

4-D Tensor Voting Motion Segmentation for Obstacle Detection in Autonomous Guided Vehicle

Yann Dumortier, Isabelle L. Herlin, André Ducrot

▶ **To cite this version:**

Yann Dumortier, Isabelle L. Herlin, André Ducrot. 4-D Tensor Voting Motion Segmentation for Obstacle Detection in Autonomous Guided Vehicle. Intelligent Vehicles Symposium, IEEE, Jun 2008, Eindhoven, Netherlands. pp.379-384, 10.1109/IVS.2008.4621203 . inria-00292702

HAL Id: inria-00292702
https://inria.hal.science/inria-00292702

Submitted on 2 Jul 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

4-D Tensor Voting Motion Segmentation for Obstacle Detection in Autonomous Guided Vehicle

Yann Dumortier, Isabelle Herlin, André Ducrot
IMARA project team, INRIA
B.P 105 78153 Le Chesnay Cedex France
Email: {firstname}.{lastname}@inria.fr

***Abstract*— Creating an obstacle detection system is an important challenge to improve safety for road vehicles. A way to meet the industrial cost requirements is to gather a monocular vision sensor. This paper tackles this problem and defines an highly parallelisable image motion segmentation method for taking into account the current evolution of multi processor computer technology. A complete and modular solution is proposed, based on the Tensor Voting framework extended to the 4D space (x, y, dx, dy), where surfaces describe homogeneous moving areas in the image plan. Watershed segmentation is applied on the result to obtain closed boundaries. Cells are then clustered and labeled with respect to planar parallax rigidity constraints. A visual odometry method, based on texture learning and tracking, is used to estimate residual parallax displacement.**

I. Introduction

A. Context

This contribution addresses moving obstacle detection for intelligent transport systems with a monocular video sensor. Automotive manufacturers integrate more and more assistance systems in their new cars, like Automatic Cruise Control, Lane Crossing Detection or obstacle location. Perception issues for Autonomous Guided Vehicle are the free space estimation and the moving obstacles detection. Depending on used sensor, such systems will have different price, accuracy and speed characteristics. On one hand, there are active sensors with range-finders. Among them, lasers provide high spatial resolutions data at high scanning speeds, but are too expensive, whereas cheaper sensors like sonars, are limited to parking applications owing to their range. Furthermore, the presence of several active sensors in the same environment may disrupts the measure acquisition. On the other hand, video sensors appear well adapted for automotive applications due to the rich 2-D information contained in a single image and the 3-D information inferred by two images. Up to date, research has mostly focused on the detection of moving objects either from fix cameras or from mobile ones in static and highly constrained dynamic scenes [1][2].

The visible motion in an image sequence is due to the 3-D camera displacement (the ego-motion) and some objects'motion. In order to detect moving obstacles, various approaches compensate the ego-motion, then segment the residual motion [4]. But numerical estimations by using motion models may provide some noise. Thus, other methods prefer first to segment the global image motion and then apply rigid geometric constraints to identify moving objects [14]. Although monocular systems based on optical flow algorithms have been studied [3], the accuracy of flow fields and its impact is never discussed despite noisy or sparse estimation. Furthermore, usual optical flow filters [5] are not efficient in the context of obstacle detection, because motion models induces a smoothing of mobile obstacle displacements. Nevertheless, [6] provided an original approach, using the Tensor Voting framework [7], to evaluate the accuracy of the velocity values according to their neighborhood. Based on this work, we developed an obstacle detection solution using different velocity fields to improve the robustness of our method. Unlike most of the motion-based image segmentation methods, this work addresses the difficult case of radial image motion.

Section II-A presents the Tensor Voting framework and its use in a specific 4-D space to estimate input velocity data and perform motion segmentation. Section II-B deals with a watershed approach to provide closed boundaries. Then, III-A discusses about the planar parallax rigidity constraint to differentiate moving objects from static ones. III-B addresses a solution to estimate the camera motion in order to recover parallax displacements. In section III-C, the rigidity constraint is applied on the segmentation supplied by the watershed. At last, conclusions and directions for future works are given in section IV.

