

Chapter 2

Dense 3D Motion Field Estimation from a Moving Observer in Real Time

Clemens Rabe, Uwe Franke, and Reinhard Koch

Abstract In this chapter an approach for estimating the three-dimensional motion fields of real-world scenes is proposed. This approach combines state-of-the-art dense optical flow estimation, including spatial regularization, and dense stereo information using Kalman filters to achieve temporal smoothness and robustness. The result is a dense and accurate reconstruction of the three-dimensional motion field of the observed scene. An efficient parallel implementation using a GPU and an automotive compliant FPGA yields a real-time vision system which is directly applicable in real-world scenarios including driver assistance systems, robotics, and surveillance.

Keywords Computer vision • Driver assistance • Motion estimation

2.1 Introduction

Future driver assistance systems are expected to support the driver in complex driving situations. This requires a thorough understanding of the car's environment, which includes not only the perception of the infrastructure but also the precise detection and tracking of moving traffic participants.

The 3D structure of the scene can easily be obtained through a stereo camera system. To detect obstacles, this information is commonly accumulated in an evidence grid-like structure [13]. We refer to it as the bird's view map. Since stereo

C. Rabe (✉) • U. Franke
Research and Technology/Machine Perception, Daimler AG, Hanns-Klemm-Str. 45,
71059 Sindelfingen, Germany
e-mail: clemens.rabe@daimler.com; uwe.franke@daimler.com

R. Koch
Multimedia Information Processing, Christian-Albrechts-Universität zu Kiel,
Hermann-Rodewald-Str. 3, 24098 Kiel, Germany
e-mail: rk@mip.informatik.uni-kiel.de

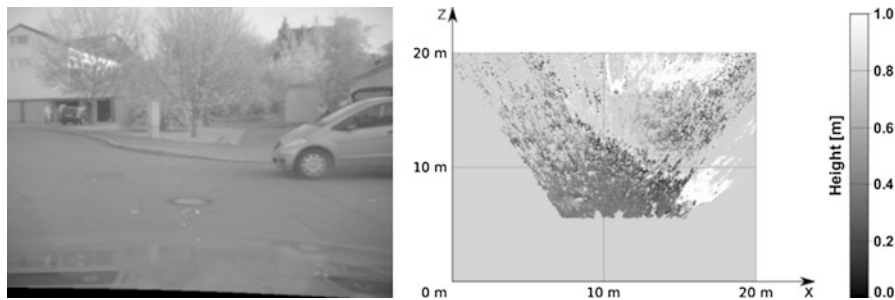


Fig. 2.1 *Left:* Typical traffic scene. The car is stationary, whereas the pedestrian runs towards the intersection. *Right:* Bird's eye view of the reconstructed 3D structure of the scene. The color encodes the height above ground (from black (0 m) to white (1.0 m)) of the observed points

does not reveal any motion information, usually this map is segmented and detected objects are tracked over time in order to obtain their motion state. The major disadvantage of this standard approach is that the performance of the detection depends highly on the correctness of the segmentation. In particular, moving objects in front of stationary ones, as illustrated in Fig. 2.1, are often bundled and therefore not detected. This causes erroneous misinterpretations and requires more powerful solutions.

In literature there are many examples using the optical flow, i.e., the apparent image motion, to detect moving objects. Argyros et al. describe a method to detect moving objects using stereo vision [1]. Comparing the normal flow of the right camera image with the normal flow between the left and the right images of the stereo cameras they detect image regions with independent object motion as inconsistencies in the flow data. Other approaches like Kellman and Kaiser [12], Mills [14], and Heinrich [8] use the geometric constraints stemming from the stereo configuration to detect independent motion. However, these approaches lack a precise measurement of the detected movements.

One of the first attempts to fuse stereo and optical flow information was studied by Waxman and Duncan [20]. They have analyzed the relationship between the optical flow fields of each camera and defined the so-called relative flow. Using this information the relative longitudinal velocity between the observer and the object is directly determined. Other approaches known as scene flow algorithms estimate the 3D motion field directly from two consecutive image pairs. The term scene flow was introduced by Vedula et al. in [19] and was defined as the three-dimensional motion of points in the world. In practice, scene flow algorithms estimate the optical flow and disparity change, optionally also the disparity map, in a combined approach. From this information, the 3D motion field is then reconstructed. Although computationally expensive, dense scene flow algorithms running in real time are available, as shown by Rannacher in [18].



Fig. 2.2 *Left:* Typical traffic scene. *Right:* Motion field, estimated by the Dense6D algorithm proposed in this chapter. The color encodes the velocity of the observed points

Although scene flow analysis provides fast detection results, it is limited with respect to robustness and accuracy due to the immanent measurement noise caused by its differential character. To get more reliable results, an integration of the observations over time is necessary. This was done in the 6D-Vision system that was first presented in [5]. The basic idea is to track points with depth known from stereo vision over two and more consecutive frames and then to fuse the spatial and temporal information using Kalman filters [11]. The result is an improved accuracy of the 3D position and an estimation of the 3D motion of the point under study at the same time. Since we get a rich 6D-state vector for each point this method is referred as 6D-Vision. Taking into account the motion information, the detection of moving objects can be carried out significantly easier and more robust than using bird view maps. In addition, using the 3D motion information a prediction of the object movement is possible. This allows a driver assistance system to warn and react to potential collisions in time.

