Tracking with Stereo-vision System for Low Speed Following Applications

Julien Morat, Frédéric Devernay, Sébastien Cornou

Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007

Tracking with Stereo-vision System for Low Speed Following Applications

J. Morat*◇, F. Devernay*, and S. Cornou◇

Abstract— **Research in adaptative cruise control (ACC) is currently one of the most important topics in the field of intelligent transportation systems. The main challenge is to perceive the environment, especialy at low speed. In this paper, we present a novel approach to track the 3-D trajectory and speed of the obstacles and the surrounding vehicles through a stereo-vision system. This tracking method extends the classical patch-based Lucas-Kanade algorithm [9], [1] by integrating the geometric constraints of the stereo system into the motion model: the epipolar constraint, which enforces the tracked patches to remain on the epipolar lines, and the magnification constraint, which links the disparity of the tracked patches to the apparent size of these patches. We report experimental results on simulated and real data showing the improvement in accuracy and robustness of our algorithm compared to the classical Lucas-Kanade tracker.**

I. Introduction

Automatic vehicle speed control is presently one of the most popular research topics throughout the automotive industry [8]. Cruise control (CC) systems, with the capability of maintaining a constant speed were the first step in this direction. The next step is adaptative cruise control (ACC), wich adds to CC the capability of keeping a safe distance from the preceding vehicle. The first ACC systems are already on the road, as pricey options on a small group of luxury cars.

Recently, new systems were developped and marketed, which added low-speed following capabilities to ACC. These systems use laser beams [12] or radar to measure the distance to the preceding car and its relative speed. LADAR and RADAR provide direct range measurements in a reliable manner, but these suffer from low angular resolution, limited field of view and cost.

Vision-based range estimations using stereo-vision techniques can now provide high resolution images yielding a wide field-of-view. Furthermore, the sensors and processing power required by these systems have become affordable, and they can also be used for other tasks such as obstacle detection or lane and road detection, creating interesting opportunities for the car industry.

Nevertheless, tracking a moving object during a long period of time is a challenging task. Most LADAR or RADAR-based ACC systems work by locating targets ahead of the vehicle, and then match these targets between two consecutive time frames. This can become difficult, especially when there are multiple targets, or when the target speed with respect to the vehicle is high compared to the frame rate of the system. Similarly, some vision-based methods generate a depth map of the scene using correlation-based stereoscopy, then extract high-level primitives representing obstacles or vehicles and finally match then between time frames [11].

A more promising approach works by combining the depth map obtained by stereoscopy with motion cues extracted by optical flow [7]. That way, the problem of matching targets between time frames becomes partially solved, and the 3-D trajectories of different targets can be estimated, although their accuracy highly depends on the accuracy of the two separate processes, which may be low in certain situations (especially for the optical flow, which suffers from the well-known aperture problem [2]).

In fact, stereo and motion cues can be extracted simultaneously by computing *scene flow* directly from the stereo image stream [13], [4], resulting in highly accurate 3-D position and speed measurements at every point in the scene. Unfortunately, the high computational cost of this process make it inadequate for real-time applications. However, the 3-D motion of *individual* features can be extracted efficiently by using an extension of the classical patch-based Lucas-Kanade tracker [9], [1] to multiple synchronized cameras [6]. In this paper, we show that this 3-D feature tracking method can be further simplified when using a stereoscopic camera setup, thanks to the image rectification process which is usually applied before stereo processing. Due to its pyramidal implementation, this tracker works in real-time, and it gives simultaneous estimates of the 3-D position and speed of features of any size in the image, such as obstacles or other vehicles, even for large motion. Besides, it can be further improved by incorporating the *magnification constraint* in the mathematical formulation of the tracker. The magnification constraint simply states that when the stereo disparity of a given feature changes, its apparent size in the images must change accordingly.

The experiments show that the performance of our tracker in terms of robustness and accuracy overcomes the performance of the classical stereo tracking method which consists in tracking each feature separately in both images, and that incorporating the magnification constraint significantly improves both its accuracy and its robustness.

The rest of the paper is organized as follows. Section II presents our extension of the classical Lucas-Kanade tracker to incorporate two constraints specific to the stereo system: the epipolar constraint, and the magnification constraint. Experimental results on synthetic data are shown, and two versions of the stereo tracker (with and without the magnification constraint) are compared to a classical Lucas-Kanade

*INRIA Rhône-Alpes, 38334 St Ismier Cedex, France {julien.morat,frederic.devernay}@inrialpes.fr
◇Environment Detection and Advanced Driving Assistance Systems, Electronic Department - Renault Research, France sebastien.cornou@renault.com

1-4244-1068-1/07/$25.00 ©2007 IEEE.