II. 2D Motion segmentation

A. Tensor Voting

The Tensor Voting is a unified framework which has been developed by G. Medioni since the 90's. This formalism, based on tensor calculus for data representation, and non-linear voting for communication, allows identification of geometrical structures from sparse and noisy N-D data. The method is non-iterative and can be processed in $O(1)$ by parallel implementation. The N-dimensional approach is an extension of the two-dimensional one.

1) Overview: The Tensor Voting attempt to recover, from a set of sparse and isotropic data, the geometrical structure such as curve elements and points in a 2-D space. The idea is to use the neighborhood layout of each considered input site to constrained its identification. The

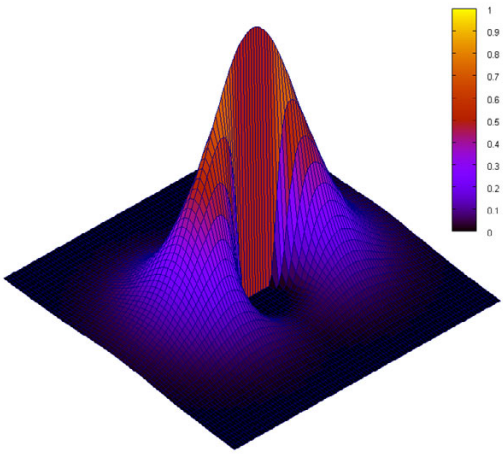


Fig. 1. Decay function of a stick tensor.

second order representation, in the form of a second order, symmetric, non-negative definite tensor, allows to encode the structural information of the input data, and to propagate this information according to a simple accumulation process.

A 2D tensor T is equivalent to a 2x2 matrix, and describes an ellipse. It can be decomposed with the following equation:

$$\begin{aligned} T &= \lambda_1 \hat{e}_1 \hat{e}_1^T + \lambda_2 \hat{e}_2 \hat{e}_2^T \\ &= (\lambda_1 - \lambda_2)\, \hat{e}_1 \hat{e}_1^T + \lambda_2 \left(\hat{e}_1 \hat{e}_1^T + \hat{e}_2 \hat{e}_2^T\right), \end{aligned} \quad (1)$$

where λ_i are the eigenvalues in decreasing order ($\lambda_i \geq \lambda_{i+1}$) and $\hat{e}_i$ the corresponding eigenvectors. The two terms correspond to elementary 2D tensors: the first one is a degenerate elongated tensor and the second is an isotropic one. They are respectively named *stick tensor* and *ball tensor*. Input data are encoded according to their isotropy by such unitary tensors. Points of a curve are coded by *stick tensors* belonging to the normal of the considered curve whereas points without structural information are described by *ball tensors*.

The communication step involves a voter and a receiver. As represented in Fig. 2, the voter infers on the receiver according to the distance l between them, and, for the stick tensor, with respect to its relative orientation θ with the receiver. Each input site communicates its structural information, coded with a tensor, to its neighborhood through a predefined voting field. All tensor voting fields derive from the stick one whose scope is defined with a decay function (Fig. 1) such as:

$$DF(\theta,l) = e^{\left(\frac{s(\theta,l)^2 + ck(\theta,l)^2}{\sigma^2}\right)}, \quad (2)$$

$$s(\theta,l) = \frac{\theta l}{sin(\theta)},\ k(\theta,l) = \frac{2sin(\theta)}{l},$$

with $s(\theta,l)$ the length of the arc between the voter and the receiver and $k(\theta,l)$ its curvature. σ is the only free

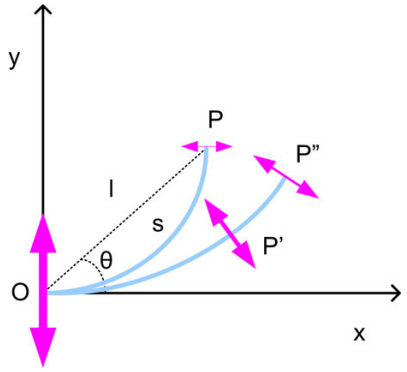


Fig. 2. Voting from a stick tensor located at the origin and aligned with the y-axis.

parameter of Tensor Voting, defining the scale of voting. c is a predefined constant, optimized not to give a round corner an advantage over a right angle.

