [28]:

$$\theta(i,j) = \cos^{-1}\left(\sin(DEC_i)\sin(DEC_j) + \cos(DEC_i)\cos(DEC_j)\cos(RA_i - RA_j)\right) \tag{2.1.3}$$

The equatorial coordinates of the stars selected for calibration of the two optical systems, and an example of their corresponding position in the left and right image, are shown in Table 2.1.1. The position of these stars in the right image of the stereo pair is shown in figure 2.1.4. The stars used for calibration are identified manually from the image, and the equatorial coordinates of these stars are extracted from a star catalog [64]. While this process may be time consuming, it must be done only once for each instrument, as the intrinsic parameters do not change during the system operation.

Table 2.1.1. Reference stars used for intrinsic calibration of the cameras.

Equatorial coordinates of the reference stars (epoch 2012.809)							Left (Feleacu) image position		Right (Marisel) image position	
Star Id.	Right Ascension, RA			Declination (Dec)						
Nr.	Hours	Minutes	Seconds	Degrees	Minutes	Seconds	x	y	x	y
1	1	7	59.992	30	27	15.523	897.5	637.7	871.8	643.3
2	1	9	18.150	30	23	40.995	1291.5	569.9	1270.6	611.6
3	1	9	34.288	30	32	23.706	1364.1	775.1	1324.3	822.8
4	1	8	43.397	30	44	32.087	1099.1	1046.7	1035.4	1069.2
5	1	7	24.985	30	15	35.513	732.1	360.9	732.4	352.5
6	1	10	0.519	30	8	29.159	1517.9	226.3	1527.5	289.7
7	1	11	27.523	30	35	21.669	1926.1	867.9	1876.3	966.5
8	1	7	11.636	30	41	28.684	644.1	959.1	590.2	940.3
9	1	6	4.526	30	18	59.142	326.7	426.1	322.4	380.1
10	1	11	17.184	30	4	40.327	1905.9	154.5	1921.0	253.4
11	1	11	35.434	30	52	12.624	1947.9	1260.5	1862.2	1359.6
12	1	6	18.646	31	1	21.167	364.9	1410.7	270.4	1365.2
13	1	8	35.059	30	28	23.732	1071.7	670.7	1042.5	692.1
14	1	9	12.212	30	26	55.228	1258.9	643.7	1231.3	682.3
15	1	10	19.527	30	41	4.653	1581.3	986.1	1521.8	1052.6
16	1	8	24.909	30	42	15.088	1008.9	990.3	950.7	1004.7
17	1	7	37.711	30	8	36.944	801.7	201.7	816.2	200.0
18	1	10	14.566	30	7	9.281	1589.5	198.5	1601.6	268.5
19	1	10	22.933	30	52	44.634	1586.7	1257.1	1502.5	1323.5
20	1	9	6.691	30	53	24.458	1207.1	1257.3	1123.9	1288.9
21	1	6	12.019	30	33	35.802	352.9	766.3	317.5	721.4
22	1	11	38.345	30	17	45.807	1998.5	462.5	1985.4	568.9
23	1	12	19.396	30	33	35.771	2186.7	838.9	2138.3	960.9
24	1	6	25.726	30	53	7.029	406.1	1221.3	329.0	1179.8

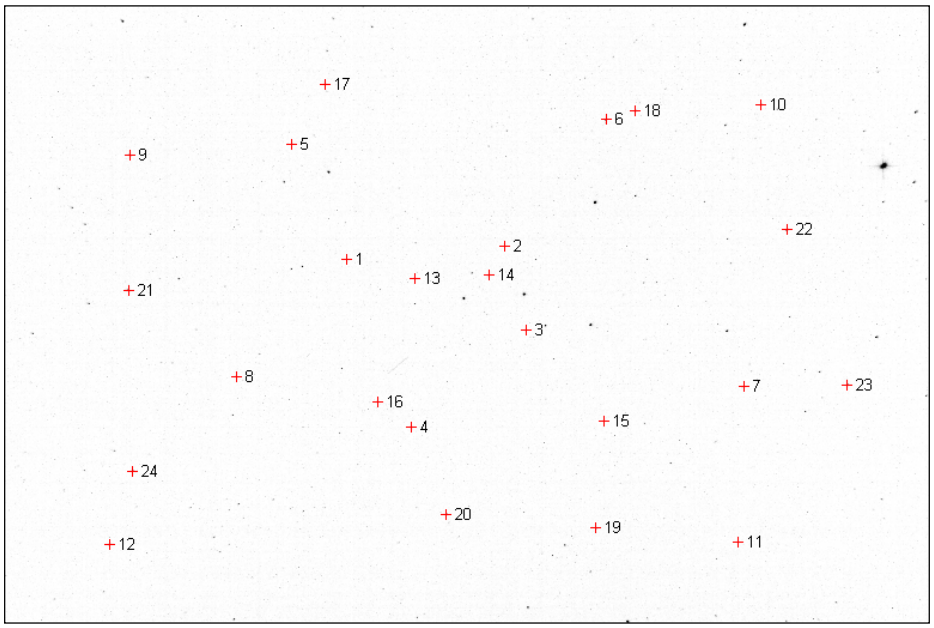


Fig. 2.1.4. Reference stars used for internal calibration - position on the Marisel image.

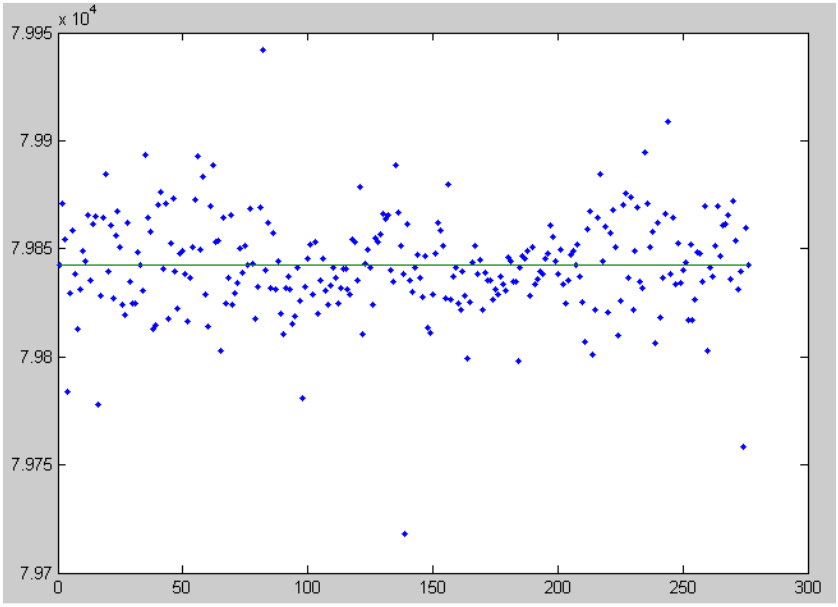


Fig. 2.1.5. Focal distances computed from all star pairs, and the median value (Marisel).

Each pair of stars can be used for focal distance computation. By analyzing all pairs that can be formed with the available reference stars, the calibration system obtains a set of candidate focal lengths, as shown in figure 2.1.5. For the final focal distance estimation, one can use the mean of these values, or the median. For our example, the focal distance obtained using the median is f=79839.60 pixels, and using the average the value obtained is f=79841.47. The standard deviation of the focal value candidates list is 22.71 pixels, which means that no significant outliers exist, a fact that is confirmed by the good agreement between the two estimated focal values. For comparison, a focal length value of 79837.02 pixels was obtained using a complex astrometrical reduction process, implemented in the AIP4Win software package [62]. This indicates that the proposed calibration method is reliable and accurate, and the assumptions we relied on are valid.

As seen from figure 2.1.5, the vast majority of the candidates are between the values of 79800 and 79900 pixels. If we take a point on the image that has a 1000 pixels distance

from the image center, the angle between the optical ray passing through this point and the optical axis of the camera would be around 0.01253 radians (43 minutes of arc), assuming a focal length of 79800. Transforming this angle back into pixel distance, assuming the other extreme of the focal distance range, 79900, gives us a value of 1001 pixels. Thus, even with the most extreme range of the focal distance candidates (well outside the standard deviation interval), and a pixel position close to the borders of the image, the violation of the assumption is minimal.

In order to additionally verify the distortion magnitude, linear, quadratic and cubic fits for the plate constants were tested using the software tool Astrometrica [63]. The linear coefficients were found to be in the order of magnitude of 10^{-5}, the second degree terms were found to be in the 10^{-10} order of magnitude, and the third degree terms were lower, in the 10^{-15} domain of values. These terms lead to a radial distortion of less than 0.01 pixels at the border of the image.

The focal length computation method is run for both the left camera system (located in the village Feleacu, Cluj county, Romania) and for the right camera system (located in Marisel, Cluj county, Romania). The estimated focal distances, which are used for the rest of the stereovision process, are f_L= 79772.22 pixels, and f_R=79839.60 pixels.

2.1.7. The translation vectors

The translation vectors can be determined from the GPS coordinates of the cameras' locations. Even if the commercial GPS devices have a low precision, with errors in the range of meters, this error is not critical for our situation, as the distance between cameras (the baseline) is several orders of magnitude greater.

