ObjectFlow: A Descriptor for Classifying Traffic Motion

Andreas Geiger and Bernd Kitt
Institute of Measurement and Control Systems
Karlsruhe Institute of Technology
{geiger,bernd.kitt}@kit.edu

***Abstract*— We present and evaluate a novel scene descriptor for classifying urban traffic by object motion. Atomic 3D flow vectors are extracted and compensated for the vehicle's egomotion, using stereo video sequences. Votes cast by each flow vector are accumulated in a bird's eye view histogram grid. Since we are directly using low-level object flow, no prior object detection or tracking is needed. We demonstrate the effectiveness of the proposed descriptor by comparing it to two simpler baselines on the task of classifying more than 100 challenging video sequences into intersection and non-intersection scenarios. Our experiments reveal good classification performance in busy traffic situations, making our method a valuable complement to traditional approaches based on lane markings.**

I. Introduction

Vision-based intersection detection and recognition [10], [26], [23], [20], [9] is an important task for advanced driver assistance systems and autonomous driving. However, it is also highly difficult for several reasons.

First, nature's diversity in appearance is hard to capture using a computer algorithm: While large and small intersections exist, they may be marked or unmarked. Traffic lights and signs may or may not be present. Also, the surrounding scenery can be either natural (e.g., trees and bushes) or man-made (e.g., buildings and brigdes).

Second, the tilted installation angle of wide angle car-mounted cameras offers rich details in the cars vincinity, but only little details at the region of interest (at distances ≥ 15 meters).

Third, other vehicles, which show up in a variety of different colors and shapes, often occlude major information sources of the scene such as lane markings. Hence, approaches based solely on lane markings are not applicable to busy traffic situations, which make up for most of the innercity scenarios. A typical 'intersection' and 'non-intersection' image frame is depicted in figure 1, demonstrating the aforementioned difficulties.

In recent years, much research has been conducted in vision based road geometry estimation and tracking [3], [29], [18], [5], [28], [6], [2], [8], [27], [22], [24]. However, most of the approaches use features based on lane markings and assume an unobstructed view onto the road, which is often not the case in innercity scenarios. Furthermore a simple road model based on splines or clothoids is used, which does not naturally allow for representing more complex multi-lane scenarios like intersections.

In contrast to our approach, [16], [17], [13] exploit digital road maps to generate road models which are mapped into the scene and matched against road features in the images, also assuming an unobstructed view onto the road. In [25] feature maps based on aerial images are generated for the same purpose. Deductive inference is used in [19], in combination with description logic to narrow down the space of plausible intersection hypotheses.

(a) Intersection scenario

(b) Non-intersection scenario

Fig. 1. **A typical intersection and a typical non-intersection scenario.** The task of detecting intersections is aggravated by the diversity in scene appearance, the tilted installation angle of the camera and large occlusions, which mainly caused by other traffic participants. Approaches solely based on lane markings will fail in such situations.

In [30] a Kalman-filter based tracking system is presented, which tracks vehicles and pedestrians at intersections. However, in their scenario a static camera on top of a building was used, making position estimation less noisy than in our setting, where depth estimates have to be deduced from disparities and the observer is moving.

In [23] road border discontinuities and lane markings are estimated to detect intersections from imagery. In [9] simple intersections without traffic are recognized by detecting road clothoids and their intersecting lines. An active camera setup is proposed in [20] for the same task. First classification approaches were conducted in [26], where color appearance has been used to segment the shape of the road in a rectified view into several intersection types using a binary support

