Object Detection and Tracking Pipeline

Pranav Jain
Person Number: 50208349
UBIT: pjain4

Muthuvel Palanisamy
Person Number: 50246815
UBIT: muthuvel

Manvijay Lather
Person Number: 50200437
UBIT: manvijay

Abstract

Object detection and tracking has become an integral part of Computer Vision. Efficient realization of these techniques in images and videos helped create various application in the field of automation, face detection, motion capture etc. This paper proposes an Lucas-Kanade object tracking technique initiated by Tensorflow's Object detection API and compares how this technique differs from the native tracking method employed in Tensorflow object detection.

1. Literature Review

An optimal approach to any tracking algorithm is based on two key areas in computer vision; object detection and object tracking. While a variety of methods are available for accomplishing these techniques, not all of them prove to be effective in achieving a significant efficiency. Conventional method for accomplishing this uses techniques such as Frame Differencing, Optical flow, and Background subtraction for object detection and Point-tracking techniques, SVM and Kernel bases tracking for object tracking . A brief introduction to a some of these techniques is given below

1.1. Some Object Detection Methods

1.1.1 Frame Differencing

Frame differencing suggests calculating the difference between two successive frame in an image to find the moving objects. While the idea is simple and east to implement, it is generally considered ineffective in dynamic environments where the changes in the frame of an image are not just isolated to the objects that needs to be tracked.

1.1.2 Background Subtraction

This techniques involves modelling the background and use that for background subtraction. We must choose a background model sensitive enough to uncover the objects in an image/frame and use that as a reference in tracking. This can be done by find the variation between frames and the reference models. Common filters like mean and median filters can be used to calculate the difference in pixel intensities. This is mainly achieved using recursive and non-recursive algorithms. A background subtraction algorithm based on frame differencing is explored in *Rupali S.Rakibe, Bharati D.Patil, Background Subtraction Algorithm Based Human Motion Detection,International Journal of Scientific and Research Publications, May 2013*. Similar to the previous approach, this method becomes in effective in the variation in the background becomes high. A more sophisticated method which can ignore highly changing background will be needed for detecting objects will be required for advanced models.

1.2. Some Object Tracking Methods

1.2.1 Support Vector Machine (SVM)

SVMs apply machine learning algorithms for learning, analyzing and storing the the output as positive and negative values that represent the tracked objects in an image and objects not tracked in an image. SVMs have a wider range of application in real world, that includes but not isolated to face detection, bioinformatics, handwriting or pattern recognitions etc.

1.2.2 Silhouette Bases Tracking approach

This approach is helpful in tracking complex geometric shapes present in images such as hands, fingers etc. It can be done by employed by track either the boundary(Contour) of objects in an image or the shape corresponding to those objects. Contour based tracking can be achieved using gradient descent minimization technique and using state space models to trace boundaries. Shape Matching is a kind of brute force approach applied on a region of interest in a frame to identify objects that can be matched with a reference object

1.3. Tensorflow API

Tensorflow was originally developed by Google Brain Team to help facilitate machine learning and deep learning

researches. Tensorflow's object detection model consists of a neural network trained from COCO(Common Objects in Context) dataset. It contains around 300k images of 90 commonly used objects. The library consists of 5 different models to use; classified based on their speed and accuracy of placing the bounding box (a box that encloses a detected object). We plan to use all the 5 different models and compare the results of one with the another to find the most optimal one.

1.4. Lucas-Kanade Tracker in OpenCV

Devoped by Bruce D. Lucas and Takeo Kanade, Lucas-Kanade Tracker uses an optical flow algorithm to track an object in subsequent frames. OpenCV incorporates a predefined function for implementing this algorithm namely - *visualize_boxes_and_labels_on_image_array()* which we are planning to append to the tensorflow's object detection technique

2. Method Proposed

Trained Neural Networks has been proven to produce better efficiency and detection models.

Our idea is to come up with an ideal tracking method that is able to self detect object and track them. We aim to make use of tensorflow's object detection technique and the Lucas-Kanade tracker implemented using opencv library. The desire pre-trained model used to detect an object is downloaded from the Tensorflow library.

We want to experiment three different methods in achieving a tracking algorithm which are as follows

2.1. Objection detection using Tensorflow

In this method, we employ a detection technique using object detection function from Tensorflow. The process initiates by starting with the first frame in a video or image. The object detected is visualized using a bounding box enclosing the object. This is done by the function*visualize_boxes_and_labels_on_image_array()* In every successive frame, the same object detection algorithm is employed to detect the new interesting objects in the frame. In the end, detecting objects in all the frame individually translates to an indirect object tracking method.

