Optical Flow Estimation Using Local Features

Abdulmalik Danlami Mohammed, Tim Morris

***Abstract*—The computation of optical flow by the differential method imposes additional constraints to the one already imposed in the derivation of the optical flow equation. Consequently, the computation of optical flow using differential methods is computationally expensive especially for devices such as mobile phones, which have low processing power. In this work, we propose an optical flow computation method based on local features called the nearest flow. Our nearest flow method works by estimating the distance ratio of two nearest features to find the best match for a feature point. To improve the quality of the sparsely generated flow vectors, we apply the random sampling consensus method to remove false flows that may arise as a result of noise and other imagery conditions.**

We compare the performance of our nearest flow method with that of Gunner Farneback's and the local differential method of Lucas and Kanade by evaluating the average angular error for each method in the computation of optical flow. The results obtained show that our nearest flow method is faster and more accurate than Gunner Farneback's method and it is almost at the same level of performance as the Lucas and Kanade method.

***Index Terms*— Angular error, Harris corner detector, k-nearest neighbour, Optical flow,**

I. INTRODUCTION

The estimation of optical flow from image sequences is a challenging task in computer vision and image analysis. Optical flow describes a 2D image motion obtained from the projection of 3D motion in a scene onto an image plane caused by the apparent motion between the camera and the visual scene. A reliable estimation of optical flow can be used to perform various tasks such as image segmentation, motion detection, structure from motion calculation, time-to-contact and focus of expansion calculations. The optical flow field provides not only the information about the motion of an object in a visual scene, but also information about the 3D structure of the scene. Hence, as an important motion cue, optical flow has been used for many computer vision and image processing applications. For example, Enkelman et al [9] applied the optical flow field to detect obstacles in the presence of moving vehicles. The work by Low and Wyeth [10] used optical flow to encode time to contact information in order to detect obstacles in the path of a moving robot.

Manuscript received March06, 2015; revised March 20, 2015.
Abdulmalik Danlami Mohammed is a Postgraduate Student at the School of Computer Science, The University of Manchester, Manchester, M13 9PL, UK. phone: +447741052107; e-mail: mohammea@cs.man.ac.uk).
Tim Morris is a Lecturer with the School of Computer Science, The University of Manchester, Manchester, M13 9PL UK (e-mail: tim.morris@manchester.ac.uk).

While Yacoob and Davis [12] used optical flow to recognize facial expression, Ardizzone and La Cascia [13] made use of optical flow to automatically index and retrieve video. Several methods of optical flow computation evolved from the optical flow constraint equation, which assumes that for a uniform image motion and under constant illumination of a scene, the intensity pattern of an image is unchanged. However, the equation is faced with the problem of finding two unknown variables using a single constraint, which gives rise to the aperture problem. To solve this problem, many differential based approaches imposed additional constraints to the flow vectors at the expense of computational load. In general however, optical flow computation using differential methods suffer a drawback in terms of the computational load required to compute the additional constraints. Hence, in this paper, we propose an efficient method of computing optical flow vectors from image sequences using local image features and a matching procedure to find feature correspondences. Figure 1, shows an optical flow image of corridor.

The remainder of this text is as follows: Section 2 describes some of the related works in the computation of optical flow. In Section 3, the mathematical foundation of the optical flow is presented. Section 4 briefly discusses feature detectors. While Section 5 presents our proposed method, Section 6 gives a comparative analysis of our method against two other methods. We conclude the discussion of this study in Section 7.

