Homework 1 Report

Ashwin Goyal; UID: 115526297

May 02, 2018

Introduction

For this project, I had to compute and compare optical flow on some sequences of the Middlebury Optical Flow data set. I have implemented the Lucas-Kanade optical flow algorithm as described in [1] and [2]. My output has then been compared to MATLAB outputs using Lucas-Kanade Method, Farneback Method and Horn-Schunck Method.

Concept

The Lucas–Kanade method assumes that the displacement of the image contents between two nearby instants (frames) is small and approximately constant within a neighborhood of the pixel p under consideration. Thus the optical flow equation can be assumed to hold for all pixels within a window centered at p, that is, the local image flow vector (V_x,V_y) must satisfy

$$\begin{gathered} I_x(q_1)V_x + I_y(q_1)V_y + I_t(q_1) = 0 \\ I_x(q_2)V_x + I_y(q_2)V_y + I_t(q_2) = 0 \\ \vdots \\ I_x(q_n)V_x + I_y(q_n)V_y + I_t(q_n) = 0 \end{gathered}$$

where q_1, q_2, ... , q_n are the pixels inside the window, and $I_x(q_i)$, $I_y(q_i)$, and $I_t(q_i)$ are the partial derivatives of the image I with respect to position x, y and time t, evaluated at the point q_i and at the current time. These equations can be written in matrix form $Av = B$, where

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix}; v = \begin{bmatrix} V_x \\ V_y \end{bmatrix}; B = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

This system has more equations than unknowns and thus it is usually over-determined. The Lucas–Kanade method obtains a compromise solution by the least squares principle, that is, it solves the 2 x 2 system

$$A^T Av = A^T B$$
$$\Rightarrow v = (A^T A)^{-1} A^T B$$

$$\Rightarrow \boxed{\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n I_x(q_i)^2 & \sum_{i=1}^n I_x(q_i)I_y(q_i) \\ \sum_{i=1}^n I_y(q_i)I_x(q_i) & \sum_{i=1}^n I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{i=1}^n I_x(q_i)I_t(q_i) \\ -\sum_{i=1}^n I_y(q_i)I_t(q_i) \end{bmatrix}}$$

This is a plain least squares solution which gives the same importance to all n pixels q_i in the window. In practice, it is usually better to give more weight to the pixels that are closer to the central pixel p. For that, the weighted version of the least squares equation is used.

$$A^T W A v = A^T W B$$
$$\Rightarrow v = (A^T W A)^{-1} A^T W B$$

where W is an n x n diagonal matrix containing the weights w_i to be assigned to the equation of pixel q_i.

$$\Rightarrow \boxed{\boxed{\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n w_i I_x(q_i)^2 & \sum_{i=1}^n w_i I_x(q_i)I_y(q_i) \\ \sum_{i=1}^n w_i I_x(q_i)I_y(q_i) & \sum_{i=1}^n w_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{i=1}^n w_i I_x(q_i)I_t(q_i) \\ -\sum_{i=1}^n w_i I_y(q_i)I_t(q_i) \end{bmatrix}}}$$

The weight w_i is usually set to a Gaussian function of the distance between q_i and p.

Implementation

The first step in any project in perception is to read the image. For this project, the algorithm has been tested on Grove and Wooden sets of Middlebury grayscale data set. The steps implemented to solve for u and v using the Lucas-Kanade Method are:

1. Converted the image into double for simplification of computation.
2. Computed the partial derivatives of the image with respect to the position x (I_x), and y (I_y) using the kernel $\begin{bmatrix}-1 & 8 & 0 & -8 & 1\end{bmatrix}/12$ and its transposed form.
3. Computed the partial derivatives of the image with respect to time (I_t) between images 1 and 2 using the $\begin{bmatrix}-1 & 1\end{bmatrix}$ kernel. It should be noted that for the first frame, the previous frame is assumed to be a simple black image.
4. Smoothed the partial derivatives I_x, I_y, and I_t using a separable and isotropic 5-by-5 element kernel whose effective 1-D coefficients are $\begin{bmatrix}1 & 4 & 6 & 4 & 1\end{bmatrix}/16$.
5. Solved the 2-by-2 linear equations for each pixel using the eigen-value method. Let

$$A = \begin{bmatrix} a & b \\ b & c \end{bmatrix} = \begin{bmatrix} \sum wI_x^2 & \sum wI_xI_y \\ \sum wI_xI_y & \sum wI_y^2 \end{bmatrix}$$

Then the eigenvalues of A are

$$\lambda_i = \frac{a+c}{2} \pm \frac{\sqrt{4b^2 + (a-c)^2}}{2}; i = 1, 2$$

These eigenvalues are compared to the threshold (τ) that corresponds to the value for the threshold for noise reduction. The results fall into one of the following cases:

   (a) **Case 1:** $\lambda_1 \geq \tau$ and $\lambda_2 \geq \tau$
   A is non-singular, and the system of equations is solved using Cramer's rule.