Thus, the vote of a stick tensor, located at the origin O and aligned with the y-axis, to any site P (Fig. 2), is expressed by:

$$S(\theta) = DF(\theta) \begin{bmatrix} -sin(2\theta) \\ cos(\theta) \end{bmatrix} \begin{bmatrix} -sin(2\theta) & cos(\theta) \end{bmatrix}. \quad (3)$$

The 2-D ball voting field is obtained by integrating over θ the vote of the 2-D stick tensor. Only the distance with the receiver infers on the casted information. At the end of the communication step, resulting tensors are specialized by the sum of their neighbors'vote. The geometrical structures encoded are obtained with the decomposition (1).

N-dimension *Tensor Voting* is derived from the 2-D framework: each additional dimension brings a new structural element (the surface in 3-D, the volume in 4-D, ...) to be coded. Elements of a N-D space are therefore encoded by tensors of the same dimension which can be decomposed in N-1 weighted elementary tensors. The N-D ball voting field is computed by integrating the 2-D stick voting field over all degrees of freedom.

2) 4D Tensor Voting: Our work is based on [6] which presents an original approach for motion grouping by Tensor Voting. Let us consider the 4-D space (x,y,d_x,d_y), where $(d_x,d_y)^T$ is the velocity vector of the pixel located at (x,y). In such space, image areas with coherent motion are described as surfaces, and can be identified by Tensor Voting. As in 2D, encoding unoriented input data with *ball tensors* and applying a voting step, specialize all tensors according to the layout of their neighborhood.

The decomposition of a generic tensor in 4-D is given by:

$$\begin{aligned} T &= (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) \\ &\quad + (\lambda_3 - \lambda_4)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + \hat{e}_3\hat{e}_3^T) \\ &\quad + \lambda_4(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + \hat{e}_3\hat{e}_3^T + \hat{e}_4\hat{e}_4^T), \end{aligned} \quad (4)$$

where $(\lambda_2 - \lambda_3)$ is the coefficient attributed to the elementary *S-Plate tensor* $\left(\hat{e}_2\hat{e}_2^T + \hat{e}_1\hat{e}_1^T\right)$ and $(\lambda_3 - \lambda_4)$ the coefficient

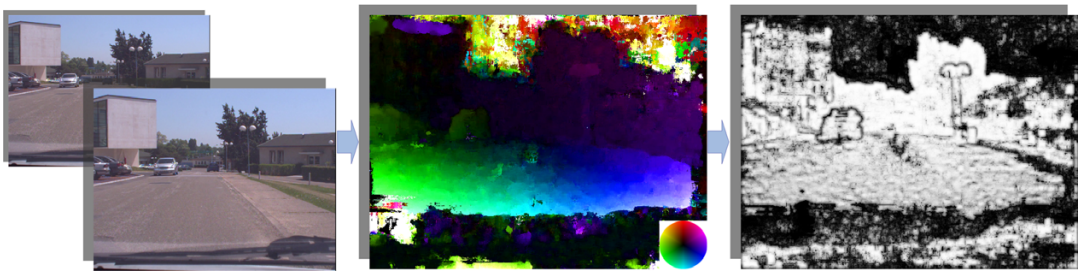


Fig. 3. From left to right, the current image, the correspondent velocity field image (with the color map on the bottom-right corner) and the surface saliency map from 4-D Tensor Voting.

to the elementary *C-Plate tensor* $\left(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + \hat{e}_3\hat{e}_3^T\right)$, which respectively describe surface and curve in the space (x, y, d_x, d_y).