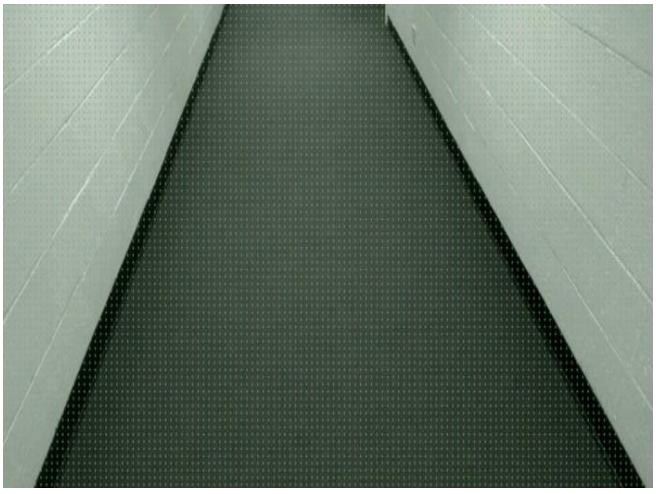


Fig 1 Optical flow image of a corridor

II. RELATED WORK

There is a wide range of methods proposed for the computation of optical flow, which includes global and local differential methods, phase based methods, correlation and feature based methods. A comprehensive survey of some of these approaches is presented in [4], the performance evaluation of most methods can be found in [1] and [5].

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

However, in this section, we review two commonly used methods: Differential and Feature Based methods. An example of the differential method for optical flow computation is presented in [3]. In their work, Horn and Schunck formulated a global smoothness constraint to minimize the square of the magnitude of the optical flow vectors, thereby generating a dense optical flow. A local differential method that proves to be computationally efficient in comparison to [3] is described in [2]. Lucas and Kanade use the constraint of the neighbourhood around a pixel in subsequent images to estimate optical flow. A feature based method of optical flow computation is presented in [8]. This method, which motivates our present approach, computes optical flow using scale invariant features and matching techniques to find the similarity between these features. The method is however robust and accurate in comparison to differential approaches. Ogata and Sato [11], proposed an optical flow estimation method using a two-stage model. In the first stage, candidate velocities are detected at the intersection of the constraint line, following which a feature matching scheme is employed to determine the velocity with the highest feature vote, which is then assigned to a pixel.

III. OPTICAL FLOW CONSTRAINT EQUATION

The optical flow constraint equation (3) is based on brightness constancy rule (1), which assumes that the intensity value of each image pixel does not change with respect to space and time, provided the scene illumination remains constant as the object moves. Images are captured at time $\boldsymbol{t}$ and at time$\boldsymbol{t} + \Delta\boldsymbol{t}$. Mathematically, this can be expressed as:

$$\mathrm{I}(x, y, t) = \mathrm{I}(x + \Delta x, y + \Delta y, t + \Delta t) \qquad (1)$$

Where, $\mathrm{I}(x, y, t)$ represents the pixel intensity on the image plane at time t, $\Delta x, \Delta y,$ denotes the displacement of the pixel with unchanged intensity value in x and y direction respectively and Δt denotes the change in time. The equation of brightness constancy (1) provides a strong mathematical foundation for the estimation of optical flow. By expanding the right-hand side using the Taylor series expansion and ignoring the high order terms, we obtain:

$$\mathrm{I}(x, y, t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t \qquad (2)$$

If we rearrange the expression then divide through by t, then use $\frac{\Delta x}{\Delta t} = V_x$, $\frac{\Delta y}{\Delta t} = V_y$ and $\frac{\Delta t}{\Delta t} = 1$ and rewrite $\frac{\partial I}{\partial x} = I_x, \frac{\partial I}{\partial y} = I_y \; and \; \frac{\partial I}{\partial t} = I_t$. we get the optical flow equation:

$$I_x V_x + I_y V_y = -I_t \qquad (3)$$

Where (I_x, I_y) denotes the spatial gradient of the image at(x, y, t), (V_x, V_y) represents the optical flow vectors in the x and y directions and I_t denotes the temporal gradient. To overcome the aperture problem, we employ a feature based computation of the optical flow and matching process to find similar image points. Hence, a stable and rotation invariant feature can be helpful in generating a reliable estimate of optical flow.