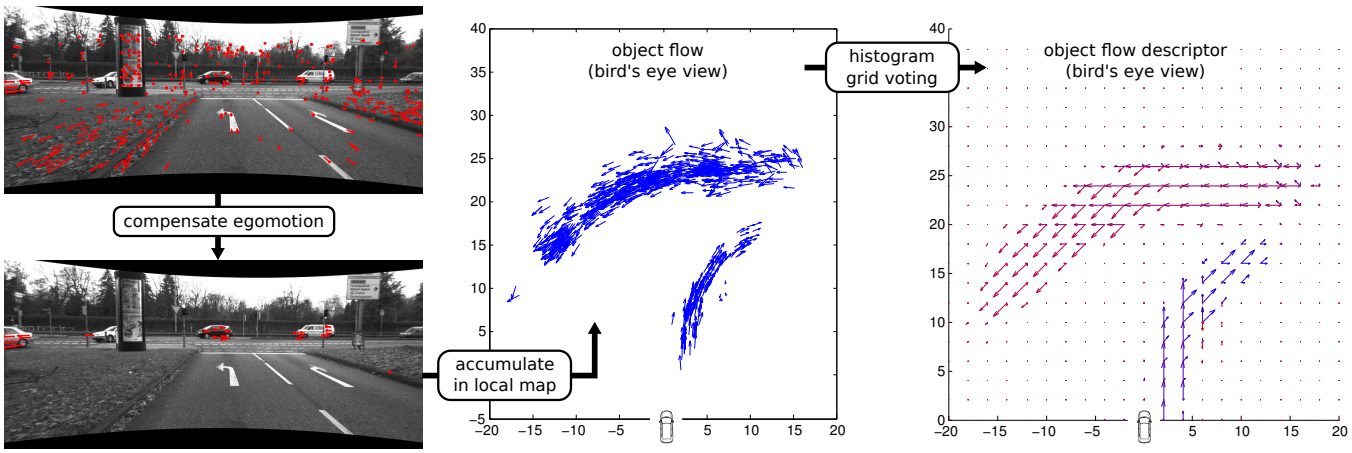


Fig. 2. **System overview.** By matching feature points and computing the vehicles egomotion using visual odometry we extract motion compensated 3D flow vectors from stereo video sequences (left). Flow vectors from the last 100 frames are accumulated into one common coordinate system and projected onto the ground plane (middle). A histogram grid is extracted by feature-based voting and serves as input to classification (right). Note that the right turning vehicle is not present at the first frame of the sequence which is shown on the left for illustration purposes.

vector machine. [10] employs super-pixel segmentation to construct feature sets, which are fed into a boosting-based classifier to distinguish between different road types and detect cars, pedestrians and crossings.

In this paper, we present a descriptor for traffic classification which is easy to compute and complementary to existing lane-marking based approaches. Instead of focusing on lane-markings and curbstones, we robustly compute the flow of scene objects like bicycles, pedestrians and cars. This 'object flow' is accumulated in a local 2D map from which histogram grid features are computed, based on a voting scheme. Unlike the map-mosaicing approach proposed in [12], this allows for a memory-efficient implementation, since only a sparse set of vectors has to be stored temporarily. Using a large margin classifier, we find optimal boundaries for classifying these features into the two classes 'intersection' and 'non-intersection' without prior streetmap knowledge.

II. System Overview

Figure 2 gives an overview over our system: First, sparse image features are extracted and matched sub-pixel accurately against spatially and temporally neighboring frames. A robust egomotion approach (section III) yields the extrinsic motion parameters between consecutive frames, which are used for extracting motion compensated 3D object flow vectors from the feature matches. We accumulate these vectors into a local map, containing approximately the last 100 frames. Voting, based on the object flow position and orientation finally gives a global histogram grid descriptor which is fed into a Support Vector Machine for classification (section IV). Our approach is evaluated on more than 100 video sequences (section V). We conclude this paper with an outlook on future work.

III. Visual Odometry

Due to our assumption of a moving observer, static parts of the scene have to be compensated for the egovehicle's

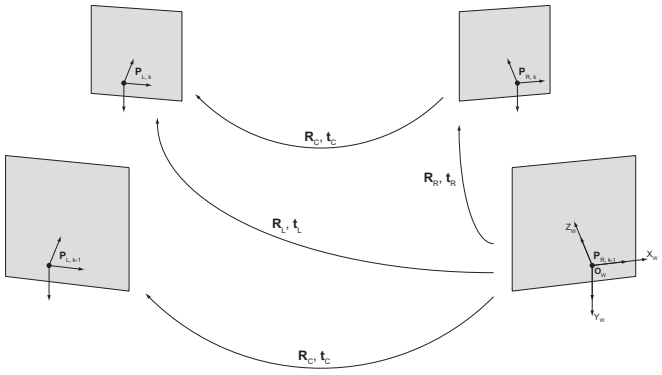


Fig. 3. **Camera configuration.** This figure illustrates the geometrical relations of our stereo-camera-rig at two consecutive time steps. The origin of the world coordinate frame is denoted by $\mathbf{O}_W$.

movement. The computation of egomotion further allows for registering object flow vectors into a local map. In our experiments we use a time span of 10 seconds, corresponding to ≈ 100 image frames.

Figure 3 shows the configuration of our moving stereo-camera-rig at two consecutive time-steps. Without loss of generality, the world reference frame coincides with the camera coordinate frame of the previous right camera. Using this assumption, the pose of the left camera with respect to the right camera is given by the extrinsic calibration of the stereo camera rig $\{\boldsymbol{R}_C, \boldsymbol{t}_C\}$ which is assumed to be fixed and known. Note that self-calibration methods like [7] could be used to relax this assumption. The poses of the current cameras with respect to the previous right camera ($\{\boldsymbol{R}_R, \boldsymbol{t}_R\}$ and $\{\boldsymbol{R}_L, \boldsymbol{t}_L\}$) are defined by the unknown egomotion and the extrinsic calibration parameters.

To parameterize the camera motion, i.e. the orientation of the right camera coordinate frame with respect to the world reference frame, we use a translation vector $\boldsymbol{t} = (t_X, t_Y, t_Z)^\mathsf{T}$ and a rotation matrix $\boldsymbol{R}(\Theta, \Phi, \Psi)$ which is a

concatenation of rotations around the coordinate axis of the world reference frame. With the knowledge of the egomotion $(V_X, V_Y, V_Z, \omega_X, \omega_Y, \omega_Z)^T$ and the time difference ΔT this transformation is readily given.

