Approximated Ground Truth for Stereo and Motion Analysis on Real-World Sequences

Zhifeng Liu and Reinhard Klette

The *.enpeda..* Project, The University of Auckland
Auckland, New Zealand

Abstract. This paper approximates ground truth for real-world stereo sequences and demonstrates its use for the performance analysis of a few selected stereo matching and optical flow techniques. Basically we assume zero roll and constant tilt of an ego-vehicle (for about 10 seconds) driving on a planar road.

Key words: performance evaluation, stereo analysis, motion analysis, real-world sequences, driver assistance

1 Introduction

Stereo matching and motion analysis are two very active research areas in computer vision. The quantitative evaluation of algorithms, as offered by the Middlebury stereo website[1] and their optical flow website[2] are often cited today as the state-of-the art of current performance evaluation of stereo or optical flow algorithms. Those evaluations have contributed considerably to significant advances in algorithm performance. However, evaluations on those Middlebury websites are based on high quality and high resolution color images, which are also only short sequences (of only a few frames) that are recorded in a lab environment, typically carefully designed to test for particular features. The challenges for stereo and optical flow algorithm in today's real world applications surpass such designed datasets and evaluation methods.

This paper deals with real-world stereo sequences. We refer to Set 1 (provided by Daimler AG) of the *.enpeda..* sequences[3], as described in [4]. These seven stereo sequences are taken with two Bosch (12-bit, gray-value) night vision cameras. Each sequence contains 250 or 300 frames (640×481), and features different driving environments, including highway (see Figure 1), urban road and rural area. Camera calibration is used for geometric rectification, such that image pairs are characterized by standard epipolar geometry as specified in [3].

[1] `http://vision.middlebury.edu/stereo/`
[2] `http://vision.middlebury.edu/flow/`
[3] `http://www.mi.auckland.ac.nz/` and follow the *data* link

Fig. 1. Sample of a stereo pair (of Sequence 1).

Intrinsic camera parameters and extrinsic calibration parameters for left and right camera (also in relation to the car) are provided. The vehicle's movement status is also given for each frame. We discuss a way to extract ground truth from these sequences.

2 Quality Metrics

To evaluate the performance of a stereo or optical flow algorithm and understand how their parameters affect results, we need a quantitative way to measure the quality of calculated stereo correspondences or motion vectors.

2.1 Stereo

The general approach of stereo evaluation is to compute error statistics based on given ground truth. (Note that any ground truth comes with some measurement error; ground truth is not truth.) We use the same error measurements as on the Middlebury stereo website, namely the *root mean squared error* between the disparity map $d(x, y)$ and the ground truth map $d_T(x, y)$, defined as follows:

$$E_R = (\frac{1}{n} \sum |d(x, y) - d_T(x, y)|^2)^{\frac{1}{2}} \tag{1}$$

where n is the total number of pixels, and the percentage of *bad matching pixels*, defined as follows:

$$E_B = \frac{1}{n} \sum (|d(x, y) - d_T(x, y)| > \delta_d) \tag{2}$$

where δ_d is the threshold of disparity tolerance.

2.2 Optical Flow

Quality metrics for optical flow evaluation have to measure the result in a 2D space. We use the common *angular error* defined as the average angle between estimated optical flow vector $\mathbf{u}$ and the true flow vector $\mathbf{u}_T$,

$$E_{AE} = \frac{1}{n} \sum \arccos\left(\frac{\mathbf{u} \cdot \mathbf{u}_T}{|\mathbf{u}||\mathbf{u}_T|}\right) \tag{3}$$

where $|\mathbf{u}|$ denotes the length (magnitude) of a vector, and the *end point error* which measures the absolute distance between end points of vectors $\mathbf{u}$ and $\mathbf{u}_T$,

$$E_{EP} = \sqrt{(u - u_T)^2 + (v - v_T)^2} \tag{4}$$

3 Approximate Ground Truth

We approximate ground truth with respect to an assumed (!) planar road surface, using known parameters of ego-vehicle and cameras (as saved in the *camera.dat* file and in the file header of every frame; see [4]).

