Non-Thesis Project

Optical flow based vehicular tracking

Anmoljeet Gill[1]*

[1] School of Computer Science, McGill University, Montreal, H3A 2A7, Canada

*To whom correspondence should be addressed.

Abstract

Motivation: The goal of this project is to determine the viability of using optical flow to track vehicles. Two optical flow algorithms, sparse optical flow and dense optical flow, were used as a basis to implement tracking on vehicles in a variety of test videos. These were compared to a simpler algorithm of background subtraction to test accuracy and frame rate, ultimately to determine which algorithm has the maximized frame rate along with high accuracy.
Results: Sparse optical flow is shown to be algorithm with the highest frame rate on all test videos with a comparable accuracy to dense optical flow and a higher accuracy compared to background subtraction.
Contact: anmoljeet.gill@mail.mcgill.ca
Supplementary information: Code is submitted along with this report at the following GitHub repository: https://github.com/aznmahou/OpticalFlowTracking.

1 Introduction

Object tracking is a field with a wide depth of usages such as surveillance, self driving cars, automation,and human-computer interactions. It is a sprawling area of research which is both very active and advancing at a breakneck pace.

Our project concerns a small part of the field of object tracking, called vehicular tracking, which focuses on tracking vehicles. With this we attempt to obtain accurate vehicular tracking with the frame rate maximized. There are multiple ways to do this, such as recent methods that involve deep learning. In deep learning, the problem and its mechanics are inputted into a Neural network to provide solution. A limitation to this approach is that it is computationally expensive. Our method avoids this limitation by focusing on the less computationally costly algorithms of Optical Flow. With Optical Flow we determine motion to ultimately track vehicles.

In this paper, we will first discuss the various Optical Flow algorithms and explain them thereafter we will talk of the methodology with using these algorithms and others to track vehicles. Finally, we will compare the results of our two optical flow algorithms to a test algorithm of Background Subtraction. All three algorithms will be then tested with sample videos for frame rate and accuracy. Then we will discuss which algorithm produced the best results and what improvements could be made to them.

2 Optical Flow

In this section we discuss what Optical Flow is and the two different Optical Flow algorithms that were used, Lucas-Kanade (1) and Gunnar-Farneback(2).

Optical Flow is the motion of objects between different frames in a sequence. The motion being the movement of the object in respect to a camera or including the cameras movement if it is not fixed in place. Optical flow can be expressed as shown in Figure 1. With the assumption of the object we are tracking does not change in pixel values, we can obtain the optical flow equation using Taylor Series Approximation shown in Figure 2.

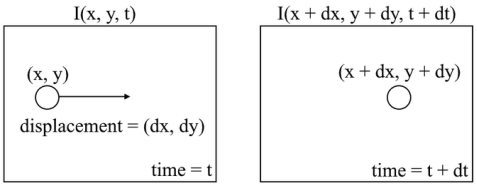

Fig. 1. Left: Initial Frame with object having position (x, y) and a movement of (dx, dy). Right: Subsequent Frame with object having position $(x + dx, y + dy)$.

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

$$u = \frac{dx}{dt} \; ; \; v = \frac{dy}{dt}$$

Fig. 2. Equation of optical flow, where $\frac{dI}{dx}, \frac{dI}{dy}$ are the image gradients, $\frac{dI}{dt}$ is the gradient along time, u,v are the unknowns to be solved.

As we have two unknowns and only one equation we must use approximation methods. We used two primary optical flow algorithms. Firstly Lucas-Kanade for sparse optical flow which is tracking a small

set of features for a faster but less accurate tracking. Secondly, Gunnar-Farneback for dense optical flow where the entire image is tracked.

2.1 Lucas-Kanade

The Lucas-Kanade algorithm computes $\frac{dI}{dx}, \frac{dI}{dy}$ and $\frac{dI}{dt}$ between two frames and solves the unknowns u and y with a least-squares approximation. This is done by assuming the displacement of an object between two different frames is small and relatively constant within a range around the feature (this being a selected pixel) we are considering.

With these assumptions, we take a small window around our feature and assume that these pixels have the same motion which now gives us the set of equations in Figure 3. These denote the equation for each pixel from 1 to n, with these equations we can organize them into a matrix form of $Av = b$, where A is our image gradients, v our unknowns and b our time gradients.

$$
\begin{aligned}
I_x(q_1)V_x + I_y(q_1)V_y &= -I_t(q_1) \\
I_x(q_2)V_x + I_y(q_2)V_y &= -I_t(q_2) \\
&\vdots \\
I_x(q_n)V_x + I_y(q_n)V_y &= -I_t(q_n)
\end{aligned}
$$

Fig. 3. Optical flow equations for a set of pixels from 1 to n.

