Optical Flow using Lucas Kanade Algorithm

Swetanjal Murati Dutta
International Institute of Information Technology, Hyderabad
Gachibowli, Telangana
swetanjal.dutta@research.iiit.ac.in

Abstract

This is a report for Assignment 5 of the Computer Vision Course taught by Dr. Avinash Sharma during Spring Semester of 2020 at IIIT-H. The goal of this assignment is to familiarise myself with the idea of Optical Flows by implementing the Lucas Kanade Algorithm from scratch.

1. Introduction

The problem definition of optical flow is to predict the apparent motion(flow) of pixels between two consecutive frames. This is done by extending the fundamental Taylor Approximation to images. We had already seen previously how gradients are computed in the context of digital images. We these ideas to compute Optical Flow, the details of which are presented in the subsequent sections of this report.

Two fundamental assumptions used in the computation of Optical Flows are:

- The color of the pixels is constant through out the track. This allows for pixel to pixel comparison and does not require the need of image features like SIFT and alike.
- Small motion of pixels between successive frames. This allows us to use first order approximation of Taylor Series i.e linearization of brightness constancy constraint(presented in Section 3).

Note the above assumptions are unique to Optical Flow. These assumptions allow us to approach the problem as "look for nearby pixels with the same color".

The other important assumption used in Lucas Kanade Algorithm is nearby pixels have the same displacement and flow is locally smooth.

2. Taylor Series Approximation

The Taylor Series Approximation is fundamental in computing Optical Flows. It states that a function can be approximated in terms of it's gradients/derivatives as follows:

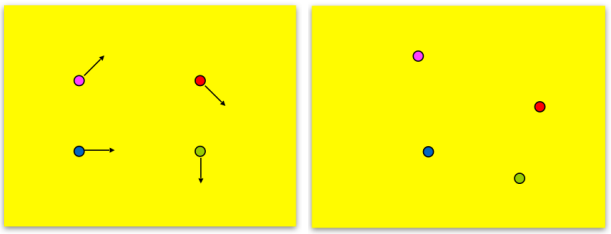


Figure 1: (a) The snapshot of $I(x, y)$ at time step t. (b) The snapshot of $I(x, y)$ at time step $t + \delta t$. The problem of Optical flow is to estimate the motion between the above two successive images

$$f(x) = f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+... \quad (1)$$

where $f'(x_0)$ is the first derivative of f with respect to x evaluated at x_0, $f''(x_0)$ is the second derivative of f with respect to x evaluated at x_0 and so on. The higher order terms are ignored as they become very small when x tends to x_0.

For the purposes of Optical Flow, we use the first order approximation. In other words, we approximate f as follows:

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) \quad (2)$$

The approximation is more and more accurate as x tends to x_0. This leads us to develop the second major assumption presented in the previous section i.e the motion is small between corresponding pixels of subsequent frames.

The Taylor Approximation holds good for functions with multiple parameters as well:

$$f(x,y) \approx f(x_0,y_0)+\frac{\partial f(x_0,y_0)}{\partial x}(x-x_0)+\frac{\partial f(x_0,y_0)}{\partial y}(y-y_0) \quad (3)$$

This allows us to directly extend Taylor Approximation to images.

2.1. Deriving the Brightness Constancy Equation

The equation developed in this section will lead us to solve the problem of Optical Flow.

Let us consider a pixel at location (x, y) at instant t. Assume this pixel moves by a small amount to location $(x + \delta x, y + \delta y)$ at instant $t + \delta t$. From the Brightness Constancy assumption,

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \tag{4}$$

From Taylor Series Approximation,

$$I(x, y, t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t \tag{5}$$

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0 \tag{6}$$

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{7}$$

$$I_x u + I_y v + I_t = 0 \tag{8}$$

where u and v are optical flow vectors along X and Y directions at pixel location (x, y) and I_x, I_y, I_t are image gradients along horizontal, vertical and time axes respectively.

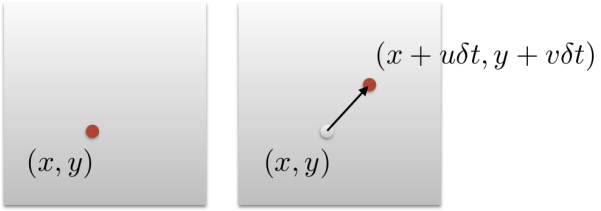


Figure 2: (a) The snapshot of $I(x, y)$ at time step t. (b) The snapshot of $I(x, y)$ at time step $t + \delta t$. This diagram is presented to make the above derivation clear along with the notations.

3. Solving the issue with Brightness Constancy Equation

The problem with the equation 8 is that there are two unknowns, u and v which are nothing but the optical flow vectors along X and Y directions at pixel coordinates (x, y) but we have only one equation to solve it.