3) Motion segmentation: From two consecutive frames, we first process an optical flow algorithm to obtain 4-D input data (x, y, d_x, d_y). Each site is coded with a *ball tensor* and accumulates the votes cast at its location. After decomposition, the analysis of eigenvalues allows to estimate the confidence with which input points belong to a surface, so to an homogeneous motion area. Fig. 3 displays a surface saliency image given by Tensor Voting process.

Two points which are close in the image plan, but belonging to different moving areas, become distant in the space (x, y, d_x, d_y) due to their different velocity. Therefore tensors from different moving areas have no impact on each other. At the end of the communication process, specialized tensors are decomposed to weighted elementary tensors according to (4). Inverse of the saliency of *S-Plate tensors* and the saliency of *C-Plate tensors*, coding respectively surface and curve information, are both used to draw the saliency map (Fig. 3) and to segment the image motion.

4) Accuracy motion estimation: Obstacle detection methods based on image motion are highly dependent on the implemented optical flow method, which is usually tuned for one specific context. 4-D Tensor Voting can assess the relevance of a point $P(x, y, d_x, d_y)$ according to its belonging to the surface formed by its neighborhood. Therefore, among several velocity values (d_x, d_y) for one input site (x, y), 4-D Tensor Voting allows to estimate which one is closest to the true motion. In practice, we have selected a pyramidal Lucas and Kanadé algorithm providing three velocity values per site. The voting step determines the best result as represented in Fig. 4. Nevertheless, the Tensor Voting does not supply a closed contour image but a confidence value, for each pixel, to its belonging to an homogeneous motion area.

B. Boundary closing

Considering the gray level value of a pixel as an height, an image can be seen as a topographical relief. The watershed segmentation provides closed boundaries by extracting the basin junction lines after flooding the relief from seeds. Usually, seeds are located at local minimas, so the method may induce an over-segmentation (Fig. 5). A filtering step has to be performed to avoid it.

Developed in the mathematical morphology framework, attribute opening (closing) is an efficient solution for area opening (closing). Whereas the classic operation is constrained by the shape of a structural element, attributes openings (closings) allow to filter a region according to some topographical criteria such as the height, the area or the volume. The process can use a tree-based image representation, where each node in the tree stands for a connected component and where the edges are weighted with the value of the specific attribute associated to the connected component (Fig. 6).

In order to use efficiently the information provided by the Tensor Voting, the watershed segmentation is applied on the inverse of the surface saliency map. Thus, seeds are located at the most confident points in each motion areas. The occlusions induced by the camera motion and the lake of texture, provide some error in the optical flow estimation. In our application, the high frequency noise is first removed by an attribute opening using a criteria of small area. Then, seeds belonging to the same moving area are merged using an attribute of type height. Finally, a wathershed segmentation is performed on the filtered saliency map to obtain closed contours.

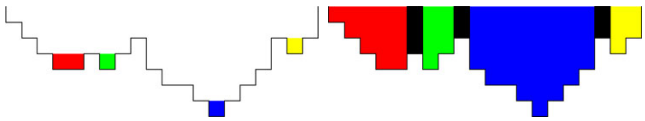


Fig. 5. Watershed segmentation (right) from colored seeds (left). The watersheds are in black.

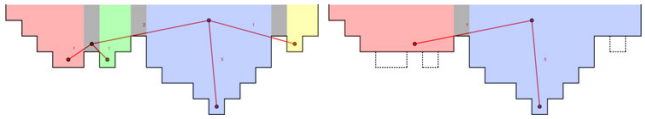


Fig. 6. Filtering step with attribute openings (height 1). Original 1D signal (left) and result (right).