3.1 Disparities on Road Surface

We consider the test sequences to be ego-motion compensated, which means that the horizon is always parallel with the row direction in the images. We conclude that pixels on the same image row have the same depth value if a projection of the planar road surface.

A side-view of the camera setting is shown in Figure 2, where θ is the known tilt angle, P is a road surface point which is projected into $p = (x_p, y_p)$ on the image plane, H is the height of the camera. It follows that

$$Z = d_e(OP_c) = d_e(OP)\cos\psi = \frac{H}{\sin(\theta + \psi)}\cos\psi \tag{5}$$

According to the stereo projection equations, the disparity d can be written as

$$d = \frac{b \cdot f}{Z} = \frac{b \cdot f}{\frac{H}{\sin(\theta+\psi)}\cos\psi} \tag{6}$$

where angle ψ can be calculated as follows, using focal length f and pixel coordinate y_p in the image:

$$\psi = \arctan\left(\frac{(y_p - y_0)s_y}{f}\right) \tag{7}$$

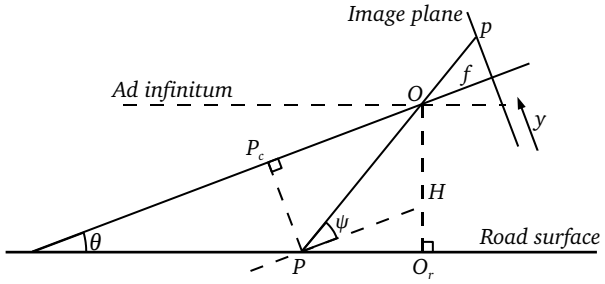

Fig. 2. Projection of a point P of the road surface.

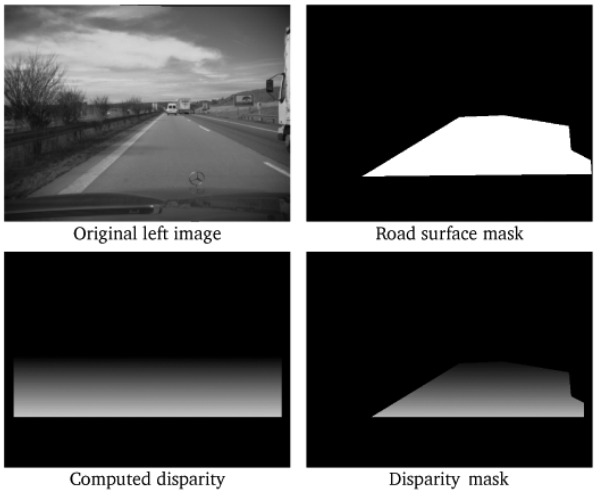

Fig. 3. Generation of a disparity mask: input image, manually generated mask, depth map of a planar road, and resulting disparity mask.

Here, y_0 is the y-coordinate of the principal point, and s_y is the pixel size in y-direction. We can also compute the y-coordinate of a line that projects to infinity

$$y_{inf} = \frac{y_0 - f \cdot \tan\theta}{s_y}$$

This is the upper limit of the road surface, and points on it should have zero disparity (if no objects block the view).

Figure 3 illustrates the process of generating an approximated disparity map on road surface areas, also using manual input for a conservative outline of the road area in a given image. In the given camera setting (of the seven sequences), there is a yaw angle (0.01 radian) which makes the cameras looking a little bit to the left. This angle can be ignored because it only defines the right camera to be about 3 mm behind the left camera.

3.2 Local Displacements of Road Surface

Speed and direction (yaw rate) of the ego-vehicle are given for all frames of those seven sequences. The road is, obviously, static, what makes the calculation of relative movement of road surface points (with respect to the camera) straight forward.