Fig 2 Corner detection on a video frame

IV. FEATURE DETECTION

Feature detection is an important task required for optical flow computation using image feature. Several image features such as colour and edges have been widely applied for tracking image points with little success due to instability of the features under varied illumination and shadow. For instance, edge features, which have low gradient values along their direction and high gradient values in other directions are vulnerable to tracking failure due to image rotation and thus generate inaccurate optical flow vectors. In comparison to edge points, corner points have high gradient values in many directions providing a stable feature for tracking in video images. Hence, this work employs image corners for tracking and a matching process to find correspondences between points in an image sequence to generate optical flow vectors. Consequently, this section gives an overview of the most common feature detection method: the Harris corner detector [7] to pave the way for further discussion in section 5. While several methods for feature detection proposed in the literature compute the first derivatives shown in equation (4) and second derivatives of the image to find interest points (corners), few of them are robust and invariant to the rotation of the feature. The Harris corner detector is widely used for corner detection because of its robustness and invariance to orientation of the data. Figure 2 shows the detected feature points in a sample video frame. The Harris corner detector finds corners at image location where the matrix of the second order derivatives eq. (5) has two large eigenvalues:

$$I_x(x, y) = \frac{\partial I}{\partial x}(x, y) \text{ and } I_y(x, y) = \frac{\partial I}{\partial y}(x, y) \qquad (4)$$

Where $\frac{\partial I}{\partial x}(x, y)$ and $\frac{\partial I}{\partial y}(x, y)$ are the first derivatives in the x and y directions respectively.

$$C = \begin{bmatrix} {I^2}_x & I_x I_y \\ I_x I_y & {I^2}_y \end{bmatrix} \qquad (5)$$

At every location of the image, the two eigenvalues that describe edge strength are extracted from the computed second order derivative, thus generating a new matrix C':

$$C' = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \qquad (6)$$

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Where λ_1 and λ_2 are the eigenvalues computed from the second order matrix C . Corners are detected when $\lambda_1 > \lambda_2$ and the small value of λ_2 is above some threshold.

V. PROPOSED METHOD

The computation of optical flow using our nearest flow (NF) begins with the detection of feature points (corners) from a captured video. In this study, we applied the Harris corner detector described in section 4 on each frame of the video to extract corner points. At this point, the detected points for each frame are transformed into row vectors to enable efficient finding of feature correspondences in image sequences. Following the extraction of corners from two video frames at different time instances and subsequent row vector transformation of the features, a k-nearest neighbor search algorithm with $k = 2$ is employed to find correspondences between the frames by comparing their vectors. Two possible matches closest to the query feature points are returned. We refer to these feature points as Most Nearest (MN) and Almost Nearest (AN) match. To find the best match between the two matches, we calculate the distance ratio between these two matches (MN and AN) (see Algorithm 1) and return the most nearest (MN) if the ratio test is big enough otherwise both matches are too close to be considered for a true match and are therefore discarded. The set of returned matches in this case represents the optical flows we are seeking. To improve the quality of the output, we applied the RANSAC method to remove false positive flow vectors.

V1. EVALUATION OF OUR APPROACH

We use a dataset of 5 different image sequences acquired from the http:vision.middlebury.edu/flow/ web site to evaluate the performance of our approach against the local differential methods of Lucas and Kanade [2] and Farneback [6]. Each dataset is composed of 8 images of the same scene captured at different times. We use the average angular error of equation (7) as the metric for the evaluation of the three methods. The average angular error as described in [5] is an important metric used to effectively evaluate the performance of many optical flow estimation methods.

$$\boldsymbol{E = \arccos \frac{U*U_e+V*V_e+1}{\sqrt{(U)^2+(V)^2+1}\sqrt{(U_e)^2+(V_e)^2+1}}} \qquad (7)$$

Where $\boldsymbol{U, V}$ denotes the current optical flow and $\boldsymbol{U_e, V_e}$ represent the estimated optical flow. $\boldsymbol{U_e, V_e}$ are the expected optical flows which are obtained by computing the flow vectors between two images of the 8 images in each datasets. In our case, we compute the expected flow vectors between the fourth and fifth images. In figure 3, we show the image dataset the computed optical flow images for each dataset. Table 1 shows the average angular errors, the standard deviation obtained from the three methods and the time in seconds for processing all frames for each of the datasets. The results obtained show that the Farneback method has the best performance on Grove and RubberWhale datasets. While the Lucas and Kanade method has the best execution time on each dataset, it has approximately the same level of performance as the nearest flow method. On average, both the Lucas and Kanade method and the nearest flow method have a better performance than the Farneback method.

