The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Road structure based scene understanding for Intelligent Vehicle Systems

Akihiro Tsukada
Toyota Motor Corporation
akihiro@tsukada.tec.toyota.co.jp

Masahiro Ogawa
Toyota Motor Corporation
masahiro@ogawa.tec.toyota.co.jp

Franck Galpin
KnowledgeNet Corporation
f.galpin@knowledgenet.jp

***Abstract*— We address the topic of intelligent vehicle systems and especially systems aiming at high level scene understanding. Our goal is to build an on-line *Behavior Map* of surrounding environment. In this case, the road structure becomes important in order to understand entities behavior. From a general viewpoint, this paper demonstrates two important concepts: 1) the smooth integration of global, absolute but partial information (the *Navi Map*) in a local and relative map (the *Behavior Map*). This is demonstrated by the building of the *Behavior Map* composed of the roads structure and pedestrian/vehicle's position and trajectory. 2) the use of both global and local information to improve the understanding of the environment. This is demonstrated by the proposed "Pedestrian Warning System". From a technical viewpoint, the main contributions are: 1) the dual ground plane/image plane approach which avoid introduction of errors in both the cost function evaluation and the constraints for minimization, 2) the smooth and efficient integration of prior data in a realistic road model computation when they are available.**

I. Introduction

A. Background

Intelligent systems can help drivers to identify dangerous traffic situation and to reduce the risk of accidents. High level information such as car navigation systems using GPS (Global Positioning System) and Global road information (the *Navi Map*) have become highly popular among car owners. On the other hand, low-level system like Driver Assist Systems (e.g. Lane Keeping or Night Vision Pedestrian Warning) are also widespread. However these systems have to make a decision using only local and low-level information (using what we denote as a *Reactive Map*). It is then difficult to apply such systems to high-level task (e.g. obstacle avoidance). In 2007 DARPA Urban challenge [9], most vehicles implemented a novel architecture combining *Navi Map*, *Reactive Map* and *Behavior Map* [1] with accurate GPS&IMU (Inertial Measurement Unit) systems. These architectures enable high-level tasks thank to the use of the *Behavior Map* (BM). Especially roads structure is quite important for drivers and pedestrians to understand a traffic scene. However, having an accurate global map and position is not always possible [5] and a map needs to be build online in order to adapt to the dynamic environment.

B. System overview

In this paper, we describe a framework for building high quality and robust BM. This map is composed of the roads structure and pedestrian/vehicle's position and trajectory. These information are retrieved from ego-sensing data, using, when available, a-priori information the *Navi Map*. The *Navi Map* is supposed absolute and global but, contrary to [9], inaccurate in terms of position and partial in terms of information (no information on dynamic entities etc.).

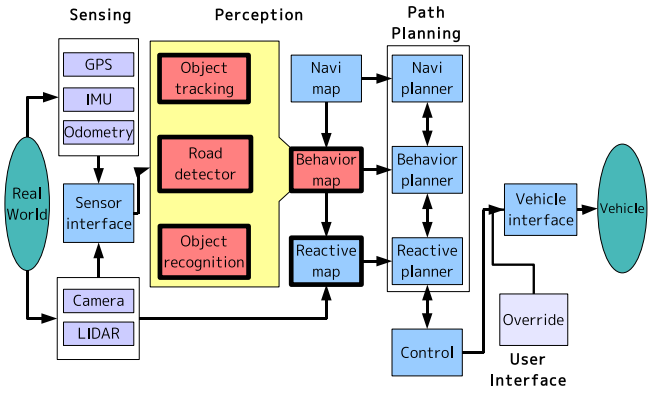

Fig. 1. System overview. Yellow area: Behavior Map modules details.