The GPS coordinates of the two observation locations are the following:

Feleacu	Latitude:	46º42'36.50"N
	Longitude:	23º35'36.74"E
	Elevation:	743 m
Marisel	Latitude:	46º40'34.362"N
	Longitude:	23º07'8.904"E
	Elevation:	1130 m

In order to extract the translation vectors (in the ECEF coordinate system) from the GPS parameters, the World Geodetic System standard is used, in its latest revision, WGS 84 [15]. This standard defines the Earth's surface as an ellipsoid with the major radius (equatorial radius) a = 6378137 meters and a flattening factor f=1/298.257223563. The minor (polar) radius can thus be computed as $b=a(1-f)$, being approximately 6356752.3142 meters.

Knowing the latitude φ, the longitude λ and the elevation h, one can extract the translation vector's components X, Y and Z using the following equations (the latitude and longitude angles are converted to radians):

$$c = \frac{1}{\sqrt{\cos^2\varphi + (1-f)^2 \sin^2\varphi}} \tag{2.1.4}$$

$$s = (1-f)^2 c \tag{2.1.5}$$

$$r = (ac+h)\cos\varphi \tag{2.1.6}$$

$$X = r\cos\lambda \tag{2.1.7}$$

$$Y = r\sin\lambda \tag{2.1.8}$$

$$Z = (as+h)\sin\varphi \tag{2.1.9}$$

At this point, the translation vectors $\mathbf{T}_{CL}$ and $\mathbf{T}_{CR}$, and the intrinsic parameters of the cameras are known. All these parameters are fixed, as the camera properties do not change, and the positions of observation remain fixed. The parameters that are still unknown are the rotation matrices $\mathbf{R}_{CL}$ and $\mathbf{R}_{CR}$, and these parameters change for each pair of acquired images, as the star tracking system continuously re-orients the cameras.

2.1.8. Online calibration of the rotation matrices

Any calibration requires a set of reliable features that have known real-world coordinates, and can be easily identified in the image space. The reference features than can be used are the stars. They are catalogued, so their position in the real world is known, and they can be easily identified in the image space by image processing algorithms.

As the observation system will be pointed towards different areas of the sky, a fresh set of reference stars must be retrieved. In order to accomplish this task, the U.S. Naval Observatory Interactive Catalog and Image Search [64] is used. The catalog allows the user to specify a sky region, in terms of the equatorial coordinates (Right Ascension and Declination) of its center, and the height and width of the region in arc minutes. Other parameters, such as the epoch of the measurement and the magnitude range of the stars to be retrieved, are provided, in order to get the most relevant stars.

A software module was developed, capable of automatically retrieving the relevant stars for any region in the sky. The approximate center of the image is specified in equatorial coordinates, along with the size of the field of view. The magnitude interval is set such that the stars do not appear too large in the image, but at the same time they should be clearly identifiable (not too faint). From the list of retrieved stars, the ones that are too close together are removed, as they may cause false matching results. The remaining stars are used for calibration.

Thus, once the telescope is oriented, the equatorial coordinates of stars that can be used for calibration, RA_i and DEC_i, i=1…N, are retrieved in a matter of seconds. In what follows, the star coordinates RA and DEC will be handled as regular angles, which means that they will be converted to radians.

After the coordinates of the stars are retrieved, a hypothetical image representation of these stars is created. This means that for each star i an image space coordinate pair (x_i, y_i) is computed.

First, the average Right Ascension and the average Declination are computed for the set of N calibration stars:

$$\overline{RA} = \sum_{i=1}^{N} RA_i \tag{2.1.10}$$

$$\overline{DEC} = \sum_{i=1}^{N} DEC_i \tag{2.1.11}$$

Then, relative angular displacements from the group center are computed for each star. The angular displacements corresponding to the right ascension are scaled so that the narrowing of the interval with the increase of the declination is accounted for:

$$\gamma_i = (RA_i - \overline{RA})\cos(\overline{\mathrm{DEC}}) \tag{2.1.12}$$

$$\varphi_i = DEC_i - \overline{DEC} \tag{2.1.13}$$

Now, the angular displacements can be converted to pixels, by using the focal distance of the camera as a scaling factor. The resulted coordinates are centered in the image plane, by adding the half width (w) and the half height (h) of the image:

$$x_i = \gamma_i f + \frac{w}{2} \tag{2.1.14}$$

$$y_i = \varphi_i f + \frac{h}{2} \tag{2.1.15}$$

The predicted star coordinates in the image space can be close to the real image position of these stars, or they can be very far apart, as shown in figure 2.1.6. The only thing one can rely on is that the relative distances between the stars in the set are correct, if the focal distance of the camera is correctly calibrated. The whole group of stars is an object that is rotated and translated with respect to its correct position in the image, and sometimes the translations and rotations can be significant (in figure 2.1.6, right, a rotation of 180 degrees is shown). Thus, at the beginning of an observation sequence for a specific region of the sky the star to image matching process must allow a broad range of displacement and rotation.

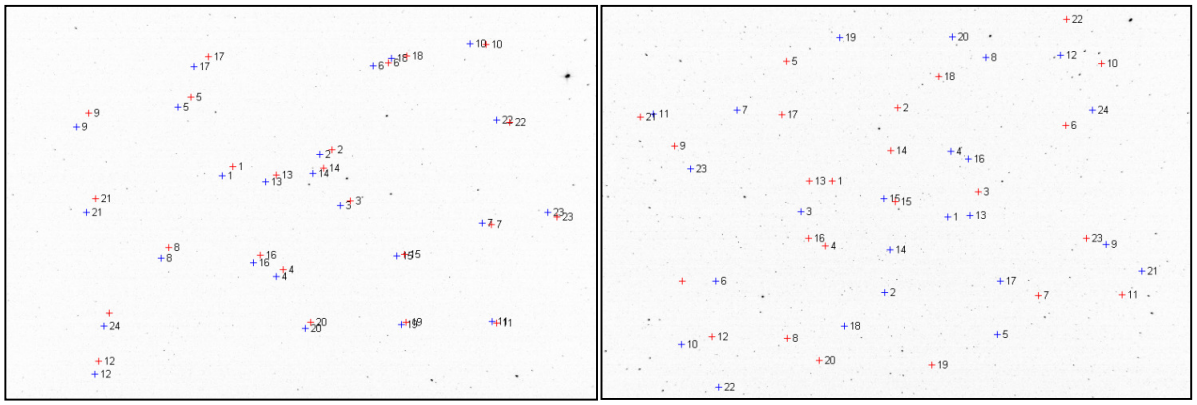


Fig. 2.1.6. Predicted image positions of reference stars (blue) versus actual positions (red). Best case scenario (left, rotation of 0°), versus worst case scenario (right, rotation of 180°).

The center of mass (of the star set object) has the image space coordinates $\bar{x}$ and $\bar{y}$, which represent the averages of the image coordinates of each star i in the set, x_i and y_i. Applying the rotation and translation transformations to the whole star set (such that the form or the scale of the object does not change) is equivalent to altering the position of each star in the image, using the following equation:

$$\begin{pmatrix} x'_i(\delta x, \delta y, \alpha) \\ y'_i(\delta x, \delta y, \alpha) \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x_i - \bar{x} \\ y_i - \bar{y} \end{pmatrix} + \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} + \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} \tag{2.1.16}$$

Using the transformation specified by equation (2.1.16), the objective function that measures the quality of the match between the hypothetic star position and the acquired grayscale image I is defined as:

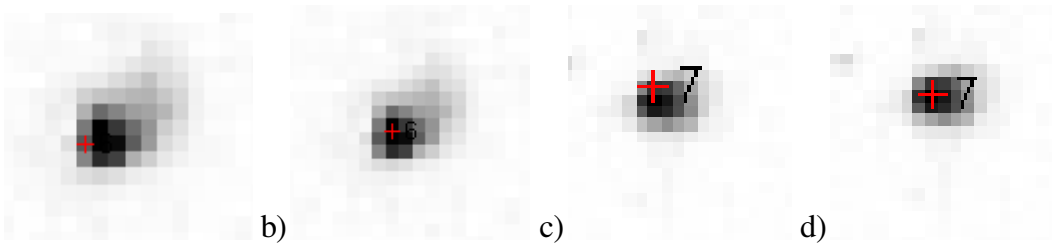
$$M(\delta x, \delta y, \alpha) = \sum_{i=1}^{N} \log(I(x'_i(\delta x, \delta y, \alpha), y'_i(\delta x, \delta y, \alpha))) \tag{2.1.17}$$

In equation (2.1.17), N denotes the number of reference stars. The best possible match between the star coordinates and the image is signaled by a maximum of the objective function M, as each star in the image has a brighter value than its surroundings. Thus, the desired parameters of the match are found as:

$$(\delta x_{match}, \delta y_{match}, \alpha_{match}) = \arg\max_{\delta x, \delta y, \alpha} M(\delta x, \delta y, \alpha) \tag{2.1.18}$$

The search for the best parameters for star matching is performed using a wide search space at the beginning of the observation sequence (for the x and y displacement, a range of plus or minus 300 pixels is allowed, and for the rotation angle the full 360 degrees angular range is explored). However, searching this space is computationally expensive, and therefore in the subsequent frames of the sequence the search space is restricted around the previously found parameters. This does not exclude the possibility of failure – sometimes the movement of the rig may be too abrupt, and a re-initialization of the parameters is required. This situation is signaled by the detection routine, which will be described in the next sections. Thus, a feedback between the detection routine and the calibration routine ensures a fast star matching process, with the possibility of recovery from failure.