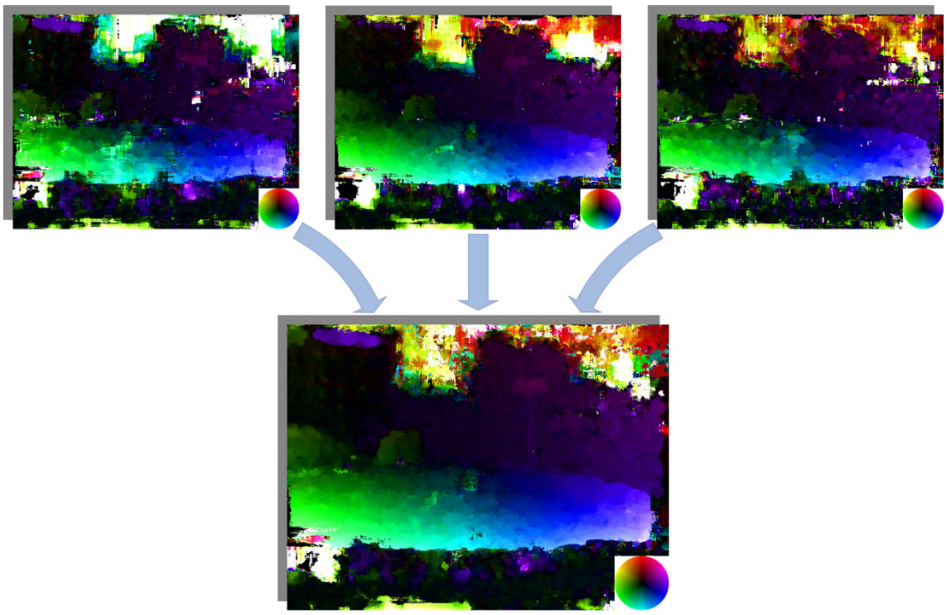


Fig. 4. Accurate motion estimation obtained (bottom) from Tensor Voting on three first estimations (top).

III. 2D Motion Grouping and Classification

After the filtering step and the watershed segmentation has been applied, a slight over-segmentation persists, so a clustering and labeling step is necessary. Objects which are static in the 3-D environment are however moving in the 2-D image plane due to the ego-motion, this prevents to differentiate correctly moving and static obstacles. [14] presents some parallax-based constraints that are used in the next subsection. It deals with rigidity constraints on pairs of points to 3-D scene analysis in the presence of camera motion.

A. Planar parallax constraint

Let us consider the coordinates $\vec{P}=(X,Y,Z)^T$ and $\vec{P'}=(X',Y',Z')^T$ of a point P of the scene, expressed in two different camera systems. $\vec{p}$ and $\vec{p'}$ are their projections in the corresponding image planes, $\mathscr{I}_1$ and $\mathscr{I}_2$. Let Π be an arbitrary static plane, d_π and d'_π its distances to the camera centers, and $\vec{p_w}$ the projection of $\vec{p'}$ in $\mathscr{I}_1$ according to the homography induced by Π. To derive the parallax constraint on each point P, we assume the decomposition of the image motion $\vec{u}$ into an homography $\vec{u_\pi}$ and a residual parallax motion $\vec{\mu}$ [15] such as:

$$\begin{aligned}\vec{u}&=\vec{u_\pi}+\vec{\mu}\ ,\\ \vec{\mu}&=-\gamma\frac{T_Z}{d'_\pi}\left(\vec{p_w}-\vec{e}\right)\ ,\end{aligned} \tag{5}$$

where $\vec{e}$ is the epipole in the first image, T_Z the z-translation from the first camera to the second one, and γ the 3-D projective structure of $\vec{P}$ with respect to Π ($\gamma=\frac{H}{Z}$, with H the distance of $\vec{P}$ from the plane Π). Note that the parallax displacements are the relative image motions of objects induced by the camera translations (rotations do not induce any parallax displacement).