The figure 1 presents an overview of a typical Automatic Driving System (ADS). The BM itself is composed of several modules (see yellow area in fig. 1) processing data in parallel: roads structure reconstruction module, object (we focus on pedestrians in this paper) detector and tracker. The outputs are then integrated into one common map. In this paper, we focus on the BM building and especially the roads structure reconstruction. Instead of presenting the application of the BM with a Behavior Planner, which is beyond the scope of this paper, we present a simple application of the BM allowing detection of potentially hazardous situation involving pedestrians. This application requires both roads *and* pedestrians recognition. An important assumption is that this layer is embedded between several other layers, especially the BM modules receive data from the layer above called *Navi Layer*. The *Navi Layer* contains high level data (approximate roads map, expected path etc.) described in an global, absolute coordinate system. As the BM is built from sensors information, the map is local and relative to the car coordinate system. We will show that the absolute/offline

[1] maps name vary a lot from one platform to the other!

information (such as approximate roads size, roads type, car position and future path) from the *Navi Layer* are smoothly integrated in the local BM, and make the system more robust and reliable.

This paper is organized as follows. In Sec. II, we explain the implementation details of the roads structure reconstruction, in Sec. III we give an overview of the object detection and multi-object tracking. In Sec. IV, we present the system integration and the experimental results that we have obtained using this system. In particular, we show the results of the "Pedestrian Warning System".

II. Roads structure reconstruction

The goal of this algorithm is to reconstruct the 2D structure of the roads (primary road and secondary roads) on the ground plane. The algorithm mainly uses image data, but LIDAR data can be used at different stage to robustify the process (typically for the ground plane estimation). We uses the following assumptions:

- known mono or multi calibrated [2] camera system,
- roads boundaries are defined by some visible gradient,
- the system *can* use information from the navigation system if available (road properties, crossroad position, expected path etc.),
- the system *can* use odometry/speed information if available,
- the road geometry follows some road model (typically clothoid, arc or line).

The figure 2 shows the main stage of the algorithm. In the following, we will focus on the core parts of the algorithm (pre-processing and minimizer part) while other stages (ground plane and secondary roads estimation and prediction/filtering) will not be discussed here.

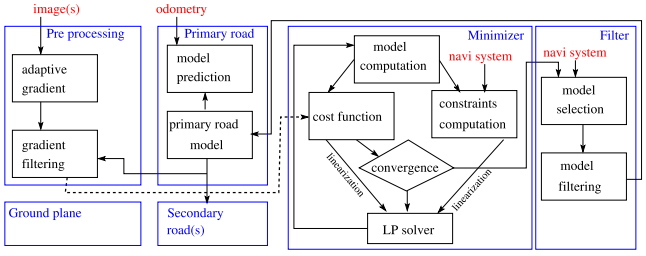

Fig. 2. Roads tracker system overview.

Compared to other systems [1] [2] the proposed approach has the following advantages:

- directly output a road model in the car coordinate system (not the image plane),
- scalable accuracy/robustness: the tracker always outputs a road model within the boundaries set by the available road data. For example, if accurate road data and position are available, the tracker will find the best road model within the margin defined by the map/localization error. If no road data are available, the tracker will find the best road model within the constraints defined by a *default* road model.
- implicit sensor fusion in the case of multi-camera system,
- roads boundaries are not necessarily defined by white lines, but only by gradient information. Furthermore, these gradients do not have to follow a particular continuous model (no segmentation required).
- multi-model for roads: single lane, double lane, secondary roads etc.
- use of a realistic road model:
  - use realistic road profiles: curve, clothoid, variable width etc.
  - use realistic constraints: maximum curvature, profile change etc.

The algorithm uses sequential convex programming method [4] to minimize iteratively a non linear objective constrained by non linear inequalities. More precisely, both objective and constraints will be linearized and a LP (Linear Program) will be solved at each step using the simplex algorithm [3]. We now present the model, cost function and constraints used by the algorithm.

A. Models

1) Camera model: We use the usual pinhole camera model using rectified images (no distortion, square pixel, no skew). The intrinsic parameters matrix is then given by:

$$K = \begin{pmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \quad (1)$$

where f denotes the camera focal, and u_0, v_0 the image center.

2) Road plane model: The road plane is supposed flat locally. It is defined by:

- h: the plane height relative to the camera coordinate,
- p: (pitch) the plane orientation around the x-axis (normal to heading direction).