A. Trifocal Constraints for Egomotion Estimation

As can be seen from figure 3, the projection matrices of all cameras $\{\mathbf{P_{R,k-1}}, \mathbf{P_{L,k-1}}, \mathbf{P_{R,k}}, \mathbf{P_{L,k}}\}$ can be computed based on the calibration and the egomotion of the stereo rig. The geometric relationship between three of these cameras, i.e. both cameras at frame $k-1$ and one of the cameras at frame k respectively, can be expressed by the *trifocal tensor* $\mathcal{T}$ [15]. This tensor encapsulates the projective relations between three views of a scene. Its entries are determined by

$$\mathcal{T}_i^{qr} = (-1)^{i+1} \cdot det \begin{pmatrix} \sim \boldsymbol{a}^i \\ \boldsymbol{b}^q \\ \boldsymbol{c}^r \end{pmatrix}, \tag{1}$$

where $\sim \boldsymbol{a}^i$ represents matrix $\boldsymbol{P}_A$ without row i and $\boldsymbol{b}^q$ and $\boldsymbol{c}^r$ are the q-th row of $\boldsymbol{P}_B$ and the r-th row of $\boldsymbol{P}_C$ respectively. Given the trifocal tensor and matching image points $\boldsymbol{x}_A \leftrightarrow \boldsymbol{x}_B$ in two images, the corresponding image point $\boldsymbol{x}_C$ in the third image can be determined using the *point-line-point transfer* [15]. This transfer can be expressed by a non-linear mapping via $\boldsymbol{x}_{j,k} = h_j\left(\mathcal{T}_j, \boldsymbol{x}_{R,k-1}, \boldsymbol{x}_{L,k-1}\right)$ with $j \in \{R, L\}$ defining the current camera.
Thus, the relationship between the trifocal tensor and point correspondences in three images can be used to determine the egomotion of the stereo rig.

In a first step, we use a Laplacian-of-Gaussian (LoG) approximating interest point detector (e.g., CenSurE [1]) to quickly detect blob-like features in the current and previous stereo pair $\{\mathbf{I_{R,k-1}}, \mathbf{I_{L,k-1}}, \mathbf{I_{R,k}}, \mathbf{I_{L,k}}\}$.

Second, we extract MuSURF (modified upright SURF) descriptors [1] and match them using the l_1-norm for robustness and efficiency. Only reliable correspondences which match in a loop $(\boldsymbol{x}_{L,k-1} \leftrightarrow \boldsymbol{x}_{R,k-1} \leftrightarrow \boldsymbol{x}_{R,k} \leftrightarrow \boldsymbol{x}_{L,k} \leftrightarrow \boldsymbol{x}_{L,k-1})$ are kept. Note that we are using rotation-variant features since exaggerated roll motion between consecutive frames is unlikely.

Third, a *bucketing* [31] technique is used to reduce the number of features. This results in several advantages: On one hand, the number of correspondences is reduced, hence also the computational complexity of the filtering algorithm. On the other hand, all image features are approximately uniformly distributed over the image plane. This is important, since far features are crucial for reliably estimating the angular velocity and near features guarantee an accurate estimation of the longitudinal vehicle velocity. Furthermore, bucketing guarantees that no biasing towards features on independently moving objects is present, which results in a more stable motion estimation especially in highly dynamic environments. The remaining feature points located on independently moving objects as well as false feature correspondences are rejected in an iterative manner using an *iterated Extended Kalman-Filter (IEKF)* described in the following section.

B. ISPKF based Motion Estimation

To reduce the impact of noisy measurements and to include knowledge about the dynamic behaviour of the vehicle we use a Kalman Filter based state estimation. Because of the non-linearities in the mappings induced by the trifocal tensor we use an *Iterated Sigma-Point Kalman-Filter (ISPKF)*, which linearizes the observation model. For discrete-time systems [14], the state and observation model are given by

$$\boldsymbol{y}_{k+1} = f\left(\boldsymbol{y}_k\right) + \boldsymbol{w}_k \tag{2}$$

$$\boldsymbol{z}_{k+1} = h\left(\boldsymbol{y}_{k+1}\right) + \boldsymbol{v}_{k+1}. \tag{3}$$

Here, $h\left(.\right)$ is the non-linear measurement equation as described in section III-A, $\boldsymbol{w}$ and $\boldsymbol{v}$ are the system noise and measurement noise, respectively. The state $\boldsymbol{y} = (V_X, V_Y, V_Z, \omega_X, \omega_Y, \omega_Z)^T$ is defined by the egomotion of the vehicle and the measurements $\boldsymbol{z}$ are the locations of the matched feature points in both current images. Since we are making a constant velocity assumption we have $f\left(\boldsymbol{y}\right) = \boldsymbol{y}$.

Compared to non-iterative filtering, the application of an ISPKF yields two central benefits: First, the linearization error can be reduced which results in more accurate motion estimates. Second, we eliminate feature correspondences not consistent with the current estimate by applying RANSAC on a random subset of the features. In a final step, all inliers are used. More details can be found in [21].

IV. Object Flow

This section describes how we compute the object flow by extracting 3D flow vectors. Note that our method is related to the approach proposed in [11], however we do not employ Kalman filters to track single features over time, but rather project atomic flow vectors onto the ground plane (i.e., bird's eye perspective). The flow is further processed in a voting-based procedure to give the final histogram descriptor which is used for classification.

