ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/224167090

Efficient Representation of Traffic Scenes by Means of Dynamic Stixels

Conference Paper · July 2010

DOI: 10.1109/IVS.2010.5548114 · Source: IEEE Xplore

CITATIONS
138

READS
1,258

2 authors:

David Pfeiffer
Daimler
29 PUBLICATIONS **2,186** CITATIONS
SEE PROFILE

Uwe Franke
Daimler
177 PUBLICATIONS **22,326** CITATIONS
SEE PROFILE

All content following this page was uploaded by David Pfeiffer on 16 May 2014.

The user has requested enhancement of the downloaded file.

Efficient Representation of Traffic Scenes by Means of Dynamic Stixels

David Pfeiffer and Uwe Franke
Daimler Research, Sindelfingen, Germany
Email: [david.pfeiffer, uwe.franke]@daimler.com

***Abstract*— Correlation based stereo vision has proven its power in commercially available driver assistance systems. Recently, real-time dense stereo vision has become available on inexpensive FPGA hardware. In order to manage the huge amount of data, a medium-level representation named "Stixel World" has been proposed for further analysis. In this representation the free space in front of the vehicle is limited by adjacent rectangular sticks of a certain width. Distance and height of each so called stixel are determined by those parts of the obstacle it represents. This Stixel World is a compact but flexible representation of the three-dimensional traffic situation. The underlying model assumption is that objects stand on the ground and have approximately vertical pose with a flat surface.**

So far, this representation is static since it is computed for each frame independently. Driver assistance, however, is most interested in pose and motion of moving obstacles. For this reason, we introduce tracking of stixels in this paper. Using the 6D-Vision Kalman filter framework, lateral as well as longitudinal motion is estimated for each stixel. That way, the grouping of stixels based on similar motion as well as the detection of moving obstacles turns out to be significantly simplified. The new *dynamic* Stixel World has proven to be well suited as a common basis for the scene understanding tasks of driver assistance and autonomous systems.

I. Introduction

Ambitious driver assistance for complex urban scenarios demands full awareness of the situation, including all moving and stationary objects that determine the free space available for driving. We are convinced that stereo vision will play an essential role for scene understanding. It delivers position, size, and shape of arbitrary objects and thus allows for their detection and recognition irrespective of their specific appearance. The tracking of those objects or even parts of them allows us to estimate their motion, helping at the same time to distinguish between stationary and moving obstacles.

Today's disparity estimation commonly relies on a correlation based scheme. ASIC as well as FPGA stereo solutions have been developed for vehicle applications.

Recently, the dense stereo algorithm "Semi-Global Matching" (SGM) has been proposed [1], which offers accurate object boundaries and smooth surfaces. Due to the computational effort, in particular the required memory bandwidth, the SGM algorithm is still too complex for a general purpose CPU. Fortunately, we were able to implement SGM on an FPGA [2]. Figure 1 shows the result of the SGM method applied to an common urban traffic situation.

The task at hand is to extract and track every object of interest captured within the stereo stream. The research of the last decades was focused on the detection of cars and pedestrians from mobile platforms. Often different object classes are recognized using independent methods, thus the image is evaluated multiple times. This common approach has several drawbacks for a practical implementation:

1) complex software structures, which remain incomplete in detection, since only objects of interest are observed,
2) porting the software implementation to hardware is expensive, due to the amount of processed data and the complexity of the required software structures which makes high demands on the hardware solution,
3) evaluation of correctness and a required integrity level can only be achieved at extraordinary expense. This point will gain importance when the upcoming ASIL (Automotive Safety Integrity Levels) norm comes into effect.

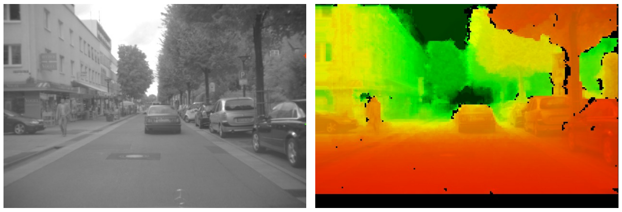


Fig. 1: The left camera image and the corresponding disparity image are illustrated. The colors encode the distance for the disparity measurements with red representing close and green representing far image points.

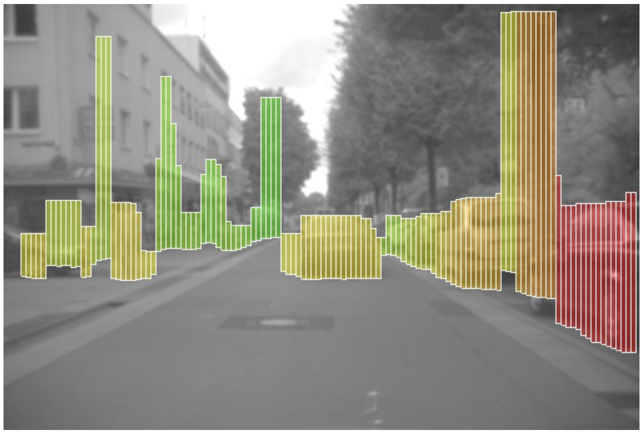


Fig. 2: Static Stixel World representation encoding the obstacles and the free space of the current traffic scene. The color scheme encodes the distance for each stixel. The stixel width is set to $w = 5\,\mathrm{px}$.