Using the above model, the relationship between image coordinates $\tilde{m}(u, v)$ and the plane coordinates $m(x, y)$ are then given by:

$$\phi : m \longrightarrow \tilde{m} \begin{cases} u = \frac{f\,x}{\cos(p)\,y} + u_0 \\ v = v_0 - f\,\tan(p) + \frac{f\,h}{\cos(p)\,y} \end{cases} \quad (2)$$

3) Road model: We defined the following parameters for the road model (see fig. 3), some of which are not used depending on the road type (single lane, double lane etc.). Fix parameters are:

- $[r_x, r_y, r_a]$: position and orientation of the road frame in the global frame coordinate (fixed for primary road depending on the camera position relative to the car, or computed for secondary roads).
- $N-1$: the number of band sections.
- $\{s_i\}_{i=0..N}$: the length of the horizontal section i. They are chosen to have a similar length in image. We denote $L = \sum_i s_i$.

Estimated parameters $Z = z_{0..N+4}$ are the following:

[2] both intrinsic and extrinsic relative to the car frame

- $z_0 = dy$: the position of the center of the road normal to the road direction in the road frame coordinate,
- $z_1 = a_w, z_2 = b_w, z_3 = c_w$: the quadratic model of the road width. At a given section, for the curvilinear abscissa l, i.e. the width is given by $a_w\, l^2 + b_w\, l + c_w$. The parameter c_w represents the width of the road at the first section.
- $z_4 = r$: the relative position of the center of the road (for double lane roads).
- $z_{5+i} = \{a_i\}_{i=0..N-1}$: the orientation of the section i relative to the previous section. The orientation of the last section extremity is set to 0.

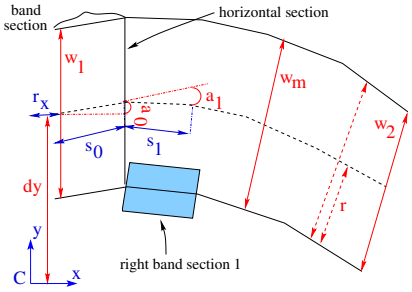

Fig. 3. Road model.

From the road model parameters, we defined the control points position of the road (center, left and right) as:

$$\psi_C^0 : Z \longrightarrow C_0 = \begin{cases} C_0^x = -\sin(r_a)\, dy + r_x \\ C_0^y = \cos(r_a)\, dy + r_y \end{cases} \tag{3}$$

$$\psi_L^0 : Z \longrightarrow L_0 = \begin{cases} C_0^x - r\, \sin(r_a + a_0)\, w_0 \\ C_0^y + r\, \cos(r_a + a_0)\, w_0 \end{cases} \tag{4}$$

$$\psi_R^0 : Z \longrightarrow R_0 = \begin{cases} C_0^x - (r-1)\, \sin(r_a + a_0)\, w_0 \\ C_0^y + (r-1)\, \cos(r_a + a_0)\, w_0 \end{cases} \tag{5}$$

The following points are then defined recursively:

$$\psi_C^i : Z \longrightarrow C_i = \begin{cases} C_{i-1}^x + \cos(A_{i-1})\, s_{i-1} \\ C_{i-1}^y + \sin(A_{i-1})\, s_{i-1} \end{cases}$$

$$\psi_L^i : Z \longrightarrow L_i = \begin{cases} C_i^x - r\, \sin(A_i)\, w_i \\ C_i^y + r\, \cos(A_i)\, w_i \end{cases} \tag{6}$$

$$\psi_R^i : Z \longrightarrow R_i = \begin{cases} C_i^x - (r-1)\, \sin(A_i)\, w_i \\ C_i^y + (r-1)\, \cos(A_i)\, w_i \end{cases} \tag{7}$$

with $A_i = r_a + \sum_{i=0}^{i} a_i$ and $w_i = a_w\, l_i^2 + b_w\, l_i + c_w$. From projection equations 2 and above equations, we can defined the projection of the control points in image $\tilde{c}_i = \phi(\psi_C^i)$:

$$\begin{cases} \dfrac{f\left(r_x - \sin(r_a)\, dy + \sum_{j=0}^{i-1} \cos\left(\sum_{k=-1}^{k=j} a_k\right) s_j\right)}{\cos(p)\left(r_y + \cos(r_a)\, dy + \sum_{j=0}^{i-1} \cos\left(\sum_{k=-1}^{k=j} a_k\right) s_j\right)} + u_0 \\ \dfrac{(\cos(p)\, v_0 - f\, \sin(p))\left(r_y + \cos(r_a)\, dy + \sum_{j=0}^{i-1} \cos\left(\sum_{k=-1}^{k=j} a_k\right) s_j\right) + f\, h}{\cos(p)\left(r_y + \cos(r_a)\, dy + \sum_{j=-1}^{i-1} \cos\left(\sum_{k=0}^{k=j} a_k\right) s_j\right)} \end{cases} \tag{8}$$

and similarly for left and right points. Using the above equation, we then directly express the projection of the road model in the image, defined with the road parameters only.