The final step for the star to image matching process is a local refinement of the position detected by rotating and translating the whole star ensemble. In a small, 7x7 sized neighborhood around the estimated position, the local maximum of the intensity image is searched. Then, the positions of the pixels in the neighborhood that have a brightness equal or 5% less than the local maximum are found, and the average position obtained from these pixels is taken as the locally refined position of the star, as shown in figure 2.1.7.



a) b) c) d)

Fig. 2.1.7. Refining the position of the stars: a) and c) are the positions estimated by rotating and translating the whole set of stars, b) and d) are the positions after local refinement.

Now that the reference stars are matched to the acquired image, the rotation matrix can be computed. For each matched star i, we now know its declination DEC_i, its right ascension RA_i, and its coordinates in the image space, x_i and y_i (for both cameras). The right ascension of the star, RA_i, is converted to the Hour Angle relative to the zero meridian, $HA_{0,i}$, which is the star's azimuth with respect to the zero meridian, and thus tied to the Earth Centered, Earth Fixed (ECEF) coordinate system. The relation between the Right Ascension and the zero meridian Hour Angle is the following:

$$HA_{0,i} = LST_0 - RA_i \tag{2.1.19}$$

The term LST_0 is the Local Sidereal Time of the zero meridian, obtained from the time and date of the observation.

The components $r_{n,k}$, n=1..3, k=1..3 of the rotation matrix **R**, the declinations DEC_i, the Hour Angles $HA_{0,i}$, and the image positions of the stars, x_i and y_i, are connected through

the following equation, in which (x_c, y_c) is the position of the principal point and f is the focal distance in pixels:

$$\begin{cases} \frac{x_i - x_c}{f}\left[r_{13}\cos DEC_i \cos HA_{0,i} + r_{23}\cos DEC_i \sin HA_{0,i} + r_{33}\sin DEC_i\right] + \left[r_{11}\cos DEC_i \cos HA_{0,i} + r_{21}\cos DEC_i \sin HA_{0,i} + r_{31}\sin DEC_i\right] = 0 \\ \frac{y_i - y_c}{f}\left[r_{13}\cos DEC_i \cos HA_{0,i} + r_{23}\cos DEC_i \sin HA_{0,i} + r_{33}\sin DEC_i\right] + \left[r_{12}\cos DEC_i \cos HA_{0,i} + r_{22}\cos DEC_i \sin HA_{0,i} + r_{32}\sin DEC_i\right] = 0 \end{cases} \quad (2.1.20)$$

For each star in the set, two equations are generated. The equation system is supra-determined, and the Gauss-Newton iterative method is used for finding the nine unknowns.

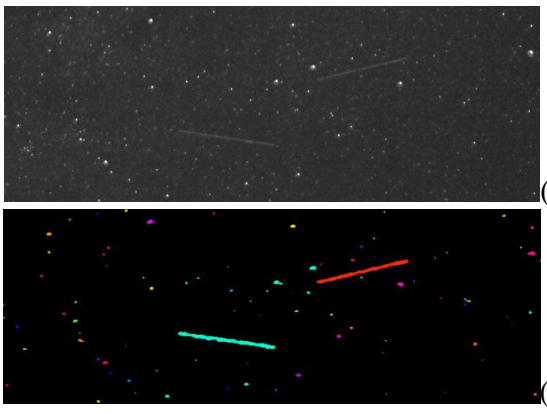
2.1.9. Detection of satellites from consecutive images

Low Earth Orbit objects (LEOs) and lower orbit MEOs have a characteristic signature streak, which can be detected even without the help of stereovision. Thus, one approach towards detection is to identify the satellites independently from the image sequence of each sensor. For each newly acquired image of a sequence, the following steps are executed:

- Background removal
- Object candidates detection in the image space
- Object classification

The image pixels that form the possible object are identified as pixels that change in time, with respect to a background. Because a star tracking system is used, the only changing pixels that are expected are those caused by a moving object. A simple difference between the frames may be the first solution of choice, but a better choice still is to estimate the background image with a moving average technique, which has a smoothing effect. The value of the averaged background is subtracted from the current frame, and the difference is thresholded using a small enough threshold that the faint contrast objects are still preserved.

Individual image pixel groups are identified by applying a labeling algorithm [65] to the binary image resulted in step 1. After that, the labeled binary objects must be classified, so that we can decide whether they are satellite streaks, planes or other objects. In order to perform that classification, we will approximate every binary object by an ellipse, and we'll compute the geometrical properties of this ellipse.



(a) (b)

Fig. 2.1.8. (a) The original image containing two satellite streaks. (b) Labeled objects on the background-removed image.

The properties for the image object classification process are: Area, Major Axis Length (L_{MAX}), Minor Axis Length (L_{MIN}) and Eccentricity (e).

In order to build a decision tree, a database for training and testing database was created by manually labeling each object. The result is a database containing all the relevant objects and their classes. Using this database, a decision tree classifier was automatically using Weka [66]:

```
Area <=85: other
Area > 85
 | Eccentricity <= 0.99
        | | Eccentricity <= 0.935: other
        | | Eccentricity > 0.935
        | | | MajorAxisLength <= 47.543: other
        | | | MajorAxisLength > 47.543: satellite
        | Eccentricity > 0.99
        | | MajorAxisLength <= 200.337: satellite
        | | MajorAxisLength > 200.337
        | | | Eccentricity <= 0.998: plane
        | | | Eccentricity > 0.998
        | | | | MajorAxisLength <= 294.676: satellite
        | | | | MajorAxisLength > 294.676: plane
```

The decision tree is able to discriminate between three classes of sky objects: satellite (or generic LEO object), plane, or other (which means any image artifact).

2.1.10. Detection of satellites from a stereo image pair

The higher orbit satellites are less bright, and also move much slower than the lower orbit ones, meaning that these satellites do not generate the characteristic streak that can be easily identified. For detecting these satellites, the parallax effect in the stereo image pair is used. This effect means that the relative position of the satellite feature with respect to the common reference stars matched in the two images, for the purpose of calibration, will be different.

Assuming that the difference in scale between the two images is negligible, as they are acquired with similar optical instruments, one can define a rotation angle and a translation vector between the left group of reference stars positions and the right group of reference stars position. Using the known image coordinates of the reference stars, the rotation angle and the translation vector (which contains the translation amounts for the x and the y coordinate) can be computed by least squares error minimization.

After the transformation parameters (rotation and translation) between the left and the right image of the stereo pair are found, they can be used to align all stars of the left image to match the ones on the right, and all the stars on the right image to match the ones on the left. This is the main idea for satellite detection: as the stars are fixed with respect to each other, transforming the whole right image using the rotation and translation parameters obtained from the reference stars analysis will get us a new image that has all the stars in the same position as the left image, but the satellite streak's position will not coincide. Figure 2.1.9 shows the warping process, for a small region of the images, containing the satellite streak.

Thus, the satellite streak detection principle is the following: warp (rotate and translate) one image of the pair, keep the other image unchanged, and look for the differences.

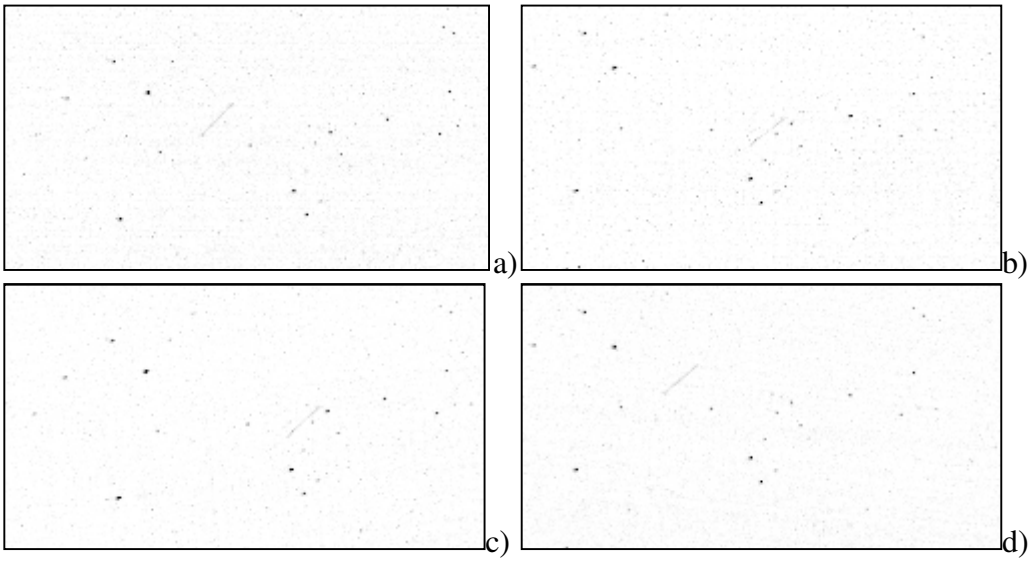


Fig. 2.1.9. Warping: aligning the star background between the images of the stereo pair. Left (a) and right (b) original images, cropped. Warped right image to match the stars of the original left image (c), and warped left image to match the stars of the right image (d).

