ENPM673-Project 4

Mushty Sri Sai Kaushik, Alexandre Filie, Santhosh Kesani

April 21, 2020

1 Introduction

The Lucas Kanade Tracker is used in computer vision applications as a differential method for estimating optical flow using brightness constancy. This method works based on the assumption of constant flow in the local neighborhood of the pixel and solving the optical flow equations for all the neighboring pixels using least squares criterion. As a part of this project, we intend to estimate this information of several such nearby pixels and resolve the inherent ambiguity of the optical flow equations extending it to tracking.

2 Fundamentals

We implement the Lucas Kanade for efficient image alignment and extending it to a consistent framework without significantly better efficiency. The approached we have adhered here is using gradient descent which is the standard and can be performed in a variety of different ways where we estimated using an additive increment to the parameters and another method where we used an incremental warp computed together with the estimated current warp. The Lucas Kanade method assumes that the displacement of the image contents is too small and almost constant with a neighborhood point under consideration. The approach of the Lucas Kanade method is as below. Starting with an initial Affine transform parameter p we can compute optimal Δ p iteratively, where in each iteration the objective function is linearized by 1st order Taylor expansion and solved by a linear system with form A Δ p where Δ p is the template offset.

$$\sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x}; \mathbf{p} + \Delta\mathbf{p})) - T(\mathbf{x}) \right]^2$$

$$\sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{x}; \mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta\mathbf{p} - T(\mathbf{x}) \right]^2$$

with respect to Δp and update the estimates of the parameters such that

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$$

3 Implementation of tracking

The basic structure of the Lucas-Kanade algorithm is given as:

The Lucas-Kanade Algorithm

Iterate:

(1) Warp I with $\mathbf{W}(\mathbf{x};\mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
(2) Compute the error image $T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
(3) Warp the gradient ∇I with $\mathbf{W}(\mathbf{x};\mathbf{p})$
(4) Evaluate the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $(\mathbf{x};\mathbf{p})$
(5) Compute the steepest descent images $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$
(6) Compute the Hessian matrix using Equation (11)
(7) Compute $\sum_{\mathbf{x}} [\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}]^{\mathrm{T}} [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))]$
(8) Compute $\Delta\mathbf{p}$ using Equation (10)
(9) Update the parameters $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$

Our goal is to minimize the sum of squared error Δ p between two images, the template T and the image I and Δ p = [p0, p1, p2, p3, p4, p5, p6]. The following steps are iteratively applied till Δ p converge.
The steepest descent and the Hessian is given as follows:

Steepest Descent Image: The Wrapped gradient is flattened and multiplied to get steepest descent image. It is of order (m*n)x6. It is calculated using the formula below.

Hessian - H: Hessian is calculated by multiplying transpose of the steepest descent and steepest descent. The resultant matrix has the order of 6x6.

$$H = \sum_{\mathbf{x}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right]^{\mathrm{T}} \left[\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \right].$$

4 Tracking Performance

During implementation of the Lucas-Kanade tracker it can be observed that the system is influenced by various factors that will need to be accounted for in more advanced implementations. Variation in image brightness and shadows present in the frame can influence the tracker and offset the estimated track point. This is visible in videos such as the car video with tree shadows changing the visible color and shape of the vehicle ahead. This influences the interpretation of the window relative to the template and can cause unwanted shifts in the track center.