B. Cost function

Using the above road model, we define the cost function which will be minimized in order to best fit the current road structure. The cost function is based on the gradient information in images. We do not require white lines or continuous gradient to define the road boundaries. Instead, the cost function uses available gradient which are supposed to be related with the road. We now give some highlights on the cost function.

1) Adaptive gradient: One issue when extracting the gradient of pixels lying on the road plane is that its norm greatly changes with the distance to the camera. Obviously, pixels corresponding to areas close to the camera have higher resolution than pixels further from it. In order to compensate for this effect, we use an adaptive gradient filter depending on the pixel resolution. The resolution of a pixel at line v in image, lying on the road plane is given by:

$$R(v) = \frac{h}{f\, \tan(p) + (v - v_0)}$$

By using a pyramid of gradient and trilinear mipmapping interpolation as an approximation for the above equation, we can compute efficiently the adaptive gradient image using GPU capabilities. The pyramid is built using a standard Sobel gradient operator on the multi-resolution version of the original image. Top images on figure 4 shows the benefit of such approach. One can notice that small, irrelevant gradients close to the camera disappear. On the opposite, despite the smooth transition between road and grass which does not appear with standard gradient, smooth but relevant edge on the left border appears when using the adaptive filter.

2) Gradient transformation: A key point of this algorithm is to define the road model in the road plane coordinate system but to evaluate the cost function using the image plane coordinate system. The sampling is performed in image coordinates so that we do not introduce artifacts due to projection in the road plane. Using ϕ^{-1} the back projection function, the gradient $\mathcal{G}_{\mathcal{R}}(m) = (g, \tilde{\theta})$ (amplitude and orientation) in the road plane are defined as:

$$\tilde{\theta} = \tan^{-1}\left(\frac{\cos(p)\, ((v - v_0)\, du - (u - u_0)\, dv) + dv\, f\, \sin(p)}{-f\, dv}\right) \tag{9}$$

with $(u, v) = \phi^{-1}(m)$ and:

$$\mathcal{G}_{\mathcal{I}} : \tilde{m} \longrightarrow (du, dv) = \begin{cases} du = g\, \cos(\theta) \\ dv = g\, \sin(\theta) \end{cases}$$

where (g, θ) are the original adaptive gradient (amplitude, orientation) in the image (note that the amplitude is still unchanged by the back projection).

3) Gradient filtering: Gradient can be sorted in 3 categories:

1) relevant gradient: the gradient related to the road edges. It can be white lines, road borders etc.
2) irrelevant gradient (out of road gradient): these gradients are not in the road area and are discarded.
3) irrelevant gradient (bad orientation gradient): these gradients are in the road area but are not oriented correctly with the road direction. These gradients are discarded.

In order to select only relevant gradients, we apply these filters:

- position filter: will select only gradient in the road edges areas and center (for double lane roads). These areas are defined as a band $\mathcal{B}$ of width b around the road edges and center line (see the blue band example in figure 3):

$$\begin{array}{l} m \in (B) \Leftrightarrow \\ \exists i \in [0, N-1], \exists P \in \{C, R, L\}, d(m, \overline{\psi_{i+1}^P \psi_i^P}) < b \end{array} \quad (10)$$