Engineers of all ages have managed the growing complexity by introducing hierarchies and abstraction layers. Following this principle we have developed a medium level representation that bridges the gap between the pixel and the object level [3]. It has the following properties required by automotive environment perception:

- **compactness:** it offers a significant reduction of the data volume, while the complete information of interest is preserved,
- **stability:** small changes of the underlying data do not cause rapid changes within the representation,
- **robustness:** outliers have very little or no impact on the resulting representation,
- **explicitness:** the information of interest is available without the need for transformation and further computation.

The Stixel World represents the 3D-situation by a set of rectangular sticks named "stixels" as shown in Figure 2. Each stixel is defined by its 3D-position in relation to the camera and stands vertically on the ground, having a definite height. Each stixel limits the free space and approximates the object boundaries. The Stixel World has been used successfully in the work of Barth et al. [4] using the silhouette of grouped stixels as a constraint for position and orientation in vehicle tracking.

Besides the position of a stixel, its motion is of utter importance for scene understanding. The static Stixel World computed from a single stereo image pair is not able to infer motion information. Therefore, we extend this approach and track the stixels over time. Using Kalman filters we are able to estimate longitudinal as well as lateral motion and obtain the pose with increased precision. This allows to distinguish between static and moving stixels and supports the subsequent grouping of stixels to objects.

Section II describes the single steps required to build the static Stixel World from raw stereo data. The new dynamic Stixel World obtained by Kalman filter based stixel tracking is described in Section III. Section IV presents results and properties of the proposed representation. Section V concludes this paper.

II. Building The Static Stixel World

Traffic scenes typically consist of a relatively planar free space limited by three-dimensional obstacles that have a nearly vertical pose. Figure 3 exemplary displays the disparity input used to extract the stixel representation as well as the current traffic situation. The different steps necessary to construct this representation are sketched first and described in detail in the following.

First of all a dense disparity image is computed using the image pair of our stereo camera system. In the second step, the free space is computed, which ends by definition at the base point of vertical obstacles. Instead of accumulating pixels above ground in an occupancy grid, we propose a new method for computing the free space directly, following an idea published in [5]. A cost image is computed (Figure 4) and subsequently analyzed. We formulate the problem in such a way that we are able to use dynamic programming, which yields a global optimum when determining the free space.

The goal of the third step is to determine the height of the stixel covering an obstacle. Stereo disparities vote for their membership to the vertical obstacle generating a membership image and a cost image (Figure 5). A second dynamic programming pass allows the optimal estimation of the height of the obstacles. An appropriate formulation of this problem allows us to reuse the same dynamic programming algorithm for this task, as applied for the free space computation. Finally, using the results of the free space computation and height segmentation the static stixel representation is extracted.

When using a stixel width of $w = 5\,\mathrm{px}$, an image with $640 \times 480\,\mathrm{px}$ is described by 128 static stixels only. Each stixel is described by just two parameters (distance/disparity and height). Their lateral position is encoded by the order from left to right. The exemplary result for the scene from Figure 1 is illustrated in Figure 2.

A. Dense Stereo

Stereo vision has been an active area of research for decades. For real-time stereo algorithms correlation-based approaches are popular (e.g. [6]). Among the top-performing algorithms in the Middlebury database [7], we found semi-global matching (SGM) [1] to be the most efficient.

Roughly speaking, SGM performs an energy minimization in a dynamic-programming fashion on multiple 1D paths crossing each pixel and thus approximating the 2D image. The energy consists of three parts: a data term for photo-consistency, a small smoothness energy term for slanted surfaces that change the disparity slightly, and a larger (constant) penalty term for depth discontinuities. Based on this algorithm, we have introduced the first real-time dense stereo implementation that runs at 25 Hz with a power consumption of less than 3 W [2]. The implementation runs on a Xilinx FPGA platform.

The following steps describe the required steps for constructing the stixel world and are illustrated on an exemplary scene depicted in Figure 1.

B. Image Based Free Space Computation

The first step when extracting the stixel representation is to determine the base point of each stixel (e.g. the coordinate where it touches the ground). By definition this point is identical to the limit of the free space along that direction. In our previous work we obtained the free space by using occupancy grids and a dynamic programming scheme as proposed by Badino et al. [8], [9]. This method has proven to be quite robust, especially when dealing with faulty disparity measurements due to bad weather conditions or other stereo artifacts. On the other hand it requires a high computational effort.

Kubota et al. presented an approach where the free space is obtained directly by warping the stereo image pairs without the need of a real stereo computation [5].

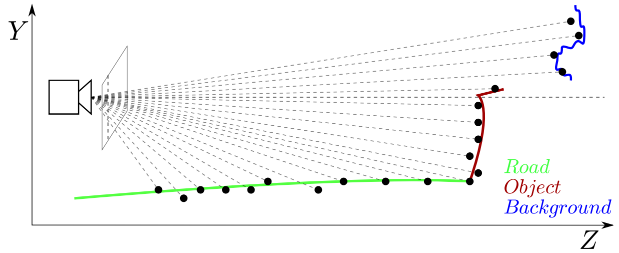


Fig. 3: Distribution of stereo measurements along one column of the image and the model assumption of road, object and background. The disparities are assumed to divide into three separate sections: road evidence (light green), object evidence (red) and background disparities (blue).

Here we propose a related method. Instead of warping the images we directly utilize the disparities to obtain a score image. This score image is then used in a dynamic programming scheme to extract the optimal free space path cutting the image from left to right while favoring temporal and spatial smoothness [8].

