ENPM 673 - Perception of Autonomous Robots

Pradeep Gopal
Sahana Anbazhagan
Srikumar Muralidharan

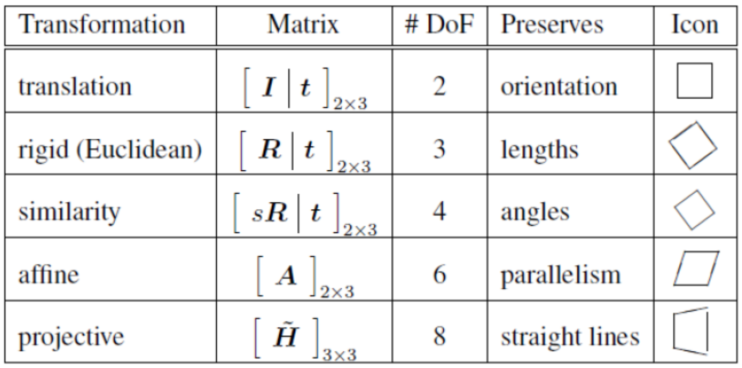
April 20, 2020

1 Introduction

We are tasked with the implementation of a Lucas-Kanade(LK) template tracker on 3 different data sets. The three data sets are obtained from the Visual Tracker benchmark database. First data set is a collection of images from Usain Bolt running data set. Second data set is the Baby fighting a real life dragon data set. The third data set is a collection of Car data set.

2 Implementation of Lucas-Kanade algorithm

The goal of the Lucas-Kanade algorithm is to minimize the sum of squared error between two images, that is template and image after it is warped with the template image. So we have to align the a template image with an input image where both are a function of x, x being the column vector with the pixel coordinates. We have calculated the inverse Affine transforms for every frame and the bounding box is computed and drawn after this using the points given in the text file. We have used inverse Affine transforms in our case because if we use Affine transforms, we have to calculate Hessian matrix multiple times. But in case of inverse Affine transforms we can do the same with just one Hessian matrix computation. There are various transformations where each accounts for different degrees of freedoms. We have used Affine transforms which accounts for 6 degrees of freedom and preserves parallelism. This helps us in optimizing the code and give



Transformation	Matrix	# DoF	Preserves	Icon
translation	$\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$	2	orientation	□
rigid (Euclidean)	$\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	3	lengths	◇
similarity	$\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	4	angles	◇
affine	$\left[\, \boldsymbol{A} \,\right]_{2\times 3}$	6	parallelism	▱
projective	$\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$	8	straight lines	⏢

Figure 1: Table that shows the various transformations with DoF

us faster results. We have taken the template image from the first image from the data set by cropping the image with the coordinates after performing the above computations. The rest of the images are the input images taken in sequence. So then we warp the input images with the template image we got from drawing the bounding boxes as per requirement. With this warped image we compute the error, by subtracting the warped image from the template image. Now the gradient of the image is calculated in the coordinate frame and we obtain,

$$\nabla I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}) \tag{1}$$

Then we compute the Jacobian of the warped image as:

$$\frac{\partial W}{\partial p} = \begin{pmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{pmatrix} \tag{2}$$

The gradient and the warped image W are both dependent on p, the parameters. We now have to find the steepest descent. For this we have to combine the computed gradient and the Jacobian. Now we computed the Hessian matrix using equation (3):

$$H = \sum_x [\nabla I \frac{\partial W}{\partial p}]^T [\nabla I \frac{\partial W}{\partial p}] \tag{3}$$

where,

$$[\nabla I \frac{\partial W}{\partial p}] = [T(x) - I(W(x;p))] \tag{4}$$

and T(x) = Template image
I(W(x;p)) = Image after the input image is warped with the template image
Further we have to compute Δp which is nothing but the product of the steepest descent and the inverse of the Hessian matrix computed. We calculated Δp using equation (5).

$$\Delta p = H^{-1} \sum_x [\nabla I \frac{\partial W}{\partial p}]^T [T(x) - I(W(x;p))] \tag{5}$$

Now, we have to iterate the above step and compute Δp until the parameters of p converges. We have to repeat all the above steps through every iteration because the parameters p varies for every iteration.
We have used Inverse LK for the implementation. The difference is that instead of performing all the operations on the warped input image, the computations are performed on the template image. The algorithm we have followed is as shown below:

1. Warp I with W(x;p)
2. Subtract T from I. That is [I(W(x;p)) - T(x)]
3. Compute gradient ∇T
4. Evaluate the Jacobian $\frac{\partial W}{\partial p}$
5. Compute steepest descent $\nabla T \frac{\partial W}{\partial p}$
6. Compute Inverse Hessian H^{-1}
7. Multiply steepest descend with error $\sum_x [\nabla T \frac{\partial W}{\partial p}]^T [I(W(x;p)) - T(x)]$
8. Compute Δp, where Δp

$$\Delta p = H^{-1} \sum_x [\nabla I \frac{\partial W}{\partial p}]^T [T(x) - I(W(x;p))] \tag{6}$$

9. Update parameters p → p + Δp

The below equations were used to update the p values.

3 Robustness and Illumination

We have implemented adaptive brightness and gamma correction in order to accommodate those instances where there is a difference in the illumination of the images. As an example in the Car4 data set, there are some places where there are either shadows causing the car we are trying to track to be in darkness. In such a situation initially we tried to create templates in various parts of the video so as to maintain the tracking

$$\mathbf{p} \leftarrow \mathbf{p} - \begin{pmatrix} 1+p_1 & p_3 & 0 & 0 & 0 & 0 \\ p_2 & 1+p_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1+p_1 & p_3 & 0 & 0 \\ 0 & 0 & p_2 & 1+p_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+p_1 & p_3 \\ 0 & 0 & 0 & 0 & p_2 & 1+p_4 \end{pmatrix} \Delta\mathbf{p}_*$$

$$\frac{1}{(1+\Delta p_1)\cdot(1+\Delta p_4) - \Delta p_2 \cdot \Delta p_3} \begin{pmatrix} -\Delta p_1 - \Delta p_1 \cdot \Delta p_4 + \Delta p_2 \cdot \Delta p_3 \\ -\Delta p_2 \\ -\Delta p_3 \\ -\Delta p_4 - \Delta p_1 \cdot \Delta p_4 + \Delta p_2 \cdot \Delta p_3 \\ -\Delta p_5 - \Delta p_1 \cdot \Delta p_5 + \Delta p_3 \cdot \Delta p_6 \\ -\Delta p_6 - \Delta p_1 \cdot \Delta p_6 + \Delta p_2 \cdot \Delta p_5 \end{pmatrix} = \begin{pmatrix} -\Delta p_1 \\ -\Delta p_2 \\ -\Delta p_3 \\ -\Delta p_4 \\ -\Delta p_5 \\ -\Delta p_6 \end{pmatrix}$$

in place at all varied situations. Later we found adaptive brightness methodology to be more accurate and thus we have incorporated the same to keep the tracking of the car going on constantly.
In case of adaptive brightness we have to shift the pixel values in order to adapt to the changes in the brightness. We need to calculate the standard deviation for the cropped image using the below mentioned formula.

$$\boldsymbol{\sigma_{cropped}} = \frac{\sum_{i=0}^{N}\left(x_i - \bar{x}_{cropped}\right)^2}{N}$$

So in order to accommodate the changes in the pixel values the change in the standard deviations should be considered. Using the below equation the z-score value can be calculated which is nothing but the difference between the mean and the standard deviation. So by computing the z-score for all the frames using the below equations we were able to overcome the brightness issue.

$$\boldsymbol{Z} = \frac{\boldsymbol{X} - \bar{\boldsymbol{X}}_{template}}{\boldsymbol{\sigma_{cropped}}} \qquad \boldsymbol{X_{new}} = \boldsymbol{Z} * \boldsymbol{\sigma_{cropped}} + \bar{\boldsymbol{X}}_{cropped}$$

Gamma correction is also known as the Power Law Transform.we obtain our output gamma corrected image by applying the following equation:

$$O = I^{1/G} \tag{7}$$

where I is our input image and G is our gamma value. Gamma values ¡ 1 will shift the image towards the darker end of the spectrum while gamma values ¿ 1 will make the image appear lighter. A gamma value of G=1 will have no affect on the input image.Since we are trying to correct the darker images we need to have gamma value greater than 1. So we have taken the gamma value as 1.5 in our case.