Given a pixel p on the image plane at time t, which is projected to a road surface point P. Let P move to a new position P' at time $t + \delta t$, where δt is the time interval between two consecutive frames (called `CycleTime` in the seven sequences, either equals 0.04 s or 0.08 s). Then, P' is projected back to the image plane at p'; see Figure 4. The approximation of local displacement at a pixel can then proceed as follows:

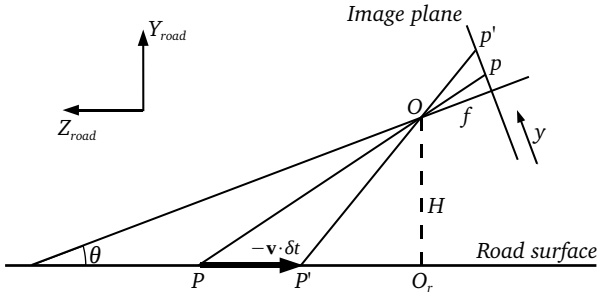

Fig. 4. Approximation of local displacement in y-direction: P and P' is the same road surface point, just in two consecutive frames. P is projected into $p = (x, y)$ in the image plane, P' is projected into $p' = (x', y')$.

First, assume that the vehicle speed equals $\mathbf{v}$ at time t, and $\mathbf{v}'$ at time $t + \delta t$; the average speed during this time interval equals δt is $\overline{\mathbf{v}} = \frac{\mathbf{v}+\mathbf{v}'}{2}$, having δt very small in the sequences. Distances (in Z_{road} coordinates) of moving points are defined as follows:

$$d_Z(P, P') = |\overline{\mathbf{v}}| \cos(\overline{\varphi} + \varphi_c)\delta t = \frac{|\mathbf{v}_1| + |\mathbf{v}_2|}{2} \cos(\frac{\varphi_1 + \varphi_2}{2} + \varphi_c)\delta t$$

where φ_1 and φ_2 are the yaw angles of the ego-vehicle at t and $t + 1$, and φ_c is the yaw angle of the camera installation (see Figure 5). Therefore, the distance between the point P and the host vehicle becomes

$$Z_{P'} = d_Z(O_r, P') = d_Z(O_r, P) - d_Z(PP') = \frac{H}{\tan(\theta + \psi)} - d_Z(PP')$$

Then, the angle between the projection ray OP' and the optical axis of the camera may be determined as follows:

$$\psi' = \arctan\big(\frac{H}{d_Z(O_r, P')}\big) - \theta = \arctan\big(\frac{H}{d_Z(O_r, P) - d_Z(P, P')}\big) - \theta$$

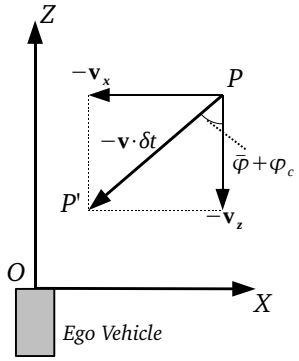

Fig. 5. Change in relative position between road surface point P and ego-vehicle.

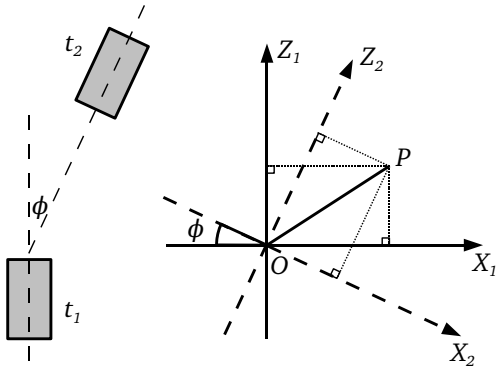

Fig. 6. A rotation of the ego-vehicle.

where $d_Z(O_r, P) = \frac{H}{\tan(\theta+\psi)}$.

Therefore, according to Equation (7), the y-coordinate of the local displacement $\mathbf{u}$ at point P' can be written as

$$v = \big(\frac{f \cdot \tan(\psi')}{s_y} + y_0\big) - y_p$$

Thus, we are also able to specify the position of point P in x-direction as follows

$$X_P = \frac{Z_P \cdot x_p}{f}$$

with $Z_P = \frac{H}{\sin(\theta+\psi)} \cos\psi$, which is actually already a known value from the previous stereo ground truth approximation.

