Estimating 3D Scene Flow from Multiple 2D Optical Flows

Jonathan Ruttle
ruttlejl@cs.tcd.ie

Michael Manzke
Michael.Manzke@cs.tcd.ie

Rozenn Dahyot
Rozenn.Dahyot@cs.tcd.ie

Trinity College Dublin
School of Computer Science and Statistics
Graphics Vision and Visualisation Group

Abstract

Scene flow is the motion of the surface points in the 3D world. For a camera, it is seen as a 2D optical flow in the image plane. Knowing the scene flow can be very useful as it gives an idea of the surface geometry of the objects in the scene and how those objects are moving. Four methods for calculating the scene flow given multiple optical flows have been explored and detailed in this paper along with the basic mathematics surrounding multi-view geometry. It was found that given multiple optical flows it is possible to estimate the scene flow to different levels of detail depending on the level of prior information present.

1 Introduction

Optical flow is the 2D projection of the 3D scene flow onto the image plane. Optical flow has been around for nearly the last three decades [7], [6] and in that time the methods and techniques for calculate optical flow has improved greatly [4], [1]. It has been shown that it is possible to back-project the optical flow to compute the scene flow [11].

Knowledge of scene flow can help with a number of applications including human motion capture [8] and can also help in determining the structure of the scene [12]. Combining multiple optical flows in a single scene flow could possible detect and correct error in individual optical flows, this corrected optical flow could then be used for better video compression.

Traditionally calculating scene flow is done by matching pixels across two views of the same scene. This allows the depth information to be calculated and the two optical flows to be combined to generate the scene flow.

Vedula et al. presents three different scenarios for calculate scene flow [11]. The first scenario occurs when there is complete knowledge of the surface of the object in the scene. In this case, Vedula shows that it is possible to calculate scene flow from a single camera. The second scenario occurs when there is knowledge of corresponding information between two image views and in this case it is possible to determine the scene flow from these two views. The third scenario is when there is no knowledge of the scene. In this situation, Vedula shows that, by using 51 cameras, possible scene flows can be generated and the results can be thresholded to reveal the moving objects in the scene.

Tian and Shah present a method for modelling the translational and rotational motion of the objects in the scene and the scene flow is estimated by using a 5D histogram [9]. Torr et al. calculate matching feature points in each image to calculate the scene flow for the matched points and the scene structure [10].

This paper summaries some of the basic projection mathematics and multi-view geometry as presented in [5] and customises it to optical flow and scene flow situations. We show how it is possible, given a surface point in a scene and two camera views of that point, to calculate the scene flow. Four methods for combining multiple optical flow images to generate a 3D scene flow are presented. The first method uses background subtraction to determine the surface points of the objects in space and then uses projection matrix to calculate the scene flow for each surface point given the corresponding optical flow from each pixel. The second method estimates the background subtraction by thresholding the optical flow, under the assumption that only the moving objects in the scene are of interest. The third method performs the thresholding on the scene flow instead again assuming that the areas of largest scene flow are the areas of interest. The last method defines a 6D histogram space where all possible point scene flows are considered. This histogram is searched to reveal the point scene flows where there is the most concurrence between the multiple views.

2 Notation

All calculations assume knowledge of all extrinsic and intrinsic camera parameters. This includes the camera projection matrix P_i and the camera centre $\mathbf{c}_i$ for each camera i.

$$\mathrm{P}_i = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{pmatrix} \tag{1}$$

$$\mathbf{c}_i = \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} \tag{2}$$

Homogeneous coordinates are used for all 2D pixels $\mathbf{u}_i^t$ in the image plane of camera i at time t and all 3D points $\mathbf{x}^t$ in world space at time t.

$$\mathbf{u}_i^t = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{3}$$

$$\mathbf{x}^t = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4}$$

Optical flow $\dot{\mathbf{u}}_i^t$ is the movement of a pixel over a time interval from some previous time t' to the current time t where the interval $|t-t'|$ is assumed to be of short duration.

$$\dot{\mathbf{u}}_i^t = \frac{\mathbf{u}_i^t - \mathbf{u}_i^{t'}}{t-t'} = \begin{pmatrix} \dot{u} \\ \dot{v} \\ 0 \end{pmatrix} \tag{5}$$