Histogram equalization is a method in image processing of contrast adjustment using the image's histogram. This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values.

4 Tracking of car

Using the above pipeline we have tracked a car from the given data set to obtain the following results.

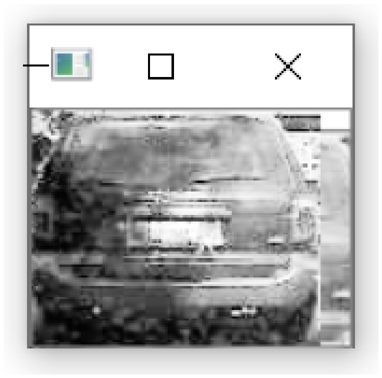


Figure 2: A chosen template for tracking the car

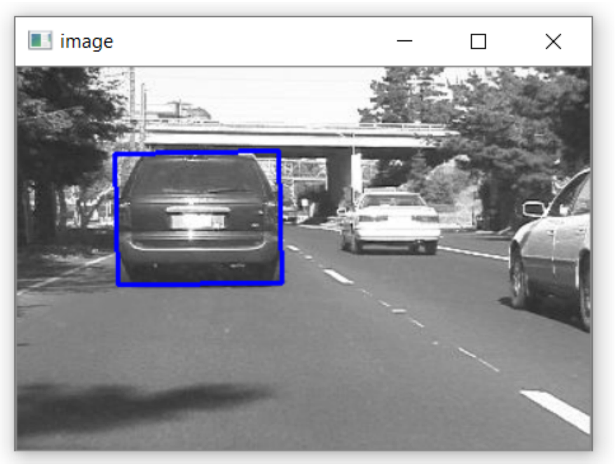


Figure 3: Showing the car being tracked

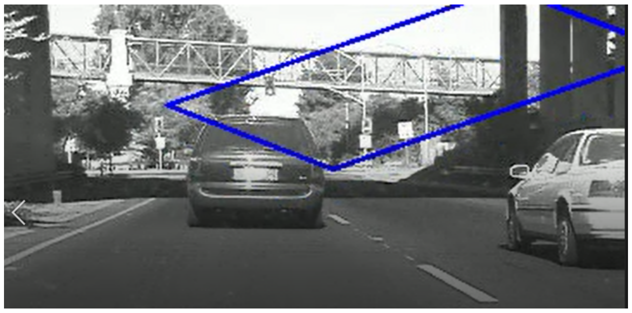


Figure 4: Showing the car being tracked when light is less and there is no correction

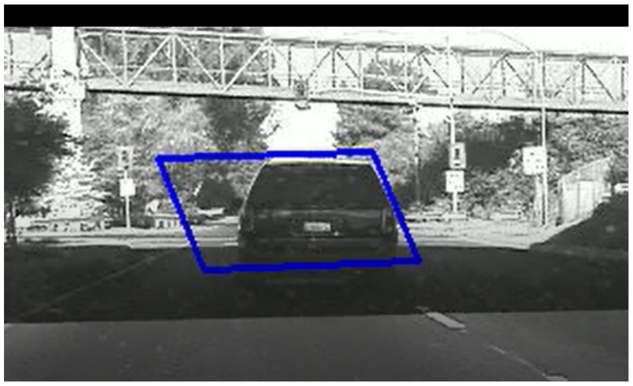


Figure 5: Showing the car being tracked after correcting the lighting using gamma correction

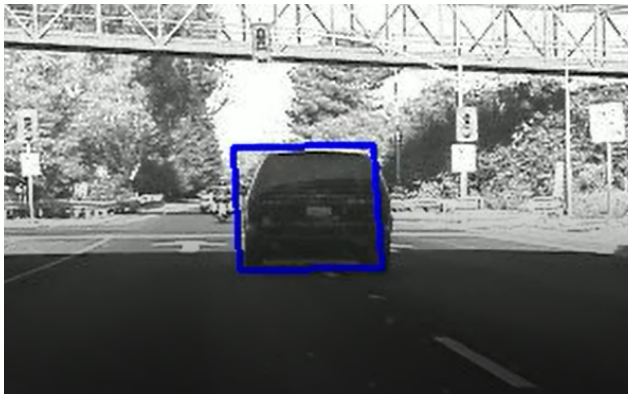


Figure 6: Showing the car being tracked after correcting the lighting. Using both gamma correction and adaptive brightness

5 Tracking of Bolt

Similarly we have tracked Usian Bolt from the given data set to see the below results.