The improved system presented in [16] is able to track up to 10,000 points in real time on modern hardware devices. However, in safety relevant applications, robustness, density, and volume of information are of utmost importance. Inspired by the recent progress of dense optical flow algorithms, Rabe et al. applied the 6D-Vision principle to dense optical flow and dense stereo data [17]. Implemented on the Graphics Processing Unit (GPU), this system is able to analyze 640×480 pixels at a frame rate of 20 Hz. A typical result for a traffic scene is shown in Fig. 2.2. Since the 3D motion information is estimated for nearly every pixel of the image, this system is called Dense6D.

In this contribution, the Dense6D system is described in more detail, focusing on the implementation aspects for a real-time system. The chapter is organized as follows: Section 2.2 describes the core elements of the system, namely, the dense stereo and dense optical flow algorithms, as well as the Kalman filter-based fusion to obtain the 3D motion estimates. In Sect. 2.3, the quantitative evaluation results on synthetic ground-truth image sequences and qualitative results on real-world sequences are presented, followed by the conclusion in Sect. 2.4.

2.2 Estimation of the 3D Motion Field

The Dense6D algorithm estimates a 3D motion field based on a dense disparity map and a dense optical flow field that can be computed by any stereo and optical flow algorithms, respectively. However, automotive applications demand robust and real-time capable algorithms. Therefore, we use two state-of-the-art algorithms meeting these requirements: the Semi-Global-Matching (SGM) stereo algorithm and the TV- L^1 optical flow algorithm. In the following, both algorithms are presented, followed by a detailed description of the Dense6D algorithm.

Throughout this chapter, we assume that the camera system is calibrated, and the captured images are preprocessed by a rectification module that performs a lens-correction and establishes a standard stereo configuration.

2.2.1 Dense Stereo

In [9] Hirschmüller presented an algorithm to obtain dense disparity maps by using mutual information as a cost function and minimizing the energy functional

$$\begin{aligned}
 E = & \sum_{x \in \Omega} C(x, d_x) \\
 & + \sum_{x \in \Omega} \sum_{y \in N_x} p_1 T[|d_y - d_x| = 1] \\
 & + \sum_{x \in \Omega} \sum_{y \in N_x} p_2 T[|d_y - d_x| > 1]
 \end{aligned} \tag{2.1}$$

with $C(x, d_x)$ being the matching cost function of the disparity d_x at the image position x of the image domain $\Omega = \{x\} \subset \mathbb{R}^2$, N_x the neighborhood of x , p_1 and p_2 the smoothness penalties, and $T[\cdot]$ a function returning 1 if the inner expression is true and 0 otherwise. The first term simply sums all matching costs and can be interpreted as the data term. The second and third terms act as smoothness terms: Small deviations of neighboring disparities are penalized by a penalty p_1 , and large deviations are penalized by the (constant) penalty p_2 . Since the penalty p_1 is smaller than the penalty p_2 , slanted or curved surfaces are preferred over disparity discontinuities.

Since a global solution to the energy minimization problem is computationally expensive, Hirschmüller proposed to resolve it approximately using dynamic programming, thus giving it the name SGM. The recursive scheme for the costs of the applied dynamic programming is defined as

$$L_r(x, d) = C(x, d) + \min \begin{cases} L_r(x - \mathbf{r}, d) \\ L_r(x - \mathbf{r}, d - 1) + p_1 \\ L_r(x - \mathbf{r}, d + 1) + p_1 \\ L_r(x - \mathbf{r}, i) + p_2 & i < d - 1 \\ L_r(x - \mathbf{r}, i) + p_2 & i > d + 1 \end{cases} - \min_k L_r(x - \mathbf{r}, k) \quad (2.2)$$

along a path defined by the step vector \mathbf{r} . The costs over multiple paths of different directions (horizontal, vertical, and diagonal) are accumulated, and the resulting disparity is then found as the one with the minimum accumulated cost. Since paths of different directions are used, the typical streaking effects known from stereo algorithms evaluating only one path can be removed almost completely.

The resulting disparity map is only pixel-discrete, because the costs are calculated for discrete disparity values only. To obtain sub-pixel accuracy, the costs near the obtained minimal disparity are taken and the refined disparity is then found at the minimum of the parabola passing through these points.

Although the original algorithm used mutual information to determine the costs of a disparity match, the above energy minimization scheme can be used with almost any matching score. In practice, the zero-mean sum of absolute differences (ZSAD) and the Census operator [23] proved to be most robust even in situations of large illumination changes and are also less computationally expensive compared to the mutual information measure.