The position of P' (for the next frame) can then be calculated by using speed $\mathbf{v}$ and time interval δt,

$$X_{P'} = X_P - |\mathbf{v}| \sin(\overline{\varphi} + \varphi_c)\delta t$$

Now we have the new relative position between the road surface point and the vehicle at time $t + \delta t$. - In a next step, we need to rotate the vehicle coordinate system by an angle according to the yaw rate given in the vehicle movement parameters; see Figure 6. Therefore, the final (relative) position is given as follows:

$$\begin{bmatrix} X^{\phi}_{P'} \\ Z^{\phi}_{P'} \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} X_{P'} \\ Z_{P'} \end{bmatrix}$$

In a final step, point P is projected back to a pixel p' on the camera's image plane. Then, the local displacement is obtained by comparing locations of p and p', as follows:

$$\psi' = \arctan\big(\frac{H}{Z^{\phi}_{P'}}\big) - \theta$$

$$v = y'_p - y_p = \big(\frac{f \cdot \tan(\psi')}{s_y} + y_0\big) - y_p \quad \text{and} \quad u = x_{p'} - x_p = \frac{f \cdot X^{\phi}_{P'}}{\frac{H}{\sin(\theta+\psi')}\cos(\psi')} - x_p$$

4 Recalibration of Tilt Angle

Although a camera tilt angle is already given for these sequences, we noticed that the angle is not always true when verifying the data. This problem might be caused by several reasons, for example, the road surface is changing (downhill, uphill), the car coordinate system is not parallel to the road surface in some situations (acceleration, braking), drivers of different weight, or driving with flat tires, or the installation of cameras may change for some reasons. (Actually, changes are easy to detect by reading the position of the Mercedes star in the given images.)

The outlined process for obtaining approximate stereo ground truth identified the importance of the tilt angle for the estimated values. We propose a method to estimate the average tilt angle for a given sequence of frames. This method is similar to the road surface stereo approximation, just in a reverse order. We estimate the tilt angle based on given depth at some feature points (i.e., with known disparities) which can be measured or identified manually.

See Figure 3 and assume a given pair of corresponding points, with disparity d. By Equation (6) we have that the tilt angle can be written as follows:

$$\theta = \arcsin\big(\frac{H\cos\psi \cdot d}{b \cdot f}\big) - \psi \tag{8}$$

where ψ is as given in Equation (7).

Altogether, at first, we randomly select five or six frames from a sequence of frames, then, we calculate or choose pairs of corresponding pixels on the road surface area, and obtain disparities between those. Each disparity (of one pixel pair) can be used to calculate a tilt angle using Equation (8), and a mean of those provides a tilt angle estimation; see Table 1 for results for the seven sequences. A more refined estimation is illustrated by Table 2; here, overlapping intervals of 20 frames are used to estimate tilt in each interval, and the table illustrates the actual variations.

Table 1. Results of tilt angle estimation for the given seven sequences.

Sequence name	Tilt angle (radian)
1: 2007-03-06_121807	0.01608
2: 2007-03-07_144703	0.01312
3: 2007-03-15_182043	0.02050
4: 2007-04-20_083101	0.06126
5: 2007-04-27_145842	0.06223
6: 2007-04-27_155554	0.06944
7: 2007-05-08_132636	0.05961

Table 2. Results of tilt angle estimation for short frame periods (first pair of frames and the following 19 pairs of frames) of sequence 2007-04-27_155554.

First pair of frames	1	11	21	31	41	51	61	71	81	91	101	111
Tilt angle (10^{-3} of a radian)	80	71	60	60	62	63	65	70	77	71	63	66
First pair of frames	121	131	141	151	161	171	181	191	201	211	221	231
Tilt angle (10^{-3} of a radian)	60	50	50	59	58	54	55	56	58	53	53	42

5 Results for Selected Techniques

We illustrate the proposed ground truth estimation by providing resulting evaluation data for a few stereo and optical flow techniques, as available on [5], or re-implementing published methods.

5.1 Stereo

For stereo, we run a standard stereo dynamic programming (DP) approach (e.g., see [3]), also modified by using some spatial propagation of disparities (from previous row to the current row, with a weight of 20%) or some temporal propagation of disparities (from the same row in the previous pair of frames, again with a weight of 20%).