   (b) **Case 2:** $\lambda_1 \geq \tau$ and $\lambda_2 < \tau$
   A is singular (non-invertible), and the gradient flow is normalized to calculate u and v.

   (c) **Case 3:** $\lambda_1 < \tau$ and $\lambda_2 < \tau$
   The optical flow, u and v, is 0.

Result

Using the aforementioned algorithm, the optical flow was computed for all the frames in the Grove data set and Wooden data set. An example output from Grove set and wooden set is shown in Figure 1 and Figure 2 respectively.

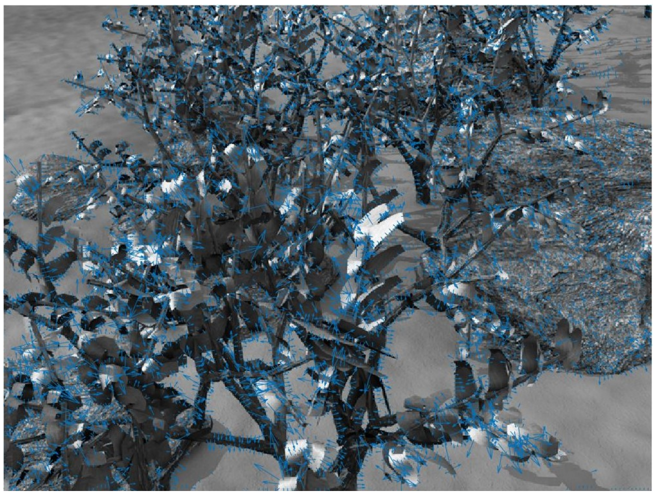
Figure 1: 4th Frame of the Grove set

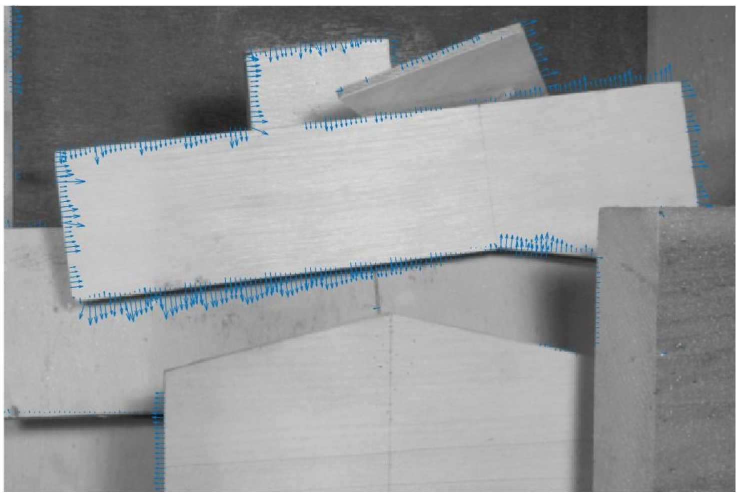
Figure 2: 5th Frame of the Wooden set

This output was compared with the outputs from the MATLAB built-in functions **opticalFlowLK** (Lucas-Kanade Method), **opticalFlowFarneback** (Farneback Method), **opticalFlowHS** (Horn-Schunck Method). An example output of these functions are shown from Figure 3 to Figure 8.

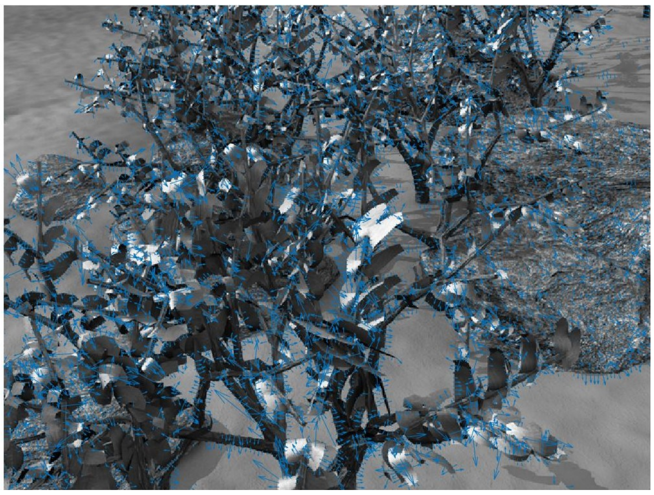
Figure 3: Lucas-Kanade Method (Grove)