- orientation filter: will select only gradient for which the orientation is consistent with the road orientation (relative to some threshold η) for a given section:

$$|\mathcal{G}_{\mathcal{R}}^{\theta}(m) - A_i| < \eta \quad (11)$$

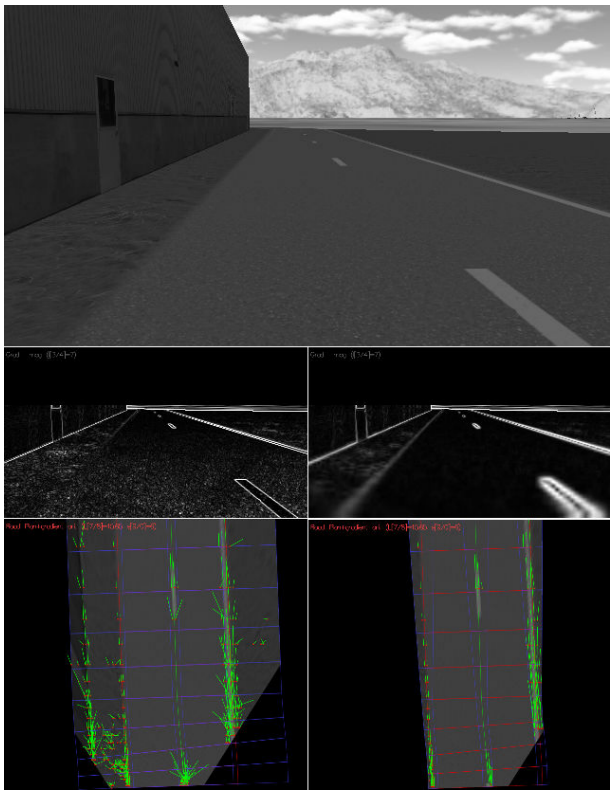

Fig. 4. Gradient computation and filtering on a synthetic image. From top to bottom: original image. Left: without adaptive filter, right: with adaptive filter. Left: without position and orientation filtering, right: with filtering.

Results are shown on figure 4: bottom right image shows that only gradients (figure shows vectors normal to the gradient) related to the road are used for the cost function computation, even in low gradient areas.

4) Cost function: We want the edges of the road to be centered on gradient of maximum values, we thus use the correlation between a triangular function $\mathcal{T}$ of width b, the band width, centered on the road edges and the gradient values:

$$\mathcal{C}^{pos} = \sum_{\tilde{m} \in \phi(\mathcal{B})} \mathcal{G}_{\mathcal{R}}^{g}(\phi^{-1}(\tilde{m})).\mathcal{T}(\phi^{-1}(\tilde{m}))$$

Notice again that the point $\tilde{m}$ are set in the image plane, while the function values are set in the road plane.

5) Cost function linearization: As we transform the problem as a LP at each step, we need to linearize the cost function. Using automatic differentiation, the gradient based direction is given by:

$$G = -\left[\frac{\partial \mathcal{C}}{\partial z_i}\right] \quad (12)$$

where z_i is a road model parameter as defined in II-A.3. From one iteration p to the next one $p+1$, we search the μ_i in:

$$z_i^{p+1} = z_i^p + \mu_i, \text{ with } \mu_i = k_i - \lambda M_i \quad (13)$$

with M_i the range of the step for parameter i. The maximum step length λ is used during the line search algorithm of the iterative search. The value μ_i is found by solving the LP:

$$\begin{array}{l} \text{max.} \sum_i k_i G_i \\ \text{s.t. } \sum_i k_i h_i^j \leq h_c^j \end{array} \quad (14)$$

We now described how we define the linear inequalities h^j.

C. Minimizer

Having defined the above cost function, we now need to minimize the function with respect to some non-linear constraints. As a use a LP, these constraints will also be linearize at each step of the minimization. The figure 5

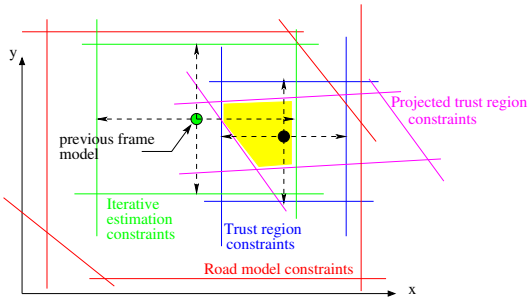

Fig. 5. Simplified overview of the constraints (after linearization). Black point: current model M(x,y). Yellow area: resulting simplex after intersection of all the constraints.

summarizes the different constraints with a simplified view with 2 parameters only (instead of $\mathbb{R}^{N+5}$).