The disparities measurements are assumed to be partitioned into three connected sections: road disparities, obstacle disparities and background disparities. This is illustrated in Figure 3. In the current version, we do not require the road to be absolutely planar but use the approach proposed by Wedel et al. [10]. They presented a general technique for robust modeling of non-planar ground surfaces using B-Splines that allows to estimate a disparity profile $d_R(v)$ giving the disparity of the road surface at each row v (assuming a camera roll angle of ~ 0).

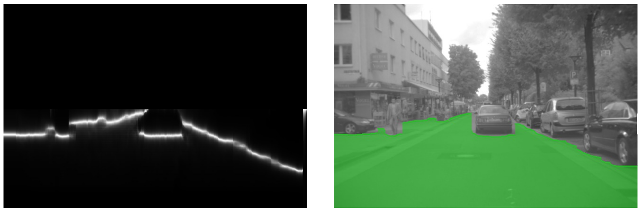
For each pixel d_v of each column in the disparity image Ω_u a score Γ_v is computed that consists of two sub-scores: one for road evidence Γ_v^R and one for obstacle evidence Γ_v^O.

$$\begin{aligned}\Gamma_v &= \alpha_1\Gamma_v^R + \alpha_2\Gamma_v^O, \text{ with}\\ \Gamma_v^O &= \sum_{v=v_b-h_v}^{v_b} |d_R(v_b) - d_v|\\ \Gamma_v^R &= \sum_{v=v_b}^{V} |d_R(v) - d_v| \qquad (1)\end{aligned}$$

The score for the object penalizes derivations from the base point disparity $d_R(v_b)$, while the score for street surface penalizes derivations from the given disparity profile $d_R(v)$. V corresponds to the height of the image and h_v gives the number of pixels to determine the object score. h_v is dynamically generated as a sum of a constant minimum number h_c and the required number of pixels h_o of an assumed virtual obstacle with the height parameter $H \approx 1\,\mathrm{m}$ standing on the ground at row v.

$$h_v = h_c + h_o \qquad (2)$$

The resulting score image is shown in Figure 4 and corresponds to the scene shown in Figure 1. The free space that is obtained when using this score image for dynamic programming is given beside the score image.



(a) Cost image for free space. (b) Resulting free space

Fig. 4: The cost image for the free space computation is shown on the left side. The right image illustrates the result of the free space computation after applying dynamic programming to the cost image.

C. Height Segmentation

The height of the objects which interrupt the free space is obtained by finding the optimal segmentation between object and background disparities. Just like to the computation of the free space, this is achieved by first computing a cost image and then applying dynamic programming again to find the upper boundary of the objects.

Given the set of free space points (v_u, d_u), d_u being the disparity value of the base point and their corresponding triangulated coordinate vectors (X_u, Z_u), the task is to find the optimal row position v_T where the upper boundary of the object at (X_u, Z_u) is located.

Every disparity $d_v \in \Omega_u$ of each column from the disparity image votes for its membership to the foreground object. In the simplest case a disparity votes positively or negatively. Positively, if it does not deviate more than a maximal distance from the expected object disparity d_u and negatively otherwise. However, such a Boolean assignment makes the threshold for the distance very sensitive: if it is too large, all disparities vote for the foreground membership, if it is too small, they all vote for the background. A better alternative is to approximate the Boolean membership in a continuous variation with an exponential function of the form

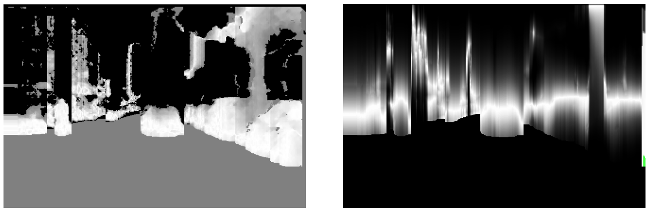
$$M_{u,v} = 2^{\left(1-\left(\frac{d_{u,v}-d_u}{\Delta D_u}\right)^2\right)} - 1. \qquad (3)$$

$$\Delta D_u = d_u - f_d(Z_u + \Delta Z_u) \text{ , with } f_d(Z) = \frac{b \cdot f_x}{Z} \qquad (4)$$

ΔD_u is a computed parameter. ΔZ_u represents the allowed deviation in $[m]$ to the base point Z_u before $M_{u,v}$ turns into a negative value. b corresponds to the base length and f_x to the focal length of the stereo camera system. Our experiments show that the explicit choice of the function (3) is not crucial as long as it is continuous. From the membership values the cost image is computed:

$$C_{u,v} = \sum_{f=v_b}^{i=v} M_{u,f} - \sum_{b=v+1}^{i=V} M_{u,b}. \qquad (5)$$

This formula expresses the idea that the maximum C_{max}, and thus the highest likelihood for the height segmentation,



(a) Membership votes.

(b) Height costs.

Fig. 5: The left side depicts the membership voting with white meaning positive votes and black meaning negative votes. Gray stands for neutral. The right image shows the resulting cost image for the height segmentation.

for each column is supposed to be reached at row v_{max} when most positive membership votes lie below (foreground) and the most negative membership values lie above v_{max} (background). Figure 5 shows an exemplary membership voting and a cost image for the scene shown in Figure 1.