A. Local Flow Accumulation and Filtering

Since we are using a stereo system with disparities estimated at sub-pixel accuracy, we can use triangulation to map the already extracted image features $\boldsymbol{x}_{L,k-1}, \boldsymbol{x}_{R,k-1}, \boldsymbol{x}_{R,k}, \boldsymbol{x}_{L,k}, \boldsymbol{x}_{L,k-1}$ (section III) to 3D points $\mathbf{X}_{k-1}, \mathbf{X}_k$ with $\mathbf{X} \in \mathbb{R}^3$. Without loss of generality we define those points in the coordinate system of the right camera. To compensate for the observer's motion and to separate the static scene part from dynamic objects, 3D points from frame $k-1$ are mapped into the coordinate system of frame k according to

$$\mathbf{X}_{k-1}^{+} = \begin{pmatrix} \mathbf{R}_{k-1\to k} & \mathbf{t}_{k-1\to k} \end{pmatrix} \begin{pmatrix} \mathbf{X}_{k-1} \\ 1 \end{pmatrix} \tag{4}$$

where $\{\mathbf{R}_{k-1\to k}, \mathbf{t}_{k-1\to k}\}$ denotes the vehicle's egomotion between frames $k-1$ and k. The 3D flow vector at frame k

is finally given by

$$\xi_k^{3D} = \begin{pmatrix} \mathbf{X}_k \\ \dfrac{\mathbf{X}_k - \mathbf{X}_{k-1}^+}{\Delta T_{k-1\to k}} \end{pmatrix} \in \mathbb{R}^6 \tag{5}$$

with time difference $\Delta T_{k-1\to k}$. The first 3 elements of ξ_k^{3D} represent the flow vector's location (in meters) in frame k's coordinate system and the last 3 elements are its velocity (in meters per second). To account for outliers and static objects in the scene we apply three additional steps:

First, we reject all flow vectors for which the compensated optical flow (reprojection into the image plane) does not exceed 8 pixels. We further keep only flow vectors which are positioned at plausible distances (5-50 meters). Last, we limit the flow vector speed to values between $0.3\frac{m}{s}$ and $30\frac{m}{s}$. This filtering step is illustrated in figure 2 (left).

Since disparity depth estimates are usually highly sensitive to calibration errors and noisy at large distances, a 3D median filter is adopted at each frame with a kernel size of 3 meters. Finally, temporal integration is achieved by building a local 2D map. To this end, all flow vectors ξ^{3D} from the last 10 seconds are projected into one common coordinate system, namely the current one at frame k, making use of the egomotion estimates from section III. Knowing the camera's pitch and roll angle, we project the aggregated flow vectors into 2D (bird's eye perspective), giving N object flow vectors $\left\{\xi_i^{2D} = (p_x^i, p_z^i, v_x^i, v_z^i)^T\right\}_{i=1}^N$ at position $\mathbf{p}_i = (p_x^i, p_z^i)^T$ and with velocity $\mathbf{v}_i = (v_x^i, v_z^i)^T$. Figure 2 (middle) illustrates the object flow vectors extracted from a typical intersection scenario.

B. Voting-based Flow Description

Since our final goal is classification, a fixed-sized descriptor must be extracted from the variable-sized object flow vectors described in the previous section. We tackle this problem by employing a voting scheme, in which each flow vector casts votes for its own direction on a fixed 2D grid of size $M_x \times M_z$. Flow vectors are sparse, but can be extrapolated to a certain extend. Thus we decided for anisotropic voting weights according to the motion direction (see figure 4). In the following, we denote the grid locations by $\left\{\mathbf{q}_j = (q_x^j, q_z^j)^T\right\}_{j=1}^{M_x \times M_z}$. In order to accumulate all votes, each grid point j is represented by a histogram $\mathbf{h}_j \in \mathbb{R}^{M_\theta}$, quantizing the object flow into M_θ orientations. Thus, the proposed descriptor becomes a vector of size $M_x \times M_z \times M_\theta$.

Every object flow instance $\xi_i^{2D} = (p_x^i, p_z^i, v_x^i, v_z^i)^T$ at position $\mathbf{p}_i$ casts one vote for its orientation $\mathbf{v}_i$ to each histogram $\mathbf{h}_j$ on the grid, located at $\mathbf{q}_j = (q_x^j, q_z^j)^T$.