Similarly the scene flow $\dot{\mathbf{x}}^t$ is:

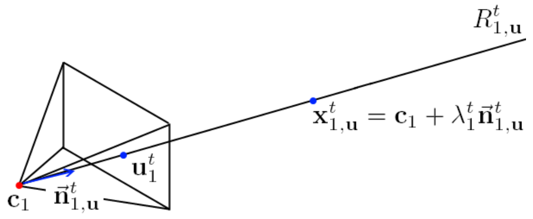
$$\dot{\mathbf{x}}^t = \frac{\mathbf{x}^t - \mathbf{x}^{t'}}{t-t'} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ 0 \end{pmatrix} \tag{6}$$

The time is usually the frame index in the video stream and the optical flow is computed with successive frames $t-t' = 1$.

2.1 Determining a 2D Pixel given a 3D Point

A 3D point $\mathbf{x}^t$ in world space can be projected onto a 2D pixel $\mathbf{u}_i^t$ in the image plane by multiplication with the projection matrix P_i.

$$\mathbf{u}_i^t = \mathrm{P}_i\, \mathbf{x}^t \tag{7}$$



$R_{1,\mathbf{u}}^t$
$\mathbf{x}_{1,\mathbf{u}}^t = \mathbf{c}_1 + \lambda_1^t \vec{\mathbf{n}}_{1,\mathbf{u}}^t$
$\mathbf{u}_1^t$
$\mathbf{c}_1$
$\vec{\mathbf{n}}_{1,\mathbf{u}}^t$

Figure 1. In camera 1, a pixel $\mathbf{u}_1^t$ and a camera centre $\mathbf{c}_1$ are used to calculate a 3D ray of all possible points $\mathbf{x}_{1,\mathbf{u}}^t$ in the 3D world that project onto pixel $\mathbf{u}_1^t$.

2.2 Determining a 3D Ray given a 2D Pixel

Given a single 2D pixel $\mathbf{u}_i^t$, it is not possible to calculate a single 3D point $\mathbf{x}_{i,\mathbf{u}}^t$ without depth information λ_i^t. It is possible to calculate a 3D ray of points where any point on the ray will be projected onto the given 2D pixel $\mathbf{u}_i^t$. Multiplying the pseudo-inverse of the projection matrix P_i^+ (defined by $\mathrm{P}^+ = \mathrm{P}^\top \left(\mathrm{P}\mathrm{P}^\top\right)^{-1}$) by the homogenous pixel coordinate will results in some point $\mathbf{x}_{i,\mathbf{u}}^t$ on the ray in the scene in front of the camera.

$$\mathbf{x}_{i,\mathbf{u}} = \mathrm{P}_i^+\, \mathbf{u}_i \tag{8}$$

Therefore a ray starting at the camera centre $\mathbf{c}_i$ and continuing on through this new point $\mathbf{x}_{i,\mathbf{u}}$ describes all possible points that could be projected onto the pixel $\mathbf{u}_i^t$. A more convenient way to describe this ray is Equation 10, where $\vec{\mathbf{n}}_{i,\mathbf{u}}^t$ is the unit vector from $\mathbf{c}_i$ to $\mathbf{x}_{i,\mathbf{u}}$ and λ is some positive number which represents the distance between the camera centre and the point that is being searched for. As seen in Figure 1.

$$\vec{\mathbf{n}}_{i,\mathbf{u}}^t = \frac{\mathbf{x}_{i,\mathbf{u}} - \mathbf{c}_i}{\|\mathbf{x}_{i,\mathbf{u}} - \mathbf{c}_i\|} \tag{9}$$

$$\mathbf{x}_{i,u}^t = \mathbf{c}_i + \lambda_i^t\, \vec{\mathbf{n}}_{i,\mathbf{u}}^t \tag{10}$$

2.3 Determining a 3D Point given two corresponding Pixels

When two cameras are used and a pixel in each image $\mathbf{u}_i^t$ and $\mathbf{u}_j^t$ are the projection of the same point $\mathbf{x}^t$, it is possible to calculate the 3D position of that point. The points position will be the intersection of the two rays $R_{i,\mathbf{u}}^t$ and $R_{j,\mathbf{u}}^t$ formed by the pixels as described in section 2.2. Unfortunately due to the quantisation of the size of the pixels