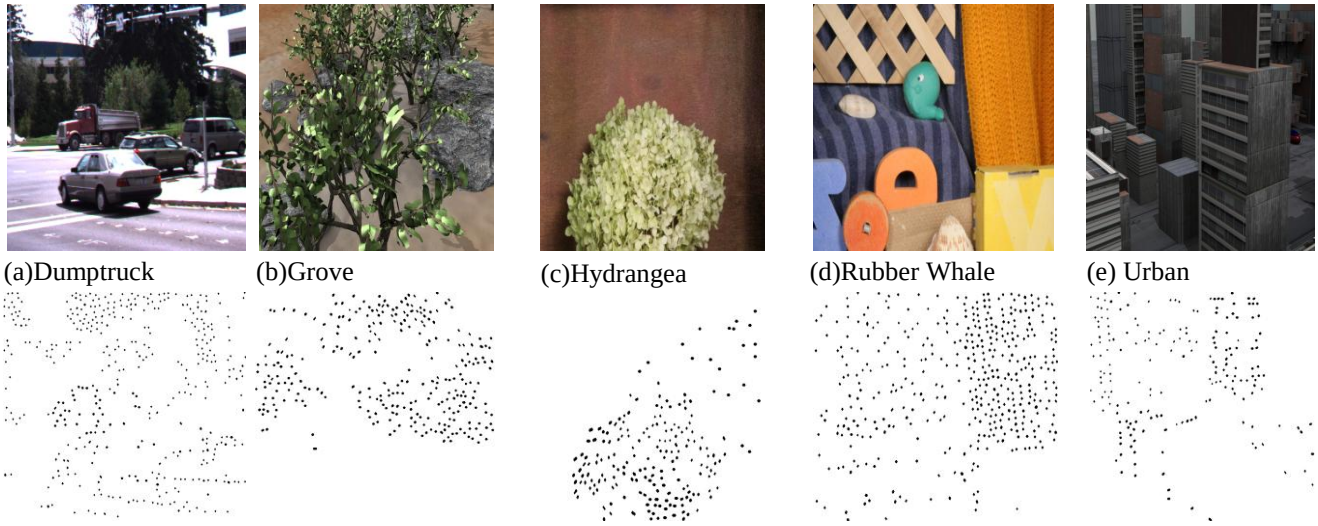
Algorithm 1 Optical flow estimation using corner points

Input: *video sequence*
Output: *flow vectors: F*
1: ***for*** *each video frame, I* ***do***
2: *Convert video frame to grayscale image, E: E ← I*
3: *Create empty lists: cornerList1← [ ] , cornerList2← [ ], f ← [ ];*
4: *invoke Harris corner detector H() on E and compute corners:* $C_i \leftarrow H(E)$
5: *Add corner points* C_i *to cornerList2: cornerList2←* $[C_i, C_i \ldots C_n]$
6: ***If*** *CornerList1 is not empty* ***Then***
7: *Transform corner1 and corner2 to row vectors:* $rV1 \leftarrow (Corner1)$, $rV2 \leftarrow (Corner2)$
8: *Create an empty list:* $m \leftarrow [\,]$
9: *invoke KNN_MATCHER(rV2, rV1, m, 2)*
for *each* $r_i \in rV2$ ***do***
10: ***If*** $\frac{m_{i,1}\ Distance\ from\ r_i}{m_{i,2} Distance\ from\ r_i} < 0.7$ ***then***
11: *Add* $m_{i,1}$ *to f:* $f \leftarrow [m_{i,1}]$
12: ***end if***
13: ***end for***
flow vectors: F $\leftarrow (f)$
14: ***end if***
15: $cornerList1 \leftarrow cornerList2$
16: ***end for***
17: ***return F***

Table I Evaluation Result of Gunner Farneback, Lucas and Kanade, and Nearest Flow Methods.