Let us consider two points, P_1 and P_2, belonging to the same solid object, and their image projection, $\vec{p_1}$ and $\vec{p_2}$. The relation between them can be expressed by:

$$\vec{\mu_1}\,\gamma_2-\vec{\mu_2}\,\gamma_1=\gamma_1\gamma_2\frac{T_Z}{d'_\pi}\left(\vec{p_{w2}}-\vec{p_{w1}}\right) \tag{6}$$

removing the epipole from the formula. Since $\gamma_1\gamma_2\frac{T_Z}{d'_\pi}$ is a scalar, $\left(\vec{\mu_1}\,\gamma_2-\vec{\mu_2}\,\gamma_1\right)$ and $\left(\vec{p_{w2}}-\vec{p_{w1}}\right)$ are collinear. Therefore,

$$\left(\vec{\mu_1}\,\gamma_2-\vec{\mu_2}\,\gamma_1\right)^T\left(\Delta\,\vec{p_w}\right)_\perp=0, \tag{7}$$

where $\vec{v}_\perp$ is a vector perpendicular to $\vec{v}$ and $\Delta\,\vec{p_w}=\left(\vec{p_{w2}}-\vec{p_{w1}}\right)$. The temporal invariant relationship γ_2/γ_1 can easily be derived from (7),

$$\frac{\gamma_2}{\gamma_1}=\frac{\vec{\mu_2}^T\left(\Delta\,\vec{p_w}\right)_\perp}{\vec{\mu_1}^T\left(\Delta\,\vec{p_w}\right)_\perp}. \tag{8}$$

Thus, (8) provides a parallax-based rigidity constraint, allowing to verify over three frames if two points belong to the same solid object.

B. Visual odometry

According to (5), the planar parallax constraint requires to compute the parallax displacement of each image point. Thus, we propose a visual odometry method, inspired by [11], looking for the homography induced by the projection of the road plane over two frames.

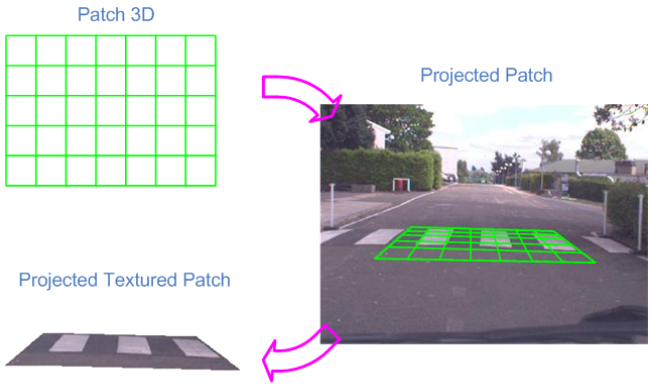


Fig. 7. Texture learning stage

Assuming known position of the camera with the road plane, a 3-D patch is projected in the image. We use the pinhole camera model to describe the relationship between $\vec{P}\ (X,Y,Z,1)^T$, a point in the camera referential system , and $\vec{p}\ (u,v,1)^T$ its projection in the image plane such as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{Z} \underbrace{\begin{pmatrix} fk_u & 0 & x_0 & 0 \\ 0 & fk_v & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\mathsf{K}} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} , \qquad (9)$$

with f the focal length, $(u_0\ v_0)^T$ the image coordinates of the intersection of the optical axis of the lens with the image plane, k_u and k_v the scale factors. These parameters are obtained by off-line calibration. The texture is directly mapped on the patch projection using current frame pixels.

Then, a second stage looks for the 3-D transformation which matches, in the next frame, the patch mapped in the current one. Since the road is assumed to be locally plane, the transformation is a warp between two planar patches describing the camera 3D motion and projection. Such a perspective projection, restricted to coplanar points, can be expressed as an homography which is a projective transformation between two planes [10].