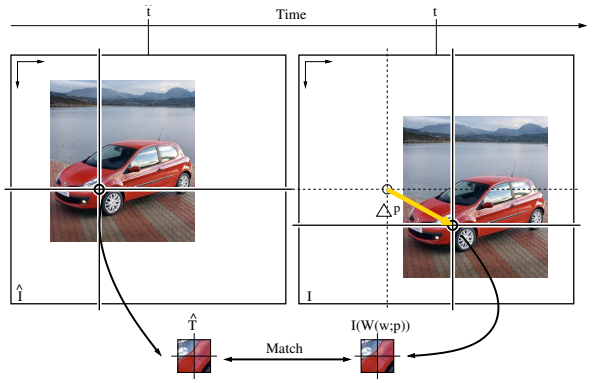

Fig. 1. The Lucas-Kanade algorithm optimizes the parameters $\mathbf{p}$ to minimize the dissimilarity between the image patch $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$ and the template $\hat{T}$ extracted from the previous image $\hat{I}$.

tracker. Section III describes how this method can be applied to ACC at low-speed, and presents results on real scenes.

II. Stereo-vision extension of Lucas-Kanade for 3-D tracking

We are interested in tracking the 3-D trajectory of vehicles using a pair of cameras. Conventional 2-D tracking techniques are designed to recover the 2-D motion of the target vehicles in the images, but do they not provide depth information. On the contrary, reconstruction from stereo-vision delivers 3-D measurements, but generally does not give any information on the motion of reconstructed objects.

Our approach is a combination of both techniques, which gives simultaneously the 3-D position and the 3-D speed of the targets (vehicles or obstacles) with respect to the cameras. We first remind the principle of the classical Lucas-Kanade 2-D tracker, on which our method relies, and we also explain how to include the epipolar constraint into it. We then explain the magnification constraint, which links the stereo disparity to the size of tracked objects, and how to incorporate it in the tracking method. Finally, results are presented, comparing the use of 2-D tracking in each image followed by reconstruction, to the two variants of our stereo tracker, with and without the magnification constraint.

A. The Lucas-Kanade algorithm and stereoscopy

The Lucas-Kanade method [9], [1] is a classical method to compute optical flow or to track 2-D points in a video sequence. Since it doesn't handle properly the well-known aperture problem, the optical flow or 2-D tracks are usually computed for a set of points of interest (typically, corners or textured areas), which can be extracted optimally [10]. From a set of 2-D points taken in image $\hat{I}$ at time $\hat{t}$ (the hat symbol denotes the previous measures), the tracker estimates the position of these points in image I at time t. Each tracked point is described by a vector of parameters $\mathbf{p}$, composed by its image coordinates in the 2-D tracking case (additional parameters, describing local geometric or photometric variations, can be added). A tracked point, also called feature, is characterized by a texture template $\hat{T}(\mathbf{x})$, where $\mathbf{x}$ is a texture point (bilinear interpolation is used for re-sampling the images, and $\mathbf{x}$ is chosen to be $(0,0)$ at the center of the template). The template is usually a square window extracted from the previous images $\hat{I}$ (cf. Fig 1). Given feature parameters $\mathbf{p}$, the *warp* function $\mathbf{W}(\mathbf{x};\mathbf{p})$ maps each point $\mathbf{x}$ in the texture template T to a point in the image I (in its simplest form for 2-D tracking, $\mathbf{W}$ is a translation by the vector $\mathbf{p}$). The Lucas-Kanade tracker optimizes the feature parameters $\mathbf{p}$ (i.e. its position) in order to minimize an energy formed by the squared differences between the texture template and the image:

$$\sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) - \hat{T}(\mathbf{x}) \right]^2, \tag{1}$$

where the sum is done over all the points in the template image.

In our case, since there are two cameras, each acquisition provides two images I_l (left) and I_r (right). The appearance of the points may be different in the left and right images, due to parallax and lighting, especially for close objects, and they should be characterized by separate texture templates $\hat{T}_l(\mathbf{x}_l)$ and $\hat{T}_r(\mathbf{x}_r)$. The energy function becomes:

$$\sum_{n\in\{l,r\}} \sum_{\mathbf{x}_n} \left[I_n(\mathbf{W}_n(\mathbf{x}_n;\mathbf{p})) - \hat{T}_n(\mathbf{x}_n) \right]^2, \tag{2}$$

where $n \in \{l,r\}$ indicates left or right camera and $\mathbf{p}$ is the feature parameters vector (detailed later). The Lucas-Kanade algorithm is an iterative method based on the Gauss-Newton algorithm. At each optimization step the goal is to find $\Delta\mathbf{p}$ which minimizes:

$$\sum_{n\in\{l,r\}} \sum_{\mathbf{x}_n} \left[I_n(\mathbf{W}_n(\mathbf{x}_n;\mathbf{p}+\Delta\mathbf{p})) - \hat{T}_n(\mathbf{x}_n) \right]^2. \tag{3}$$

This approach supposes that an initial estimate of $\mathbf{p}$ is available, usually obtained from the feature position at time $\hat{t}$. Using the first order Taylor expansion of I_n to expand eq. (3) gives:

$$\sum_{n\in\{l,r\}} \sum_{\mathbf{x}_n} \left[I_n(\mathbf{W}_n(\mathbf{x}_n;\mathbf{p})) + \nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} \Delta\mathbf{p} - \hat{T}_n(\mathbf{x}_n) \right]^2, \tag{4}$$

where ∇I_n is the gradient of I_n. The partial derivative of eq. (4) with respect to $\Delta\mathbf{p}$ is:

$$2 \sum_{n\in\{l,r\}} \sum_{\mathbf{x}_n} \left[\nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} \right]^{\mathsf{T}} \left[I_n(\mathbf{W}_n(\mathbf{x}_n;\mathbf{p})) + \nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} \Delta\mathbf{p} - \hat{T}_n(\mathbf{x}) \right]. \tag{5}$$