D. Extraction of Static Stixel Measurements

Once the free space computation and height segmentation have been applied successfully, all required input is given and extracting the static stixel measurements is straightforward. In combination with a predefined width for the stixels (e.g. $w = 5\,\mathrm{px}$), the base points $v_{bu} \dots v_{bu+w}$ and the top points $v_{tu} \dots v_{tu+w}$ span a rectangle. All covered image points and disparities within that area belong to the static stixel measurement. Instead of taking the base point disparity as reference, the covered disparity values are now used to accurately adjust the distance associated with the stixel. For this purpose we rely upon a histogram based approach, yet any other method that offers outlier rejection would suffice as well.

This process leads to (static) stixel measurements that encode the free space as well as position and height of the first obstacle along each viewing direction within the image with high precision. The resulting stixel representation is illustrated in Figure 2.

III. Building The Dynamic Stixel World

A. The 6D-Vision Principle

The precision of static stixels can be further improved if measurements of consecutive images are combined properly. In addition, tracking the stixels can reveal the motion of Stixels covering dynamic objects. Estimating motion of other objects requires knowledge of the ego-motion. In the following, we extend the basic idea of building a static stixel representation to the time domain.

In [11] Franke et al. presented a method called 6D-Vision that allows for the simultaneous estimation of 3D-position and 3D-motion for a large number of image points. Their position is tracked over time, and a rich 6D representation combining position and velocity in the state vector $\tilde{\underline{x}} = (X, Y, Z, \dot{X}, \dot{Y}, \dot{Z})^T$ is determined using Kalman filters [12]. The underlying motion model for each tracked pixel is constant velocity with $a = 0$.

For our tracking purpose we follow this principle with the restriction $\dot{Y} = 0$, since we do not expect the stixels to move vertically. Therefore our state vector is reduced to $4D$ and only position and velocity $\tilde{\underline{x}} = (X, Z, \dot{X}, \dot{Z})^T$ are estimated.

B. Ego-motion Estimation

In order to estimate the true motion of the stixels in world coordinates, the motion of the ego vehicle must be known. Certainly, this information can be taken from inertial sensors. However, we prefer to use the method described in [13] since it outperforms available standard inertial sensors. The basic idea of this approach is to track static image points over time and use their depth to estimate the motion with full six degrees of freedom.

C. Estimation of Dynamic Stixels

Assuming a constant velocity v_c and a constant yaw rate $\dot{\psi}_c$ over the time interval Δt, the movement of a vehicle can be described in this car's right handed coordinate system by

$$\Delta \underline{x}_c = \int_0^{\Delta t} \underline{v}_c(\tau) d\tau = \frac{v_c}{\dot{\psi}_c} \begin{pmatrix} 1 - \cos \dot{\psi}_c \Delta t \\ \sin \dot{\psi}_c \Delta t \end{pmatrix}. \tag{6}$$

The new position of a world point located at $\underline{x}_k = (X, Z)^T$ after the time Δt is described by

$$\underline{x}_k = R_y(\psi)(\underline{x}_{k-1} + \underline{v}_{k-1}\Delta t - \Delta \underline{x}_c).$$

Having a filter position $(X, Z)^T$ and an estimated velocity $(\dot{X}, \dot{Z})^T$ of a stixel, the system and measurement model of the extended Kalman filter is deduced as follows. The system model is given by

$$\tilde{\underline{x}}_k = A_k x_{k-1} + B_k + \omega_k. \tag{7}$$

With

$$\begin{aligned} A_k &= \begin{pmatrix} R_y(\psi) & \Delta t R_y(\psi) \\ 0_{2\times 2} & R_y(\psi) \end{pmatrix} \\ B_k &= \frac{1}{\dot{\psi}_c} \begin{pmatrix} 1 - \cos \dot{\psi}_c \Delta t \\ -\sin \dot{\psi}_c \Delta t \\ 0 \\ 0 \end{pmatrix}. \end{aligned} \tag{8}$$

R_y corresponds to the 2×2 rotational matrix revolving around the y-axis. The noise term ω_k is assumed to be Gaussian white noise with covariance matrix Q.

A measurement to update a tracked stixel consists of three parts: The disparity measurement d_{meas}, the column position u_{meas} and a height observation h_{meas}.

In addition to stereo, we also require an algorithm to estimate the 2D-motion between consecutive images. For this purpose we rely on the dense optical flow method as introduced by Zach et al. [14] that is based upon the variational approach of Horn and Schunk [15]. However, any other optical flow algorithm serves the purpose.

Since the stereo and optical flow computation works on rectified images, we are able to use the pin-hole camera model using the non-linear measurement equation

$$\underline{z} = \begin{pmatrix} u \\ v \\ d \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} X f_u \\ Y f_v \\ b f_u \end{pmatrix} + \underline{\gamma} \tag{9}$$

with the focal lengths f_u and f_v and the baseline b of the stereo camera system. The noise vector $\underline{\gamma}$ is assumed to be Gaussian white noise with a covariance matrix R. This non-linear projection from our system state to the measurements forces us to use an extended Kalman filter. The input disparity d_{meas} and column position u_{meas} is extracted from the disparity and the horizontal component of an optical flow image using the predicted position of the stixel $\hat{x} = (\hat{X}, \hat{Z})^T$ for time step k. $\hat{x}$ is projected into the image. The measured column position u_{meas} is computed by

$$u_{\text{meas}} = u_{k-1} + du(\hat{u}) \tag{10}$$

with $du(\hat{u})$ being the displacement that is extracted from the area the stixel covers within the optical flow image around the predicted column $\hat{u}$. The disparity measurement d_{meas} is taken directly from the disparity image. The measurement-state-relation for the filter is obtained by the Jacobian approximation of (9).