Solution of the minimization problem still remains computationally heavy, and a straightforward implementation takes about 2 s to compute a VGA disparity map on a state-of-the-art computer. Since the calculation of the cost cube requires access to the predecessors, each path must be computed sequentially, rendering it unsuitable for massively parallel machines like GPUs. However, Gehrig et al. proposed an implementation on an FPGA, which is able to calculate the disparity map in about 30 ms [6]. The massive speedup was achieved by calculating the disparity map on a sub-sampled image of half the resolution (overview image) and then combining it with the disparity map calculated for a portion of the image computer at the full resolution (fine image). This strategy assumes that distant objects of interest mainly occur in the center of the image, which is typical for traffic scenes. The implemented engine allows the computation of 64 disparity steps at each level, which leads to a total disparity range of 128 pixels. The implementation supports the ZSAD and Census matching costs, and the sub-pixel refinement is performed using an equi-angular fit. The sub-pixel accuracy of the engine is 1/16 pixel due to the use of fixed-point arithmetic. To remove mismatches, especially for partially occluded pixels, a right-left verification step is performed additionally. Using a CPU implementation with similar optimizations, the algorithm runs at 14 Hz [7]. The results for a typical traffic scene are displayed in Fig. 2.3.



Fig. 2.3 *Left: Traffic scene. Right: Three-dimensional visualization of the corresponding scene*

2.2.2 Dense Optical Flow

Dense optical flow algorithms calculate an optical flow vector for every pixel in the reference image by introducing a regularization term and formulating the problem as an energy minimization problem. Inspired by the seminal work of Horn and Schunck [10], a diverse range of such techniques has been developed. For a detailed review, the reader is referred to the surveys [2, 4, 22].

The method presented by Horn and Schunck solves the aperture problem by introducing a smoothness constraint, with the assumption that nearby optical flow vectors are similar in direction and magnitude. The optical flow field $\mathbf{u} = (u_x, u_y)^T : \Omega \rightarrow \mathbb{R}^2$ is found by minimizing the energy function

$$E = \int_{\Omega} \left\{ \lambda |\rho(\mathbf{x}, \mathbf{u}(\mathbf{x}))|^n + \sum_{i=x,y} |\nabla u_i(\mathbf{x})|^n \right\} d\mathbf{x} \quad (2.3)$$

with $n = 2$. The parameter λ defines the weight of the data term with respect to the regularization term. The data term $\rho(\mathbf{x}, \mathbf{u})$ is the constant brightness assumption, which is defined as

$$\rho(\mathbf{x}, \mathbf{u}) = I_1(\mathbf{x} + \mathbf{u}) - I_0(\mathbf{x}) \quad (2.4)$$

with $I_{\{0,1\}} : \Omega \rightarrow \mathbb{R}^2$ as the intensity function of the previous and current images, respectively.

The resulting optical flow field yields very encouraging results in regions of constant optical flow. However, due to the quadratic penalization in the smoothness term, the algorithm tends to over-smooth the optical flow field at flow boundaries and also over-weights the outliers. Therefore, Zach et al. presented in [24] a computational method to solve the energy function for the case $n = 1$, with

significantly improved results. However, in automotive scenarios the constant brightness assumption is often violated, e.g., shadow casts or changes of the exposure time, that results in incorrect optical flow estimates (Fig. 2.4b). Wedel et al. has proposed a structure–texture decomposition of the input images to overcome this problem [21]. Excellent results were obtained for image pairs containing only small displacements. However, this decomposition removes vital image information essential in estimating large displacements correctly (Fig. 2.4c).

Instead of a computationally expensive preprocessing step, we use a modification proposed by Müller et al. in [15]. The key idea here is to replace the original data term based on the constant brightness assumption with a robust data term $\rho_r(\mathbf{x}, \mathbf{u})$ based on the Census operator:

$$\rho_r(\mathbf{x}, \mathbf{u}) = h(c_1(\mathbf{x} + \mathbf{u}), c_0(\mathbf{x})) \quad (2.5)$$

Here, $c_{\{0,1\}}(\mathbf{x})$ gives the Census signature around the center pixel \mathbf{x} in the previous and current images, respectively. $h(c_1, c_0)$ denotes the Hamming distance of the two signatures. To solve the energy minimization problem, a variation of the framework of Zach et al. is utilized.

Implemented on the GPU, this version runs in real time (25 Hz) with a Census window of 3×3 pixels. Due to the reduced entropy caused by the Census operator, the subpixel accuracy is slightly worse compared to the original version on ideal images, but the increased robustness against illumination changes outweighs this fact easily (Fig. 2.4d).

In practice, the algorithm is implemented using an image resolution pyramid of five levels to estimate large displacement vectors, and 25 iterations are performed on each pyramid level. In each iteration step, the data term is firstly minimized using a first-order Taylor approximation, followed by a median filter and the smoothing step.

2.2.3 Temporal Integration of the Motion Field

Having established a correspondence over time for an observed image point by an optical flow or feature tracking algorithm, the 3D motion can be calculated directly from the reconstructed 3D points and the known time interval. However, such techniques suffer heavily from the immanent measurement noise, and the results are not robust. Therefore, we use a Kalman filter to estimate the 3D position and 3D motion of a point [5, 16, 17]. Due to the recursive nature of Kalman filters, the estimation keeps improving continuously with each measurement and by updating the state vector and its associated covariance matrix. This eliminates the need to save a history of measurements and is computationally very efficient. Additionally, Kalman filters take advantage of measurement uncertainties which can be considered when the flow field is evaluated for subsequent applications.