The weight of each vote $w(\mathbf{q}, \xi^{2D})$ depends on the grid point $\mathbf{q}$ and the flow ξ^{2D} itself via

$$w(\mathbf{q}, \xi^{2D}) = \exp\left\{-(\mathbf{q}-\mathbf{p})^\mathbf{T}\mathbf{W}(\mathbf{q}-\mathbf{p})\right\} \in [0..1]\,. \tag{6}$$

Here $\mathbf{W} \in \mathbb{R}^{\mathbf{2\times 2}}$ is defined by its eigenvalue decomposition

$$\mathbf{W} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^\mathbf{T} \tag{7}$$

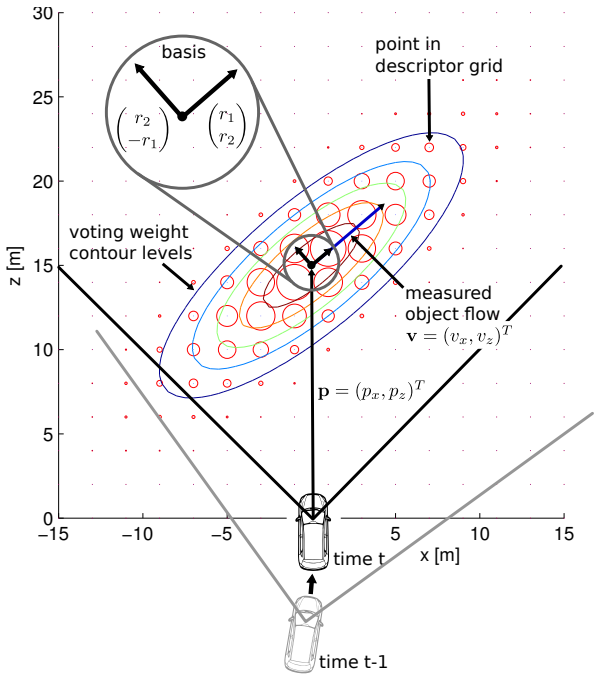


Fig. 4. **Illustration of the voting procedure.** This figure depicts the votes casted by an object flow vector at position $\mathbf{p} = (0, 15)^T$ on a grid of size $M_x = M_z = 16$. The size of the circles and the constant-level countours illustrate the weight of the vote each grid point receives for the orientation of the measured object flow.

with

$$\mathbf{R} = \begin{pmatrix} r_1 & r_2 \\ r_2 & -r_1 \end{pmatrix} \qquad \mathbf{\Lambda} = \begin{pmatrix} 1/\lambda_1^2 & 0 \\ 0 & 1/\lambda_2^2 \end{pmatrix}. \tag{8}$$

$\mathbf{\Lambda}$ and $\mathbf{W}$ are mathematically similar and $\mathbf{R}$ is a orthonormal change-of-basis matrix, defined by the orientation of the respective object flow ξ^{2D}:

$$r_1 = \frac{v_x}{\sqrt{v_x^2 + v_z^2}} \qquad r_2 = \frac{v_z}{\sqrt{v_x^2 + v_z^2}} \tag{9}$$

Here, the parameters λ_1 and λ_2 control the longitudinal and lateral influence of each flow vector.

The weight calculation step is illustrated in figure 4 for one single flow vector and a grid of size 16×16. After accumulating the votes from all object flow vectors, each of the $M_x \times M_z$ histograms $\mathbf{h}_j$ is individually normalized (i.e., divided) by $\max(10, \max(\mathbf{h}_j))$. Concatenating all histograms finally yields the object flow descriptor

$$\mathbf{d} = (\mathbf{h}_1, ..., \mathbf{h}_{M_x \times M_z})\,, \tag{10}$$

a vector of size $M_x \times M_z \times M_\theta$. Figure 2 (right) illustrates an object flow descriptor with $M_x = M_z = 16$, where the arrow directions represent $M_\theta = 8$ canonical histogram orientations and the arrow lengths depict the relative number of votes for each orientation.

V. Experiments

A qualitative assessment of the proposed method is given in figures 5,7. Figure 7 depicts in each row two frames of the sequence, the extracted object flow and the computed

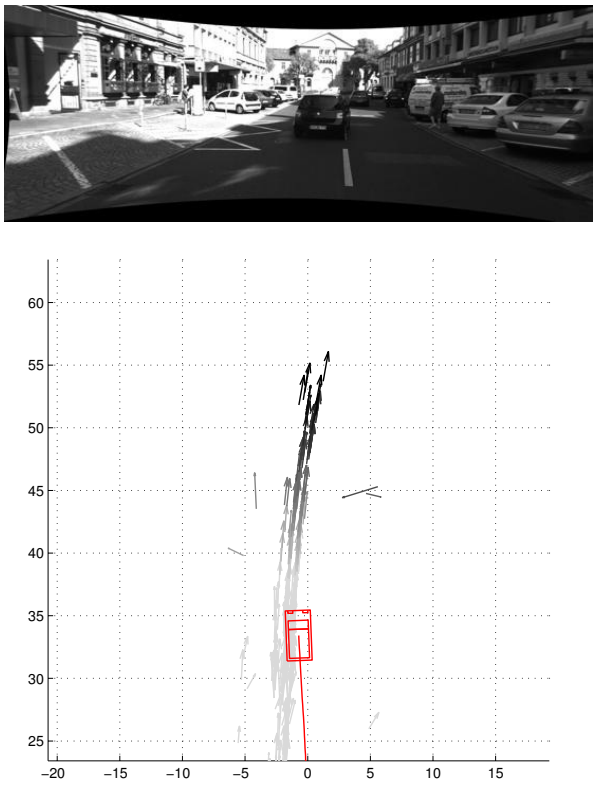
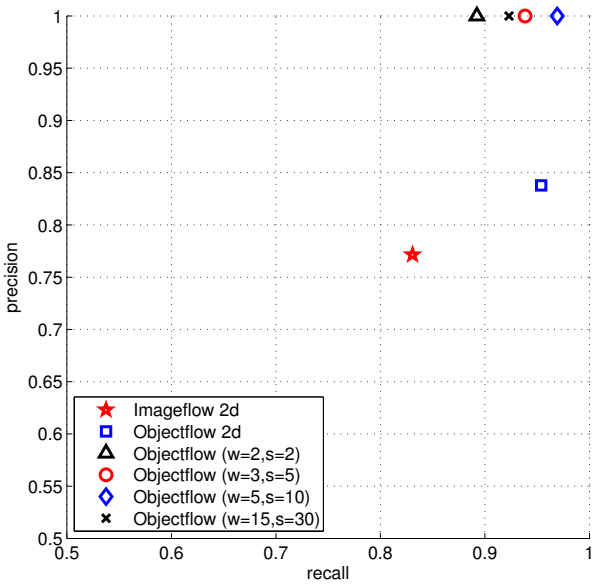