$$\mathsf{H} = \mathsf{K}\,(\mathsf{R} + \frac{\vec{t}\,\vec{n}^T}{d})\,\mathsf{K}^{-1} . \qquad (10)$$

where $\mathscr{T}_{(\mathsf{R},\mathsf{t})}$ describes the camera motion, with R the 3x3 rotation matrix and $\vec{t}$ the translation vector. d is the distance from the camera to the road plane and $\vec{n}$ a normal of this plane. At each iteration, a correlation is performed between the warped patch and the current frame patch, using a cost function applied on the red, green and blue image components:

$$f_{cost} = \frac{1}{3N} \sum_{i=1}^{N} \sum_{j=1}^{3} (|p_{patch(ij)} - p_{image(ij)}|) , \qquad (11)$$

with N denotes the number of pixel, $p_{patch(i)}$ the i^{th} pixel value of the warped patch, $p_{image(i)}$ its correspondent located at the same coordinates in the next image and j the color component. A simple gradient descent method is applied to minimize the f_{cost} cost function and find the camera displacement.

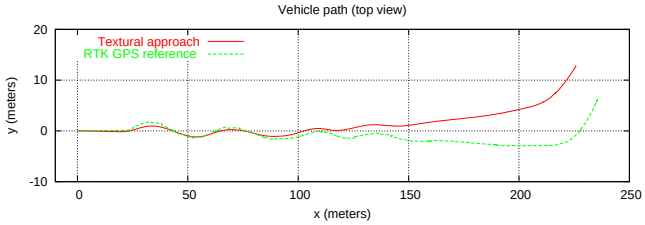


Fig. 8. Vehicle path estimation from a RTK GPS in green (reference) and from the presented visual odometry method in red.

C. Grouping and labeling

Acording to (10) and the camera motion $\mathscr{T}_{(\mathsf{R},\mathsf{t})}$ provided by the visual odometry, the homography induced by any virtual static planar surface can be computed. As some plan projections are not defined in the whole image, their homography has no meaning for all pixels for which the residual parallax displacements cannot compute using (5). Therefore, we only consider the plan perpendicular to the focal axis and located at infinity, whose homography is

$$\mathsf{H} = \mathsf{K}\,\mathsf{R}\,\mathsf{K}^{-1} .$$

The computation for all pixels would be time consuming, so the constraint is checked with the few ones selected with respect to their saliency, provided by the 4-D Tensor Voting. In such way, only relevant velocities get involved in this step, and the overall approach remains robust. Then, cells are clustering according to a threshold applied on the rigidity constraint result (8). By assuming static the lower part of the picture, we can continuously discriminate moving objects from static ones.

IV. Conclusion

In this paper, we presented an image-motion-based approach for moving obstacle detection from one embedded



Fig. 10. Estimation of the parallax-based rigidity constraint with respect to the point circled in red. In black, the points consistent with the reference one according to the constraint given by (8).



Fig. 9. Correct patch tracking sequence (we can notice that very few texture information enable a good tracking)

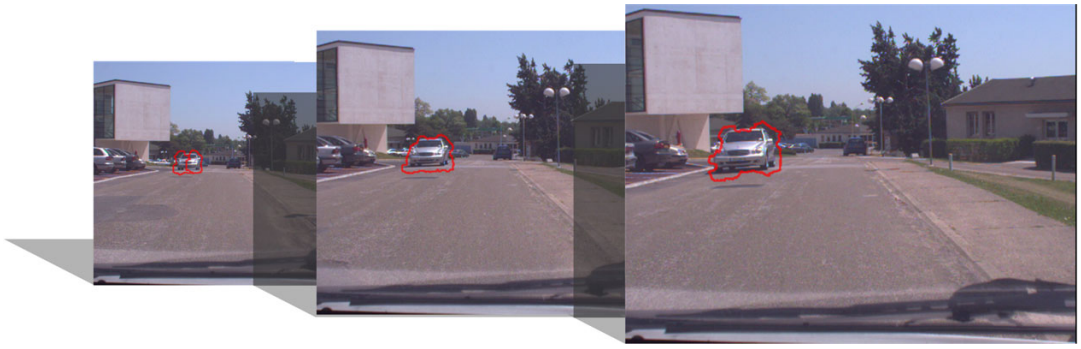


Fig. 11. Obstacle detection though three successive frames in the case of a radial motion.

monocular vision sensor. The 4-D Tensor Voting allows to appraise many set of input data together, to select the best velocity value for each image coordinate. Furthermore, a single parameter enable to tune the whole approach: the scope which is the only free parameter of the Tensor Voting. Its value is directly related to the topography magnitude of the resulting surface saliency map. Thus, the watershed associated to the attribute filtering, is particularly well-adapted to perform the closing stage. The visual odometry, used for clustering cells ensuing, has been evaluate with a centimetric RTK GPS, as illustrated fig. 8.