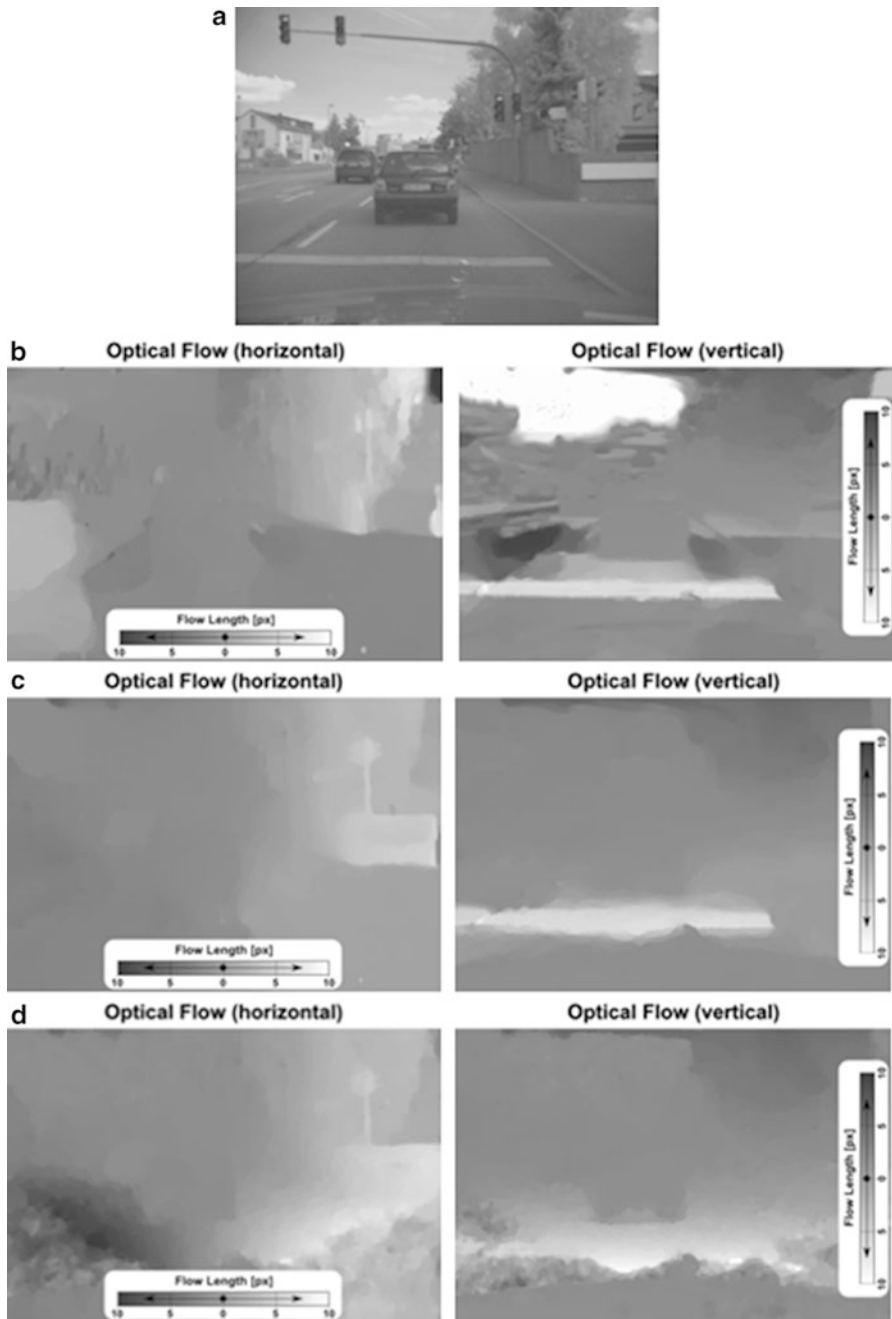


Fig. 2.4 Estimated optical flow fields under strong illumination changes. **(a)** Original image. **(b)** Without compensation. **(c)** With structure-texture decomposition. **(d)** Modified data term based on the Census operator

Using a stereo camera system, the 3D structure of the observed scene is readily reconstructed by the stereo algorithm. Here, the left image point $\mathbf{x} = (x, y)^T$ is the projection of a world point $\mathbf{X} = (X, Y, Z)^T$. Expressed in homogeneous coordinates, this relation is given by

$$\begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} \simeq \mathbf{\Pi} \cdot \tilde{\mathbf{X}} \quad (2.6)$$

with the positive disparity $d \equiv d(x)$, related to the left image. The extended projection matrix $\mathbf{\Pi}$ is written as

$$\mathbf{\Pi} = \begin{pmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 0 & b \cdot f_x \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{R}_c & \mathbf{t}_c \\ \mathbf{0} & 1 \end{pmatrix} \quad (2.7)$$

with f_x and f_y as the focal lengths in pixel, $(c_x, c_y)^T$ as the principal point in pixels, and b as the base width of the stereo camera system. The rotation matrix \mathbf{R}_c and the translation vector \mathbf{t}_c describe the extrinsic orientation of the camera system to the world and to the car coordinate system, respectively. To determine the 3D position for an observed image point \mathbf{x} with known disparity d , (2.6) has to be inverted.