At the minimum, this partial derivative must be zero, wich leads to the following ewpression for the parameters update:

$$\Delta\mathbf{p} = H^{-1}\mathbf{b}, \tag{6}$$

where H is the Gauss-Newton approximation of the Hessian matrix:

$$H = \sum_{n\in\{l,r\}} \sum_{\mathbf{x}} \left[\nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} \right]^{\mathsf{T}} \left[\nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} \right], \tag{7}$$

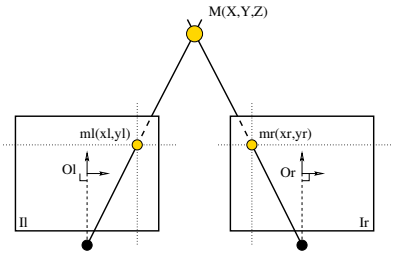

Fig. 2. In a rectified stereo pair, a 3-D point M is projected onto the 2-D image points $\mathbf{m}_l = (x, y)$ and $\mathbf{m}_r = (x + d, y)$.

and b is:

$$\mathbf{b} = \sum_{n\in\{l,r\}} \sum_{\mathbf{x}_n} \left[\nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}}\right]^{\mathsf{T}} \left[\hat{T}_n(\mathbf{x}_n) - I_n\left(\mathbf{W}_n\left(\mathbf{x}_n;\mathbf{p}\right)\right)\right]. \quad (8)$$

The first term in the sum of eq. (8) is a vector with the same dimensions as $\mathbf{p}$, and the images formed by each of its components are called the *steepest descent images*. The second term is the difference between the template and the warped image. The parameter update of eq. (6) is then repeated for each feature until convergence.

Now that we have written all the equations for the tracker, we still have to make the right choices for the parameter vector $\mathbf{p}$ describing the tracked feature, and the warps $\mathbf{W}_n$ which map the image patches at time $\hat{t}$ onto the images at time t. The simplest solution is to track the feature separately in both images, and then to reconstruct its 3-D position from the tracked 2-D positions. This corresponds to using $\mathbf{p} = (x_l, y_l, x_r, y_r)$ for the parameters vector, the translation by (x_l, y_r) for $\mathbf{W}_l$, and the translation by (x_r, y_r) for the right warp. In fact, with these choices, eq. (2) can be separated in two independent equations for the left and right images, and this stereo tracker is strictly identical to applying 2-D Lucas-Kanade tracking separately to each camera. In the rest of this paper, we refer to this method as the unconstrained Lucas-Kanade tracker.

However, by doing so, we are tracking the 3-D motion of a feature using a 4-dimension parameters vector, which means that the formulation is under-constrained. The missing constraint is in fact re-applied when the 3-D position is reconstructed from the two image positions: it is the well-known epipolar constraint. We will now show how this constraint can be taken into account during the optimization step.

B. Adding the epipolar constraint to stereo tracking

Using the unconstrained Lucas-Kanade tracker, the feature parameters $\mathbf{p}$ are four-dimensional, whereas the object being tracked has 3 degrees of freedom. If we know the projection functions of each camera, we can use for the feature parameters the 3-D coordinates of the tracked object relative to the cameras. However, this leads to more complicated warp functions and Jacobian, due to the non-linearity of the projection functions [6]. In the stereo case, a much simpler approach is to rectify the images prior to tracking (Fig. 2), so that for any 3-D point, its projections in the two cameras have the same y coordinate, and the difference between the x coordinate in the right and left images is called the stereo disparity d. For any set of stereo cameras with perspective projections, the rectification can easily be computed, and it is a 2D transform which only depends on the geometry of the cameras, not on the observed scene. Rectification is equivalent to a re-projection of the two image planes onto a plane parallel to the 3-D line joining the two optical centers, as shown Fig. 2. In that situation, the projections of any 3-D point have the same y coordinates in the images.

Since the 3-D parameters (X, Y, Z) can be computed from the image parameters (x, y, d) using the camera projection functions, we propose to use $\mathbf{p} = (x, y, d)$ as the feature parameters. The image coordinates are chosen to be zero at the principal point in each camera, for reasons that will be explained in the next section, but any image reference frame could be chosen for the calculations presented in this section. With this choice for the feature parameters, the problem is not under-constrained anymore, since the features have exactly three degrees of freedom. In fact, we integrated the so-called *epipolar constraint* into feature parametrization. The epipolar constraint states that for any point $\mathbf{m}_l$ in the left image, its matching point in the right image must lie on the projection in that image of the line passing through the left optical center and $\mathbf{m}_l$, and in the case on rectified images the epipolar line is horizontal.