The height of each stixel is assumed to be constant. Still - when approaching an observed object - the quality of the height measurement tends to increase and an additional update is reasonable. To obtain a height observation for a dynamic stixel, the closest static stixel measurement in a certain range is determined and its height is used, so no additional height segmentation for the dynamic stixels is required. The height information is updated by means of a low pass filter via

$$h_k = \alpha \cdot h_{k-1} + (1 - \alpha) \cdot h_{\text{meas}} \tag{11}$$

with $\alpha \approx 0.95$ as a fixed parameter.

Due to the possible lateral movement, stixels are no longer bound to equidistant columns. As a consequence, their maximum total number is not fixed, since they start to overlap partially and/or belong to objects in different depths.

IV. Results

The video material is recorded in our test vehicle using an 640×480 imager with a $42°$ lens. The SGM stereo algorithm is running asynchronously on an FPGA within 40 ms causing one frame delay. The dense optical flow is calculated on the GPU (NVidia 285GTX) using a CUDA implementation that takes ~ 35 ms to compute. This step is made asynchronously as well. The remaining algorithms are running on the CPU (Core2Quad 3.0Ghz, 333Mhz FSB, 4GB DDR2 800) in real-time within ~ 40 ms. This time splits as follows:

- rectification: 3 ms
- free space computation: 12 ms - 4 ms for the cost image, 8 ms for the dynamic programming step
- height segmentation: 11 ms - 3 ms for the cost image, 8 ms for the dynamic programming step
- static stixel measurements: 5 ms (including distance refinement with histogram based approach)
- dynamic stixels: 10 ms, Note that this is a mean value. The time taken depends on the actual number of tracked stixels, which can grow up to about four times the amount of static stixels.

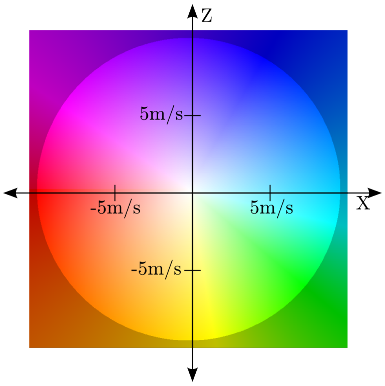


Fig. 7: Illustrates the color encoding of speed and direction. Maximum saturation is reached with 10 m/s. The color value represents the direction the stixel moves with respect to our ego-vehicle that is in line with the Z-axis.

A. Result of the Static Stixel World

The images depicted in Figure 6 show a couple of situations and their corresponding static stixel representations. These include urban environments as well as humans at close range. More examples for static stixels including further scenarios such as highways or rural roads can be found in [3].

Note how the stixel representation is able to follow the contour of the objects and thus approximating them very precisely. For all images within the following sub-sections a stixel width of 5 pixels has been chosen. By varying this parameter one has the freedom to determine the trade-off between compactness and the level of detail of the approximation.

The stixel extraction scheme proves to be quite robust. All screen shots have been generated with the same set of parameters without any additional tuning was performed.