Now with these, we have nine equations to solve two unknowns which results in an over-determined system that has no solution. To find a solution, least-squares approximation is applied resulting in the final equation of Figure 4.

$$
\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_y(q_i) I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix}
$$

Fig. 4. Optical Flow Matrix solution with least-squares approximation.

A limitation of this algorithm is that as Lucas-Kanade only works for small movements based on our assumptions, it will fail when there is large motion such as a entire vehicle moving. To solve this problem we use the method of pyramids, this method involves reducing the resolution of the image so that the large motion becomes a small motion, and repeat this to obtain a more accurate prediction. This method solves the problem but results in a higher amount of computations being required which directly can decrease frame rate. A representation of this is shown in Figure 5.

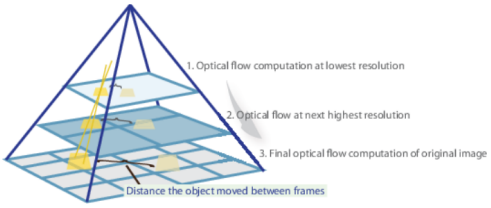

Fig. 5. Pyramid method, involving starting at a low resolution to get a solution and iterate this through higher resolution scales to obtain more accurate solutions.

2.2 Gunnar-Farneback

For Gunnar-Farneback, it is a two step algorithm. The first step, polynomial expansion is used to approximate around the neighborhood of every pixel, mainly using a quadratic polynomial. This is done on both images and equating both coefficients in the polynomials allows us to solve for the displacement or $u = \frac{dx}{dt}$ and $y = \frac{dy}{dt}$ for a pixel. The polynomials, and solution is shown in Figures 6 and 7.

$$
\begin{aligned}
f_1(\mathbf{x}) &= \mathbf{x}^T \mathbf{A}_1 \mathbf{x} + \mathbf{b}_1^T \mathbf{x} + c_1 \\
f_2(\mathbf{x}) &= f_1(\mathbf{x} - \mathbf{d}) = (\mathbf{x} - \mathbf{d})^T \mathbf{A}_1 (\mathbf{x} - \mathbf{d}) + \mathbf{b}_1^T (\mathbf{x} - \mathbf{d}) + c_1 \\
&= \mathbf{x}^T \mathbf{A}_1 \mathbf{x} + (\mathbf{b}_1 - 2\mathbf{A}_1 \mathbf{d})^T \mathbf{x} + \mathbf{d}^T \mathbf{A}_1 \mathbf{d} - \mathbf{b}_1^T \mathbf{d} + c_1 \\
&= \mathbf{x}^T \mathbf{A}_2 \mathbf{x} + \mathbf{b}_2^T \mathbf{x} + c_2.
\end{aligned}
$$

Fig. 6. Quadratic polynomials for frame 1 and frame 2, with A being an symmetric matrix, b a vector and c a scalar and being estimated from a weighted least-squares approximation.

$$
\begin{aligned}
\mathbf{A}_2 &= \mathbf{A}_1, \\
\mathbf{b}_2 &= \mathbf{b}_1 - 2\mathbf{A}_1 \mathbf{d}, \\
c_2 &= \mathbf{d}^T \mathbf{A}_1 \mathbf{d} - \mathbf{b}_1^T \mathbf{d} + c_1. \\
\mathbf{d} &= -\frac{1}{2} \mathbf{A}_1^{-1} (\mathbf{b}_2 - \mathbf{b}_1).
\end{aligned}
$$

Fig. 7. Coefficients of both frames being equated and the displacement d being solved.

The limitation of the algorithm only working for small movements in Lucas-Kanade is also present in Gunnar-Farneback. To solve the issue we once again use a pyramid method of decreasing the resolution.

3 Methodology

In this section we will discuss the full methodology of how we implemented sparse optical flow tracking and dense optical flow tracking. In both optical flow implementations, the input video was shrunk down from its original size to 600x600 pixel size to increase processing speed and thus frame rate.

3.1 Sparse Optical Flow

Sparse optical flow is where an optical flow algorithm, in our case Lucas-Kanade is run on a few (thousand or so) key features. In our case we used a algorithm called Shi-Tomasi corner detection (3) to determine our features. Corners are valuable features as they are invariant to translation, scale, rotation and intensity changes. Corner detection algorithms work in general in the following three steps.

1. Determine which window (small section of the image) produces a large variation in pixel intensity when moved in both horizontal and vertical directions.
2. For each window found, calculate a score value.
3. With this score value, each window can be classified as a corner, edge or a uniform region. Take the corners and edges with the largest score value.

