ENPM673 - Perception for Autonomous Robots
Project 4

Cheng Chen, Haixiang Fang, Timothy Werder

Due April 20, 2020

Introduction

In computer vision, one of the most controversial and widely used aspects is tracking. However, the ability to track and object autonomously has large applications in the world including security, analysis, and robotic interaction.
As of 2020 many techniques for visual recognition have skewed close to the realm of artificial intelligence which implicitly determines the mathematical processes allowing for tracking. In this project, we implement and evaluate a mathematically explicit method for tracking three objects, a runner, a car, and a baby fighting a dragon. Each of these videos comes with their own challenges, the evaluation of each will explain the differences and difficulties associated with each when implementing the Lucas-Kanade (LK) algorithm.

Output Video Link

https://www.youtube.com/watch?v=DtVvEdBBy3c&feature=youtu.be

Part 1: Developing the Tracker

Creating the Template

The first part of the LK tracker is the creation of the template, the desired object to be tracked. This is as simple as cropping the desired object and convert to grayscale. The used templates can be seen below.

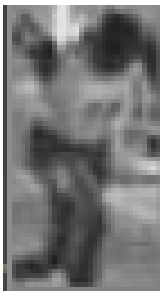

Figure 1: Bolt template.

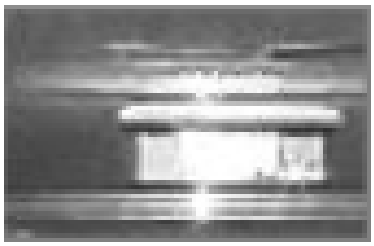

Figure 2: Car template.

Figure 3: Baby template.

The Inverse Composition Algorithm

Pre-computation

Before estimate the Affine transformation between a feature in a frame and a template feature, the following parameter:

1. Evaluate the gradient ∇T of the template $T(x)$
2. Evaluate the Jacobian $\frac{\partial T}{\partial p}$ at $W(x;0)$ and $p = 0$

3. Compute the steepest descent images $\nabla T \frac{\partial T}{\partial p}$ of the template $T(x)$
4. Compute the Hessian matrix of the template T

where

- The gradient $\nabla T = [\frac{\partial T}{\partial x}, \frac{\partial T}{\partial y}]$, which was computed by OpenCv SobelX and SobelY.
- $\frac{\partial T}{\partial p} = \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix}$, as $p = [p_1, p_2, p_3, p_4, p_5, p_6]$ is variables of Affine transformation $W(x; p)$.
- The Affine transformation $W(x; p) = \frac{\partial T}{\partial p} = \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
- The Hessian matrix of template $H = \sum_x [\nabla T \frac{\partial T}{\partial p}]^T [\nabla T \frac{\partial T}{\partial p}]$

Iteration

for each iteration, update the following parameters:

1. Warp the current frame with $W(x; p)$ to compute $I(W(x; p))$ of the template
2. Compute the error between the current frame and template $I(W(x; p)) - T(x)$
3. Evaluate $\sum_x [\nabla T \frac{\partial T}{\partial p}]^T [I(W(x; p)) - T(x)]$ of the template
4. Compute Δp
5. Update the warp transformation $W(x; p) \leftarrow W(x; p) \circ W(x; \Delta p)^{-1}$

where

- Affine transformation variables changes are $\Delta p = H^{-1} \sum_x [\nabla T \frac{\partial T}{\partial p}]^T [I(W(x; p)) - T(x)]$
- The composition of previous warp and inverse of warp is $W(x; p) \circ W(x; \Delta p)^{-1} = \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1+\Delta p_1 & \Delta p_3 & \Delta p_5 \\ \Delta p_2 & 1+\Delta p_4 & \Delta p_6 \\ 0 & 0 & 1 \end{bmatrix} \right)^{-1}$

Part 2: Evaluation of the Tracker

Evaluation of the Tracker: Bolt

For the (Usain) Bolt tracker we are initially able to accurately track the position to approximately the halfway point of the video. At this point the camera makes a sudden jerk to the right in an attempt to keep up with the runners. At this point the tracker has an issue due to the sudden change in the position of the gradient. We observe that the box initially loses Bolt but is able to find the runner after triggering a robust measure that re-positions the tracker at the most recent best tracker position and resets the warp and p_{prev} values. The results of this tracker can be seen below.

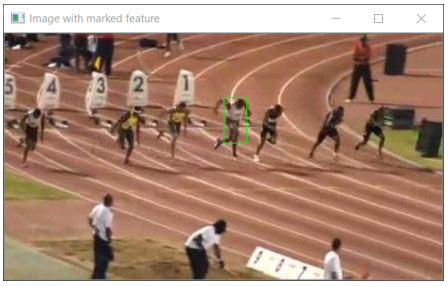

Figure 4: Bolt being tracked 20 frames into the video.

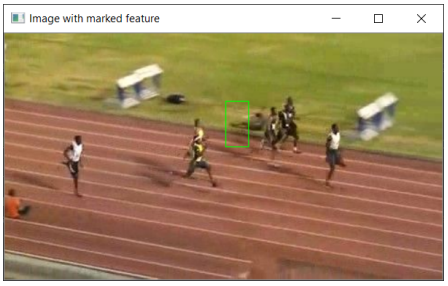

Figure 5: Bolt being tracked 200 frames into the video.

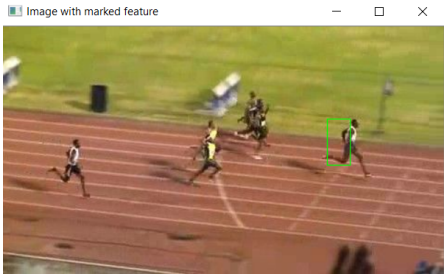

Figure 6: Bolt being tracked 233 frames into the video.