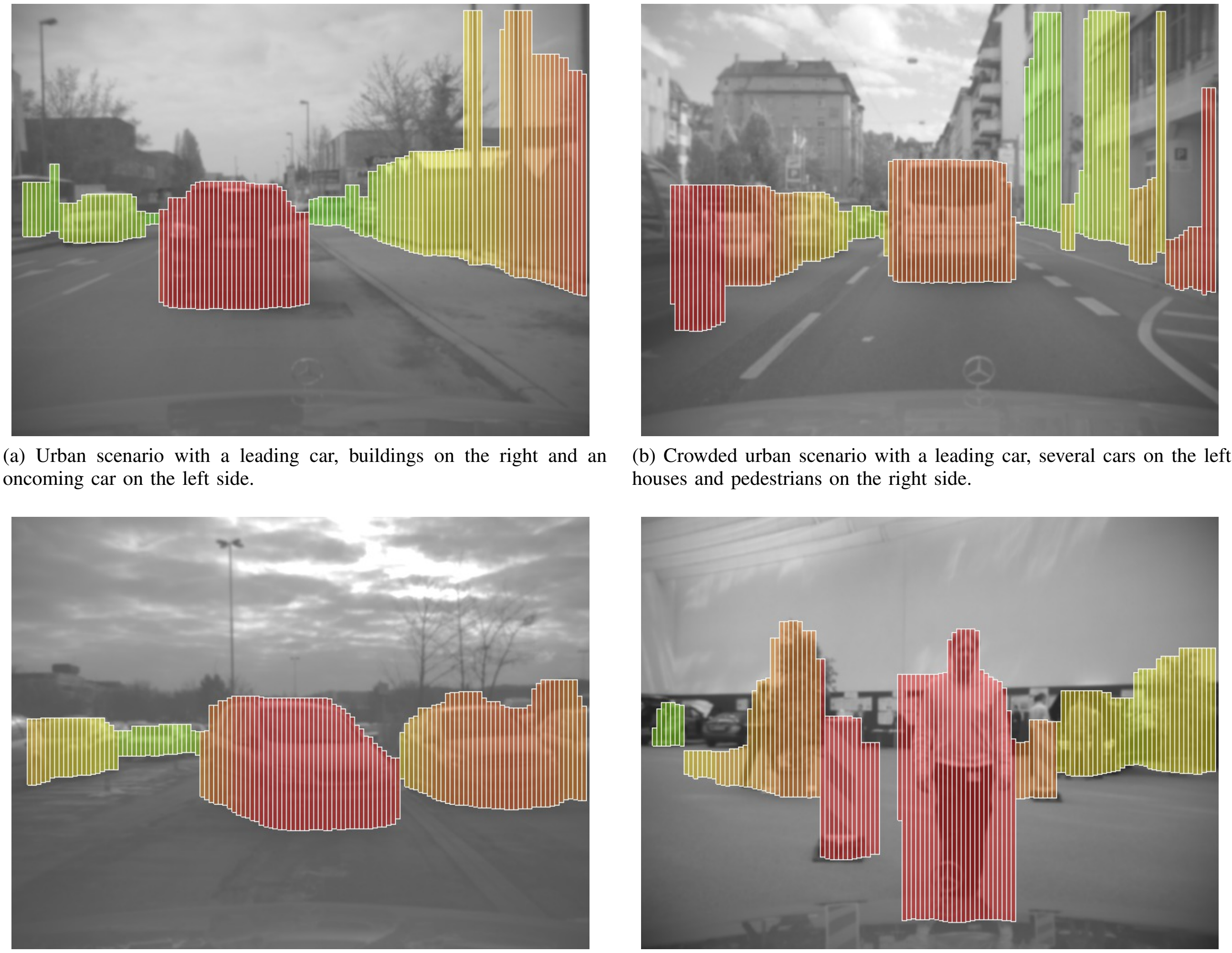
B. Results of the Dynamic Stixel World

The recorded images shown in Figure 8 illustrate some basic examples by featuring only a few moving objects. The coloring for the dynamic stixels encodes the movement along the XZ plane and is explained in Figure 7. The Kalman filters are used without any preconditioned initialization.

Additional scenarios are presented in Figure 9. These include 3D-views and correspond to more complex traffic situations with either several moving objects, a higher ego-motion or even non-rigid motion of the objects themselves (e.g. a turning truck).

A short explanation regarding the motivation for each scene as well as a description of contents is given below each sub-figure. Note that these are just exemplary examples, though the Stixel algorithm was successfully evaluated in our test vehicle for hours in open road traffic.

The Kalman filters used for these results estimate position and velocity for every Stixel independently. Depending on the filter configuration a reliable motion estimate is available within 3 update steps. To obtain further state information like de- and acceleration these filters must be extended



(a) Urban scenario with a leading car, buildings on the right and an oncoming car on the left side.

(b) Crowded urban scenario with a leading car, several cars on the left, houses and pedestrians on the right side.

(c) Parking site, a car is turning left.

(d) A person and various other objects are located ahead.

Fig. 6: A set of exemplary scenarios are illustrated together with their corresponding static stixel representation. The colors of the stixels encode the distance with red representing close and green representing far objects.

accordingly. Due to the complexity of the whole system this has not been tested yet.

V. Summary and Conclusion

A stereo-based method was presented that allows for the computation and representation of the free space as well as obstacles including information of their world motion by means of dynamic stixels.

The proposed stixel representation serves well as a compact medium level representation with significant compression rates compared to the raw input that is generated from dense stereo and optical flow computation.

Dense disparity images are used to compute cost images for the free space computation and the height segmentation. Dynamic programming is applied to these cost images to ensure spatial and temporal smoothness of the results. This information is used to extract the static stixel representation. Motion information is derived from tracking dynamic stixels over time by means of Kalman filters and the use of dense optical flow.

In conjunction with the motion information, dynamic stixels form a powerful intermediate level representation to support further processing steps such as object clustering, control of attention and reasoning. The estimation of motion works quite precisely thus delivering reliable state information. Experimental results show the robustness of our real-time capable method.