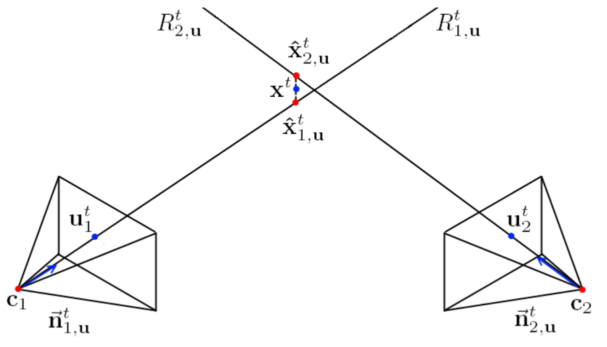

$R^t_{2,\mathbf{u}}$
$R^t_{1,\mathbf{u}}$
$\hat{\mathbf{x}}^t_{2,\mathbf{u}}$
$\mathbf{x}^t$
$\hat{\mathbf{x}}^t_{1,\mathbf{u}}$
$\mathbf{u}^t_1$
$\mathbf{u}^t_2$
$\mathbf{c}_1$
$\vec{\mathbf{n}}^t_{1,\mathbf{u}}$
$\mathbf{c}_2$
$\vec{\mathbf{n}}^t_{2,\mathbf{u}}$

Figure 2. Two pixels $\mathbf{u}^t_i$ and $\mathbf{u}^t_j$ used to calculate point $\mathbf{x}^t$ by intersection the 2 rays $R^t_{i,\mathbf{u}}$ and $R^t_{j,\mathbf{u}}$. Where camera $i = 1$ and camera $j = 2$.

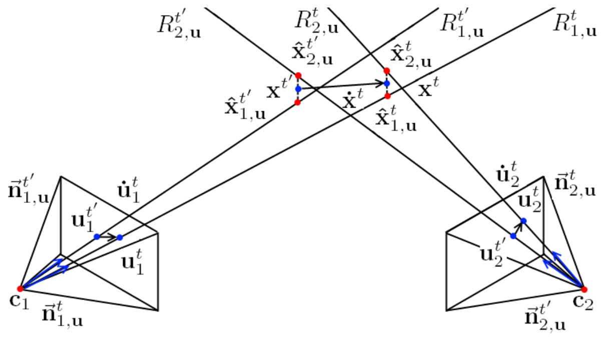

$R^{t'}_{2,\mathbf{u}}$
$R^t_{2,\mathbf{u}}$
$R^{t'}_{1,\mathbf{u}}$
$R^t_{1,\mathbf{u}}$
$\hat{\mathbf{x}}^{t'}_{2,\mathbf{u}}$
$\hat{\mathbf{x}}^t_{2,\mathbf{u}}$
$\mathbf{x}^{t'}$
$\dot{\mathbf{x}}^t$
$\mathbf{x}^t$
$\hat{\mathbf{x}}^{t'}_{1,\mathbf{u}}$
$\hat{\mathbf{x}}^t_{1,\mathbf{u}}$
$\vec{\mathbf{n}}^{t'}_{1,\mathbf{u}}$
$\dot{\mathbf{u}}^t_1$
$\mathbf{u}^{t'}_1$
$\mathbf{u}^t_1$
$\dot{\mathbf{u}}^t_2$
$\vec{\mathbf{n}}^t_{2,\mathbf{u}}$
$\mathbf{u}^t_2$
$\mathbf{u}^{t'}_2$
$\mathbf{c}_1$
$\vec{\mathbf{n}}^t_{1,\mathbf{u}}$
$\mathbf{c}_2$
$\vec{\mathbf{n}}^{t'}_{2,\mathbf{u}}$

Figure 3. Two optical flows used to calculate the scene flow of a point using two sets of intersection rays. Where camera $i = 1$ and camera $j = 2$.

and floating-point arithmetic it is quite unlikely that the rays will actually intersect, but they will pass close to each other. Therefore the midpoint of the two closest points $\hat{\mathbf{x}}^t_{i,\mathbf{u}}$ and $\hat{\mathbf{x}}^t_{j,\mathbf{u}}$ on the two rays can be used to give an approximate solution $\mathbf{x}^t$. If the distance between the two closest points $\hat{\mathbf{x}}^t_{i,\mathbf{u}}$ and $\hat{\mathbf{x}}^t_{j,\mathbf{u}}$ is very large for example larger then the size of a pixel then this may indicate that the two pixels do not refer to the same point and there may be error in the pixel matching.