The original image (left or right) will be considered as foreground, and the corresponding warped image (right to left, or left to right), smoothed by a convolution with a Gaussian kernel G, will be considered as the background. For example, for detecting the satellite pixels of the left image, the foreground and the background images are defined as:

$$I_F = I_L$$
$$I_B = I_{RL} * G \tag{2.1.21}$$

In what follows, the detection of the satellite pixels from the left image, using the warped right image as the background, will be presented. For detecting the satellite pixels on the right image, the same steps are applied, using as foreground the right image I_R and as background the warped left image I_{LR} convolved by the Gaussian kernel G.

Identifying the satellite pixels in the foreground image means identifying the relevant differences between the foreground and the background. This process is, however, more complex than simply computing the difference between the intensity values of the pixels of the two images, because the two images are acquired from different locations, with non-identical cameras, and the satellite streak signal is not very strong with respect to the stars and the noise in the image. In order to overcome these difficulties, an elaborate strategy for finding the satellite pixels was devised.

First, a threshold image I_T is defined. Each pixel of the threshold image at image coordinates (x, y) is defined as a fraction of the maximum between the foreground image pixel and the background image pixel at the same coordinates:

$$I_T(x, y) = \eta \max(I_F(x, y), I_B(x, y)) \tag{2.1.22}$$

The fraction coefficient η is adjusted by trial and error, currently being set to $\eta = 0.4$.

The difference between the foreground pixel intensity value and the background pixel intensity value at coordinates (x, y) is compared with the threshold image pixel value at the same coordinates. The pixel difference value above the pixel threshold indicates a possible

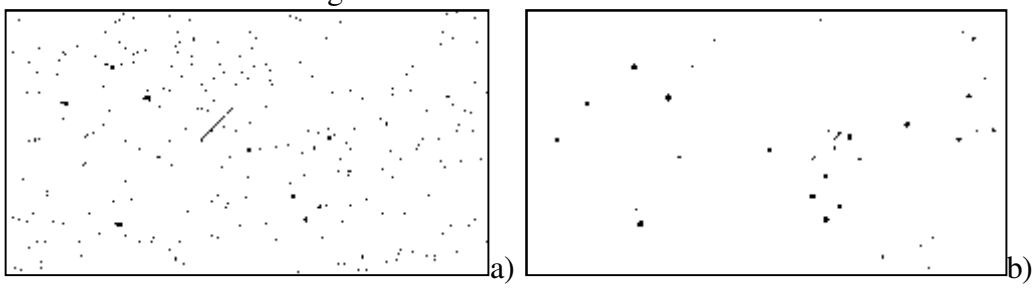
satellite pixel at the specified coordinates. Unfortunately, this does not remove all possible false positives.

The false positives are the strong differences between foreground and background which may arise at the location of bright stars. These bright stars are not point-like in the image, and sometimes, due to chromatic aberrations or CCD saturation, they are not even circular (their shape may be elliptic, or they may have linear steaks emanating from the central point). Corrupted star shape, or uneven back lighting of the location, may lead to significant differences around the bright stars, even when their centers are aligned by warping. In order to avoid these false positives, a mask image for the regions most likely to cause false positives is created.

First, a static threshold T is applied to both the foreground and the background images. The value of this threshold is tuned experimentally (currently T=10). The foreground mask M_F and the background mask M_B are binary (logical) 2D arrays, defined as:

$$\begin{aligned} M_F(x,y) &= (I_F(x,y) > T) \\ M_B(x,y) &= (I_B(x,y) > T) \end{aligned} \tag{2.1.23}$$

The two masks are shown in figure 2.1.10.



a) b)

Fig. 2.1.10. The foreground star mask (a) and the background star mask (b).

In order to increase the safety margin, the two masks are dilated with a circular structured element D, with the radius of 4 pixels. This will account for possible small misalignments in the warping process, and for incomplete identification of all star pixels in the thresholding process.

$$\begin{aligned} M_{FD} &= M_B \oplus D \\ M_{BD} &= M_D \oplus D \end{aligned} \tag{2.1.24}$$

The two masks are then combined using a pixel wise $\wedge$ (and) operation. The position of the bright stars is the same in the foreground and in the background, the dilation accounts for small misalignments, and thus we expect that the significantly bright stars will have non-zero (logical true) pixels in both masks. The final star mask M is, for each pixel position (x, y):

$$M(x,y) = M_{DF}(x,y) \wedge M_{DB}(x,y) \tag{2.1.25}$$

The final star mask is shown in figure 2.1.11.

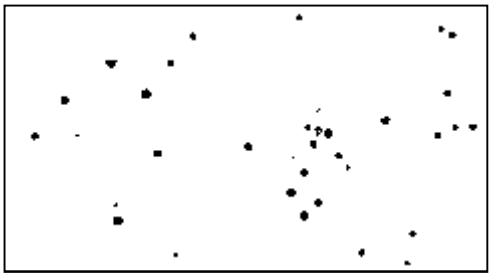


Fig. 2.1.11. Combined star mask, showing the exclusion zones for avoiding false positives.

Now, the difference between the background and the foreground can be computed. The arithmetical difference is compared with the threshold image I_T, and with the fixed, low threshold T, retaining the pixels that pass both thresholds, excluding those that correspond to the exclusion mask M. Formally:

$$I_{\Delta}(x, y) = I_F(x, y) - I_B(x, y) \tag{2.1.26}$$

$$I_{\mathrm{R}}(x, y) = (I_{\Delta}(x, y) > I_T(x, y)) \wedge (I_{\Delta}(x, y) > T) \wedge \neg M(x, y) \tag{2.1.27}$$

The result binary (logical) image I_R contains all pixels for which the difference I_{Δ} between foreground and background passes both the global threshold T and the location-specific threshold $I_T(x, y)$, and do not fall inside the exclusion zone. The result image is depicted in figure 2.1.12.

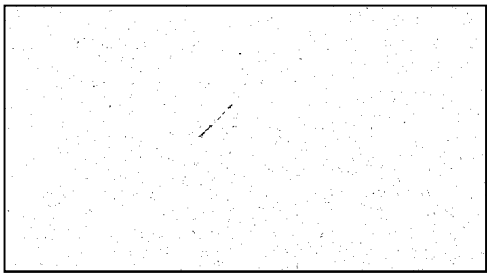
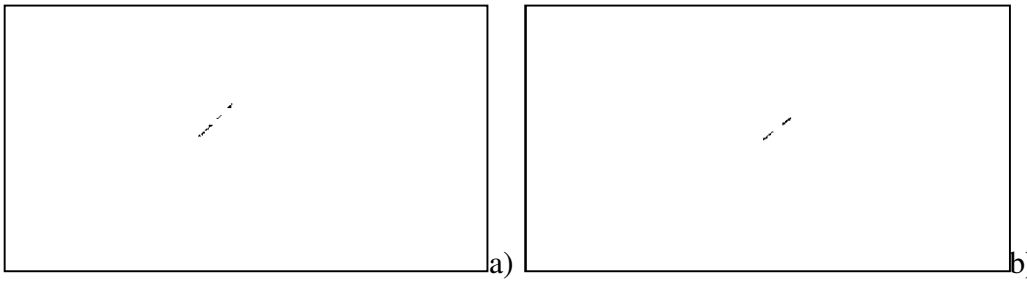


Fig. 2.1.12. Candidate satellite pixels obtained by foreground-background difference analysis.

It can be seen from figure 2.1.12 that even if the satellite's pixels are clearly identified, they are not the only pixels that have passed the established conditions. Fortunately, the other non-false pixels are usually isolated, while the pixels belonging to the satellite are grouped together into larger clusters. For this reason, a simple connected component identification process (binary image labeling) is applied, and the clusters with a pixel count of less than 10 pixels are excluded. Figure 2.1.13 shows the remaining pixels, after labeling and area based validation, for the left image as foreground (the situation for which all the processing steps have been described), and also for the right image as foreground (results obtained by executing the same algorithm steps, with the right image as the foreground and the warped left image as background).



a) b)

Fig. 2.1.13. Satellite pixels, validated by area analysis. Detection results on the left image (a) and on the right image (b).

A final step in the detection process is to cluster all streak fragments that are close together into a single object. This step is necessary because, as can be seen from figure 2.1.13, most of the streaks will not be detected as a compact object, due to the low contrast and the fact that they may be close to a significantly bright star, which will exert its exclusion zone and cut some of the pixels from the result.

This detection algorithm has a very high tolerance to noise and different background illumination between the images of the stereo pair, while retaining a high sensitivity. The number of false positives resulted from this processing stage is very low, easily removable in the next step, the stereo measurement. If a very high number of satellite candidates are found in the detection step, this is a signal that the reference stars are not matched properly in the two frames, and the star matching process is re-initialized by searching for rotations and translations in the whole parameter space, instead of reusing the position of the past frames in the sequence as the starting point, and a reduced range for rotation and translation.

2.1.11. Establishing the stereo correspondence

Stereovision-based 3D reconstruction relies on matching features from the left and right images, followed by triangulation. If the sky objects were point-like in the image space, the correspondence search process would be straightforward. However, we have already seen that this is not the case, and the image signature of these objects is a line segment, having a length proportional to the speed of the object, and to the exposure time of the camera.