1) Direct trust region: As the above linearization of the cost function is only valid around the current point (typically in the range $[-M_i, M_i]$), we need to define a set of constraints called the trust region. Classically, we define a box around the point with the following linear constraints on k_i (blue lines in 5).

$$\forall i \in [0..N+4], 0 \leq k_i \leq 2.\lambda.M_i$$

As the gradient descent algorithm is not affine invariant to the parameters, we usually need to take special care when defining the box. However, we will see in the next section that these constraints are superseded by other constraints coming from motion in the image plane. This set of constraints adds $N+5$ linear constraints to the linear program (the lower bound are implicit in the simplex algorithm).

2) Projected trust region: Recalling that the terms of the cost function are originally computed in the image plane, we actually want to limit the motion of the control points *in the image plane*. We only limit the motion on left and right points (since the center point will then be intrinsically constraint). Since the motion of the points in image is not a linear function of the road model parameters, we perform a linearization of the control points image coordinates. The resulting Jacobian expresses the relationship between a small displacement in parameter space and the displacement in pixel space for each control points:

$$\Gamma = \left[\begin{array}{cccc} \frac{\partial \psi_L^i}{\partial z_j} \frac{\partial \phi^u}{\partial m} & \frac{\partial \psi_L^i}{\partial z_j} \frac{\partial \phi^v}{\partial m} & \frac{\partial \psi_R^i}{\partial z_j} \frac{\partial \phi^u}{\partial m} & \frac{\partial \psi_R^i}{\partial z_j} \frac{\partial \phi^v}{\partial m} \end{array} \right] \quad (15)$$

with $i \in [0, N], j \in [0, N+4]$. The resulting $4N \times (N+4)$ matrix can be expressed only using the fix parameters and

the control points coordinates (the analytical development is not detailed here).

The projected trust region constraints are then given by (pink lines in 5):

$$\begin{cases} \Gamma_i.(K - \lambda M) \leq d_{max} \\ \Gamma_i.(K - \lambda M) \geq -d_{max} \end{cases}$$

where Γ_i the i^{th} line of the Jacobian, d_{max} the maximum displacement allowed in pixel, $K = [k_i]$ the parameters vector and $M = [M_i]$ the step vector. This gives $8\,N$ additional linear constraints to the system (N horizontal sections, left and right points, u and v motion, lower bound and upper bounds).

3) Road model constraints: Besides the constraints related to the sequential linear program, we also want to enforce some property on the road profile. One key point of this algorithm is the ability to smoothly integrate navigation data when available. This integration is done through the road model constraints (red lines in 5):

- when few or no information on the road and position are available, a set of default constraints is used by the minimizer,
- when accurate road parameters are available, the algorithm can be used to *localized* the car on the current road,
- when accurate position is available and few road information, the best road model is search inside the bounds and robust filtering can be applied,
- when both position and map are accurately known, the model parameters are tight inside a *small area* and robust and reliable results are expected.

The road model is chosen based on the type of road expected (city street, highway, single lane, double lane etc.).

a) Constraint on dy: This constraint bounds the position of the car on the road (default $dy_{max} = 1 \sim 6$m):

$$|k_0 + dy| \leq dy_{max} \tag{16}$$

b) Constraint on the road width: The parameters a_w, b_w, c_w described the road width profile. We derived 3 linear constraints by bounding the width of the road for the first section, the middle section and the last section:

$$|(a_w + k_1)\,X^2 + (b_w + k_2)\,X + (c_w + k_2)| \leq w_{max} \tag{17}$$

(default $w_{max} = 3 \sim 8$m).

c) Constraint on the center line: For double lane road, the center line position can vary slightly depending on the road type (default $dr_{max} = 0.1$).

$$|k_3 + r - 0.5| \leq dr_{max} \tag{18}$$

d) Constraint on sections angles: Constraints on angles usually arise when considering the road maximum curvature. The maximum angular difference between 2 successive sections can be approximate by:

$$\Delta a_{i,max} = |a_{i+1} - a_i| \leq \frac{s_i}{R} \tag{19}$$

where R the minimum curvature radius of the current road [3].

[3]the curvature value can usually be deduced using the maximum allowed speed of the road and some charts depending of the road type.