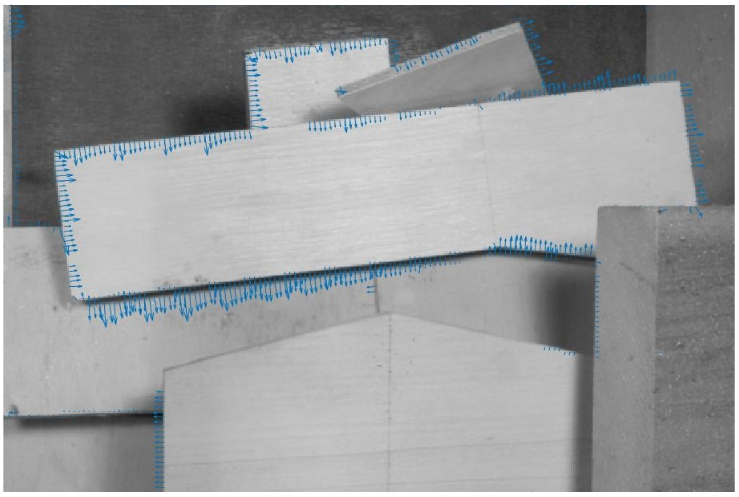
Figure 4: Lucas-Kanade Method (Wooden)

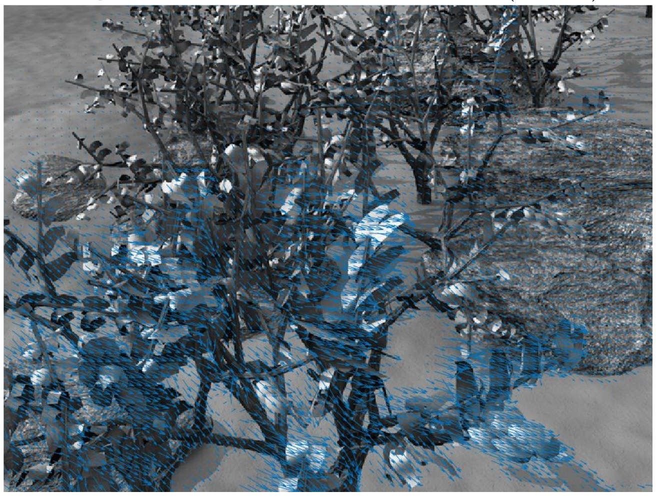
Figure 5: Farneback Method (Grove)

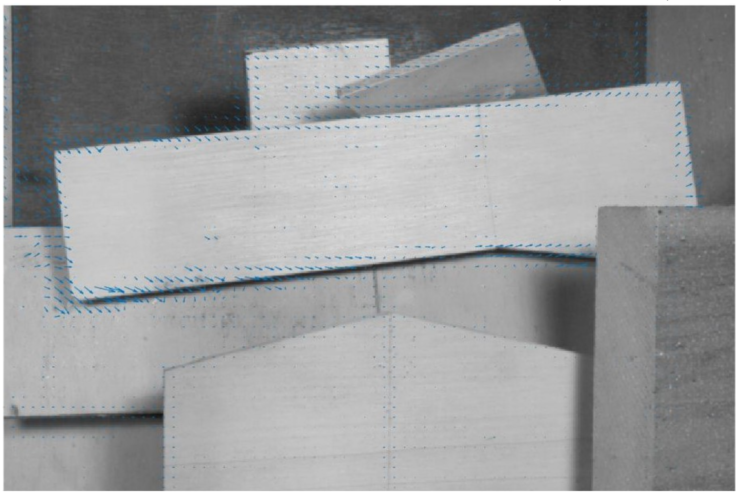
Figure 6: Farneback Method (Wooden)