The only modifications to the Lucas-Kanade equations of the previous section are different expressions for the warp functions $\mathbf{W}_{\{l,r\}}$ and their Jacobian matrices (the texture templates are still square windows extracted around the feature position in the previous images). Recall that the warps $\mathbf{W}_n$ are 2-D functionals that map template coordinates to image coordinates in image n. In this case, each warp $\mathbf{W}_n$ is the translation in image n by the 2-D coordinates of the feature position in that image:

$$\mathbf{W}_l(\mathbf{x}_l;\mathbf{p}) = \mathbf{x}_l + (x, y),\ \mathbf{W}_r(\mathbf{x}_r;\mathbf{p}) = \mathbf{x}_r + (x + d, y), \quad (9)$$

so that the Jacobian of the warps in each image are simply:

$$\frac{\partial \mathbf{W}_l}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \frac{\partial \mathbf{W}_r}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad (10)$$

and the steepest descent images from eq. (8) are:

$$\begin{aligned} \nabla I_l \frac{\partial \mathbf{W}_l}{\partial \mathbf{p}} &= \begin{bmatrix} I_{lx} & I_{ly} & 0 \end{bmatrix}, \\ \nabla I_r \frac{\partial \mathbf{W}_r}{\partial \mathbf{p}} &= \begin{bmatrix} I_{rx} & I_{ry} & I_{rx} \end{bmatrix}, \end{aligned} \quad (11)$$

where $\nabla I_l = (I_{lx}, I_{ly})$ and $\nabla I_r = (I_{rx}, I_{ry})$ are the gradients of the two images.

The Gauss-Newton approximation of the Hessian from eq. (7) also has a very simple expression:

$$H = \sum_{\mathbf{x}_l} H_l(\mathbf{x}_l) + \sum_{\mathbf{x}_r} H_r(\mathbf{x}_r), \text{ with} \quad (12)$$

$$H_l(\mathbf{x}_l) = \begin{bmatrix} I_{lx}^2 & I_{lx}I_{ly} & 0 \\ I_{lx}I_{ly} & I_{ly}^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \text{ and} \quad (13)$$

$$H_r(\mathbf{x}_l) = \begin{bmatrix} I_{rx}^2 & I_{rx}I_{ry} & I_{rx}^2 \\ I_{rx}I_{ry} & I_{ry}^2 & I_{rx}I_{ry} \\ I_{rx}^2 & I_{rx}I_{ry} & I_{rx}^2 \end{bmatrix}. \quad (14)$$

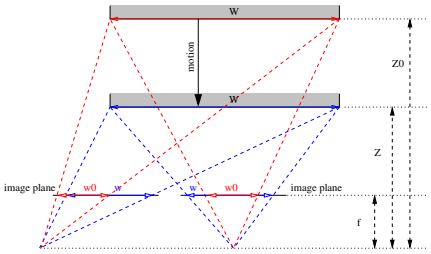

Fig. 3. Aspect size w is dependent of real size W, focal length f and object depth Z.

Using all these ingredients, we repeat the parameter update of eq. (6) until convergence. As we can see, taking into account the epipolar constraint in the Lucas-Kanade tracker leads to quite simple equations, and it has a computational cost which is not higher than applying the 2-D Lucas-Kanade tracker in each image, thanks to the simple geometry of the rectified images.

C. Objects become bigger when coming closer: adding the magnification constraint

The stereo trackers described previously make the assumption that an image patch centerer around the object at two consecutive time frames are similar (see eq. (2)). However, in automotive applications, vehicles or obstacle are usually moving fast *towards* the camera, so that their apparent size changes drastically accross time (see Fig. 3). In the following, we describe how to handle properly this change in apparent size within the tracker, by adding the *magnification constraint*.

We chose the image coordinates systems so that the coordinates of the principal point in each image are zero. With this convention, the disparity d is a simple function of the depth Z of the object, the focal length f of the cameras expressed in pixels, and the baseline B, which is the distance between the two optical centers:

$$d = Bf/Z. \tag{15}$$

Since B and f are constant, a variation of d only depends on the variation of the depth Z. There is also a simple relation between the apparent size of a fronto-parallel object in an image and its depth, defined by:

$$w = Wf/Z, \tag{16}$$

where W is the size (in world units) of a fronto-parallel object, and w is its apparent size in the images. As we can see, a strong variation of Z due to a high longitudinal speed will cause a strong variation of the apparent size w, and the closer the object, the bigger the variation of the apparent size. This strong variation may cause the previous stereo trackers to fail, which may be critical for low-speed following or obstacle avoidance.

However, is it possible to handle this size variation when tracked features are fronto-parallel (i.e. parallel to the rectified image planes). Non-fronto-parallel features or objects also follow a similar rule, although there are high order effects, which can be neglected when the depth range covered by the feature is small with respect to Z,

Equations (15) and (16) show that the disparity d and the apparent size w both depend on depth Z, leading to the following relation:

$$\frac{d}{\hat{d}} = \frac{Bf/Z}{Bf/\hat{Z}} = \frac{\hat{Z}}{Z} = \frac{Wf/Z}{Wf/\hat{Z}} = \frac{w}{\hat{w}}, \tag{17}$$

where $\hat{d}$ and $\hat{w}$ are the disparity and the apparent size at the previous time frame $\hat{t}$. We call this simple relation between the disparity and the apparent size the *magnification constraint*.