References

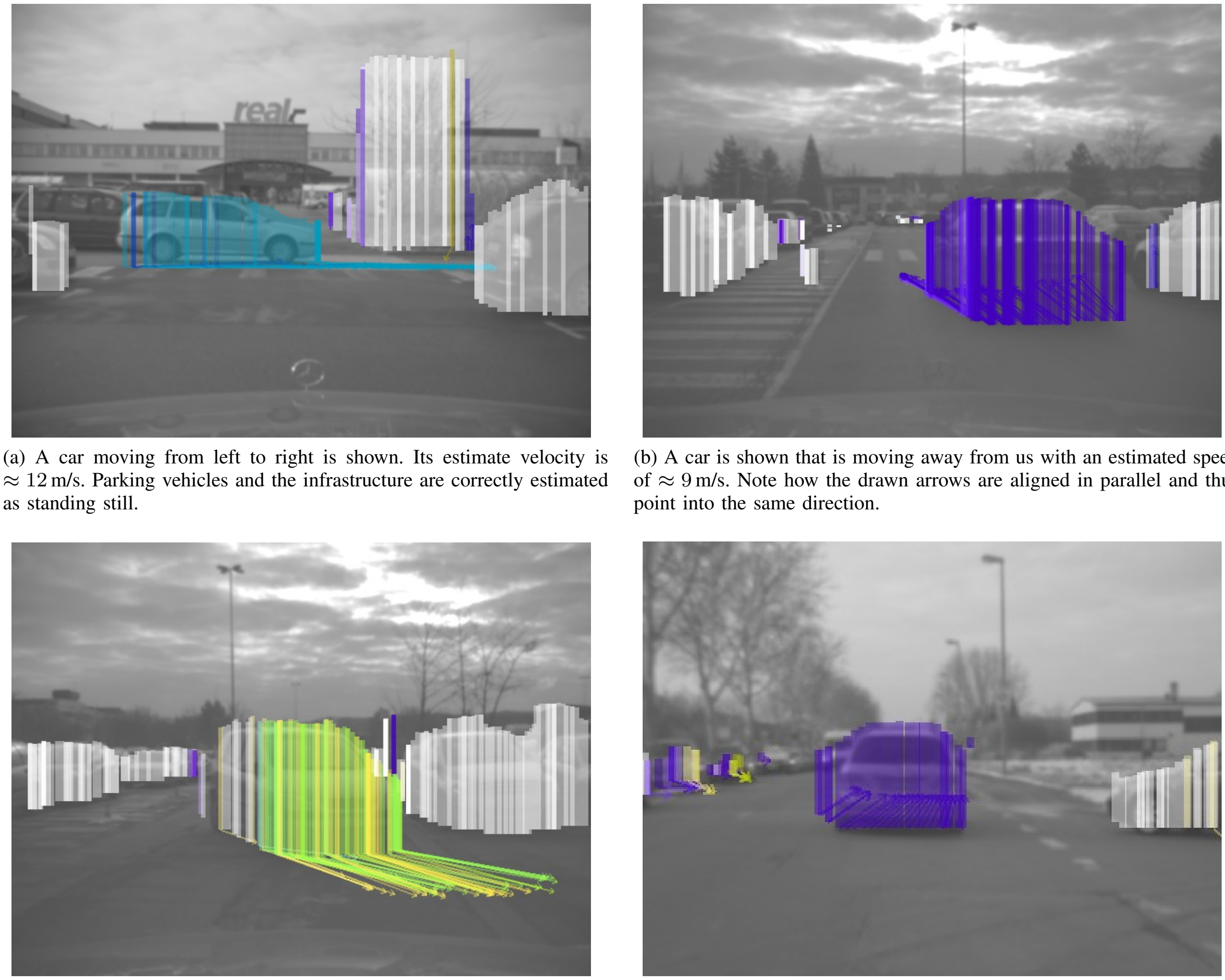
[1] H. Hirschmüller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *CVPR*, 2005.

[2] S. Gehrig, F. Eberli, and T. Meyer, "A real-time low-power stereo vision engine using semi-global matching," in *International Conference on Computer Vision Systems*, 2009.

[3] H. Badino, U. Franke, and D. Pfeiffer, "The stixel world - a compact medium level representation of the 3d-world," in *DAGM Symposium*, (Jena, Germany), September 2009.

[4] A. Barth, D. Pfeiffer, and U. Franke, "Vehicle tracking at urban intersections using dense stereo," in *Submitted*, 2009.

[5] S. Kubota, T. Nakano, and Y. Okamoto, "A global optimization algorithm for real-time on-board stereo obstacle detection systems," in *Intelligent Vehicles Symposium, IEEE*, 2007.



(a) A car moving from left to right is shown. Its estimate velocity is ≈ 12 m/s. Parking vehicles and the infrastructure are correctly estimated as standing still.

(b) A car is shown that is moving away from us with an estimated speed of ≈ 9 m/s. Note how the drawn arrows are aligned in parallel and thus point into the same direction.

(c) This illustration features an approaching car that takes a left turn just in front of our vehicle. The stixels approximating its front correctly point into that direction. The rear of this car has just been visible a few frames. Therefore the stixels on the rear are not yet recognized correctly as moving. This car moves with ≈ 5 m/s. This frame is identical to the one shown in Figure 6c.

(d) We are driving at ≈ $13\,m/s$ behind a leading car trying to keep a constant distance of ≈ 18 m. Our yaw rate is ≈ 0. All stixels covering the car point in the same direction. Their velocity estimate corresponds to our ego-motion. The car on the right is correctly estimated as standing still. Some stixels on the left side approximating parking cars are wrongly estimated as moving.

Fig. 8: Four different exemplary scenes are illustrated showing rather clearly arranged traffic scenes with only a few moving obstacles. The sequences within the parking site were taken while standing still. If stixels are estimated as moving, arrows are drawn that point into the direction where they are expected within the next half second. The color encoding is explained in Figure 7.

[6] U. Franke, "Real-time stereo vision for urban traffic scene understanding," in *Intelligent Vehicles 2000*, 2000.

[7] D. Scharstein and R. Szeliski, "Middlebury online stereo evaluation," 2002. http://vision.middlebury.edu/stereo.

[8] H. Badino, U. Franke, and R. Mester, "Free space computation using stochastic occupancy grids and dynamic programming," in *Workshop on Dynamical Vision, ICCV*, (Rio de Janeiro, Brazil), October 2007.

[9] H. Badino, R. Mester, T. Vaudrey, and U. Franke, "Stereo-based free space computation in complex traffic scenarios," in *Image Analysis and Interpretation*, pp. 189–192, 2008.

[10] A. Wedel, U. Franke, H. Badino, and D. Cremers, "B-spline modeling of road surfaces for freespace estimation," in *Intelligent Vehicles Symposium, IEEE*, 2008.