Furthermore, we run Birchfield-Tomasi (BT, designed to be an improvement of standard stereo DP), and also implemented a semi-global matching technique using mutual information (SGM-MI) as a cost function (with three pyramid levels and 16 paths). See the Middlebury website for related references. (The experiment on Sequence 7 is only performed on the first 220 frames, instead of the total number of 250, because the road surface is reduced to a very small area after the ego-vehicle makes a large turn to the left.)

Regarding the standard DP algorithm, sequence 1 returns smallest RMS errors and bad matching percentages. In contrast, Sequence 6 returns the largest error values out of the seven sequences.

In spatial propagation (DPs), the method takes 20% of the disparity value from the previous scanline into the final result. In other words, we apply

$$d'_{y,t} = (1 - \lambda_1)d_{y,t} + \lambda_1 d_{y-1,t} \quad \text{where} \quad \lambda_1 = 0.2$$

Temporal propagation (DPt) uses

$$d'_{y,t} = (1 - \lambda_2)d_{y,t} + \lambda_2 d_{y,t-1} \quad \text{where} \quad \lambda_2 = 0.2$$

and temporal and spatial propagation combined (DPts) uses

$$d'_{y,t} = (1 - \lambda_1 - \lambda_2)d_{y,t} + \lambda_1 d_{y-1,t} + \lambda_2 d_{y,t-1}$$

where $\lambda_1 = 0.1$ and $\lambda_2 = 0.1$.

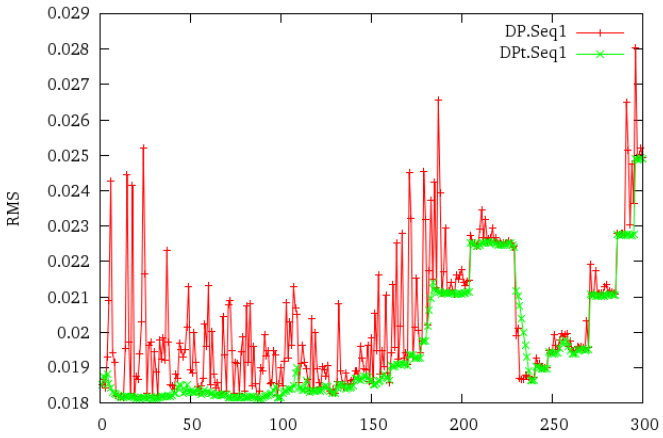

Fig. 7. Comparing RMS error results between DP and DPt.

Figure 7 shows a comparison between DP and DPt for all the frames of Sequence 1. Time propagation shows here (and for the other sequences) an obvious improvement by keeping the RMS error about at the local minimum of the standard DP. Of course, driving on a plane means that disparity values should remain constant, and any deviation from this may be used to detect a change, such as a 'bumpy' road.

A comparison with respect to the second quality metric (percentage of bad matches) is shown in Figure 8. Similar to RMS errors, DPt shows best results. (Note that this evaluation is only restricted to the road surface area.)

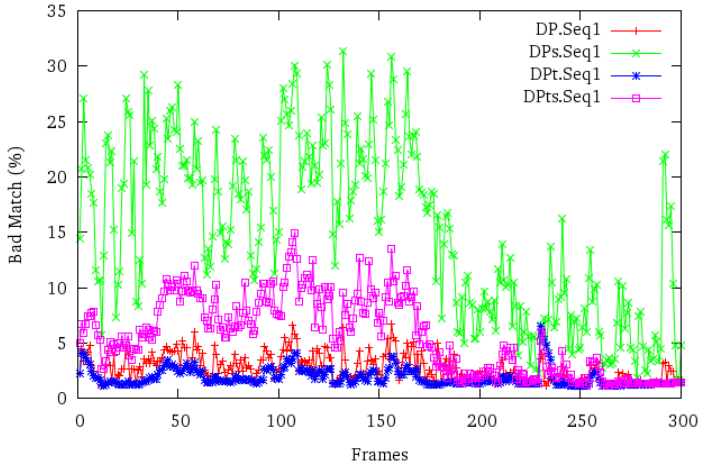

Fig. 8. Percentages of bad matches for DP and its variants.