Using the same feature parameters as before, $\mathbf{p} = (x, y, d)$, we can rewrite the warp functions in order to take into account the magnification constraint (note that $\mathbf{x}_l$ and $\mathbf{x}_r$ are template coordinates, and their origin is at the center of each template):

$$\mathbf{W}_l(\mathbf{x}_l; \mathbf{p}) = \frac{d}{\hat{d}}\mathbf{x}_l + (x, y), \tag{18}$$

$$\mathbf{W}_r(\mathbf{x}_r; \mathbf{p}) = \frac{d}{\hat{d}}\mathbf{x}_r + (x + d, y), \tag{19}$$

and the Jacobian matrices become (i and j are the coordinates of $\mathbf{x}_l$ and $\mathbf{x}_r$):

$$\frac{\partial \mathbf{W}_l}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & \frac{i}{\hat{d}} \\ 0 & 1 & \frac{j}{\hat{d}} \end{bmatrix}, \quad \frac{\partial \mathbf{W}_r}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & 1 + \frac{i}{\hat{d}} \\ 0 & 1 & \frac{j}{\hat{d}} \end{bmatrix}, \tag{20}$$

and the steepest descent images become:

$$\nabla I_n \frac{\partial \mathbf{W}_n}{\partial \mathbf{p}} = \begin{bmatrix} I_{nx} & I_{ny} & S_n \end{bmatrix}, \tag{21}$$

with $n \in \{l, r\}$ and

$$S_r = \frac{iI_{rx} + jI_{ry}}{\hat{d}} + I_{rx}, \qquad S_l = \frac{iI_{lx} + jI_{ly}}{\hat{d}}. \tag{22}$$

Using Eq. (21), the Hessian matrices H_l and H_r from Eq. (12) become:

$$H_n(\mathbf{x}_n) = \begin{bmatrix} I_{nx}^2 & I_{nx}I_{ny} & I_{nx}S_n \\ I_{nx}I_{ny} & I_{ny}^2 & I_{ny}S_n \\ I_{nx}S_n & I_{ny}S_n & S_n^2 \end{bmatrix}. \tag{23}$$

The full tracking algorithm is then built by repeating the parameter update of eq. (6) until convergence for each tracked feature. This stereo tracker, altough it takes into account both the epipolar constraint and the magnification constraint, does not add much complexity to the overall computational time. In fact, the only notable difference is that the coefficients for the bilinear interpolation of the images I_l, I_r and their gradients are not constant, since the warps are not translations anymore (bilinear interpolation of these images in used in the computation of H and b, in equations (7) and (8)).

D. Implementation and Experimental Results

Our implementation is based on the real-time pyramidal Lucas-Kanade tracker included in the *OpenCV* library [5]. The pyramidal method works as follows: four image pyramids are built with a reduction factor of 0.5, from the stereo pairs at time t and $t+1$. The coarsest pyramid level is chosen

so that the motion of individual features is below 0.5 pixels, which is the the largest motion the Lucas-Kanade tracker can handle robustly. The features are first tracked in the coarsest pyramid level, as described in the previous sections, and the extracted feature motion (both in position and disparity) is scaled to the next resolution and used as the initialization for tracking at the next level. With this implementation, we get real-time performance for tracking a few hundred features from 640×480 images at 25fps on a standard PC using any of the three trackers presented in this paper.

We compare the performance in terms of accuracy and robustness of the unconstrained Lucas-Kanade algorithm (described in Sec. II-B) and of the two versions of our enhanced tracker incorporating the epipolar constraint (Sec. II-B) and the epipolar & magnification constraints (Sec. II-C). We generated a set of synthetic stereo sequences with a resolution of 1024×768, of a textured plane moving in the depth (Z) direction at a constant speed. In the initial stereo pair (at $t = 0$), the size of the textured plane is about 512×512 pixels, and a set of 400 features are generated over the plane. The disparity is initialized from the true disparity of the plane (in a real setup, it would be initialized by stereo matching). The template size is set to 21×21 pixels, and 5 pyramid levels are used in order to track high speed motion. Raw results comparing the position of tracked features for the various trackers are shown Fig. 4, and show that for a few features the different trackers behave differently, although most features are tracked correctly. For each feature, the ground truth motion is known and can be compared to the tracked feature motion.

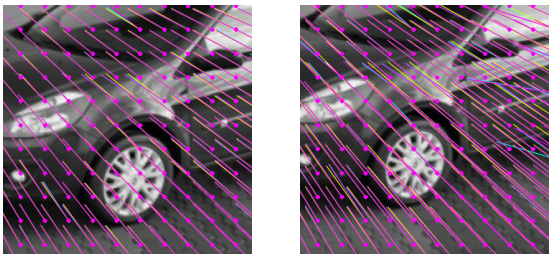

Fig. 4. Enlargement of one of the test sequences showing the points tracked with the unconstrained method (magenta), incorporating the 3-D (cyan) and incorporating the 3-D with scale (yellow). Top image corresponds to $2\times$ longitudinal speed and bottom image corresponds to $3b\times$ longitudinal speed.