All these constraints add $2\,(N + 5)$ linear constraints to the system.

4) Iterative estimation constraints: As the road model is refined for each new image, we also want to limit the variation of the model from frame to frame. This is achieved by adding $2\,(N+5)$ new constraints on the parameters variation from frame to frame (green lines in 5). The constraints are defined in a way similar to the ones in II-C.3 except that instead of being defined as an absolute constraint of type $|z_i + \mu_i| < \tau_i$, it is defined relatively to the previous model parameters $|z_i + \mu_i - z_i^{t-1}| < \Delta_i$.

D. Implementation

The current implementation is balanced between:

- GPU part: compute the adaptive gradients, filter the gradient, project the model and compute the cost function,
- CPU part: compute the LP system and solve it using the simplex algorithm.

For example, a model with $N = 8$ sections (about 35m lenght) is composed of 13 parameters and 119 constraints. Combined with other features not presented here (dynamic road length, model prediction and filtering), it allows to run the estimator between 5 and 10Hz (3GHz CPU, GPU Nvidia GTX 200 series).

III. Objects Detection and Tracking

We briefly present the detection and multi-objects tracking used in the current system. Although the system aimed at tracking both pedestrians and vehicles, we focus here on pedestrians as it will demonstrate the "Pedestrian Warning System".

a) Pedestrian detection: The detection is done using HoG features and a linear-SVM classifier[7], using a GPU-based implementation. False positive are rejected using the following criteria:

- we use the road structure and ground plane as a prior in order to reject unexpected entities position and scale,
- LIDAR data allows further rejection based on scale and position of entities,
- tracker output is used to check the temporal consistency of the detection (distribution of SVM score) for a few frames.

b) Multi-object tracking: We apply Rao-Blackwellized Particle Filter[6] to multi-object tracking. We add the appearance information during data association. The features are based on a SIFT-like [8] descriptor and the matching is done using the Normalized Cross-Correlation. We also added a template update feature in order to deal with the problems arising during long term tracking.

IV. Results

A. Roads Tracker

The figure 6 shows an example of road tracking on real data. One can notice that the road is successfully tracked despite of the non relevant gradient (white arrows). The road model also handles correctly the clothoid model of the road. The tracker is also robust to other cars occultation of the road (right lane).

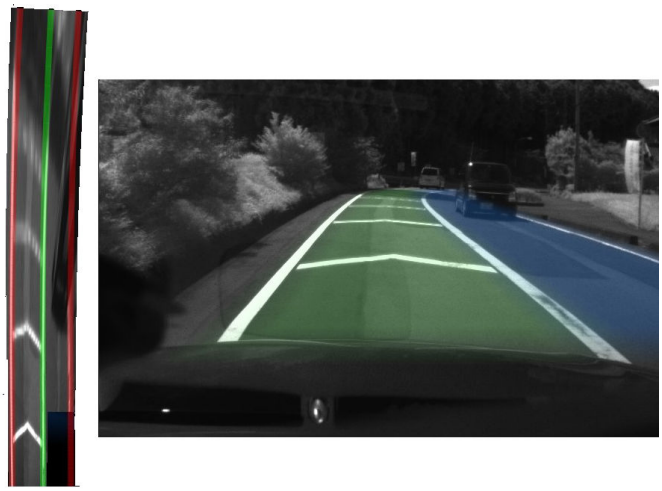

Fig. 6. Left: bird view of the road. Right: original image and road structure.