2.4 Determining a 3D Scene Flow given two corresponding Optical Flows

Similar to the situation in section 2.3 two 2D pixels $\mathbf{u}^t_i$ and $\mathbf{u}^t_j$ with optical flows $\dot{\mathbf{u}}^t_i$ and $\dot{\mathbf{u}}^t_j$ both visualising the same 3D point and scene flow can be used to calculate the position of that point $\mathbf{x}^t$ and the direction and magnitude of that scene flow $\dot{\mathbf{x}}^t$. As shown in Figure 3 the pixels can be converted into rays $R^t_{i,\mathbf{u}}$ and $R^t_{j,\mathbf{u}}$ and the midpoint of the two closest points reveal the position of the 3D point $\mathbf{x}^t$. The optical flow can be used to calculate the last position of each of the pixels in the previous frame $\mathbf{u}^{t'}_i$ and $\mathbf{u}^{t'}_j$, by subtracting the optical flow from the current position. These new pixels can be converted into rays $R^{t'}_{i,\mathbf{u}}$ and $R^{t'}_{j,\mathbf{u}}$ and the previous 3D point $\mathbf{x}^{t'}$ can be calculated in the same way as in section 2.3. The vector from this previous point $\mathbf{x}^{t'}$ to the current point $\mathbf{x}^t$ is the scene flow $\dot{\mathbf{x}}^t$.

2.5 Determining a range of possible points and scene flows from a single pixel

Given a single pixel $\mathbf{u}^t_i$ with optical flow $\dot{\mathbf{u}}^t_i$ it is possible to calculate a range of possible points $\mathbf{x}^t$ and scene flows $\dot{\mathbf{x}}^t$ which would project onto that pixel and optical flow. Using

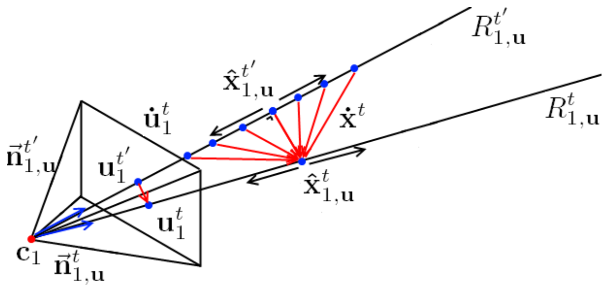

$R^{t'}_{1,\mathbf{u}}$
$\hat{\mathbf{x}}^{t'}_{1,\mathbf{u}}$
$\dot{\mathbf{u}}^t_1$
$\dot{\mathbf{x}}^t$
$R^t_{1,\mathbf{u}}$
$\vec{\mathbf{n}}^{t'}_{1,\mathbf{u}}$
$\mathbf{u}^{t'}_1$
$\hat{\mathbf{x}}^t_{1,\mathbf{u}}$
$\mathbf{u}^t_1$
$\mathbf{c}_1$
$\vec{\mathbf{n}}^t_{1,\mathbf{u}}$

Figure 4. A range of possible scene flows $\dot{\mathbf{x}}^t$ given one optical flow $\dot{\mathbf{u}}^t_i$ after picking one possible point $\mathbf{x}^t$. Where camera $i = 1$.

the optical flow it is possible to calculate the position of the pixel in the previous frame $\mathbf{u}^{t'}_i = \mathbf{u}^t_i - \dot{\mathbf{u}}^t_i$. The 3D rays $R^t_{i,\mathbf{u}}$ and $R^{t'}_{i,\mathbf{u}}$ can be calculated for both the current and previous pixel using the technique described in section 2.2.