The state vector of the Kalman filter is defined as $\xi = (X, Y, Z, \dot{X}, \dot{Y}, \dot{Z})$, the combination of the 3D position and the 3D velocity vector. The system model describes the propagation of the state vector ξ at the previous time step $t - 1$ to the current time t , assuming a linear motion, which is described by the linear equation

$$\xi_t = \begin{pmatrix} \mathbf{R}_e & \Delta t \cdot \mathbf{R}_e \\ \mathbf{0} & \mathbf{R}_e \end{pmatrix} \xi_{t-1} + \begin{pmatrix} \mathbf{t}_e \\ \mathbf{0} \end{pmatrix} \quad (2.8)$$

with \mathbf{R}_e and \mathbf{t}_e denoting the rotation and the translation component of the inverse motion of the observer, also called the ego-motion. Δt is the time between any two time steps.

The measurement model of the Kalman filter describes the relation between the measurement vector $\mathbf{z} = (x, y, d)^T$, consisting of the current image position and the disparity of the analyzed world point, and the state vector ξ . Here, only the position components of the state vector are directly measured, and the relation between the measured projection and the reconstructed 3D point is given by (2.6). Since the measurement model must be formulated in Euclidean space rather than in projective space, the measurement model is nonlinear:

$$\tilde{\mathbf{z}} = w \begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} = \mathbf{\Pi} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.9)$$

$$\mathbf{z} = \frac{1}{w} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \tilde{\mathbf{z}} \quad (2.10)$$

The current measurement vector \mathbf{z}_t is given as

$$\mathbf{z}_t = (x_t, d_t)^T, \quad x_t = x_{t-1} + u_t \quad (2.11)$$

with u_t denoting the optical flow related to the previous position x_{t-1} of the image point and the corresponding disparity d_t at the new image position x_t . It is worth noting that x_{t-1} in (2.11) depicts the old measured image position at the previous frame, not the projection of the filtered state ξ_{t-1} . That means the image position of the features is only determined by the optical flow algorithm, while the filtering only influences the velocity and the disparity estimation. This way, undesired low-pass filtering effects of the Kalman filter are avoided.

To estimate the 3D motion field for all pixels, we associate every pixel $\hat{\mathbf{x}}$ on the discrete pixel grid with one Kalman filter $\mathcal{K}(x_t)$. In addition to the internal state of the Kalman filter, the image position x is stored with subpixel accuracy in the same data structure. At each time step, the prediction step of all Kalman filters is performed first. Here, we use the inertial sensors of the vehicle—the speed and yaw rate information—to calculate the ego-motion required for the state transition. For the optical flow field u_t , the Kalman filter field $\mathcal{K}_{t-1}(\hat{\mathbf{x}}_{t-1})$ is warped to the filter field $\mathcal{K}_{t-1}(\hat{\mathbf{x}}_t)$, with $\hat{\mathbf{x}}_t$ as the new pixel discrete image position, which is derived from x_t , i.e., calculated according to (2.11). After this resampling step, the Kalman filters are updated according to the given measurement model, resulting in the updated Kalman filter field $\mathcal{K}_t(x_t)$.

During the resampling step it is possible that not every pixel x_t of the current image is referred by a flow vector u_t . In this case, a new filter has to be created with initial values and connected to the empty pixel. An initialization based on the states and the covariances of the surrounding pixels is beneficial.

If one pixel x_t of the current image is referred by more than one flow vector set u_t , one either has to decide which one of the filters to use with the corresponding pixel on the next frame or has to combine them into a new one. In this case, the covariances of the concurring filters can be used as weights to generate the new filter state. It is also reasonable to use the depth information so that the nearest filter survives, while the others are reset. In our implementation, the filter which is assigned as the last one remains alive. A more complex solution decreases the real-time capability significantly and thus outweighs the benefit in practice.

To achieve maximum performance, this Dense6D system is implemented on a GPU. The Kalman filter code is based on an implementation according to Bierman [3] and was automatically generated by a code generator that exploits the sparse

structure of our matrices. The overall system achieves real time (20 Hz) on VGA images and consists of the following phases: image acquisition, rectification, stereo computation on an FPGA, optical flow calculation and Kalman filter calculation on the GPU, and visualization using a 3D viewer.

2.3 Evaluation

2.3.1 Evaluation on Ground Truth Sequences

In the first experimental phase, we study our vision system on a synthetic stereo image sequence of 250 frames rendered with the ray tracing software Povray. The images have a resolution of 640×480 pixels with an intensity resolution of 12 bits. In our test sequence (Fig. 2.5), the camera moves through an artificial traffic scene containing crossing and turning vehicles.

Figure 2.6a shows the ground truth motion field for a single image of this sequence. The vectors point from the current position of the world point to the position in 0.250 s. The velocity is in gray scale: white corresponding to 0.0 m/s, whereas black encodes a velocity of 8.0 m/s. Figure 2.6b shows the result of a direct combination of the stereo information and the optical flow. It is obvious that this naive approach performs very poorly due to the temporal noise of the depth estimation. Also, it should be noted that the prediction horizon had to be reduced to 0.050 s since the calculated motion vectors are extremely noisy.