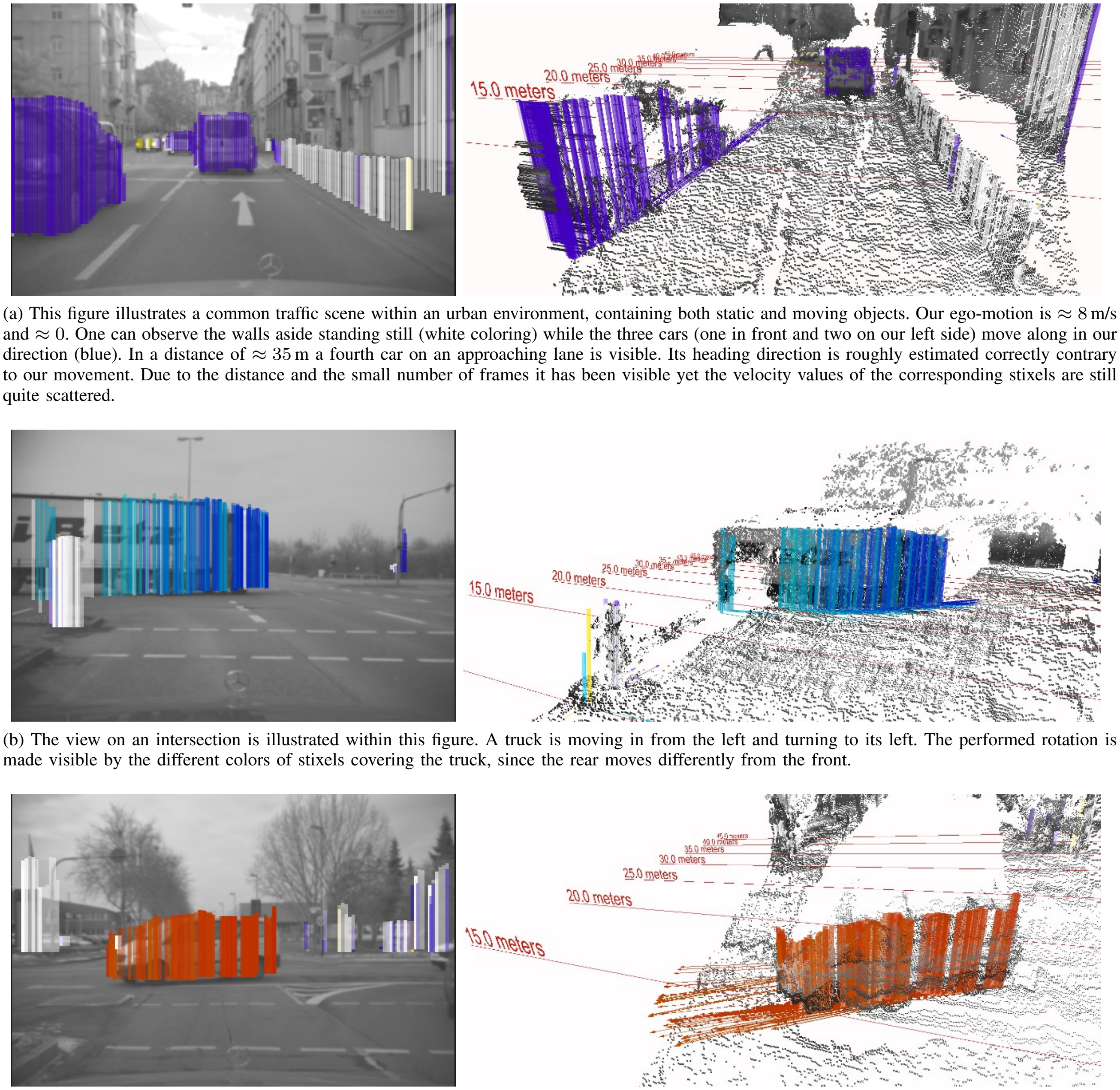
[11] U. Franke, C. Rabe, H. Badino, and S. Gehrig, "6d-vision: Fusion of stereo and motion for robust environment perception," in *DAGM Symposium*, 2005.

[12] G. Welch and G. Bishop, "An introduction to the kalman filter," tech. rep., Department of Computer Science, University of North Carolina at Chapel Hill, 1995.

[13] H. Badino, "A robust approach for ego-motion estimation using a mobile stereo platform," in 1^{st} *International Workshop on Complex Motion (IWCM'04)*, (Günzburg, Germany), Springer, October 2004.

[14] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l1 optical flow," in *DAGM*, 2007.

[15] B. K. P. Horn and B. G. Schunk, "Determining optical flow," in *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.



(a) This figure illustrates a common traffic scene within an urban environment, containing both static and moving objects. Our ego-motion is ≈ 8 m/s and ≈ 0. One can observe the walls aside standing still (white coloring) while the three cars (one in front and two on our left side) move along in our direction (blue). In a distance of ≈ 35 m a fourth car on an approaching lane is visible. Its heading direction is roughly estimated correctly contrary to our movement. Due to the distance and the small number of frames it has been visible yet the velocity values of the corresponding stixels are still quite scattered.

(b) The view on an intersection is illustrated within this figure. A truck is moving in from the left and turning to its left. The performed rotation is made visible by the different colors of stixels covering the truck, since the rear moves differently from the front.

(c) A car is moving from right to left and performs a left turn at an intersection. It moves rather rigidly, which is denoted by the uniform orange coloring.

Fig. 9: More complex results of dynamic stixels are shown. The color encoding is explained in Figure 7. To be able to illustrate the motion properties and spatial alignment in detail, 3D-views of the scene are given besides the pictures.

View publication stats