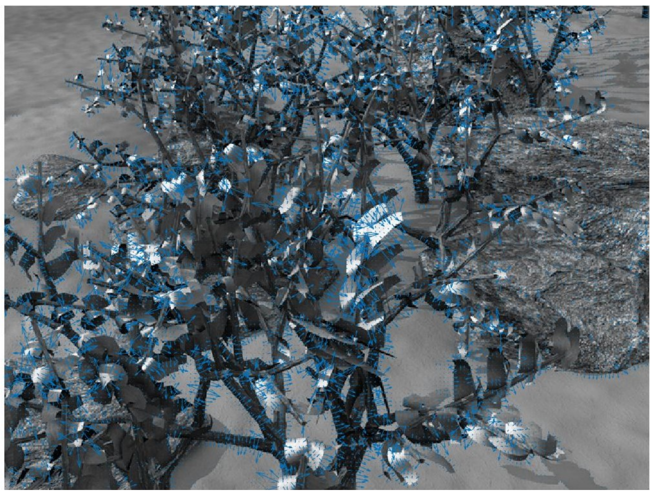
Figure 7: Horn-Schunck Method (Grove)

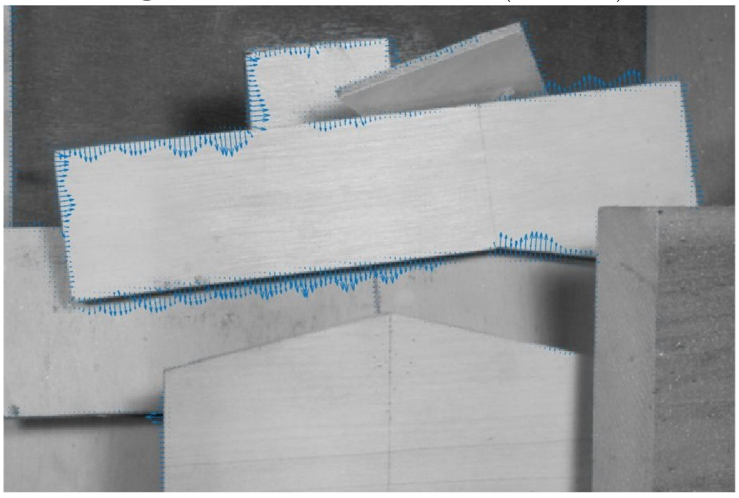
Figure 8: Horn-Schunck Method (Wooden)

Discussion

This section of the report compares the results achieved using different methods.

The first comparison is being done between my implementation and MATLAB's implementation of Lucas-Kanade Method. On comparing Figure 1 with Figure 3, or Figure 2 with Figure 4, it can be concluded that my implementation of the Lucas-Kanade method is correct. This inference is achieved as the only difference between the two images is the magnitude of the vectors which actually varies according to the scaling factor set by the algorithm.

Now, the three methods of optical flow estimation are compared to each other based on the outputs in the three different regions of the image.

1. **Textured Region.** To compare the outputs in textured region, I compared Figure 3, Figure 5, and Figure 7. It can be observed that the strength of the optical flow vectors from Horn-Schunck Method is much weaker than the strength of the vectors from Lucas-Kanade Method. This difference has been accounted for in the algorithm and still the strength of optical flow vector from Horn-Schunck Method is weaker. As for the Farneback Method, it works on the whole textured region as single body, that is, it has accounted for the continuous nature of the body, which has not been done by either of the two remaining methods.

2. **Non-Textured Region.** To compare the outputs in the non-textured region, I compared Figure 4, Figure 6, and Figure 8. It can be clearly observed that there are no optical vectors identified in non-textured regions in case of Lucas-Kanade method and Horn-Schunck Method. This is because these methods are based on the partial derivatives of the images with respect to position. In non-textured regions, the partial derivatives are close to zero. But, in case of Farneback method, there are weak optical vectors present in the non-textured region as well.

3. **Object Boundaries.** To compare the outputs at the object boundaries, I observed the differences in Figure 4, Figure 6, and Figure 8. It is evident that to get the optical flow vectors at the boundaries, Lucas-Kanade method or Horn-Schunck method should be used. In Farneback method, it does not identify the boundaries at all. It just computes the optical flow vectors in the global image. But as the former two methods are based on partial derivatives of the image with respect to position, they have strong optical flow vectors at the object boundaries. Also, among Lucas-Kanade and Horn-Schunck, Lucas-kanade has a stronger optical flow vector but Horn-Schunck has a more uniform output.

So, it can be concluded, that no method is superior than the other, but the most desirable output in this case is achieved from Lucas-Kanade Method.

References

[1] Lucas-Kanade Method - Wikiwand, *https://www.wikiwand.com/en/Lucas%E2%80%93Kanade_method*

[2] MATLAB Documentation, *https://www.mathworks.com/help/matlab/*

[3] ShareLaTeX, Online LaTeXEditor, *https://www.sharelatex.com/*