Two sets of sequences are generated:

- In the first set, we generate sequences at five different longitudinal speed values (reference speed, $2\times$, $3\times$, $4\times$, and $5\times$ the reference speed), without image noise. The initial frame is the same for all sequences. With these sequences, we will be able to check the performance of the trackers with respect to motion speed.
- In the second set, we generate sequences at the reference speed, with different levels of image noise (white Gaussian noise is added to the synthetic image data). With these sequences, we will be able to check the performance of the trackers with respect to image noise.

In each sequence and for each feature, the tracking error is

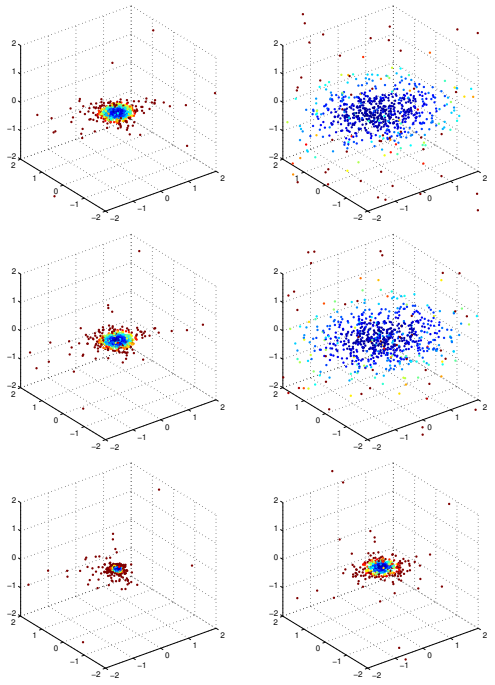

Fig. 5. Error in (x, y, d) pixel for the unconstrained Lucas a& Kanade algorithm (top), incorporating the epipolar constraint (middle) and incorporating the epipolar & magnification constraints (bottom). Left and right column correpond to small ($1\times$) and large ($5\times$) depth speed in Fig. 6.

measured in the parameter space (x, y, d), as shown in Fig. 5. From this raw data, we want to evaluate both the robustness of each tracker, measuring if the tracker fails completely or succeeds in tracking the feature, and its accuracy, measuring the accuracy of the tracked feature when the tracker succeeds. Unfortunately, the RMS error mixes the two, and a non-robust highly accurate tracker may give the same RMS error as a very robust but inaccurate tracker. For example, from the distributions shown in Fig. 5, the unconstrained tracker at $1\times$ seems to give less accurate results than the tracker with epipolar & magnification constraints at $5\times$, but its RMS error is much lower.

In order to separate outliers (i.e. wrongly tracked points) from inliers, we chose to model the error distributions of Fig. 5 by a mixture of Gaussians: a narrow Gaussian that models the accuracy of the results, and a wide Gaussian which models the outliers. The RMS error is estimated from the trace of the covariance matrix of the narrow Gaussian, and the relative weight of both Gaussians gives the percentage of outliers. This Gaussian mixture was estimated using the EM (Expectation Maximization) algorithm. The results are presented in Fig. 6 for the first set of sequences (with different speed values), and in Fig. 7 for the second set of sequences (with different noise values). The total RMS error is also shown for comparison purposes.

The numerical results shown in Fig. 6 show that both the inaccuracy and the outlier ratio for the unconstrained tracker increase with speed. The use of the epipolar constraint does not enhance the tracking accuracy, but reduces the outliers ratio. This may be explained by the reduction of the degrees of freedom from the four image coordinates

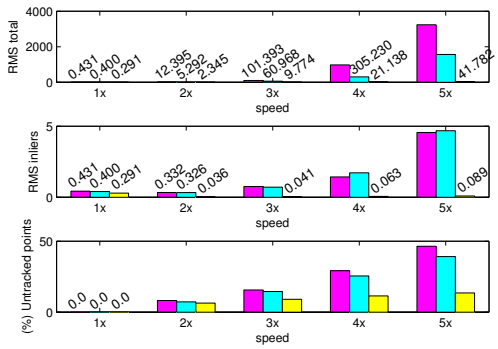

Fig. 6. Performance comparison of the unconstrained tracker (magenta) and the and the trackers with epipolar constraint (cyan) and epipolar & magnification constraints (yellow) as a function of depth speed. The RMS is expressed in pixels.