The two rays can be seen in Figure 4, basically any vector that connects $R^{t'}_{i,\mathbf{u}}$ to $R^t_{i,\mathbf{u}}$ is a possible scene flow. The Equation 10 gives you a range of possible points $\mathbf{x}^t = \mathbf{c}_i + \lambda^t_i\ \vec{\mathbf{n}}^t_{i,\mathbf{u}}$ by exploring all valid λ^t_i. For each point $\mathbf{x}^t$ on $R^t_{i,\mathbf{u}}$ a range of possible previous points $\mathbf{x}^{t'}$ can be found the same way on $R^{t'}_{i,\mathbf{u}}$. The time between frames will be very small therefore it is possible to use this prior knowledge to place a maximum magnitude on the scene flow. It is possible to calculate the closest point $\hat{\mathbf{x}}^{t'}_{i,\mathbf{u}}$ on the previous ray $R^{t'}_{i,\mathbf{u}}$ to the currently selected point $\mathbf{x}^t$ on $R^t_{i,\mathbf{u}}$. Then possible previous points $\mathbf{x}^{t'} = \hat{\mathbf{x}}^{t'}_{i,\mathbf{u}} + \lambda^{t'}_i\ \vec{\mathbf{n}}^{t'}_{i,\mathbf{u}}$ by varying $\lambda^{t'}_i$ between plus and minus the maximum magnitude of the scene flow.

3 Calculating 3D Scene Flow

Previously to calculate scene flow required knowledge of corresponding pixels, this in general requires pixels matching. The following four algorithms try to avoid this costly approach and use alternative methods by either finding surface points which can be projected into the cameras to find corresponding pixels or by overlapping all possible scene flows to find areas of concurrence.

3.1 Background Subtraction

Background subtraction is performed on each of the images to determine which pixels belong to foreground objects and which pixels belong to the background. A dense selection of equally distributed points is generated in the 3D world to be explored. Each point is projected into each of the cameras using Equation 7. If all the pixels relating to a point belong to a foreground object then the point is kept, otherwise the point is discarded. The remaining points form the solid shape of the object in the scene. The more cameras and the more points used the better the shape will be in approximating the object, but the more computationally expensive it will be.

To extract the surface points or visual hull, the remaining points are examined and inside points are discarded and surface points are kept. This is done by stepping through all the points and each time a border is crossed the point is kept otherwise it is discarded. At this point it is also noted which relative side of the object the point is on, this allows the relative cameras to be selected in determining the scene flow. The remaining points can be projected into the appropriate cameras using the method described in section 2.1 and then the method described in section 2.3 can be used to determine the scene flow at that point.

3.2 Thresholding Optical Flow

Assuming that the only thing moving in the scene is the object then the background subtraction can be approximated by thresholding the optical flow results. The same method described in section 3.1 can be used but thresholded optical flow is used instead of the background subtraction results. This solution suffers from problems when the scene flow is in-line with one of the cameras. When this is the case then very little optical flow is seen in that camera and so it falls below the threshold, and therefore disappear in the final estimation of the scene flow.

3.3 Thresholding Scene Flow

This time the scene flow is calculated for all the points in the scene. The magnitude of each scene flow is measure and the ones below a threshold are discarded.

3.4 Histogram Data

A 6D histogram is created with the first three dimension as world position x y z, and the last three dimensions as scene flow components $\dot{x}$ $\dot{y}$, $\dot{z}$. For every pixel in every camera the method described in section 2.4 is used to generate possible scene flows, for every scene flow explored and relevant entry in the histogram is incremented.

The histogram is reduced to a 3D histogram by taking the maximum scene flow for each point, this is because no single point can have more than one scene flow. This new histogram is then thresholded reducing the points to the ones with the highest values, these should be the surface points of the object as that is where the maximum concurrence should be.

4 Results

All four methods where implemented on the Human Eva II dataset [3]. Black et al. [2] optical flow algorithm was used to generate the optical flow for each of the four cameras. The two frames for the four cameras can be seen in Figure 5, this includes the result of a background subtraction and the thresholded optical flow. The results of the four methods can be seen in Figure 6.