Now we discuss the Birchfield-Tomasi algorithm (BT). Surprisingly (?), compared with DP techniques, the disparity maps and the quality metrics indicate bad results for BT; disparity values are typically incorrect on the road surface.

This bad performance may be due to the following two reasons. First, the BT algorithm is developed on the concept of the existence of depth discontinuities. However, depth discontinuities may not exist in many real world situations, such as on the road. Second, the BT algorithm uses a disparity propagation method to fill in untextured areas, both in horizontal and vertical directions. However, within the road surface area, the true disparities only change very smoothly in vertical direction.

We also run the BT algorithm (as implemented) on Middlebury stereo data. The same problem, as widely visible in the road scenes, occurs in the untextured area in the upper right corner.

The third stereo algorithm, to be discussed here for the provided approximated ground truth, is SGM-MI algorithm (16 directions for cost aggregation, three pyramid levels to calculate mutual information iteratively). Again, this experiment does not follow performance evaluations results as published on the Middlebury stereo page, this time for SGM: Calculated disparity maps are quite sparse in untextured road surface areas, but more dense in other areas, like vehicles, buildings or trees. Possibly we simply did not use a sufficient number of iterations for mutual information calculations. Because SGM shows more reasonable results on objects or textured areas, it might be recommended to have DPt in lower regions of the image sequence, and an SGM technique (probably not MI) in upper parts of the images.

5.2 Optical Flow

This section reports about experimental results for three 'classical' optical flow algorithms (Horn-Schunck, Lucas-Kanade, and Pyramid Lucas Kanade) on those seven real-world sequences, using the evaluation metrics as given in Section 2.2.

At first we discuss the original Horn-Schunck (HS) and the Lucas-Kanade (LK) algorithm. Results obtained for HS and LK are meaningless and unsuitable for comparison, certainly because of the given 'structureless' images (Sobel

Sequence name	Number of frames	Angular error (degrees)	End point error (pixels)
1: 2007-03-06_121807	300	73	20.7
2: 2007-03-07_144703	300	97	8.9
3: 2007-03-15_182043	300	64	9.5
4: 2007-04-20_083101	250	45	14.4
5: 2007-04-27_145842	250	66	13.4
6: 2007-04-27_155554	250	32	20.9
7: 2007-05-08_132636	220	32	6.5

Fig. 9. Mean angular errors and end point errors for the PyrLK optical flow algorithm.

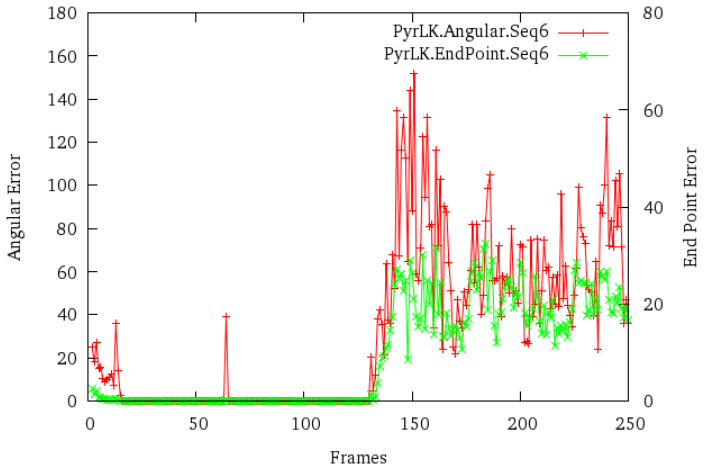

Fig. 10. Angular errors and endpoint errors for PyrLK on Sequence 6.

preprocessing improves the results, similar to [2], where this was noticed for stereo belief propagation on those seven sequences). There might be two reasons for this problem. First, these seven real-world sequences are captured by night-vision cameras which produce somehow blurry and low contrast images, with large non-textured areas on road surfaces. Second, these two classic algorithms may only handle small displacements properly (say, magnitudes of 3-5 pixels), but most pixels in the road mask have actually much larger local displacements than that.