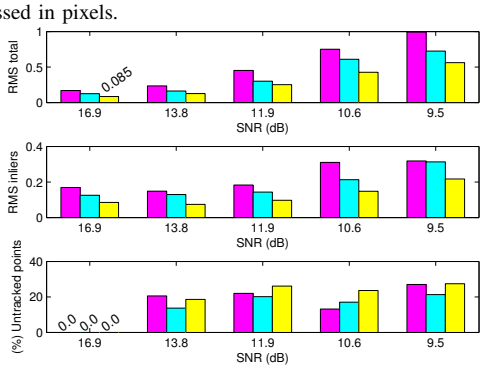

Fig. 7. Performance comparison of the unconstrained tracker (magenta) and the and the trackers with epipolar constraint (cyan) and epipolar & magnification constraints (yellow) as a function of Signal to Noise Ratio (in dB) applied at the reference speed. The RMS is expressed in pixels.

of the stereo feature to the possible solutions only (i.e. projections of a 3-D point). The joint use of the epipolar and the magnification constraints significantly improves the tracker accuracy and the outliers ratio. For higher speeds, the improvement is even more significant (up to $100\times$ more accurate for the highest tested speed). This enhancement is explained by the fact that the motion model used by the tracker follows more closely what is observed in the images.

The results obtained on the second set of sequences, with Gaussian white noise, show that at low speed, the ratio of outliers for the three trackers is similar, but the tracker with the epipolar and the magnification constraints gives more accurate results. The overall performance of the latter (measured by the the total RMS), is also better than the two other trackers. At higher speeds and on noisy images, we can expect to get a better improvement from that tracker, from the results shown in Fig. 6.

The stereo tracker could be further improved, by taking into account higher order effects in the tracker, such as illumination changes, rotation in the image, etc., but this would require adding more components to the parameter vector, and may result in a slower and less robust tracker [1]. It would be slower because of the additional computational cost, and it would probably be less robust due to over-fitting a tracking model with more degrees of freedom to the observed images. The tracker with the epipolar and the magnification constraints incorporates the maximum number of constraints

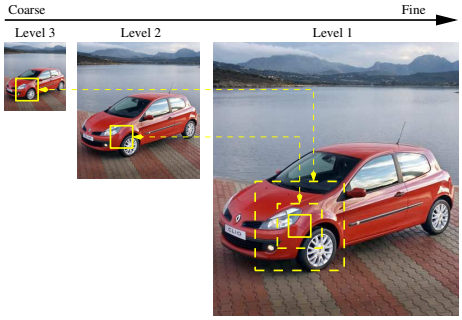

Fig. 8. Pyramidal approach by Bouguet [3] use a single window size for the different levels of the pyramid. The tracked region is bigger in coarse level than in fine level.

in a tracker with the minimum number of degrees of freedom, making it the best candidate for real-time Lucas-Kanade tracking in stereo sequences.

III. Tracking Vehicles

For the reasons explained in Sec. II-D, we used a pyramidal implementation based on the 2-D tracker by Bouguet [3], which first estimates the motion at a coarse resolution (typically $1/8$ of the original image size), and then improves it at each finer scale ($1/4$, $1/2$, and 1). When switching from one scale to another and changing the image size, the parameter vector is rescaled, however the template size in pixels remains the same, so that the template covers larger image regions in the coarser levels of the pyramid (see Fig. 8).

This trick enables tracking of small features in the full-resolution images, but it may give wrong results when the tracked features are near a motion or depth discontinuity: at coarse resolutions, the tracker may be attracted by the background motion (e.g. the road) instead of the object (e.g. car) motion. This effect is visible in the left column of the Fig. 10, where the tracked points that were initially on the rear of the preceding vehicle gradually drift on the road. The solution we propose is to track fixed-size regions instead of points. These regions may be for example the result of an obstacle detection process.

A. Tracking 3-D regions

Instead of tracking points, we choose to track rectangular regions in images, corresponding to 3-D fronto-parallel planes in the world. The objects we are interested in are the rear of vehicles, for which the fronto-parallel assumption is acceptable. The pyramidal approach is modified in order to scale the template $\hat{T}$ according to the level of the pyramid and the disparity variation. The tracked region size is memorized at each time frame (taking into account the magnification), in order to use the same region size at the next time frame.

If the region size is small in the original images, it is even smaller in the coarser pyramid levels, so that it may become impossible to track. For this reason, if at a given pyramid level the region size is below 5×5 pixels, we do not track the region in that resolution, but only in higher resolutions.

Similarly, when the region size is too big in a given resolution (e.g. it represents a surface of more than 2500

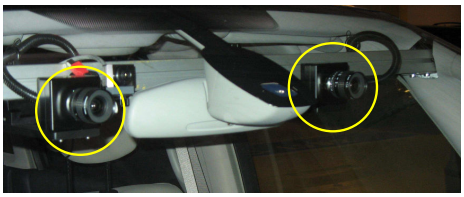

Fig. 9. Stereo-vision setup mounted behind the windshield of the test vehicle.

pixels), tracking it in that resolution may become too costly. Besides, if the region is large, since it usually does not correspond exactly to a fronto-parallel plane, the extra accuracy brought by high resolution tracking may not be meaningful. Consequently, if the surface of the tracked region is over a certain size in a given resolution, we do not track the feature in that resolution, but only in coarser resolutions.

B. Experimentation on real sequences

We present some experimental results obtained on a stereo-vision setup mounted behind the windshield of a test vehicle (Fig. 9). The cameras are 640×480 B&W $1/3$" CCD sensors with 6mm lenses. The baseline is about 40cm. Images are rectified and the image reference frame is chosen so that zero disparity corresponds to infinite depth.