With our important features, the corners of the vehicles, they are fed into Lucas-Kanade and a optical flow is calculated for each of them. We then update our features based on our new position. To do actual tracking we draw over the patch the features take to find the path the vehicles take. One final important step is to recalculate our features with corner detection again after some time has passed. This step is for handling cases where a vehicle has moved out of frame as the final features of that vehicle will end at the borders of the image.

Below in Figure 8, we show feature detection along with motion lines on a frame of one test video. By increasing or decreasing the number of corners we allow to be detected we can alter how much is detected.

Fig. 8. The green circles are our features detected by Shi-Tomasi corner detection and the green lines following the circles are the motion of those corners.

3.2 Dense Optical Flow

Dense optical flow is the motion of the entire image (in our case the frame of a video) being calculated, this is costly compared to sparse but allows a much higher accuracy.

Firstly, we take our frame and run the Gunnar-Farneback algorithm and this gives us the optical flow of the frame in a 2d array of $\frac{dx}{dt}$, $\frac{dy}{dt}$ for every pixel. With this we then convert it from the x,y into polar coordinates. This gives us the magnitude or distance of the flow and the direction or angle of the flow. We then use these two values to create a HSV(Hue,Saturation,Value) color representation, with the hue being the angle of our flow movement and the value being the magnitude of our flow. This is afterwards converted into RGB and is shown below in Figure 9.

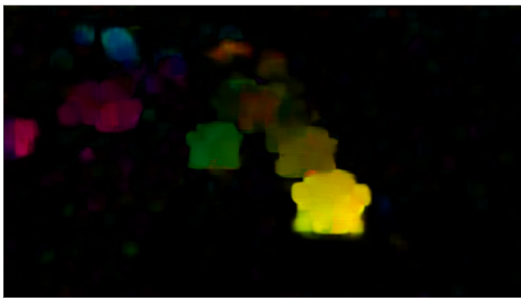

Fig. 9. The optical flow of a frame converted to RGB and shown.

One issue that occurred during the project was that the flow created by the algorithm were blurred together making it difficult to recognize separate objects. This was solved by using morphological transformations (4). First, erosion was used over the flow to reduce the contact between the separate vehicles. Afterwards the technique of opening (erosion followed by dilation) to reduce noisy pixels. Then closing (dilation followed by erosion) was applied to make sure the flows has no holes left by the previous operations. We then layer our flow over the initial frame and our tracking is done.

In Figure 10, one frame of a test video is shown with the dense flow being show.

Fig. 10. The vehicles are covered in a white outline of the dense flow based on each vehicle.

4 Results

Our main goal for this project was to obtain vehicular object tracking with a priority on maximized frame rate with sufficiently high accuracy. As a benchmark was required we used a algorithm called background subtraction (5). This algorithm takes an image and calculates the foreground, the objects of interest from the current image ,and a model of the background.

The three algorithms of background subtractor, sparse optical flow ,and dense optical flow were tested each for two different metrics: frame rate and accuracy. They were all tested on five different test videos (provided in the GitHub repository) obtained from various sources. The videos are varied, with them having both horizontal and vertical movement. Another feature of the videos was they varied in the density of vehicles going from a empty highway to a gridlocked road. The videos were as following:

1. 1 minute long, average vehicular density, vertical movement.
2. 57 seconds long, average vehicular density, vertical movement.
3. 13 seconds long, average vehicular density, horizontal movement.
4. 21 seconds long, low vehicular density, vertical movement.
5. V 1 minute long, high vehicular density, vertical movement.

For the first metric, the algorithms were ran on the various videos and the frame rate was calculated based on the total number of frames in a video and the run-time of the algorithm to complete. The results are shown in Table 1, Overall, we can judge that the sparse optical flow algorithm has the highest frame rate in all test cases. This is likely due to the background subtractor requiring more calculations and the dense optical flow calculating based on the entire frame. Meanwhile the sparse optical flow calculates a limited number of features much less than every pixel.

	Video 1	Video 2	Video 3	Video 4	Video 5
Background Subtractor	15.2	26.7	7.3	7.9	5.3
Sparse Optical Flow	37.3	38.1	16.0	14.3	12.7
Dense Optical Flow	6.6	3.6	3.1	2.7	3.3

Table 1. Frame rate for each different algorithm on all five test videos. The frame rate is in frames per second.

The second metric, the accuracy was slightly harder to test. As the three methods used different tracking, it was decided the accuracy was best tested by human testers who can judge the accuracy. Four different individuals were shown the videos (including the author) and were asked to judge the accuracy of the algorithms on the different videos. They were asked to judge which of the three different algorithms for each video was the most accurate in terms of vehicular tracking. Below in Table 2 the results are shown. The most accurate method was dense optical flow. This was the expected result due to the methodology of the algorithm taking in the entire frame and using all available information to determine motion.