Figure 1: Image of cropped template

Figure 2: Image of cropped and warped template

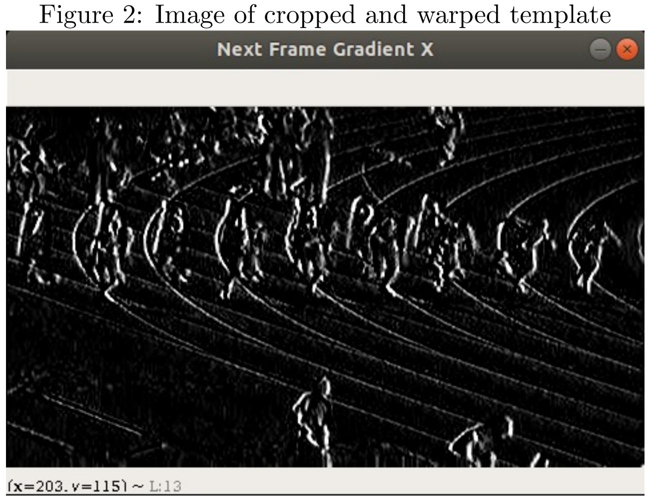

Figure 3: Image of gradient X

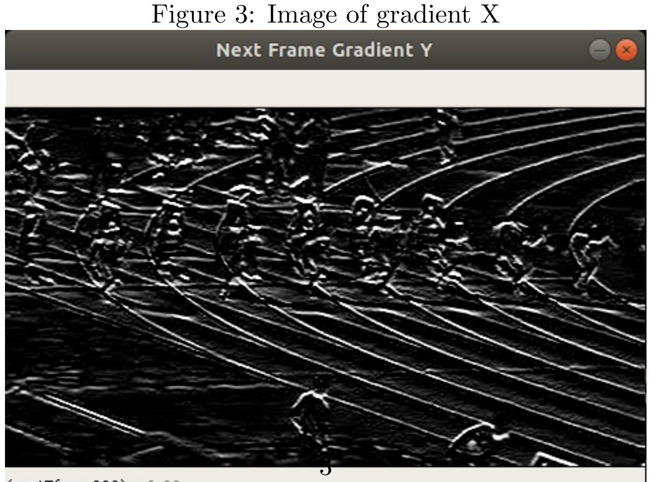

Figure 4: Image of gradient y

A similar scenario is observed in the Bolt video as there are other "objects" around the primary target that can change the visible window relative to the template, causing the tracker to shift. In some cases, it is possible this interference cause the tracker to lose track of the original target and start tracking surrounding objects that are moving in a similar trajectory and speed.

Another condition that affects tracking performance as observed in this project is variation in speed. It can be observed in some conditions that the tracker can be thrown off by the predicted speed/acceleration of the target. This causes variations in the tracking center until the tracker can focus on and catch up to the intended target. The car video did not exhibit these issues as the relative change in speed is small compared to the other videos.

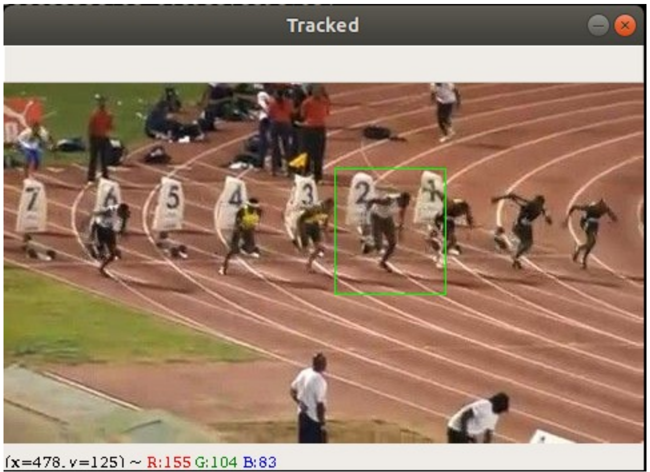

Figure 5: Tracked image of bolt

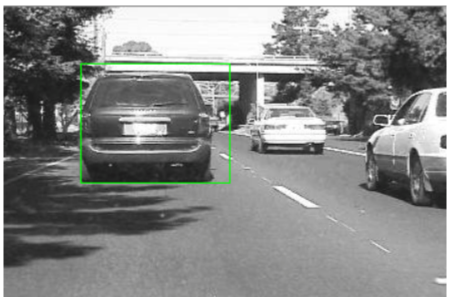

Figure 6: Tracked image of car

Figure 7: Tracked image of car

5 Roubstness to Illumination

The least-squares approach of the Lucas Kanade tracker assumes that the errors obtained in the image data have a Gaussian with mean zero. Although, if the window contains outliers, we opt to statistical analysis to detect them and reducing their weights respectively. Also, the Lucas Kanade tracker does not hold good when there is a change in illumination. This is mainly because the sum of all the squared distance errors we try to minimize have become sensitive to these illumination changes. We implement the following the approaches to handle this scenario.

1. **Scaling By Average Brightness**:

The traditional Lucas Kanade tracker is sensitive to brightness changes. We computed the average brightness of the pixels in the template and scale the brightness of the pixels in each frame such that average brightness of the tracked region is the same as the template's average brightness. By doing this, we remove some potential error scenarios for changes in illumination as we first normalized the template and then the tracked window by subtracting off mean and divided by standard deviation. This improves the Lucas Kanade tracker's efficiency in scenarios where there is a sudden illumination change.
We have used the gamma function to perform image brightening.

2. **Huber loss**:

Huber loss is a better robust M-estimator compared to the least squares approach as it is less sensitive to the outliers in the data than the squared error loss and both represented as a function of y - f(x) is represented below to contrast and compare.
Huber loss has a differentiable extension to an affine function and its properties allow it to combine most of the sensitivity and robustness of the M-estimation. Now this becomes a weighted least square problem where each residual term has a corresponding weight and the minimization function