If the conditions were ideal, the moving object would become visible at the same time in both images of the stereo pair, and it will also disappear at the same time. If this were the case, the easiest way to find the corresponding points is to look at the ends of the line segments. Unfortunately, there are several causes that make this approach a bad idea. First, the two observation sites have different background illumination conditions, a condition that affects the thresholding of the object's features and may cause different length of segments in the two images. Second, the line segment's length may be influenced by the Gaussian noise of the image, or by the background stars that happen to be near the object, and these conditions may be slightly different for the two cameras. Third, due to the low cost equipment used for camera triggering (off the shelf GPS receivers connected to PC's) there may be some errors in the camera synchronization process, errors that may influence the start of the exposure period (which influences the position of the first end of the linear segment) or the duration of the exposure (which influences the segment length). All these conditions make a compelling argument against using the ends or the middle of the segment for matching. Fortunately, there is a more reliable property of the moving object related line segment, which facilitates accurate, sub-pixel based matching between the images: *the trajectory line itself*.

While the trajectory line is unbounded, and therefore the correspondence points are difficult to find, there is another restriction that drastically limits the search space: the epipolar constraints. The epipolar constraint says that for each point $\mathbf{P_L}$ of the left image, the possible correspondences in the right image are located on a line called the epipolar line. This constraint is shown in the figure below: the optical centers of the two cameras form, together with the unknown 3D point **X** that needs to be measured, a plane that intersects the two image planes forming the epipolar lines. One can see that the plane is completely determined by one projection ray (caused by one point in one of the images) and the line joining the cameras optical centers. This means that knowing a point in the left image and the camera parameters (intrinsic and extrinsic), the epipolar line that will contain the corresponding right point $\mathbf{P_R}$ can be computed.

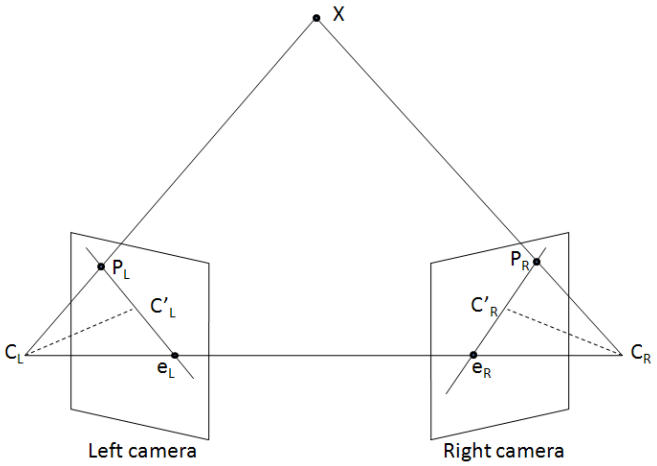


Fig. 2.1.14. The epipolar geometry.

The epipolar line for each point of the left image can be computed from the intrinsic and extrinsic parameters of the cameras, via the fundamental matrix [67]. The process of correspondence finding is depicted in figure 2.1.15, and has the following steps:

For each candidate object on the left image:

Compute the center of mass of the candidate object, C_L;

Using the fundamental matrix computed from cameras' parameters, compute the epipolar line on the right image, the geometric locus of the stereo correspondents of the left center of mass;

For all candidate objects in the right image:

Compute the distance between their centers of mass and the epipolar line (distances d_1, d_2, d_3 in figure 13);

If the distance of a right candidate object to the epipolar line is below a threshold (i.e. 50 pixels), then:

Compute the elongation axis of the candidate object;

Compute the intersection between the elongation axis and the epipolar line. This will be the stereo match of C_L, denoted C_R;

Using C_L and C_R, apply stereo triangulation and compute the 3D coordinates. Accept the object as valid if the distance to the observer is in the accepted range.

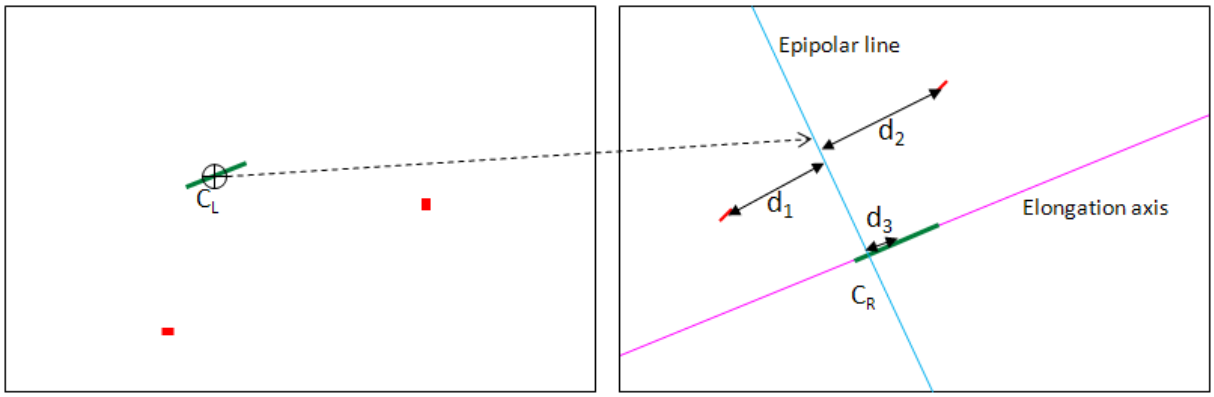


Fig. 2.1.15. Searching for the left-right stereo correspondence. The green objects are the satellite streaks, and the red objects are the false candidates.

Using the distance to the epipolar line as a correspondence filter, and applying a range validation on the stereo 3D reconstruction results, has the effect of preventing the possible false positives of the detection stage to propagate to the final results. This is a great advantage of stereovision – imposing geometrical constraints severely limits the valid left-right pairs, and thus occasional false positives in the detection phase are not critical.

The correspondent point for the left center of mass is, as described, not the right center of mass of the object that passes the distance to the epipolar line test, but the intersection between the epipolar line and the elongation axis. This point was preferred because the center of mass is not sufficiently stable for matching – due to weak contrast or due to bright stars in the vicinity of the object, some object pixels may be lost, and the center of mass will be displaced. The elongation axis, on the other hand, does not change much, as long as an adequate number of pixels for the object are found.

2.1.12. Computation of the 3D coordinates of the satellite

The corresponding points in the left and right images are now available, and thus the triangulation process can be applied. This triangulation will transform the image point pair ($\mathbf{P}_L$, $\mathbf{P}_R$) into a single 3D point $\mathbf{P}_W$. This 3D point lies at the intersection of the projection rays passing through $\mathbf{P}_L$ and $\mathbf{P}_R$ (the lines passing through the image points and the optical centers).

Because the camera parameters are known, one can write the equation of the line passing through the 3D point $\mathbf{P}_W$ and the optical center of the left camera $\mathbf{C}_L$ [68]. The perspective projection leads to the following relation:

$$\begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} = \mu \mathbf{R}_L \begin{bmatrix} x_L - x_{CL} \\ y_L - y_{CL} \\ -f_L \end{bmatrix} + \begin{bmatrix} X_{CL} \\ Y_{CL} \\ Z_{CL} \end{bmatrix} \tag{2.1.28}$$

where μ is a scaling factor depending on the distance Z_W. Multiplying with $\mathbf{R}_L^T$ leads to:

$$\begin{bmatrix} x_L - x_{CL} \\ y_L - y_{CL} \\ -f_L \end{bmatrix} = \mu^{-1}\mathbf{R}_L^T \begin{bmatrix} X_W - X_{CL} \\ Y_W - Y_{CL} \\ Z_W - Z_{CL} \end{bmatrix} \tag{2.1.29}$$

which can be detailed as:

$$\begin{bmatrix} x_L - x_{CL} \\ y_L - y_{CL} \\ -f_L \end{bmatrix} = \mu^{-1} \begin{bmatrix} r_{11} & r_{21} & r_{31} \\ r_{12} & r_{22} & r_{32} \\ r_{13} & r_{23} & r_{33} \end{bmatrix} \begin{bmatrix} X_W - X_{CL} \\ Y_W - Y_{CL} \\ Z_W - Z_{CL} \end{bmatrix} \tag{2.1.30}$$

The above equation is a system of three equations with four unknowns. The third equation can be used to write μ^{-1} as a function of the other unknowns:

$$\mu^{-1} = \frac{-f_L}{r_{13}(X_W - X_{CL}) + r_{23}(Y_W - Y_{CL}) + r_{33}(Z_W - Z_{CL})} \tag{2.1.31}$$

Replacing μ^{-1} in the first two equations, one obtains the following system:

$$\begin{cases} x_L - x_{CL} = -f_L \dfrac{r_{11}(X_W - X_{CL}) + r_{21}(Y_W - Y_{CL}) + r_{31}(Z_W - Z_{CL})}{r_{13}(X_W - X_{CL}) + r_{23}(Y_W - Y_{CL}) + r_{33}(Z_W - Z_{CL})} \\ y_L - y_{CL} = -f_L \dfrac{r_{12}(X_W - X_{CL}) + r_{22}(Y_W - Y_{CL}) + r_{32}(Z_W - Z_{CL})}{r_{13}(X_W - X_{CL}) + r_{23}(Y_W - Y_{CL}) + r_{33}(Z_W - Z_{CL})} \end{cases} \tag{2.1.32}$$

This system has two equations and three unknowns. Writing the same equations for the right camera system leads to a total of four equations with three unknowns, a supra-determined system that can be solved by a least squares approach. From a geometrical point of view, the least squares solution will find the 3D point that has a minimum distance from the two projection lines, even if small calibration or matching errors will prevent these lines from intersecting [67].