Datasets	Farneback			Lucas & Kanade			NearestFlow		
	Av.Angle Err.	Std.Deviation	8-Frames/s	Av. Angle Err.	Std.Deviation	8-Frames/s	Av.Angle Err.	Std.Deviation	8-Frames/s
Dumptruck	**1.81**	**0.13**	**3.05**	**1.62**	**0.24**	**1.09**	**1.61**	**0.25**	**2.31**
Grove	**1.24**	**0.19**	**3.03**	**1.6**	**0.24**	**1.25**	**1.61**	**0.25**	**2.13**
Hydrangea	**1.78**	**0.16**	**2.26**	**1.7**	**0.17**	**1.06**	**1.69**	**0.17**	**2.1**
RubberWhale	**1.77**	**0.13**	**2.28**	**1.63**	**0.24**	**1.06**	**1.64**	**0.24**	**2.18**
Urban	**1.4**	**0.31**	**3.04**	**1.62**	**0.22**	**1.31**	**1.56**	**0.25**	**2.22**

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)



(a)Dumptruck (b)Grove (c)Hydrangea (d)Rubber Whale (e) Urban

(f) Dumptruck flows (e)Grove flows (h)Hydrangea flows (i) RubberWhale flows (j)Urban flows

Fig 3 Images and the estimated flows of data (http:vision.middlebury.ed/flow/)

VII. DISCUSSION

As depicted in Table 1, our nearest flow method is faster in terms of the number of frames being processed per second in comparison to the Farneback method. Except for two datasets, our nearest flow method outperformed the Farneback method. The reason for this discrepancy is that, while the nearest flow method computes optical flow at areas with high curvature representing the most stable and salient areas in an image using second order image derivatives as described in section 4 of this paper, the Farneback [6] method uses a quadratic polynomial to estimate each neighborhood of two frames which may also include other noisy regions such as shadows and thus generate a dense optical flow. The Lucas and Kanade method has the best throughput in comparison to both Farneback and our nearest flow method. However, its performance rate is at about the same level as our method. Our future work will focus on optimizing the nearest flow method to improve the execution time. The nearest flow method has potential to work in mobile devices with very low computation capability.

REFERENCES

[1] J.L Baron, D.J Fleet, and S.S Beauchemin “Performance of Optical Flow Technique” International Journal of Computer Vision, Vol. 12 No, 1 pp43-77, 1994.

[2] Bruce D. Lucas, Takeo Kanade “An Iterative Image Registration Technique with application to Stereo Vision”, In Proc. Of 7th Intl. Joint Conference on Artificial Intelligence, pp121-130. 1981(IJCAI)

[3] Berthold K.P, Horn, Brian G.Schunk,“Determining Optical Flow”, Techniques and Applications of Image Understanding, vol. 281, pp319-326. 1981.

[4] Peter O Donovan, “Optical Flow: Techniques and Application”, 2005.

[5] Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, Michael J. Black and Richard Szeliski, “A Database and Evaluation Methodology for Optical Flow”, International Journal of Computer Vision, vol. 92, pp1-31. 2011.

[6] Gunner Farneback, “Two-Frame Motion Estimation Based on Polynomial Expansion”, Image Analysis, pp363-370. 2003.

[7] Chris Harris and Mike Stephens, “A Combined Corner and Edge Detector”,Alvey Vision Conference,147-15. 1988.

[8] Li Xu, Zhenlong Dai and Jiaya Jia, “A Scale Invariant Optical Flow”, Proc ECCV pp385-399. 2012.

[9] W.Enkelmann, V.Gengenbach, W. Kruger, S. Rossle, W.Tolle, “Obstacle Detection BY Real-Time Optical Flow Evaluation”, Imaging and Vision Computing, Vol.1 No.3 pp160-168. 1990.

[10] Toby Low and Gordon Wyeth “Obstacle Detection using Optical Flow”, Australasian Conference on Robotics and Automation, 2005.

[11] Masami Ogata and Takao Sato, “Motion-detection model with two stages: spatiotemporal Filtering and Feature Matching”, Journal of Optical Society of America vol.9 No. 3 pp377-387. 1992.

[12] Yaser Yacoob and Larry S. Davis, “Recognizing Human Facial Expressions From Long Image Sequences Using Optical Flow”, IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 18 ,No. 6, pp638-642. 1996.

[13] Edoardo Ardizzone and Marco La Cascia “Video Indexing Using Optical Flow Field” In Proc. International Conference on Image Processing, Vol.3 pp831-834. 1996

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)