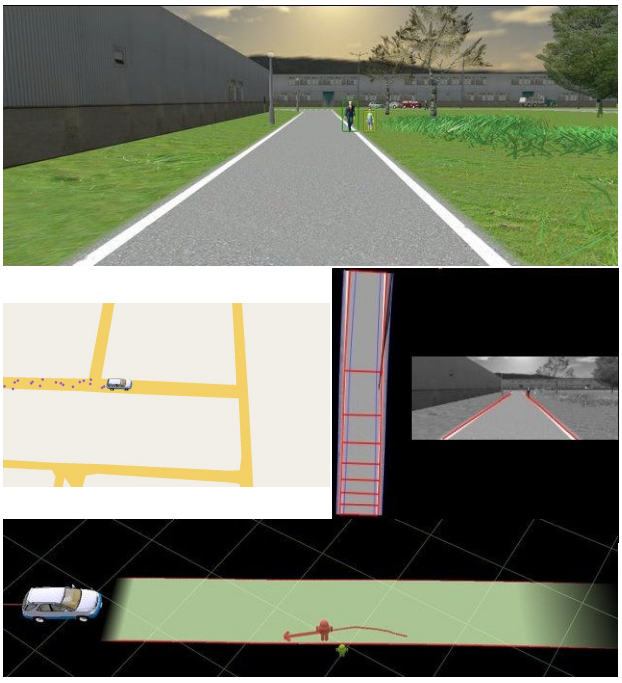

Fig. 7. Top: original image and HOG detected pedestrians. Middle: Navi map (GPS position in blue), Roads Tracker. Bottom: Final Behavior Map.

B. Integration

The presented modules are integrated using a customized middleware which allows to run all modules in parallel while maintaining the synchronization between them. A navigation module containing the *Navi Map* and current approximate GPS position is also run and send data to the Roads Tracker. The Roads Tracker also allows to track secondary roads which are initiated using GPS trigger on approximate secondary roads position (the tracking algorithm is essentially the same as the one presented before).

C. Pedestrian Warning System

For testing purpose, we use synthetic data from a World Simulator which simulate cameras, LIDAR and positioning device in a realistic way. Devices use realistic noise model (based on real sensors noise). Images are realistic enough to obtain algorithms performance *similar* to the one with real data. The navigation module contains the global path to the goal and send it to the Warning System. The pedestrians are consider dangerous when they are, or will be, in the boundaries of the road used by the car. Pedestrian speed and tracker confidence also influence the danger level of the pedestrian. The figure 7 and video shows the results with a situation detected as dangerous. In this scenario, a pedestrian is passing another one by walking on the road. From the trajectory and pedestrian position relatively to the road, the system is able to produce a warning (red color) for the walking pedestrian who is intersecting the current vehicle trajectory.

V. Conclusion

We presented several recognition modules in the area of automotive system. These modules, composed of a roads reconstruction system and an objects tracking system, are merged together to build a BM. Beside the obvious application of ADS, an application of this map was presented to demonstrated the power of high level recognition modules combination. A key point was the integration of such modules inside a global hierarchy where both high level (*Navi Map* and planning) and low level systems cooperate in order to improve robustness and reliability.

Following this direction, we are currently enriching the BM, integrating a car tracker and a *Behavior Path Planner* using the proposed method. The method is also being extended to take into account multi-camera system in order to deal with the crossroads.

References

[1] M. Aly, Real Time Detection of Lane Markers in Urban Streets, *IEEE Intelligent Vehicles Symposium*, June 2008, Eindhoven, The Netherlands.

[2] S. Sehestedt, S. Kodagoda, A. Alempijevic, and G. Dissanayake, Efficient lane detection and tracking in urban environments, *in Proc. European Conf. Mobile Robots*, 2007, pp 126131.

[3] J. A. Nelder and R. Mead, A simplex method for function minimization, *Computer Journal*, vol.7, 1965, pp 308313.

[4] J. Nocedal and S. Wright, Numerical Optimization, *Springer Series in Operations Research*, Springer, 1999.

[5] T. Luettel, M. Himmelsbach, F. v. Hundelshausen, M. Manz, A. Mueller and H.-J. Wuensche, "Autonomous Offroad Navigation Under Poor GPS Conditions", *International Conference on Intelligent Robots Systems*, 3rd Workshop:Planning, Perception and Navigation for Intelligent Vehicles (PPNIV), 2009.

[6] Simo Sarkka, Aki Vehtari and Jouko Lampinen, "Rao-Blackwellized Particle Filter for Multiple Target Tracking", *Information Fusion Journal*, Volume 8, Issue 1, 2007, pages 2-15.

[7] Navneet Dalal and Bill Triggs, "Histograms of Oriented Gradients for Human Detection", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. II, June 2005, pages 886-893.

[8] David G. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, Volume 60(2), 2004, pages 91-110.

[9] *Journal of Field Robotics*, Special Issue on 2007 DARPA Urban Challenge, 2008, Part 1,2,3.