4.1 Background Subtraction

The scene flow derived using background subtraction is shown in Figure 6 (a). This can be visually inspected and it appears to be very accurate to when compared with the two frames of the scene. This method does require a framework which calculates the background subtraction. While a lot of human motion tracking algorithms use some form of background subtraction it is not always possible to have a robust background subtraction. So while the result is very good it has its limitations in terms of the required prior knowledge of the background of the scene. Figure 7 shows the difference between the original optical flow and the projection of the new scene flow in camera four. Most of the difference is very small, but there are bands and patches of large difference. It is unknown if this difference is due to error in the scene flow or in the original optical flow which has been corrected or a combination of the two.

4.2 Thresholding Optical Flow

The results of the thresholding of the optical flow can be seen in Figure 5. When compared with the background subtraction a good detection of objects is observed but the edges are not as clean and some parts of the body disappear. The scene flow derived from this method can be seen in Figure 6 (b). Due to the error in segmentation the resulting scene flow is not as good as when using background subtraction but the general outline of the body is visible. The motion of the head in camera 3 is in-line with the camera therefore the head disappears in the thresholded optical flow this therefore causes the head to disappear in the scene flow. The advantage to this method is that no prior information about the scene is required. In theory using some better adaptive thresholding techniques and building information up over time could produce better results.

4.3 Thresholding Scene Flow

The scene flow results obtained by thresholding the scene flow can be seen in Figure 6 (c). Again this is not as good as the scene flow derived from the background subtraction, but the outline of the person is visible. This method solves the problem of the previous method of when the motion is in-line with a camera, but it still suffers where the body is not moving, again over time this could be filled in as knowledge of the scene builds up.

4.4 Histogram Data

The scene flow results obtained by using a histogram of all possible scene flows can be seen in Figure 6 (d). This results is also not as good as the scene flow derived from the background subtraction, but again the outline of the person can be seen. This method assumes no prior knowledge of the scene. This method can be computationally intensive due to the 6D histogram, and the large number of scene flows required. To get good results requires fine tuning to pick the correct number and sizes of each of the dimensions of the histogram. Also the resulting scene flow is quantised and limited to the size of the dimensions of the histogram.

One overall disadvantage of using the Human Eva II dataset was the position of the cameras which is four cameras each at a corner of a room. This arrangement means that a surface point is only every visible from two cameras, this only allows one estimation of the scene flow. With more cameras it might be possible to average the scene flow over multiple estimations and achieve a better result.

5 Conclusions and Future Work

In this paper, we have presented four methods for determining the scene flow given multiple optical flows. From visual inspection of the results it can be seen that the background subtraction method performed the best, but the general motion can be seen using the other methods.

Further work is required to quantify the accuracy of the scene flow, this can be done using the motion capture data supplied with the Human Eva II dataset. Further work is required in experimenting with more cameras to have a surface point visible to more then just two cameras. Further work is also required to see how both colour information and temporal information can improve the result.

Acknowledgement

This work has been supported by an Innovation Partnership between Sony-Toshiba-IBM and Enterprise Ireland (IP-2007-505) and forms part of Trinity's Sony-Toshiba-IBM European Cell/B.E. Center of Competence.

References

[1] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Comput. Surv.*, 27(3):433466, 1995.

[2] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 296–302, 1991.

[3] M. J. Black and L. Sigal. HumanEva. http://vision.cs.brown.edu/humaneva/index.html.

[4] J. Bouguet. Pyramidal implementation of the lucas kanade feature tracker: Description of the algorithm. Intel Corporation, Microprocessor Research Labs, 2002.

[5] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[6] B. K. P. Horn and B. G. Schunck. *Determining optical flow*, page 389407. Jones and Bartlett Publishers, Inc., 1992.

[7] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *Proceedings of Imaging Understanding Workshop*, page 121130, 1981.

[8] C. Theobalt, J. Carranza, M. A. Magnor, and H. Seidel. Combining 3D flow fields with silhouette-based human motion capture for immersive video. *Graphical Models*, 66(6):333351, Nov. 2004.

[9] T. Y. Tian and M. Shah. Recovering 3D motion of multiple objects using adaptive hough transform. In *Proceedings of the Fifth International Conference on Computer Vision*, page 284. IEEE Computer Society, 1995.

[10] P. H. S. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, page 278294. Springer-Verlag, 2000.

[11] S. Vedula and S. Baker. Three-Dimensional scene flow. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(3):475480, 2005.