The result from the proposed Dense6D approach is shown in Fig. 2.6c. Here, the prediction horizon is again 0.250 s. As it can be seen, the static points on the street are correctly estimated, and the moving vehicles are clearly visible. The estimation of the motion fields of the preceding and the crossing car is quite accurate. Only the motion of the turning vehicle on the right seems to be underestimated. This is primarily caused by the violation of the linear motion model and the integrated



Fig. 2.5 Stereo image pair of the ground truth stereo sequence used in this evaluation

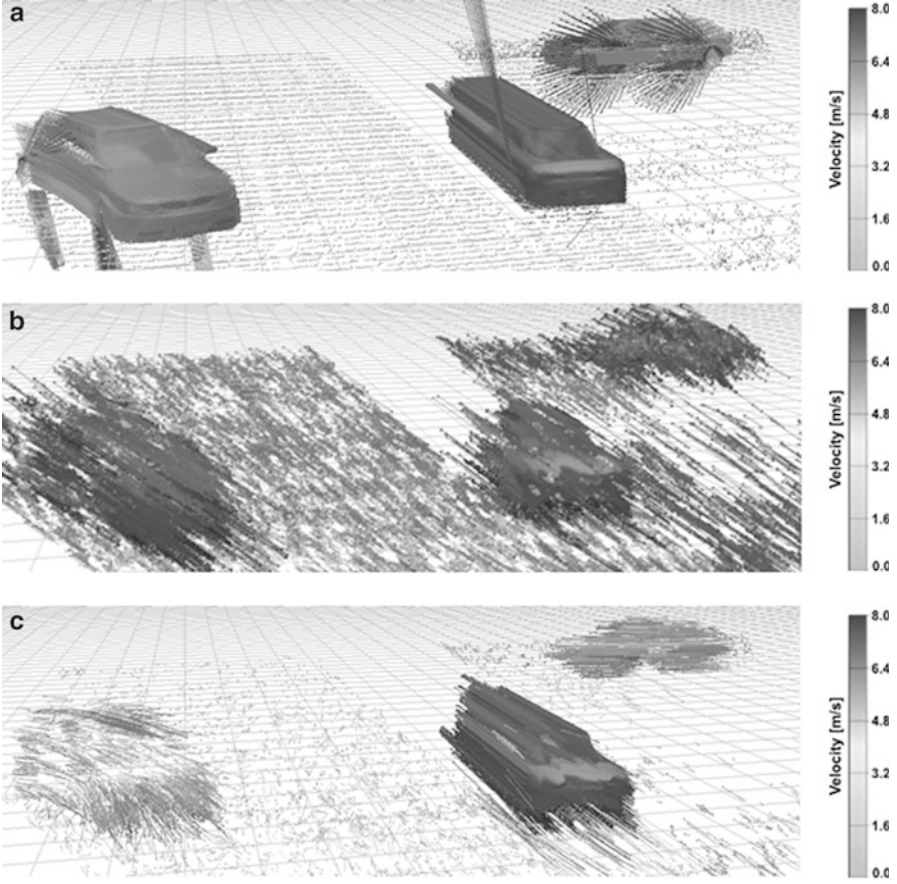


Fig. 2.6 Estimated motion field of the described methods. (a) Ground truth. (b) Direct combination of optical flow and stereo. (c) Dense6D. The vectors point to the predicted 3D position in 0.250 s (a) and (c) respectively 0.050 s (b)

outlier detection method (3σ -test). All estimates violating the motion model are rejected, which results in constantly re-initialized filters for the turning car. Hence, the visible motion vectors of the turning car correspond mainly to filters that have not yet reached their steady states.

The three-dimensional ground truth position and the motion field are used for the calculation of the error distributions $\rho[\chi]$ with $\chi = Z, \dot{X}, \dot{Y}, \dot{Z}$ as the quantities to analyze and χ^* as the corresponding ground truth. The error distributions, accumulated over the whole image Ω and the whole sequence $[0, T]$, are shown in Fig. 2.7. Again, the superiority of the Dense6D system compared to the direct approach is clearly visible.

In addition, the median of the error distribution of χ (ME) and the root mean squared error (RMS).