Fig. 5. **Object flow of a lane change maneuver.** This figure depicts the object flow field of a preceding car. While the first car changes lanes from the left to the right, the observer changes lanes in the opposite direction.



Method (w,s)	TP	TN	FP	FN	Precision	Recall
Img.Flow 2d	54	32	16	11	0.77	0.83
Obj.Flow 2d	62	36	12	3	0.84	0.95
Obj.Flow (2,2)	58	48	0	7	1.00	0.89
Obj.Flow (3,5)	61	48	0	4	1.00	0.94
Obj.Flow (5,10)	63	48	0	2	1.00	0.97
Obj.Flow (15,30)	60	48	0	5	1.00	0.92

Fig. 6. **Classification results.** This figure illustrates our classification results on 113 recorded sequences for both baselines (*ImageFlow 2d* and *ObjectFlow 2d*) and our method (*ObjectFlow*) with 4 parameter sets, varying the size of the exponential voting kernel w and the grid step size s.

descriptor using different scenarios. Note that not only a binary classification, but also regressing different intersection types should be possible and promises worthwhile avenues for future research. Figure 5 shows the egovehicle, its motion computed using the visual odometry approach described in section III and the object flow of a preceding vehicle. Both vehicles perform a lane change in the opposite direction of each other. While some outliers remain, most flow vectors are correct.

To evaluate the proposed descriptor quantitatively, we recorded 113 grayscale video sequences using our car-mounted stereo rig at a framerate of 10 fps, with 1382×512 pixels and an opening angle of $90\,^{\circ}$. Each sequence contained approximately 10 seconds, which was empircally found to be well suited for our method. We manually classified the sequences into 65 positive (intersections) and 48 negative (non-intersections) examples.

For comparison, two baselines were created: The first baseline, called *ImageFlow 2d* accumulates sparse optical flow vectors of the left camera in a single histogram with eight bins corresponding to eight canoncial directions in the image. The second baseline, called *ObjectFlow 2d* is essentially the same, except that the optical flow vectors in the image were compensated for the vehicles egomotion prior to histogram voting, using the technique described in section III. The idea behind both baseline algorithms is to capture the main optical flow direction in the image: A high horizontal optical flow ratio should be expected at intersections while non-intersection scenarios are supposed to produce mainly vertical flow vectors in the image. However, both baselines make no use of depth information, except for compensating the egomotion (*ObjectFlow 2d*).

To account for the high dimensionality of the *ObjectFlow* descriptor proposed in section IV, we extended the training set by shifting all examples by ± 2 meters in the x/z plane. Sure enough, we did not include the translated instances of the test data into the training set of the respective run for fairness reasons.

For classification we employed a soft-margin support vector machine (SVM), making use of the libsvm library [4]. A linear kernel was chosen to avoid overfitting due to the high-dimensional nature of the feature space ($M_x \times M_z \times M_\theta$). Since the number of recorded scenarios (113) is relatively small compared to the high-dimensional feature space, leave-one-out cross-validation was employed.

Figure 6 shows the result of our experiments in terms of true positives, false positives, true negatives, false negatives, precision and recall. Here a 'true positive' denotes a correctly detected intersection. We compare the two baselines to our method using four sets of histogram grid parameters. The parameter w adjusts the influence area of the exponential voting kernel from equation 8 via