2.1.13. Experimental results

In order to estimate the accuracy of the range measurement algorithms for LEO objects, known satellites were used, because for some of them the ground truth information can be extracted from the website www.heavens-above.com. Several satellites that were visible at the time and place of the observations were manually identified, and their range information extracted.

The following table shows the computed results for several known LEO satellites. The observations were made on July 9, 2011.

Table 2.1.2. Range measurement results, LEO satellites.

Local time	**Object name**	**Orbit Min x Max (km)**	**Distance to observer (km)**	**Distance to ground - computed (km)**	**Distance to observer - computed (km)**
2:58:51	Cosmos192	704x718	714	**704**	**713.70**
3:09:07	Cosmos1743	543x564	586	**566**	**582**
3:44:32	Cosmos923Rocket	761x789	1078	**790**	**1109**
2:50:59	Cosmos2263Rocket	825x848	1161	**806**	**1131**
3:41:44	Meteor*	N.A.	N.A.	**100,25**	**105**

* The detected distance meteor-observer is in the typical range interval for meteor burning in the atmosphere.

The results for LEO detection and ranging seem to be very promising, both for the distance to observer and for the distance to the ground. With the current limitations of camera sensitivity, the system is able to reliably detect objects that have a distance from the ground in the interval of 100 to 1500 kilometers.

For testing the detection and measurement accuracy for MEO and beyond objects, several observations have been performed, covering four sky zones, specified by a central reference star, observations aimed to detect six satellites, four MEOs and two Molniyas (Highly Eccentric Orbit objects), whose apparitions and ranges were predicted using the astronomical software The Sky [69].

The aspect of the observed satellites in the image proved to be extremely variable. While the length of the streak was consistent with the satellite's nature, the brightness of the satellite's pixels in the image seems to be unique to each of the observed case. Some examples are shown in figure 2.1.16: the average brightness of the satellites ranges from 15 to 65 DNU (the brightness of a pixel can have values from 0 to 255 DNU – Digital Number Unit), which makes the satellite brighter than the background, which has an average brightness of 5 DNU, but significantly less bright than the stars in the image, which can go up to the saturation value of 255 DNU. For the higher range satellites such as the Molniyas, the lower brightness is also combined with a shorter streak.

Fig. 2.1.16. Aspect of satellites in the image space. From left to right: 733, 738, PRN10, PRN8, Molniya 3-41. Average satellites' brightness: 55, 65, 30, 22, 15 DNU. Average background brightness: 5 DNU.

A variable brightness level, and also a variable length of the satellite streak, can be also observed for the same satellite, in different frames. The reason for this behavior is a fast spin of the satellite, which changes the amount of light the object reflects, as the reflectivity of the satellite's surface is not homogeneous. This effect is shown in figure 2.1.17**,** for a Molniya type satellite, which can go from a very strong brightness (value 255 DNU – saturation) to invisibility in 3-4 frames. This behavior strongly affects the detection rate.

Fig. 2.1.17. Variable perceived brightness of Molniya 1-91 due to rapid spinning. From saturation (left) to a brightness of 36 DNU (middle) and then to 12 DNU (right).

A summary of the detection and coordinate computation results is presented in Table 2.1.3. The predicted range, with respect to the Feleacu observation station, was generated using *The Sky*. The measured range is the distance between the 3D coordinate vector of the satellite, in the ECEF coordinate system, computed by stereovision, and the 3D coordinate vector of the Feleacu observation station, computed from the GPS coordinates using the geodesic model. The observation time is the time interval between the first presence of the satellite in both images of the stereo pair and the last presence. The average time elapsed between frames is 8 seconds, which includes the exposure time of 5 seconds. The detection rate is computed as the ratio between the number of frames the satellite is correctly detected and ranged and the total number of frames the satellite was observable with both telescopes.

Table 2.1.3. Measurement results for the observed satellites.

Sky region	**Satellite name**	**Observation Time (UTC+3)**	**Mean predicted range (km)**	**Mean measured range (km)**	**Mean error (km)**	**Mean absolute error (km)**	**Detection rate (%)**
SAO37985	GLONASS 733	0:33:24-0:35:16	19223.60	19131.73	-91.87	91.93	86.66
SAO37985	GLONASS 738	0:35:40-0:37:24	19242.26	19168.26	-74.00	73.99	100
SAO54449	GPS PRN 10	1:21:32-1:23:56	20455.47	20471.77	16.30	41.66	89.47
SAO54449	GPS PRN 8	1:47:48-1:49:40	20860.97	20849.13	-11.84	119.89	100
SAO36361	Molniya 3-41	2:09:16-2:16:04	32054.90	32590.26	-535.36	547.48	90.90
SAO25214	Molniya 1-91	2:37:40-2:50:44	39828.90	39651.32	-177.58	485.15	40.23

The test results show variable measurement accuracy. As expected from a stereovision sensor, the accuracy is significantly better for the MEO satellites than for the Molniyas, as the accuracy drops with the distance. The measurement error corresponds to a pixel uncertainty of matching of 1-3 pixels, which, taking into account the detection method, is expected. There are, however, several cases where the measurement errors seem to be systematic, pointing to a permanent offset between the predicted position and the measurement, which may be caused by outdated predictions.

Several experiments aimed to verify the assumption that the position of the principal point is not relevant to the measurement process were also performed. The position of this point is important in itself, as it defines the interior geometry of the camera, but it does not affect the triangulation process, due to the fact that any deviation from the true value of this point will be compensated by the rotation matrices. In order to support this claim, the position

of the principal point has been significantly altered, and the detection and measurement algorithm has been applied for the same frame, which includes a GPS satellite. The following facts were observed:

- When the principal point is assumed to be in the center of the image, the satellite is detected at the ECEF coordinates (in km) X=21131.22, Y=4908.85, Z=15042.85, having the distance to the observer 20278.07 km.
- When the principal point is displaced from the center by 50 pixels on both axes, the measured ECEF coordinates are X=21131.53, Y=4908.91, Z=15043.04, and the range is 20278.44 km.
- When the principal point is displaced from the center by 200 pixels on the horizontal axis, and by 100 pixels on the vertical, the measured ECEF coordinates are X=21132.30, Y=4909.05, Z=15043.52, the range being 20279.36 km.
- When the principal point is displaced to the top left corner of the image, more than 1000 pixels away from the original position, the measured ECEF coordinates are X=21143.52, Y=4911.16, Z=15050.34, the range being 20292.65.

All these displacements are extreme, well beyond the normal displacements of a principal point with respect to the image center. However, the results on the 3D reconstruction process are minor, well below the uncertainty expected from triangulation, assuming image space positioning errors of sub-pixel accuracy. Thus, the assumption about the principal point is valid.

2.1.14. Conclusions

This chapter presented solutions for the surveillance of the Earth orbits using large baseline stereovision, relying on low cost equipment, able to detect and range objects having a wide range of altitudes, almost in real time, with a high detection rate. The main advantages of applying stereovision to space surveillance are the following:

- It requires no elaborate, lengthy surveillance of the same sky region. The detection using stereovision is instantaneous – if an orbiting object is in the left and right frame, it will be detected instantly.
- It requires no strong assumptions about the orbits to be surveyed: the same setup can detect a 19000 km altitude satellite and a 40 000 km satellite, or both, if they happen to be in the same image. Only when switching between highly different ranges, such as between LEO and MEO, the optical setup needs to be changed.
- Automatic, instantaneous ranging: the system is able to produce a 3D coordinate vector of the object detected. While the accuracy of the measurement is not comparable to what one can obtain using multiple observations and employing elaborate orbit calculation tools, having an instant 3D position approximation may be highly valuable in the process of large sky area surveillance, and can provide a quick working estimate that can be further refined.
- Works with low cost equipment, in less than ideal working conditions. The cheap and lightweight equipment can be easily set up to a new site, and the algorithms have proven resilient to image noise and background pollution. In fact, one of the observation sites is quite close to the busy city of Cluj-Napoca, and this has not impaired the detection performance.

The methods can be easily deployed on multiple observation sites, for different types of instruments, with different focal lengths and apertures. This way, many regions of the sky can be surveyed at the same time, with detection results delivered for each acquired image pair.

(b-ii) Scientific, professional and academic future development plans

The candidate's activity in the future will rely on the experience, the results and the reputation obtained from the activity that lead to the present habilitation thesis.

For the near future, the research activity will be related to the research projects the candidate is involved in, including a research project in which the candidate has the role of manager. This means that the near future research activity will be focused on the two main directions that have produced the results described in this thesis: modeling and tracking complex, dynamic 3D environments, and stereovision-based space surveillance.

The field of dynamic environment modeling and tracking is strongly related to the field of sensorial perception for driving assistance, a field in which the research group that the candidate is part of, the Image Processing and Pattern Recognition Group (IPPRG) is involved in since 2001, mainly through contracts with industrial partners. Thus, the results that will be pursued will need to have real world application.

The main challenges that will be tackled in the context of dynamic environment modeling and tracking are the following:

1) Developing a dynamic world model and tracking solution that will integrate the stereovision information in the measurement process without first transforming it into a raw elevation map. The raw stereo information will include disparity and grayscale values for each pixel in the image, and the measurement model will relate these values and their uncertainty directly to the tracking mechanism. In this way the error of the measurement can be estimated with much better precision, which will improve the tracking results.