Evaluation of the Tracker: Car

The car tracker is one of the more difficult objects to track. In this video the tracker is set to follow a van as it goes under a bridge and makes a lane change. We break this video down into three different sections for our analysis: initial, under the bridge and after the lane change.

The first section, the car largely does not make any movements that the tracker needs to adjust for. The tracker does begin the respond as the car begin to make it lane change. This lane change begins in the light but largely occurs during the under the bridge sequence.

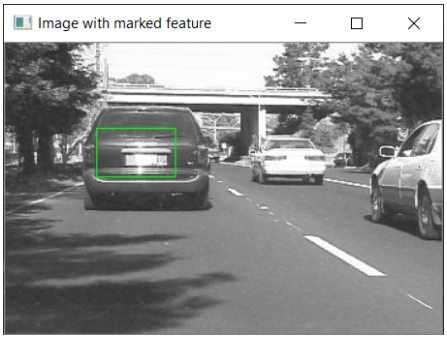

Figure 7: Car being tracked 20 frames into the video.

When car goes under the bridge the tracker encounters a breakdown (loss of information) in the gradient. This is some-what mitigated by the cars licence plate reflecting allowing for the gradient to be computed within a reasonable measure . At this point the tracker attempts to adapt to the perceived large movement (rapidly changing gradient) and moves towards the steepest descent.

Figure 8: Car being tracked 186 frames into the video.

Finally after passing under the bridge the car is now in the right most lane for the remainder of the video. Here the tracker largely is able to follow the minimal movements ,although it's accuracy is not quite right as it lost much of the information during the under the bridge sequence.

Figure 9: Car being tracked 542 frames into the video.

Evaluation of the Tracker: Baby

The final video we attempt to use our tracker on is of a fight sequence between a dragon and a baby. As we've discussed with the previous videos a rapid shift in movement is one of the largest influences to the tracker failing. This video uses fast movement which at some points can cause a loss of information from rapid displacement of relevant information. For the most part we are able to track the baby's position on the screen especially with the handicap of the baby taking up a large portion of the video itself.

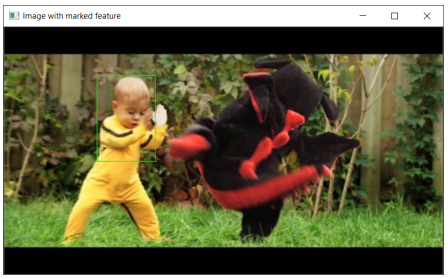

Figure 10: Baby being tracked 1 frame into the video.

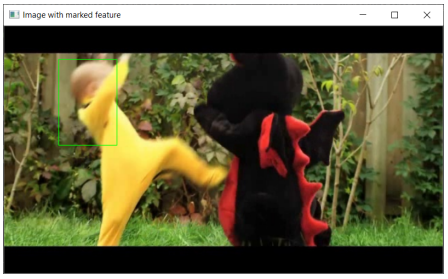

Figure 11: Baby being tracked 63 frames into the video.

Part 3: Robustness and Improvements

Scaling of Intensities

The LK registration breaks down when there is a change in illumination, because the registration formulated on optical flow, and the sum of squared distances error needed to minimize is sensitive to illumination changes, such as a car going from light to dark. In this project, in order to eliminate the illumination swift, a brightness resale applied to the warped feature being registered, making the average brightness of feature in each frame match up the average brightness of the template. Based on the cars on the road video, while the car go to under bridge and become dark, the brightness resale function make the dark intensities. Therefore, the LK registration could match up the intensity between feature in current template and template, and is able to capture up the car.

Weighted Least Squares

Instead of giving same weight to all the errors, we compensated them with a M-estimator. First, we calculated the mean and variance of the errors for each iteration and considered them as a normal distribution. Then a relatively higher

value was given to the errors within one standard deviation interval. A lower coefficient was given to the outliers. Thus, a weighted Δp will be updated with the corresponding vector to minimize the goal function of inverse compositional algorithm:

$$\sum_{x} \Lambda_{ii}[T(W(x;\Delta P)) - I(W(x;p))]^2 \tag{1}$$

Improved Output

An example of how illumination correction aided our program can be seen specifically when the the car from the second part of the project goes under the bridge. Prior to illumination aid the tracker would often get lost trying to track non-valued gradients. Example figure 12. However after adding the illumination aid, the tracker performed significantly better as seen in figure 13.

Figure 12: Car being tracked 186 frames before correction.

Figure 13: Car being tracked 186 frames with correction.

Part4: Conclusion and Lessons Learned

Solution to the problem of lost tracking would be applying the pyramid method with the existing algorithm. The Pyramid concept is implemented when the video frames are under continuous swift motion (input video of Bolt in our case). The down pyramid will decrease the frame resolution by multiple times and set them as layers respectively. The change in position of the object will be less at the top layer of the pyramid than the bottom layer, which is the original frame. By computing the change in parameter thorough all such layers we can compensate the larger change in motion and detect the object.

In a large sense this project is the first project to start fully within the Coronavirus lock down. The inability for the entire team to meet was the cause of major slowdown and code adjustments. Additionally the push for isolated coding and the greater reliance on GitHub lead to the teams first major technological difficulty of a branch-master merge. What we garnered from this difficulty was the importance on keeping updated on pushed code to the database to avoid branches that require intensive merge attention.