To solve this problem, we take a small patch(say 5 x 5 neighbourhood) around every pixel and make use of all the neighbouring pixels' equations. However, in doing so we have to assume that the neighbouring pixels also undergo the same displacement as the pixel (x, y).

4. Mathematical Formulation of Lucas Kanade Method

If we use a m x m window, we get a stack of $n = m^2$ equations as follows:

$I_x(q_1)u + I_y(q_1)v = -I_t(q_1)$
$I_x(q_2)u + I_y(q_2)v = -I_t(q2)$
.
.
.
$I_x(q_n)u + I_y(q_n)v = -I_t(q_n)$

We have n equations but only 2 unknowns. This leads us to the popular Optimization problem of least squares. Stacking up the above equations in matrix form, we get:

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1)) \\ I_x(q_2) & I_y(q_2) \\ . & . \\ . & . \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \quad x = \begin{bmatrix} u \\ v \end{bmatrix} \quad b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ . \\ . \\ -I_t(q_n) \end{bmatrix}$$

Mathematically, the problem of least squares can formulated as follows,

$\min_x ||Ax - b||$

The above objective can lead to unconstrained optimization. Hence, we optimize subject to the constraint $x^T x = 1$. Adding Lagrangian and taking derivatives and equation them to 0, we get the following solution to the above problem,

$x = (A^T A)^{-1} A^T b$

One can observe that $A^T A$ is nothing but Covariance matrix. Simplifying further, we find that the least squares solution can be obtained by solving:

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum -I_t I_x \\ \sum -I_t I_y \end{bmatrix}$$

The solution exists when inverse of $A^T A$ exists. We know inverse of a matrix exists when the columns are independent. Such a condition is met when both the eigen values of the covariance matrix $A^T A$ are very large. This is familiar to us from the derivation in Harris Corner Detector. In other words, for corner points the solution would exist. We define these points for which solution exists as 'good features to track'.

5. Results

We present a few results from Middlebury Dataset provided for this assignment. We also present results of tracking and segmentation in videos.

The videos are attached in the zip folder.

6. Dataset

Due to file size constraints in submission portal, please download the input data and output data and other relevant videos from here

(a) Arrows showing optical flow

(b) Image segmentation computed from gradients

Figure 3: Illustration of optical flow predicted from Lukas Kanade Algorithm

(a) Arrows showing optical flow

(b) Image segmentation computed from gradients

Figure 4: Illustration of optical flow predicted from Lukas Kanade Algorithm

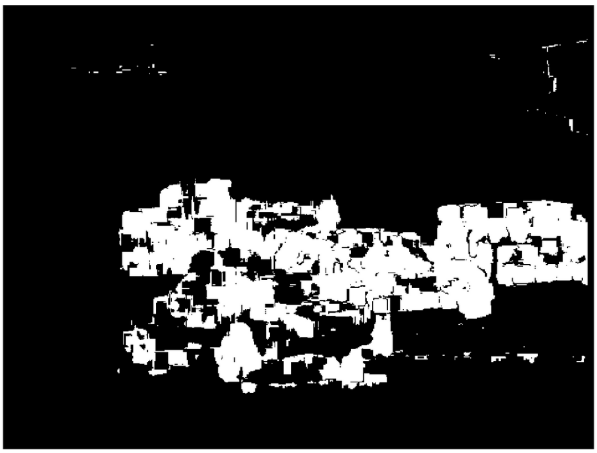
The official repository containing the source code and results can be found here. Note that this repository is currently private. It would be made public after the end of the semester. TAs can be given access on special request.

7. Some important observations

- The Lucas Kanade Algorithm doesn't perform well on flat surfaces. The gradients must be prominent in order for the optical flow arrows and segmentation to be correct.
- In almost all practical purposes, the assumption that movement between successive frames is small is violated. For this reason, implementing a Hierarchical Lucas Kanade Algorithm is essential. This Hierarchical Lucas Kanade Algorithm makes use of the image Pyramid. Lesser the resolution, lesser is the movement of pixels.
- In some cases, the linearity assumption is violated. In such a situation we should warp and iteratively compute the gradients for each pixel.