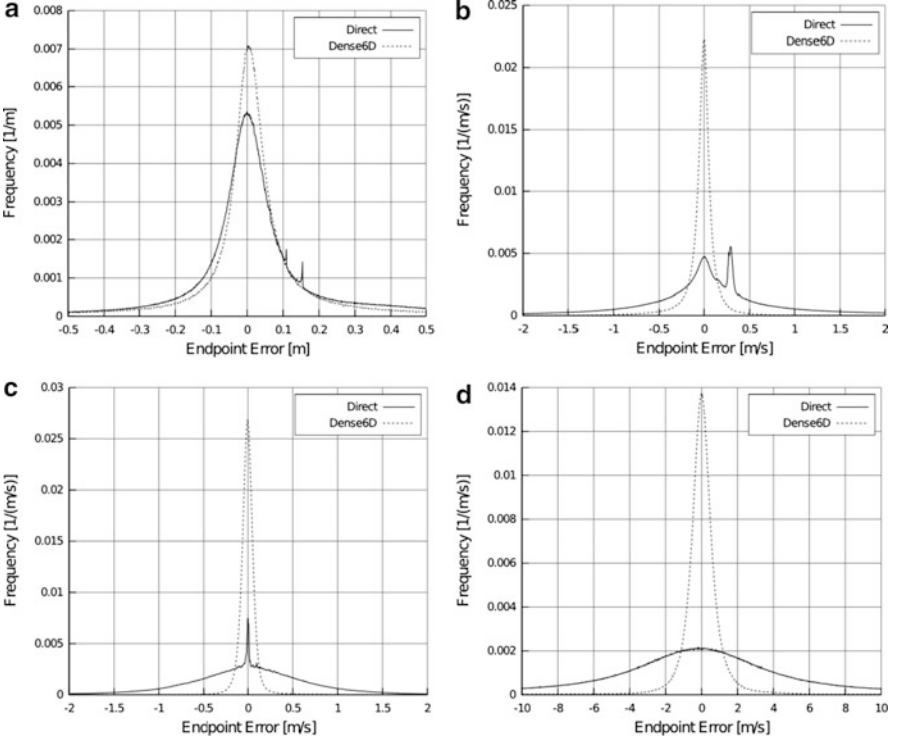


Fig. 2.7 Error distributions of the Z-position and the velocity components calculated from the direct combination of optical flow and stereo (*solid*) and the Dense6D method (*dashed*). (a) Position component Z. (b) Velocity component Z. (c) Position component γ . (d) Velocity component γ

$$E_{\text{RMS}} = \sqrt{\frac{1}{N} \sum_{t=0}^T \int [\chi_t(\vec{x}) - \chi_t^*(\vec{x})]^2 d\vec{x}} \quad (2.12)$$

are computed and given for the individual components in Table 2.1.

2.3.2 Real-World Results

The Dense6D system is deployed in our instrumented research car to evaluate its performance in real-world scenarios. Figure 2.8 shows the estimated motion field for a turning vehicle at a distance of about 30 m. Here, the observer was moving at a speed of about 3 m/s. Besides the turning car, a pedestrian and the shopping cart of the pedestrian behind are visible. In Fig. 2.9 multiple moving pedestrians are visible. The observer was again moving at a speed of about 3 m/s while performing

Table 2.1 Median error (ME) and root mean square error (RMS) of the Z-position and the velocity components calculated by a direct combination of the optical flow and the stereo information (direct) and the proposed Dense6D algorithm

	Z (m)		\dot{X} (m/s)		\dot{Y} (m/s)		\dot{Z} (m/s)	
	ME	RMS	ME	RMS	ME	RMS	ME	RMS
Direct	0.0010	2.749	0.0462	42.0093	0.0004	15.370	0.4374	141.442
Dense6D	0.0104	1.068	-0.0065	0.3623	-0.0044	0.339	0.0107	2.538



Fig. 2.8 *Left:* Typical traffic scene. *Right:* Corresponding 3D motion field, estimated by the Dense6D algorithm proposed in this chapter

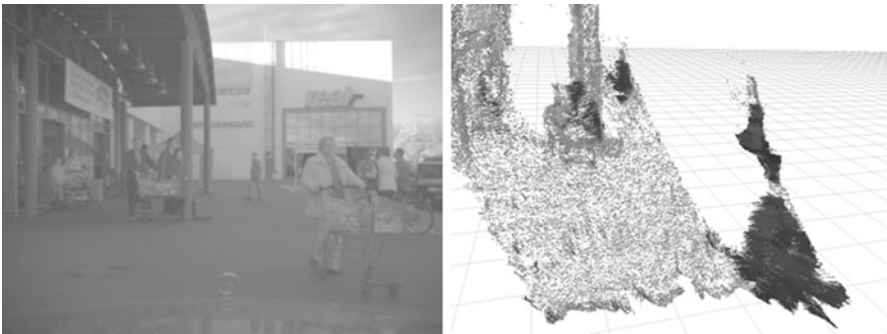


Fig. 2.9 *Left:* Typical traffic scene. *Right:* Corresponding 3D motion field, estimated by the Dense6D algorithm proposed in this chapter

a strong turning maneuver. Since the Kalman filters estimate the motion with respect to the fixed world coordinate system, the motion induced by the moving observer is completely compensated.

Obviously, the proposed Dense6D method is directly applicable in real-world scenarios and the excellent results on synthetic sequences shown in the previous section are validated.

Future work will include a multi-filter implementation and an image-based ego-motion estimation on the GPU as already known from [16]. In addition, methods to estimate the uncertainties of the stereo and optical flow information are currently under investigation.

2.4 Conclusions

In this chapter, we have presented the Dense6D system for dense, robust, accurate motion field estimation operating in real time. We have combined the state-of-the-art dense stereo and variational optical flow estimation techniques with Kalman filters under a linear motion model assumption. Evaluation of the relevant error quantities compared to simulated ground truth data shows that this approach shows far better results in real time compared to what is known in the literature so far. In real-world scenarios the technique shows its potential rather well. The Dense6D system is currently implemented in an instrumented research vehicle and is anticipated to become a key feature in emerging driver assistance systems.

Future work will include a multi-filter implementation and an image-based ego-motion estimation on the GPU as already known from [16]. In addition, methods to estimate the uncertainties of the stereo and optical flow information are currently under investigation.