$$\lambda_1 = 3w \qquad \lambda_2 = w \tag{11}$$

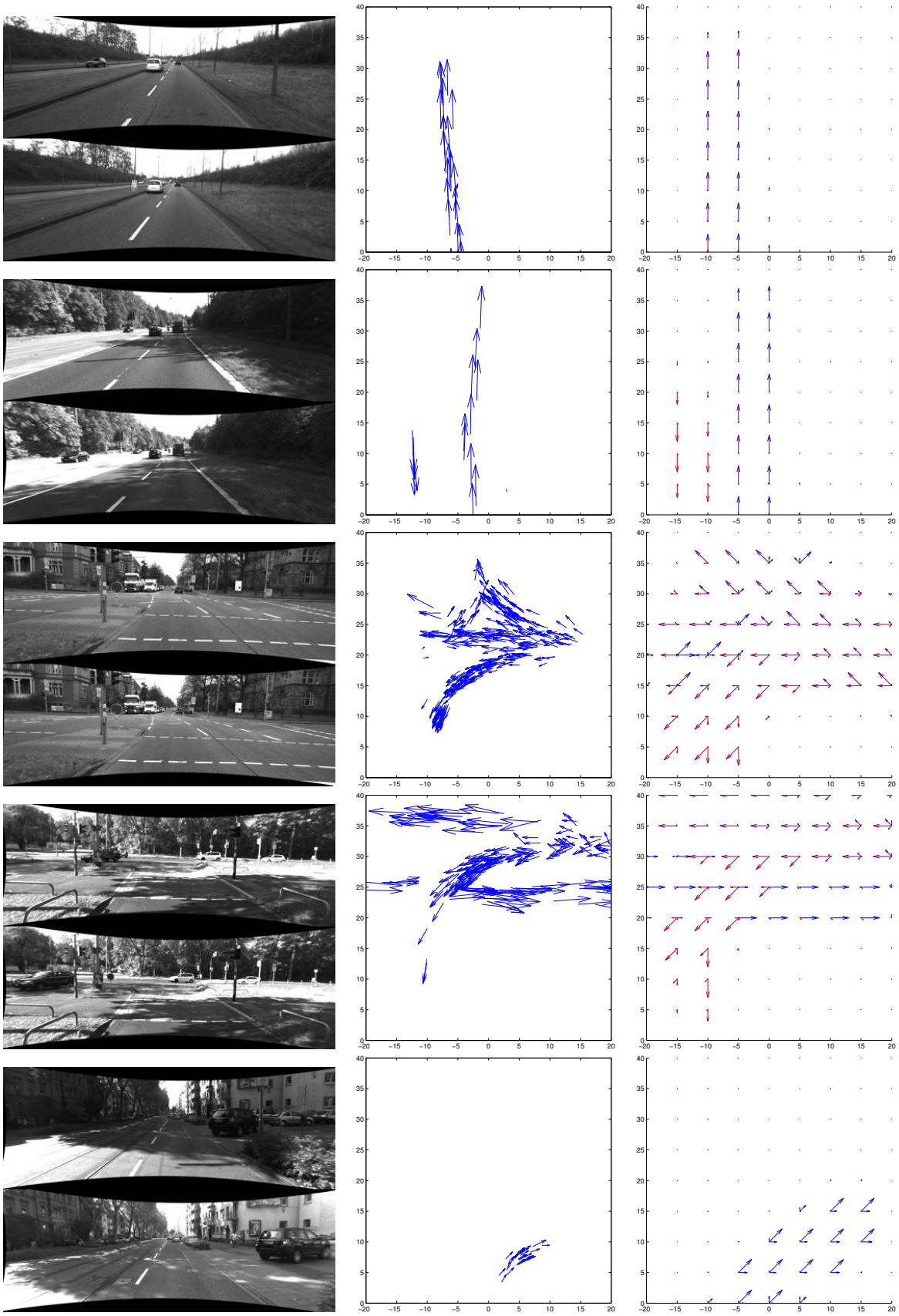


Fig. 7. **Results on 2 non-intersection and 3 intersection sequences.** Each row depicts one scenario: The left column shows two image frames of the left camera, the middle column displays the extracted object flow from a bird's eye perspective (unit: meter) and the right column depicts the histogram grid descriptor using a step size of $s = 5$ meters.

and s is the histogram grid step size, e.g. $s = 2$ in figure 4 and $s = 5$ in figure 7.

While the *ImageFlow 2d* and *ObjectFlow 2d* baselines perform quite well already, adding depth information clearly helps in discriminating intersection scenarios versus non-intersection scenarios. For all parameter sets of the *ObjectFlow* descriptor no false positive has been reported. The false negative rate differs slightly among the parameter settings with the lowest rate for $w = 5$ and $s = 10$. We believe that this 'averaging' parameter set is mainly preferred as a consequence of the relatively small amount of training examples. Still, all results clearly indicate the benefits of stereo-based measurements with a precision of 1.0 and a recall ≥ 0.89. Our system currently runs on a single CPU core at about 2 frames per second. Since the most computationally expensive part is feature matching, real-time can be achieved with a GPU or FPGA-based feature matcher.

VI. Conclusion and Future Work

In this paper a complementary novel descriptor for visually classifying traffic motion into intersection and non-intersection scenarios from within a moving vehicle has been proposed and evaluated against two baseline algorithms. Our findings indicate that including depth information into the classification process clearly outperforms classifiers with image based motion features. Though our results are promising, we intend to include more image sequences to the training data base in the future. Further, we plan to investigate other features for matching as well as combining the complementary object flow features with road based ones into local maps. Robustly fitting road topography models to this data promises valuable directions for future research towords advanced driver assistance systems and autonomous driving in difficult urban scenarios.