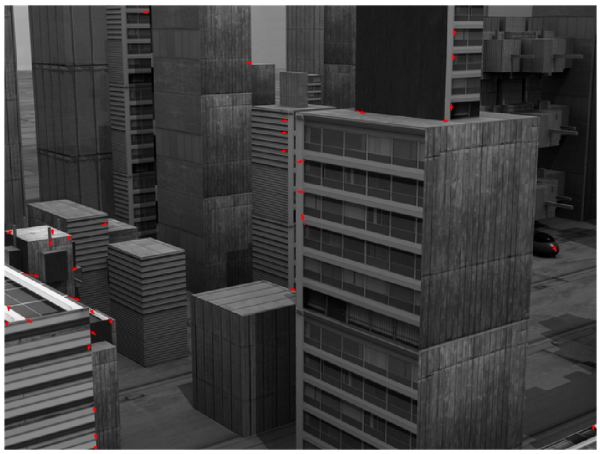
References

(a) Arrows showing optical flow

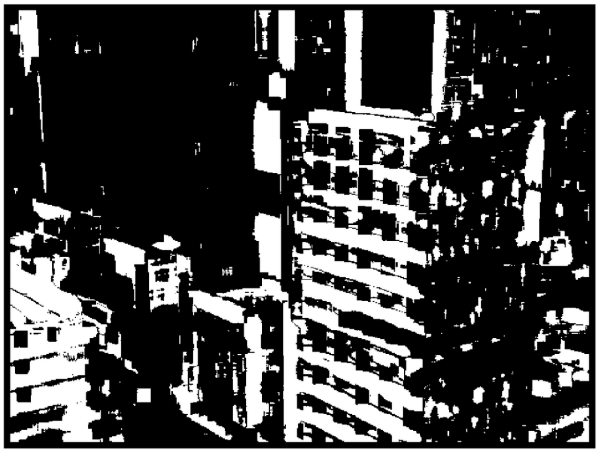


(b) Image segmentation computed from gradients

Figure 5: Illustration of optical flow predicted from Lukas Kanade Algorithm

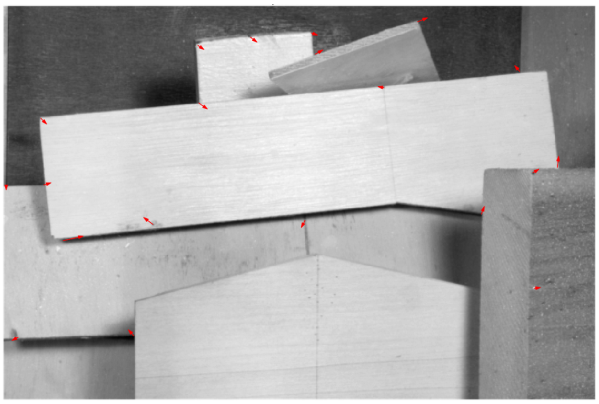


(a) Arrows showing optical flow

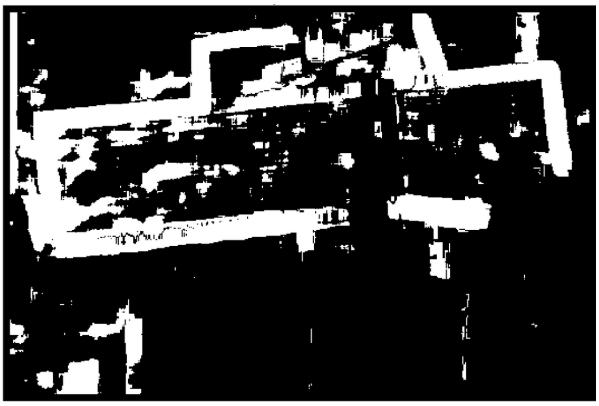


(b) Image segmentation computed from gradients

Figure 6: Illustration of optical flow predicted from Lukas Kanade Algorithm

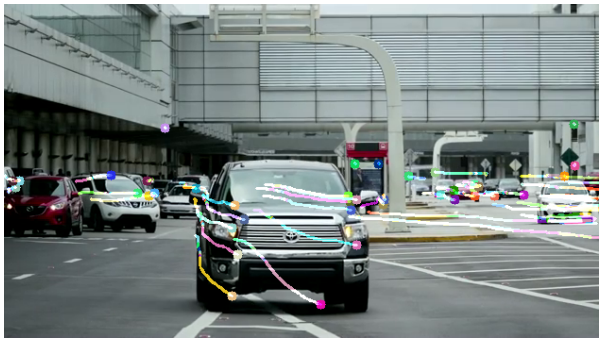


(a) Arrows showing optical flow

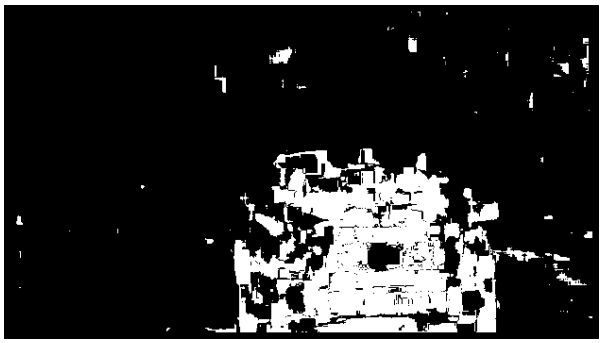


(b) Image segmentation computed from gradients

Figure 7: Illustration of optical flow predicted from Lukas Kanade Algorithm



(a) Tracks generated from a video. Refer video(tracks_video.mp4) attached in this zip folder for a better visualization



(b) Image segmentation mask generated from a video. Refer video(segmentation_video.mp4) attached in this zip folder for a better visualization

Figure 8: Lucas Kanade Optical Flow Results from Video