	Video 1	Video 2	Video 3	Video 4	Video 5
Tester 1	DOF	SOF	SOF	DOF	SOF
Tester 2	DOF	BF	DOF	DOF	DOF
Tester 3	SOF	DOF	DOF	DOF	DOF
Tester 4	DOF	SOF	DOF	DOF	SOF

Table 2. Accuracy test results, where the participants were asked to pick the best algorithm for each video. An approximate clip of 15 seconds was shown for each algorithm. The abbreviations are as follows: BG = Background subtractor, SOF = Sparse optical flow, DOF = Dense optical flow.

4.1 Conclusion

The fastest method in terms of frame rate was sparse optical flow but the most accurate method was dense optical flow.

In terms of frame rate the Background Subtractor was a fast algorithm that ran smoothly and without much issues. The key issue with this algorithm was its lack of accuracy on most videos. One issue that it faced was in video 3, the algorithm was unable to properly distinguish between separate objects leading to a large bounding box on multiple objects. This problem propagated further and the bounding boxes went erratic and started covering non moving objects. Another issue was the severe issue with clumping objects in video 5 as it was a congested roadway. Overall this algorithm is accurate for low objects moving across the field of interest but once things become heavily dense the algorithm proceeds to fail.

These issues of objects clumping were also faced in dense optical flow as many vehicles were showing as one continuous object but were partially mitigated by our usage of morphological operations to reduce the connection between objects. An additional limitation of dense optical flow was the amount of processing time taken to run over a video, this is likely unreliable in a real world situation as the frame rates produced were low. A final issue was in some videos it was occasionally having a flashing effect, this seems to be when the video is large slowing down the algorithm. One strength of dense optical flow was its high accuracy as it accurately tracked all moving vehicles which is once again due to it calculating the entire motion gradient. This is a costly but accurate algorithm.

Finally for sparse optical flow, it had the highest frame rate of all the algorithms, but it had a set of issues itself. One issue was that with the corner detection done to obtain the features to track, there were many wasted corners on objects that were not moving. One such example of this was in video 3, the corners of a railing were being detected. A possible solution to this is to check if a corner has moved or not by comparing its position in a frame and then in the subsequent frame. In a combination of both accuracy and frame rate the best algorithm is sparse optical flow.

5 Discussion

To accomplish the goal of vehicular object tracking with a preference on maximizing frame rate, we used optical flow methods in two different ways and compared our results to the results of the algorithm of background subtraction.

Sparse optical flow was determined to be the most accurate of both optical flow methods and background subtraction. A few potential improvements that could be made to our methodology is improved corner detection for obtaining features. As we had many wasted features on non moving objects, implementing a motion check every few frames to remove the non moving features would free up many features to improve accuracy. Object recognition could be used preferentially over corner detection for feature selection, but this would be costly as you require expensive deep learning to be able to identify specific objects.

Dense optical flow while accurate had a hefty cost of low frame rates. Some improvements on this would be implementing some concurrency, that is being able to break up the total process of calculating the motion gradient of the entire frame into smaller parts. One possible way this could be accomplished is distributing the calculations for each neighbourhood of pixels to a GPU so each calculation is run at the same time. Then later combining the calculations to create a single resulting flow. This however still leaves the issue of blurring in dense flow. A way around this is once again object recognition to differentiate objects but once again it is a costly method.

Moving forward, it is best to use sparse optical flow for simple vehicular object tracking where we do not care to identify the object but instead wish to determine such details as position and motion compared to the camera. This would be useful is tasks such as self driving or road intersection management.

References

[1]Lucas, B. and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision. In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 674– 679.

[2]Farnebäck, Gunnar. "Two-frame motion estimation based on polynomial expansion." Scandinavian conference on Image analysis. Springer, Berlin, Heidelberg, 2003.

[3]Jianbo Shi and Tomasi, "Good features to track," 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 1994, pp. 593-600, doi: 10.1109/CVPR.1994.323794.

[4]Modvinstev, Alexander, and Abid K. Revision. “Morphological Transformations¶.” OpenCV, opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html.

[5]Software. Author: Ujj Wal; The source code for the background subtractor is available at

https://github.com/ujjwal3067/BGsubtractor?fbclid=IwAR0VwhprgUS6Cc3ZTgvaM7CKbSoCUbWAq9AAIEDoABjnrUdYtoKiL_9tzOs

[6]Mendes, Paulo AS, et al. "Movement detection and moving object distinction based on optical flow." Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2019. 2019.

[7]Hua, Shuai, Manika Kapoor, and David C. Anastasiu. "Vehicle tracking and speed estimation from traffic videos." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2018.