The main problem with using the raw elevation map as the source of measurement is that the process of achieving this map is complex and its errors are difficult to model. The raw stereo information is composed of disparity and gray level information for each pixel in the left or the right image. This information is then transformed into 3D points, which means that the disparity information, which has a constant error related to the uncertainty of the stereo matching process, is transformed into distance, whose uncertainty is variable, increasing as the distance itself increases. This distance error influences the errors in the other coordinates as well, producing an unevenly spread set of 3D points in the scene. From these points, the elevation map is built, which is another complex process, which can filter some of the point errors, or can amplify others. All these transformations lead to a high difficulty in producing an accurate measurement model. If the tracked scene could be related directly to the (u – pixel column, v – pixel row, d – pixel disparity, g – pixel gray value) space of the stereo measurement, the errors affecting all these values would be constant and easily modeled. However, switching to this model is not easy. First, it could mean a high computational load, if each particle of the environment model is projected into the image space, and second, one should be aware of the occlusions, which prevent some existing entities in the scene to be visible in the image space. The occlusion problem is further complicated by the dynamic nature of the environment, which could transform a visible object into an occluded one during the tracking process.

2) Transforming a world model and tracking method into a sensor fusion technique, by integrating multiple measurement sources in the measurement process. As an intermediate representation, either the dynamic occupancy grid or the dynamic elevation maps are suitable for this attempt.

A complex dynamic environment is better observed if multiple sensors, each with its own strengths and weaknesses, are used. Stereovision has a large point density, but weaker measurement accuracy for these points; laserscanners have an extremely accurate measurement precision, but a low point density (excepting the very high end laser sensors,

which cost in the range of tens of thousands of euro) and suffer from the variable timestamp of each scanning ray; radars only detect metal targets, but have the advantage of delivering the speed of the target as raw measurement, extracted using the Doppler effect, without the need of tracking. Besides using different sensors, one could also use different type of data extracted from the same sensor, using different processing algorithms (maybe even on different computing platforms). For example, from a stereo system one can compute the optical flow, which is another computationally demanding task, but which can add speed information to the tracked features of the scene, allowing the tracker to estimate the speeds of the dynamic entities of the environment faster.

Handling multiple sensors means facing a series of challenges. The errors of each sensor have to be modeled, and the world model has to be related to each of the sensors through customized projection routines. The sensors may not be synchronized with each other, meaning that they deliver data at different time instants. For accurate fusion, the time instants of the sensors must be known with respect to a common time base, and the time differences will have to be compensated using the prediction of the world model.

Due to the fact that the candidate is the manager of the PNII-PCCA project "Automatic Medium and High Earth Orbit Observation System Based on Stereovision", set to end in 2016, another short term research field requiring his attention is the stereovision-based space surveillance. While many problems have already been solved in the first years of the project, there are still issues that need to be addressed, so that the resulting system may become a robust infrastructure for space surveillance, able to open our path towards collaboration with the interested international institutions such as the European Space Agency.

The main challenges that are still open in the field of stereovision based space surveillance are:

1) Improving the quality of the range estimation, by an in depth analysis of the sources of uncertainty in the measurement process and devising solutions to remove these uncertainties.

The stereovision process is highly sensitive to a series of factors: accurate matching of the features from the left and right image, precise synchronization of the acquisition process, both at starting the image capture process and in terms of the length of the exposure time, and highly accurate knowledge of the camera parameters, both intrinsic and extrinsic. All these premises are challenged by the nature of the observation system: each station has to be triggered independently, and no common signal can be applied, the exposure time is of the order of seconds, which means that the target is not point-like and thus the exact matching of the corresponding points is difficult and relies on very precise estimation of the target trajectory and of the epipolar lines, the rotation matrices are continuously changing and require constant re-calibration. All these issues lead to uncertainties, which lead to errors in measurement, so a significant effort has to be made towards understanding and compensating them.

2) Designing a tracking algorithm based on estimating the state of the target, state which is in fact made out of the orbital parameters. A method for determining the orbit parameters from the stereo results, and a prediction system capable of propagating these parameters, have to be designed.

Any object orbiting the Earth has a trajectory described by a set of parameters that are named by the astronomical community Two Line Elements (TLE). Knowledge of the object's TLE allows for the prediction of the object's position with respect to any point on Earth. Conversely, the TLE elements can be refined from the measurements that can be associated to the orbiting object, and the timestamps of these measurements. A very important objective

of the research project is to estimate the TLE parameters of the observed targets, while using these parameters to predict the position of the targets in the image space. Basically, a tracking algorithm, with the target state described by the TLE parameters, will be designed and implemented.

For the long term, the candidate estimates that his research will be focused on perception systems for robotics, driving assistance applications and space surveillance, but also on generic computer vision topics. The candidate will be involved in his research group's projects, but he will also continuously look for new projects, proposing research topics of his own and participating in national and international grant competitions.

As a full time faculty member, the candidate will also be involved in activities related to teaching and advising. While continuing his lectures on the topics of Image Processing and Design with Microprocessors, and the laboratory activities related to these subjects, the candidate hopes to expand his teaching activity with a new and original lecture, related to generic methods of environment modeling, perception and understanding, which can be set up perhaps in the masters curriculum.

The candidate will continue to advise diploma and master theses, on subjects related to his research activities. However, if this habilitation will be successful, the candidate will also advise doctoral research, a new challenge implying increased responsibility. The candidate will use his experience to help the PhD students pursue meaningful and fruitful research topics, will work with the students for getting original and publishable results, and will assist them in the publication process. Also, it is the habilitation candidate's opinion that it is the task of the advisor to continuously look for sources of financing for the PhD students he advises, and to help them in the future academic or professional career they choose.

(b-iii) References

[1] A. Elfes, "A Sonar-Based Mapping and Navigation System", in proc of *IEEE International Conference on Robotics and Automation*, April 1986, pp. 1151-1156.

[2] A. Elfes, "Using Occupancy Grids for Mobile Robot Perception and Navigation", *Computer*, vol. 22, No. 6, June 1989, pp. 46-57.

[3] C. Coue, C.Pradalier, C.Laugier, T.Fraichard, P.Bessiere, "Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application", *The International Journal of Robotics Research*, Vol 25, No 1, 2006, pp. 19-30.

[4] C. Chen, C. Tay, K. Mekhnacha, C. Laugier, "Dynamic environment modeling with gridmap: a multiple-object tracking application", in proc of *International Conference on Automation, Robotics and Computer Vision (ICARCV) 2006*, pp. 1-6.

[5] T. Weiss, B. Schiele, K. Dietmayer, "Robust Driving Path Detection in Urban and Highway Scenarios Using a Laser Scanner and Online Occupancy Grids", in proc of *IEEE Intelligent Vehicles Symposium 2007*, pp. 184-189.

[6] S. Pietzch, T. D. Vu, J. Burtlet, O. Aycard, T. Hackbarth, N. Appenrodt, J. Dickmann, B. Radig, "Results of a Precrash Application based on Laser Scanner and Short Range Radars", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 10, No. 4, 2009, pp. 584-593.

[7] T. Gindele, S. Brechtel, J. Schroeder, R. Dillmann, "Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map Knowledge", in proc of IEEE Intelligent Vehicles Symposium 2009, pp. 669 – 676.

[8] S. Thrun, "Learning Occupancy Grids With Forward Sensor Models", *Autonomous Robots*, Vol. 15, No 2, 2003, pp. 111-127.

[9] H. Badino, U. Franke, R. Mester, "Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming", *Workshop on Dynamical Vision, ICCV,* 2007, pp. 1-12.

[10] F. Oniga, S. Nedevschi, "Processing Dense Stereo Data Using Elevation Maps: Road Surface, Traffic Isle, and Obstacle Detection", *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 3, March 2010, pp. 1172-1182.

[11] A. Rosenfeld, J. L. Pfaltz, "Sequential Operations in Digital Picture Processing", *Journal of the Association for Computing Machinery*, Vol. 13, No. 4, October 1966, pp. 471-494.

[12] C. Pantilie, S. Nedevschi, "Real-time Obstacle Detection in Complex Scenarios Using Dense Stereo Vision and Optical Flow", *IEEE Conference on Intelligent Transportation Systems (IEEE-ITSC)*, 2010, pp. 439-444.

[13] D. F. Huber and M. Hebert, "A new approach to 3-D terrain mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '99)* 1999, pp. 1121-1127 vol.2.

[14] M. Vergauwen, M. Pollefeys, and L. Van Gool, "A stereo-vision system for support of planetary surface exploration," *Machine Vision Applications*, vol. 14, pp. 5-14, 2003.

[15] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, and R. Chatila, "Autonomous Rover Navigation on Unknown Terrains: Functions and Integration," *International Journal of Robotics Research*, vol. 21, pp. 917-942, October 1, 2002.

[16] M. Harville, "Stereo person tracking with adaptive plan-view templates of height and occupancy statistics," *Image and Vision Computing*, vol. 22, pp. 127-142, 2004.

[17] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application," *International Journal of Robotics Research*, vol. 25, pp. 19-30, January 1, 2006.

[18] P. Pfaff, R. Triebel, and W. Burgard, "An Efficient Extension to Elevation Maps for Outdoor Terrain Mapping and Loop Closing," *International Journal of Robotics Research*, vol. 26, pp. 217-230, February 1, 2007.

[19] R. Triebel, P. Pfaff, and W. Burgard, "Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing," in IEEE*/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)* 2006, pp. 2276-2282.