Initial tracked 3-D features were manualy selected on the first image. In the sequence depicted in Fig. 10, the vehicle has been tracked over 500 frames (until it disapears) with the 3-D points tracker (left column) and the 3-D region tracker (right column). The sequences presented in this section were recorded within the French funded project *DO30*.

IV. Conclusions and Perspectives

The accurate measure of position and speed of the target vehicle in low-speed-following applications is a serious challenge for stereo-vision systems. The proposed algorithms efficiently extend the Lucas-Kanade algorithm [9] to a stereo-vision setup, taking advantages of stereo and motion tracking. The main strength of that method is that the speed measurement is obtained directly from captured images without interpretations on the tracked object, thus limiting the error sources. The most important contribution of the proposed algorithms are the consideration of the epipolar and the magnification constraints into the tracker formulation. We applied the proposed scheme to synthetic and real data. The result showed the effectiveness and robustness of the scheme. Real-time implementation of the vehicle tracker is also presented. In order to perform a full evaluation of our system versus active sensors, like RADAR or LADAR, we will also compare the outputs of these sensors, both qualitatively and quantitatively.

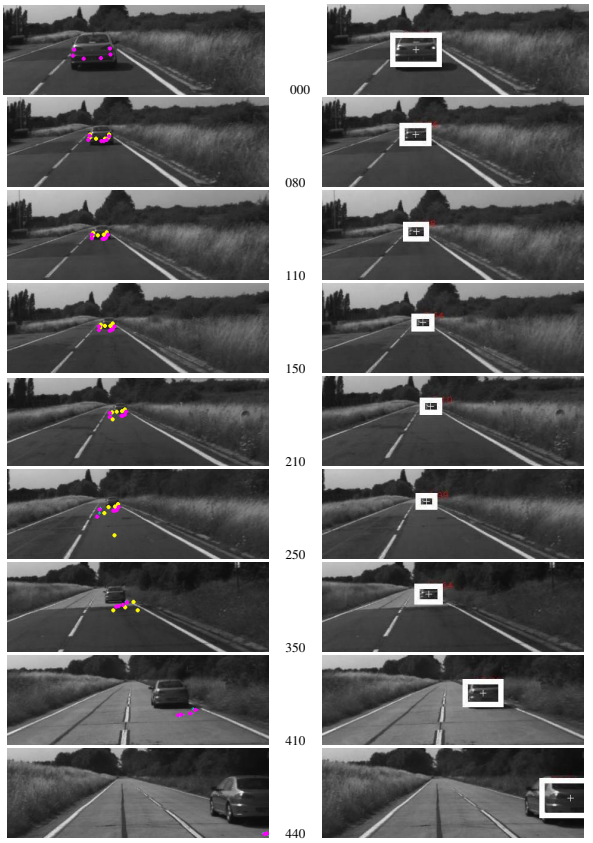

Fig. 10. Stereo-vision sequence acquired on board of the vehicle overtaking another. Left column presents the three stereo trackers (with the same colors as in Fig.5) versus the vehicle tracker shown in the right column. Center column gives the frame numbers. The feature trackers lose the target easily, because the template also covers the road in the coarse levels of the pyramid, xhereas the region tracker works correctly, although the tracked region is not perfectly fronto-parallel.

References

[1] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *IJCV*, 56(3):221–255, February 2004.

[2] J. L. Barron, D. J. Beauchemin Fleet, S.S., and T. A. Burkitt. Performance of optical flow techniques. In *Proceedings of Conference on Computer Vision and Pattern Recognition CVPR92*, pages 236–242, Urbana-Champaign, USA, 1992.

[3] Jean-Yves Bouguet. Pyramidal implementation of the Lucas-Kanade feature tracker. Technical report, Intel Corp., Microprocessor Research Labs, 2000.

[4] Rodrigo L. Carceroni and Kiriakos N. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3D motion, shape and reflectance. *IJCV*, 49(2-3):175–214, 2002.

[5] Intel CORP. *Opencv Library*, 2006.

[6] Frederic Devernay, Diana Mateus, and Matthieu Guilbert. Multi-camera scene flow by tracking 3-d points and surfels. In *CVPR (2)*, pages 2203–2212. IEEE Computer Society, 2006.

[7] HEINRICH S. FRANKE U. Fast obstacle detection for urban traffic situations. In *IEEE Transactions on intelligent transportation systems*, 2002.

[8] W.D. Jones. Keeping cars from crashing. *IEEE Spectrum*, pages 40–45, 2001.

[9] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[10] J. Shi and C. Tomasi. Good features to track. In *CVPR*, 1994.

[11] K. Sobottka, P. Zuber, and H. Bunke. Shape-based template matching for robust obstacle tracking in low-resolution range image sequences. In *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 1999.

[12] Y. Teguri. Laser sensor for low-speed cruise control. *Convergence Transportation Electronics Association*, 2004.

[13] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. *IEEE*, 27(3):475–480, 2005.