The third algorithm is the pyramid Lucas-Kanade (PyrLK) algorithm. This algorithm runs at first a Canny edge detector on the input images, and uses then edge points as feature points. Finally. it uses five pyramid levels and a

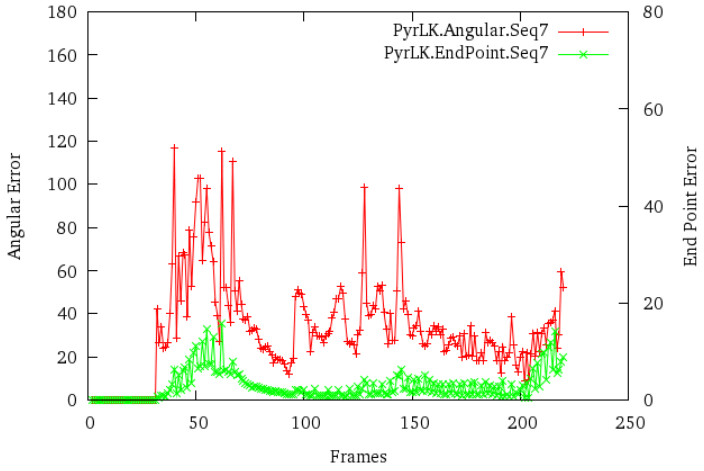

Fig. 11. Angular errors and endpoint errors for PyrLK on Sequence 7.

3×3 window to compute optical flow vectors for those feature points (only). Table 9 shows the evaluation results of the PyrLK algorithm on each sequence. Average angular errors range from about 30 to 100. This is because many of the computed optical flow vectors are actually normal flow, which is perpendicular to the local edge.

Figure 10 shows the quality metrics for PyrLK results on Sequence 6. In the first 135 frames, the ego vehicle stops, and both errors are close to zero. After that, the vehicle starts doing 'snake-like' movements, and causes that the quality of optical flow result drops significantly. Figure 11 shows the quality metrics for Sequence 7. In the first 30 frames, the ego-vehicle is parked besides the road, then, in the next 70 frames, the ego-vehicle turns to the right. During this period, the quality also drops, because the road surface is bumpy. After that the vehicle starts moving straight on a flat road surface, and the errors reduce smoothly.

By comparing quality metrics with vehicle data (see image headers), we notice that the occurrence of a local maximum in errors is most likely to match with a change in the `CycleTime`, normally equals 0.04, but occasionally 0.08.

6 Conclusions

The difficulty for the evaluation of stereo and motion techniques on real-world sequences is the lack of ground truth. This problem is partially solved in this paper by approximating the 3D geometry of the road.

Algorithms (and parameters for those) have mainly be selected for illustrating the proposed evaluation methodology. Further approximate ground truth (such as estimated poses of simple objects, such as rectangular faces in the scene) might be accumulated, to go, step by step, towards a 3D modeling of the actually recorded real scene. Of course, some objects or features are not of interest with respect to applications such as driver assistance or traffic monitoring.

The order of the algorithms' performance is clearly inconsistent to that reported on the Middlebury stereo or optical flow website. This difference shows the necessity for establishing performance evaluation methods also (!) on various real-world sequences.

References

1. *.enpeda..* Image Sequence Analysis Test Site, `http://www.citr.auckland.ac.nz/6D/`
2. Guan, S., Klette, R.: Belief-propagation for stereo analysis of image sequences. CITR-TR-208, Computer Science, The University of Auckland, Auckland, 2007.
3. Klette, R., Schlüns, K., Koschan, A.: *Computer Vision.* Springer, Singapore, 1998.
4. Liu, Z., Klette, R.: Performance evaluation of stereo and motion analysis on rectified image sequences. Technical report, Computer Science Department, The University of Auckland (2007).
5. Intel Open Source Computer Vision Library, `http://www.intel.com/research/mrl/research/opencv/`