[12] Y. Zhang and C. Kambhamettu. On 3-D scene flow and structure recovery from multiview image sequences. *IEEE Trans. on Systems, Man, and Cybernetics - Part B*, 33(4):592606, 2003.

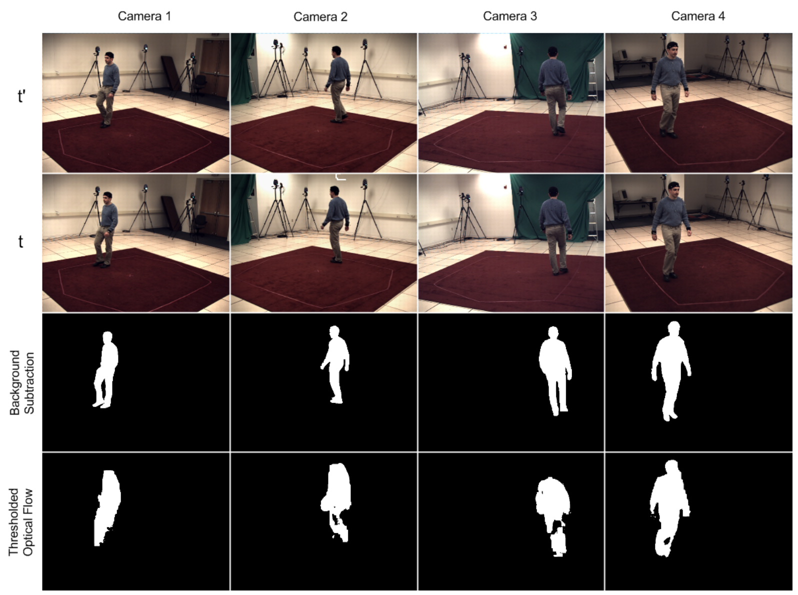


Figure 5. Two consecutive frames of the Human Eva II dataset consisting of 4 cameras also the result of the background subtraction and the thresholding of the optical flow.

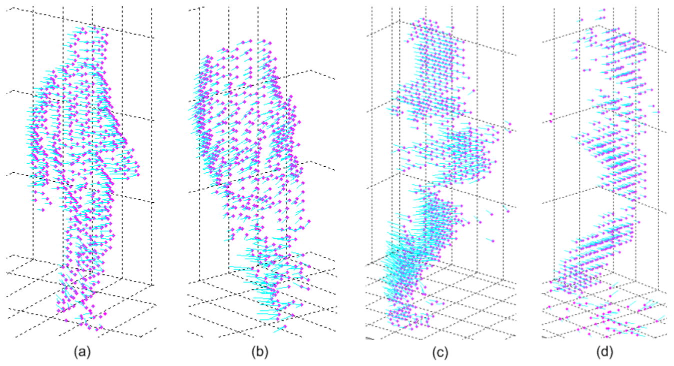


Figure 6. Results of the 4 scene flow algorithms. (a) using the background subtraction, (b) thresholding the optical flow, (c) thresholding the scene flow and (d) exploring the histogram data. The computation time for each scenario is (a) 1.8 seconds (b) 1.4 seconds (c) 7.9 seconds and (d) 117 seconds. This does not included the time taken to calculate the optical flow or background subtraction.

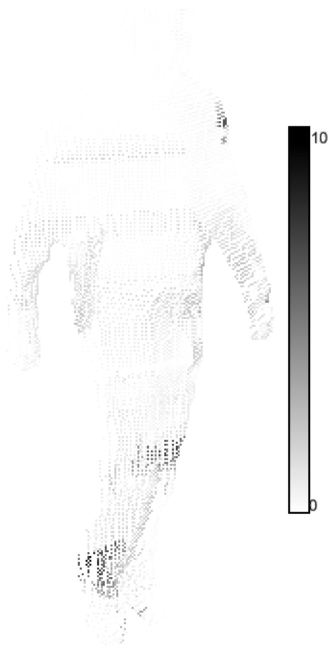


Figure 7. The scene flow result obtained using the background subtraction is projected back into camera 4, and the magnitude of the difference between the original optical flow and the new optical flow is plotted.