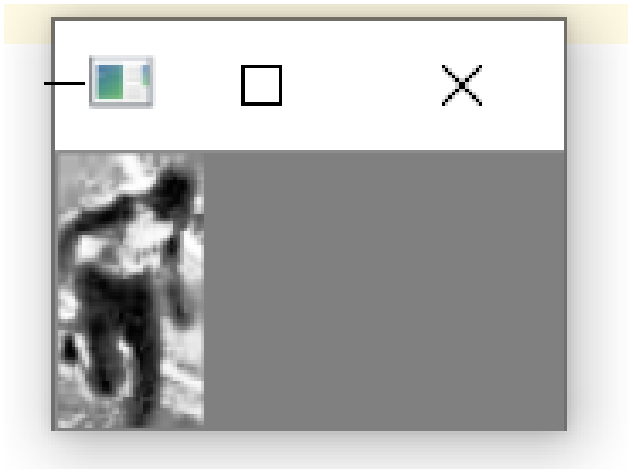


Figure 7: A chosen template for tracking the Bolt

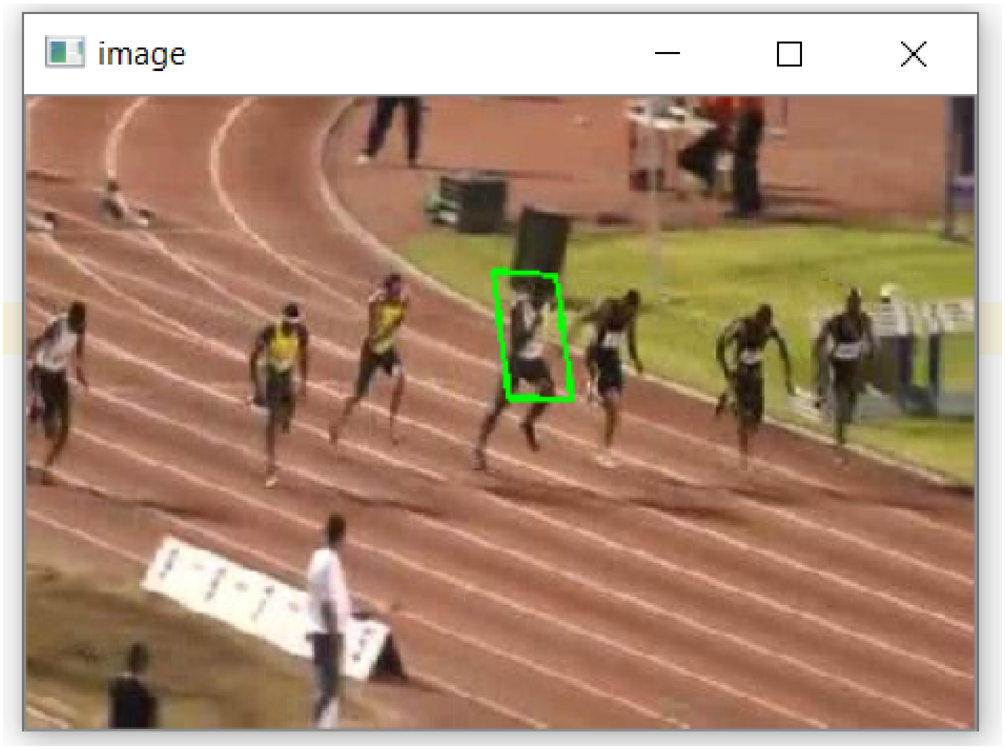


Figure 8: Showing Bolt being tracked

6 Tracking of Baby

Similarly we have tracked the baby face from the given data set to get the shown outputs.