References

1. A.A. Argyros, M.I. Lourakis, P.E. Trahanias, S.C. Orphanoudakis, Qualitative detection of 3D motion discontinuities, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'96)*, vol. 3, pp. 1630–1637, Nov 1996
2. S.S. Beauchemin, J.L. Barron, The computation of optical flow. *ACM Comput. Surv.* **27**, 433–466 (1995)
3. G.J. Bierman, *Factorization Methods for Discrete Sequential Estimation* (Academic Press, Inc., New York, 1977)
4. D.J. Fleet, Y. Weiss, Optical flow estimation, Chapter 15, in *Handbook of Mathematical Models in Computer Vision*, ed. by N. Paragios, Y. Chen, O. Faugeras (Springer, Berlin, 2006), pp. 239–258
5. U. Franke, C. Rabe, H. Badino, S. Gehrig, 6DVision: fusion of stereo and motion for robust environment perception, in *Proceedings of the 27th DAGM Symposium*, pp. 216–223, 2005
6. S. Gehrig, F. Eberli, T. Meyer, A real-time low-power stereo vision engine using semi-global matching, in *Proceedings of the 7th International Conference on Computer Vision Systems*, Liège, Belgium, Oct 2009
7. S.K. Gehrig, C. Rabe, Real-time semi-global matching on the CPU, in *Proceedings of the IEEE Workshop on Embedded Computer Vision*, 2010
8. S. Heinrich, Fast obstacle detection using flow/depth constraint, in *Proceedings of the IEEE Intelligent Vehicles Symposium 2002*, vol. 2, pp. 658–665, Jun 2002

9. H. Hirschmüller, Accurate and efficient stereo processing by semi-global matching and mutual information, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, no. 2, San Diego, CA, USA, pp. 807–814, Jun 2005
10. B.K.P. Horn, B.G. Schunck, Determining optical flow. *Artif. Intell.* **17**, 185–203 (1981)
11. R.E. Kalman, A new approach to linear filtering and prediction problems. *Trans. ASME J. Basic Eng.* **82**(Series D), 35–45 (1960)
12. P.J. Kellman, M.K. Kaiser, Extracting object motion during observer motion: combining constraints from optic flow and binocular disparity. *J. Opt. Soc. Am. A* **12**, 623–625 (1995)
13. M.C. Martin, H. Moravec, Robot evidence grids. Tech. Rep. CMU-RI-TR-96-06, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Mar 1996
14. S. Mills, Stereo-motion analysis of image sequences, in *Proceedings of the first joint Australia and New Zealand conference on Digital Image and Vision Computing: Techniques and Applications, DICTA'97/IVCNZ'97*, Dec 1997
15. T. Müller, C. Rabe, J. Rannacher, U. Franke, R. Mester, Illumination robust dense optical flow using census signatures, in *Proceedings of the 33th DAGM Symposium*, 2011
16. C. Rabe, U. Franke, S. Gehrig, Fast detection of moving objects in complex scenarios, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 398–403, Jun 2007
17. C. Rabe, T. Müller, A. Wedel, U. Franke, Dense, robust, and accurate motion field estimation from stereo image sequences in real-time, in *Proceedings of the 11th European Conference on Computer Vision*, ed. by K. Daniilidis, P. Maragos, N. Paragios. Lecture Notes in Computer Science, vol. 6314 (Springer, Berlin, 2010), pp. 582–595
18. J. Rannacher, Realtime 3D motion estimation on graphics hardware, Bachelor thesis, Heidelberg University, 2009
19. S. Vedula, S. Baker, P. Rander, R. Collins, T. Kanade, Three-dimensional scene flow, in *Seventh International Conference on Computer Vision (ICCV'99)*, vol. 2, pp. 722–729, 1999
20. A.M. Waxman, J.H. Duncan, Binocular image flows: steps toward stereo-motion fusion. *IEEE Trans. Pattern Anal. Mach. Intell.* **8**(6), 715–729 (1986)
21. A. Wedel, T. Pock, C. Zach, H. Bischof, D. Cremers, Statistical and geometrical approaches to visual motion analysis, in *An Improved Algorithm for TV-L1 Optical Flow*, ed. by D. Cremers, B. Rosenhahn, A.L. Yuille, F.R. Schmidt (Springer, Berlin, 2009), pp. 23–45
22. J. Weickert, A. Bruhn, T. Brox, N. Papenberg, A survey on variational optic flow methods for small displacements, in *Mathematical Models for Registration and Applications to Medical Imaging*, ed. by O. Scherzer (Springer, New York, 2006)
23. R. Zabih, J. Woodfill, Non-parametric local transforms for computing visual correspondence, in *Proceedings of the Third European Conference on Computer Vision*, pp. 151–158, May 1994
24. C. Zach, T. Pock, H. Bischof, A duality based approach for realtime TV-L1 optical flow, in *Proceedings of the 29th DAGM Symposium on Pattern Recognition*, pp. 214–223, 2007

Smart Mobile In-Vehicle Systems

Next Generation Advancements

Schmidt, G.; Abut, H.; Takeda, K.; Hansen, J.H.L. (Eds.)

2014, XX, 292 p. 165 illus., 79 illus. in color., Hardcover

ISBN: 978-1-4614-9119-4