The method offers a robust solution to the problem of dynamic scene perception for autonomous guided vehicles. It has been successfully assessed in different situations including both translational and radial motions. In this last case, obstacles have been detected up to 60 meters with an image resolution of 640x480. As the *Tensor Voting* process is highly paralellisable and can be executed with a complexity of $O(1)$ in parallel architectures, we are currently working on integrating all the process on graphic chipset to a frame rate execution. Acquisition frequency depends on relative autonomous vehicle and obstacle speeds, our goal is to provide a system working at 15 Hz.

Acknowledgment

The authors wish to acknowledge Mr. G. Medioni (professor at the Institute for Robotics and Intelligence Systems, University of Southern California) for his contribution to this work through private discussions.

References

[1] M. Irani and P. Anandan, *A Unified Approach to Moving Object Detection in 2D and 3D Scene* PAMI, vol. 20, No. 6, pp. 577-589, 1998

[2] A. Talukder, S. Goldberg, L. Matthies and A. Ansar, *Real-time detection of moving objects in a dynamic scene from moving robots vehicles* IROS, 2003

[3] J.L. Barron, D.J Fleet and S.S. Beauchemin, *Performance of optical Flow Techniques* IJCV, vol. 12, n. 1, pp. 43-77, 1994

[4] B. Jung and G. Sukhatme, *Detecting Moving Object using a Single Camera on a Mobile Robot in an Outdoor Environment* Conf. on Intelligent Autonomous Systems

[5] D. Calow, N. Krger, F. Wrgtter and M. Lappe, *Space variant filtering of optical flow for robust three dimensional motion estimation* ICSC, 2004

[6] M. Nicolescu and G. Médioni, *Perceptual Grouping from Motion Cues Using Tensor Voting in 4-D* Morgan & Claypool publishers, 2006

[7] P. Mordohai and G. Médioni, *Tensor Voting: A Perceptual Organization Approach to Computer Vision and Machine Learning* ECCV 2002, pp. 423-437, 2002

[8] L. Najman and M. Couprie, *Watershed algorithms and contrast preservation* Proceedings DGCI 2003

[9] M. Couprie and G. Bertrand, *Topological Grayscale Watershed Transformation*, SPIE Vision Geometry V Proceedings, Vol. 3168, pp. 136-146, 1997

[10] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision* Cambridge University Press, 2000

[11] P. Gérard and A. Gagaglowicz, *Three Dimensional Model-based Tracking Using Texture Learning and Matching* Pattern Recognition Letters, 2000

[12] Y. Dumortier, M. Kais and R. Benenson, *Real-time Vehicle Motion Estimation Using Texture Learning and Monocular Vision* ICCVG, 2006

[13] C. Yuan, G. Médioni, J. Kang and I. Cohen, *Detecting Motion Regions in Presence of Strong Parallax from a Moving Camera by Multi-view Geometric Constraints* IEEE Trans Pattern Anal Mach Intell, Vol. 29, n. 9, pp. 1627-41, 2007

[14] M. Irani and P. Anandan, *Parallax Geometry of Pairs of Points for 3D Scene Analysis* European Conf. on Computer Vision, Vol. 1, pp. 17-30, 1996

[15] R. Kumar, P. Anandan and K. Hanna, *Shape recovery from multiple views: a parallax based approach*hskip 1em plus 0.5em minus 0.4em-DARPA IU Workshop, 1994