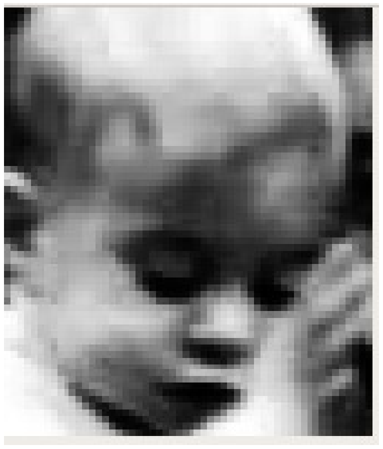


Figure 9: A chosen template for tracking the Baby

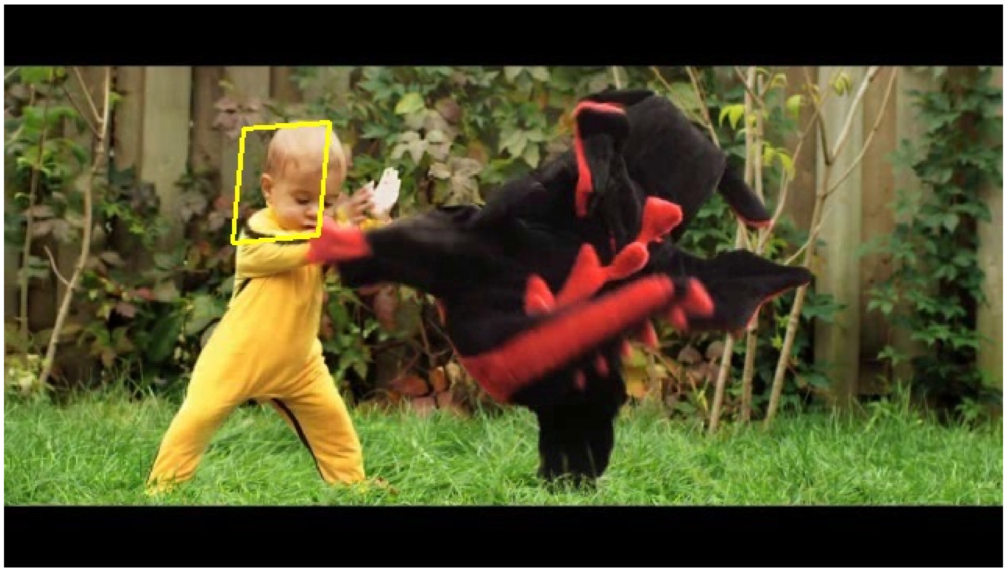


Figure 10: Showing the baby being tracked

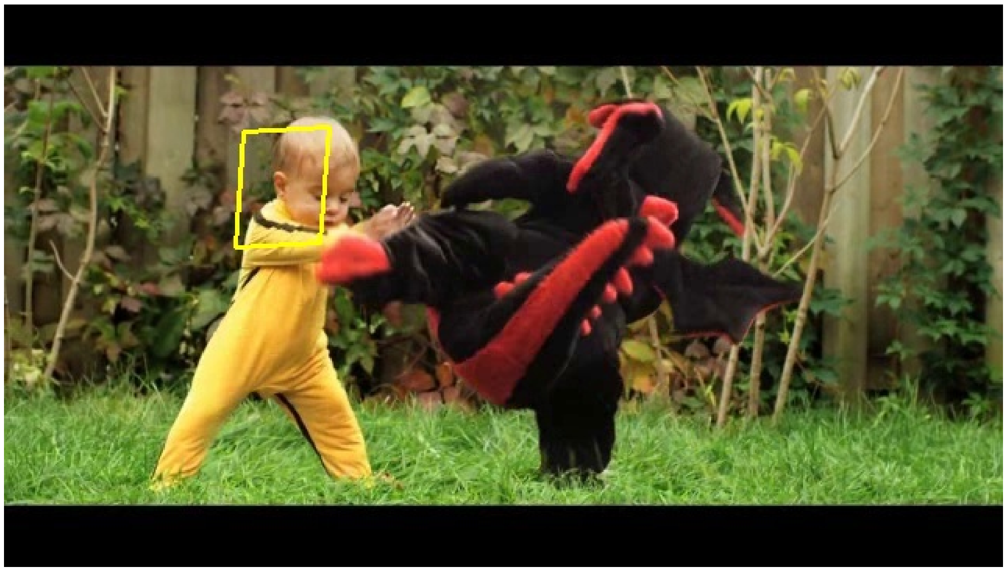


Figure 11: Showing the baby being tracked

The final output videos are uploaded on Google drive