VII. Acknowledgments

We would like to thank the *Karlsruhe School of Optics and Photonics* and the *Deutsche Forschungsgemeinschaft* for supporting this work and the reviewers for their comments.

References

[1] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *ECCV (4)*, 2008, pp. 102–115.

[2] M. Aly, "Real time detection of lane markers in urban streets," *Intelligent Vehicles Symposium, 2008 IEEE*, pp. 7–12, June 2008.

[3] A. Broggi, "A massively parallel approach to real-time vision-based road markings detection," in *In Proceedings of the Intelligent Vehicles '95 Symposium*. IEEE Computer Society, 1995, pp. 84–89.

[4] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/cjlin/libsvm.

[5] R. Danescu, S. Nedevschi, M. M. Meinecke, and T. B. To, "Lane geometry estimation in urban environments using a stereovision system," *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pp. 271–276, 30 2007-Oct. 3 2007.

[6] R. Danescu and S. Nedevschi, "Probabilistic lane tracking in difficult road scenarios using stereovision," *Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 272–282, 2009.

[7] T. Dang, C. Hoffmann, and C. Stiller, "Continuous stereo self-calibration by camera parameter tracking," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1536–50, 2009.

[8] E. D. Dickmanns and B. D. Mysliwetz, "Recursive 3-d road and relative ego-state recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, no. 2, pp. 199–213, Feb 1992.

[9] W. Enkelmann, G. Struck, and J. Geisler, "Roma - a system for model-based analysis of road markings," *Intelligent Vehicles '95 Symposium., Proceedings of the*, pp. 356–360, Sep 1995.

[10] A. Ess, T. Mueller, H. Grabner, and L. van Gool, "Segmentation-based urban traffic scene understanding," in *BMVC*, 2009.

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

[12] A. Geiger, "Monocular road mosaicing for urban environments," in *Intelligent Vehicles Symposium, 2009 IEEE*, June 2009, pp. 140–145.

[13] V. Gengenbach, H. H. Nagel, F. Heimes, G. Struck, and H. Kollnig, "Model-based recognition of intersections and lane structures," *Intelligent Vehicles '95 Symposium., Proceedings of the*, pp. 512–517, Sep 1995.

[14] M. S. Grewal and A. P. Andrews, *Kalman Filtering Theory and Practice Using MATLAB*, 3rd ed. Wiley, 2008.

[15] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[16] F. Heimes, K. Fleischer, and H. H. Nagel, "Automatic generation of intersection models from digital maps for vision-based driving on inner city intersections," *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pp. 498–503, 2000.

[17] F. Heimes and H. H. Nagel, "Towards active machine-vision-based driver assistance for urban areas," *Int. J. Comput. Vision*, vol. 50, no. 1, pp. 5–34, 2002.

[18] A. S. Huang, D. Moore, M. Antone, E. Olson, and S. Teller, "Finding multiple lanes in urban road networks with vision and lidar," *Auton. Robots*, vol. 26, no. 2-3, pp. 103–122, 2009.

[19] B. Hummel, W. Thiemann, and I. Lulcheva, "Description logic for intersection understanding," *In Proc. Cognitive Systems with Interactive Sensors (COGIS), Stanford*, 2007.

[20] T. M. Jochem, D. A. Pomerleau, and C. E. Thorpe, "Vision based intersection navigation," in *Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE*, Sep 1996, pp. 391–396.

[21] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *IEEE Intelligent Vehicles Symposium*, 2010.

[22] D. Koller, Q. T. Luong, and J. Malik, "Using binocular stereopsis for vision-based vehicle control," *Intelligent Vehicles '94 Symposium, Proceedings of the*, pp. 237–242, Oct. 1994.

[23] M. Lutzeler and E. D. Dickmanns, "Ems-vision: recognition of intersections on unmarked road networks," 2000, pp. 302–307.

[24] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, no. 1, pp. 20–37, March 2006.

[25] O. Pink, F. Moosmann, and A. Bachmann, "Visual features for vehicle localization and ego-motion estimation," in *Intelligent Vehicles Symposium, 2009 IEEE*, June 2009, pp. 254–260.

[26] C. Rasmussen, "Road shape classification for detecting and negotiating intersections," *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pp. 422–427, June 2003.

[27] B. Southall and C. J. Taylor, "Stochastic road shape estimation," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, 2001, pp. 205–212.

[28] A. Takahashi and Y. Ninomiya, "Model-based lane recognition," *Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE*, pp. 201–206, Sep 1996.

[29] C. J. Taylor, J. Malik, and J. Weber, "A real-time approach to stereopsis and lane-finding," *Intelligent Vehicles Symposium, 1996., Proceedings of the 1996 IEEE*, pp. 207–212, Sep 1996.

[30] H. Veeraraghavan, O. Masoud, and N. P. Papanikolopoulos, "Computer vision algorithms for intersection monitoring," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 4, no. 2, pp. 78–89, June 2003.

[31] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial Intelligence*, vol. 78, no. 1-2, pp. 87–119, 1995.