[20] I. Dryanovski, W. Morris, and X. Jizhong, "Multi-volume occupancy grids: An efficient probabilistic 3D mapping model for micro aerial vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)*, 2010, pp. 1553-1559.

[21] B. Gassmann, L. Frommberger, R. Dillmann, and K. Berns, "Real-time 3D map building for local navigation of a walking robot in unstructured terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, 2003, pp. 2185-2190, vol.3.

[22] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, pp. 174-188, 2002.
[23] M. Isard and A. Blake, "CONDENSATION—Conditional Density Propagation for Visual Tracking," *International Journal of Computer Vision*, vol. 29, pp. 5-28, 1998.
[24] C. D. Pantilie and S. Nedevschi, "SORT-SGM: Subpixel Optimized Real-Time Semiglobal Matching for Intelligent Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 61, pp. 1032-1042, 2012.
[25] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*, Mit Press, 1993.
[26] WWWConsortium, "VRML Virtual Reality Modeling Language", Available online: http://www.w3.org/MarkUp/VRML/
[27] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2012)*, 2012, pp. 3354-3361.
[28] A. Geiger, "The KITTI Vision Benchmark Suite - Raw Data", Available online: http://www.cvlibs.net/datasets/kitti/raw_data.php
[29] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision*, vol. 47, pp. 7-42, 2002.
[30] R. Danescu. (2013). Dynamic Elevation Map Tracking Demo Video. Available online: http://users.utcluj.ro/~rdanescu/dynamap.avi
[31] T. N. Nguyen, B. Michaelis, A. Al-Hamadi, M. Tornow, and M. M. Meinecke, "Stereo-Camera-Based Urban Environment Perception Using Occupancy Grid and Object Tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 154-165, March 2012.
[32] R. Danescu, F. Oniga, and S. Nedevschi, "Modeling and Tracking the Driving Environment With a Particle-Based Occupancy Grid," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, pp. 1331-1342, 2011.
[33] U. Franke, C. Rabe, H. Badino, and S. K. Gehrig, "6D-Vision: Fusion of Stereo and Motion for Robust Environment Perception," in *DAGM-Symposium* vol. 3663, ed: Springer, 2005, pp. 216-223.
[34] D. Pfeiffer and U. Franke, "Efficient representation of traffic scenes by means of dynamic stixels," in *IEEE Intelligent Vehicles Symposium (IV 2010)*, 2010, pp. 217-224.
[35] T. Schildknecht, "Optical surveys for space debris", *The Astronomy and Astrophysics Review*, Vol. 14, 2007, pp. 41-111.
[36] H. Klinkrad, *Space Debris. Models and Risk Analysis,* Springer & Praxis Publishing Inc.: Chichester, UK, 2006.
[37] IADC, "IADC Space Debris Mitigation Guidelines", Available online: http://www.iadc-online.org/Documents/IADC-2002-01,%20IADC%20Space%20Debris%20Guidelines,%20Revision%201.pdf .
[38] H. Riebeek, "Catalog of earth satellite orbits", Available online: http://earthobservatory.nasa.gov/Features/OrbitsCatalog/ .
[39] T. Donath, T. Schildknecht, P. Brousse, J. Laycock, T. Michal, P. Ameline, L. Leushacke, "Proposal for a European Space Surveillance System", in *Proceedings of the 4th European Conference on Space Debris*, 2005, Vol. 587, pp. 31-38.
[40] NASA Earth Observatory, "Three Classes of Orbit", Available online: http://earthobservatory.nasa.gov/Features/OrbitsCatalog/page2.php.
[41] V. Agapov, I. Molotov, Z. Khutorovsky, "Analysis of Situation in GEO Protected Region", in *Advanced Maui Optical and Space Surveillance Technologies Conference*, Maui, Hawaii, 2009, pp. 180-189.
[42] R. Sridharan, A. F. Pensa, "U.S. Space Surveillance Network capabilities", in *Proc. SPIE*, San Diego, CA, 1998, pp. 88-100.
[43] H. Klinkrad, R. Tremayne-Smith, F. Alby, D. Alwes, "Europe's eyes on the skies: The proposal for a European Space Surveillance System", *ESA Bulletin*, European Space Agency, 2008, pp. 42-48.
[44] FAS Space Policy Project, "GEODSS: Ground Based-Electro-Optical Deep Space Surveillance", Available online: http://www.fas.org/spp/military/program/track/geodss.htm .
[45] DARPA, "Space Surveillance Telescope (SST)", Available online: http://www.darpa.mil/Our_Work/TTO/Programs/Space_Surveillance_Telescope_%28SST%29.aspx .
[46] D. F. Woods, R. Y. Shah, J. A. Johnson, A. Szabo, E. C. Pearce, R. L. Lambour, W. J. Faccenda, "Space Surveillance Telescope: focus and alignment of a three mirror telescope", *Optical Engineering*, Vol. 52, No. 5, 2013, id. 053604 .
[47] ***, "ESA Space Debris Telescope", Available online: *http://en.wikipedia.org/wiki/ESA_Space_Debris_Telescope*.

[48] T. Flohrer, J. Peltonen, A. Kramer, T. Eronen, J. Kuusela, E. Riihonen, T. Schildknecht, E. Stoveken, E. Valtonen, F. Wokke, W. Flury, "Space-based optical observations of space debris", in *Proceedings of the 4th European Conference on Space Debris*, 2005, Vol. 587, pp. 165-170.
[49] ***,"Télescope ROSACE du CNES", Available online: http://www.obs-hp.fr/guide/autres/rosace.html.
[50] H. Klinkrad, "Monitoring Space-Efforts Made by European Countries", in *International Colloquium on Europe and Space Debris*, 2002.
[51] SpaceInsight, "An introduction to Space Insight's Starbrook space surveillance systems", Available online: http://www.spaceinsight.co.uk/downloads/SpaceInsight_Starbrook.pdf.
[52] T. Yanagisawa, H. Kurosaki, H. Banno, Y. Kitazawa, M. Uetsuhara, T. Hanada, "Comparison Between Four Detection Algorithms For GEO Objects", in *Advanced Maui Optical and Space Surveillance Technologies Conference*, Maui, Hawaii, USA, 2012; pp. 91-97.
[53] M. P. Lévesque, S. Buteau, "Image processing technique for automatic detection of satellite streaks", *DRDC Valcartier TR 2005-386*, DRDC Valcartier, Canada, 2007.
[54] M. P. Lévesque, "Automatic Reacquisition of Satellite Positions by Detecting their Expected Streaks in Astronomical Images", in *Advanced Maui Optical and Space Surveillance Technologies Conference*, Maui, Hawaii, 2009; pp. E81.
[55] M. P. Lévesque, M. Lelièvre, "Improving satellite-streak detection by the use of false alarm rejection algorithms", *DRDC Valcartier TR 2006-587*, DRDC Valcartier, Canada, 2008.
[56] E. Stoveken, T. Schildknecht, "Algorithms for the optical detection of space debris objects", in *Proceedings of the 4th European Conference on Space Debris*, 2005; Vol. 587, pp. 637-640.
[57] A. Milani, D. Farnocchia, L. Dimare, A. Rossi, F. Bernardi, "Innovative observing strategy and orbit determination for Low Earth Orbit space debris", *Planetary and Space Science*, Vol. 62, 2012, pp. 10-22.
[58] Sigma, "SIGMA 20mm F1.8 EX DG ASP RF", Available online: http://www.sigmaphoto.com/shop/20mm-f18-ex-dg-asp-rf-sigma .
[59] Canon, "Canon EOS 50D-Instruction Manual", Available online: http://gdlp01.c-wss.com/gds/1/0300001591/02/eos50d-h2-en.pdf.
[60] Celestron, "Celestron Advanced Series CG-5 & CG-5 GT INSTRUCTION MANUAL", Available online: http://www.telescopes.com/images/pdf/CELE054.pdf .
[61] W. M. Smart, *Textbook of Spherical Astronomy*, 6th edition, Cambridge University Press, Cambridge, UK, 1986.
[62] R. Berry, J. Burnell, *The Handbook of Astronomical Image Processing with AIP4Win Software*, 2nd edition, Willmann-Bell, Inc., Richmond, Virginia, 2005.
[63] H. Raab, "Astrometrica", Available online: http://www.astrometrica.at .
[64] USNO, "USNO Interactive Catalog and Image Search", Available online: http://www.usno.navy.mil/USNO/astrometry/optical-IR-prod/icas/fchpix.
[65] R. M. Haralick, L. G. Shapiro, *Computer and Robot Vision*, Addison-Wesley Longman Publishing Co. Inc., Boston, Massachusetts, USA, 1992, p. 630.
[66] R. R. Bouckaert, E. Frank, M. A. Hall, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, "WEKA-Experiences with a Java Open-Source Project", *Journal of Machine Learning Research*, vol. 11, 2010, pp. 2533-2541.
[67] E. Trucco, A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall Publishing, Upper Saddle River, NJ, USA, 1998, p. 343.
[68] R. C. Gonzalez, R. E. Woods, *Digital Image Processing*, Addison-Wesley Longman Publishing Co. Inc., Boston, Massachusetts, USA, 2001, p. 793.
[69] Bisque, "TheSkyX Serious Astronomer Edition", Available online: http://www.bisque.com/sc/pages/TheSkyX-Professional-Edition.aspx .