2.2. Object tracking initialized by Tensorflow

In this method, we initialize the process by starting with Tensorflow's object detection. The first part of the detects the objects of interest in the image using the pre-trained model The objects detected in the first frame is used as input for tracking in the second part.

The object detection technique employed here returns an array of bounding boxes which consists of a array of coordinates for boxes enclosing an object for all interesting objects and their corresponding scores. Since, all the boxes detected may not used ideal, we set a threshold to the scores to limit the number boxes.

In the next part, we use SIFT detector on the first frame of the video/ image to detect the feature points in the image. Not all points here are required for tracking. Hence, we limit them by choosing only the feature point present inside the bounding box detected from the object detection. This will give us the keypoints that can be used to track the object.

Using the feature points extracted above, we initiate the Lucas-Kanade tracking function- *calcOpticalFlowPyrLK()* from openCV. The difference in the location of the object in the current frame to that in the previous frame in calculated. This is used to eliminate static objects from background by setting a threshold on the difference.

Finally, a bounding box is drawn over the object using the *visualize_boxes_and_labels_on_image_array()* function from tensorflow library.

Many authors misunderstand the concept of anonymizing fosr blind review. Blind review does not mean that one must remove citations to one's own work—in fact it is often impossible to review a paper unless the previous citations are known and available.

2.3. Hybrid Approach

As the name suggests, this method employs the correlation between both object detection from Tensorflow and Lucas-Kanade Tracker. The outcome visualized is to combine Tensorflow's efficiency in detection and the speed of tracking achieve by KLT algorithm. This idea is to create a fail proof technique in which the shortcomings in the KLT algorithm is balance by usage Tensorflow to detect objects periodically.

So, instead of just using the object detection technique only for initializing, we use Tensorflow to detect object for every **n** frames and pass the detected objects as input to tracker everytime. So, in short once the object detection passes the input to tracker, it waits for *n* frames to detects objects again and pass the same to the tracker. This approach has a potential benefit of detecting objects that are not present in the beginning of the video. Also, it could help rectify any pitfalls in the Tracking algorithm but constantly running object detection again while maintaining speed.

3. Experimental Analysis and Result

We compared the time taken to track an object in a video using the two different approaches discussed in the previous sections of this report. Our first approach is, keep detecting the object throughout the video using Tensorflow and the other way we have chosen to track the object is to detect it in the beginning using Tensorflow and then track it through the video using Open Cs tracker.

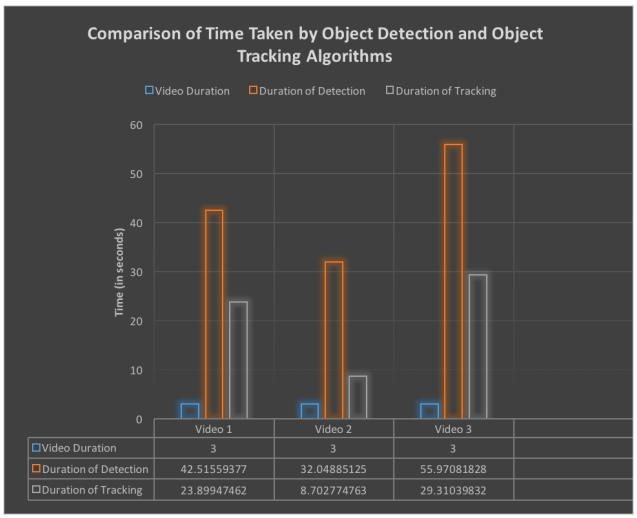

Figure 1. Comparison

The graph here represents the data we gathered after implementing these two algorithms on 3 different videos of same length, i.e. 3 seconds each. X-axis: Name of the video. Y- axis: Time in seconds.

As you can see clearly for every video, regardless of it resolution, first approach (detection algorithm) has taken longer to track the object and evidently more than twice the time taken by the tracking algorithm

Although the Open CV tracker seems like a better option in terms of speed, it has a couple of limitations as well. First limitation being tracker algorithm depending on Tensorflow to detect the object and only then tracking it throughout the video. Second limitation being the decrease in size of the box enclosing the feature points of the detected object as the time passes.

4. Conclusion and Future Work

We will be resolving the shortcomings of our work in the future. To begin with, we will be reducing the dependency on Tensorflow for object detection by look at other alternatives. The primary reason for this is to have zero chance of error while detecting objects. Tensorflow though faster and effective most of the time, it fails to detect objects at certain frame which when coinciding with the object detection frame, can prove to be a failure in the tracking process.

As mentioned earlier, hybrid approach, though proving to be better comparitvely, it does not meet the excpectation that we expected. This is because the sweet spot for reinitializing object detection varies for different videos and we will work on designing an algorithm to detected the ideal frame to initiate object detection in the hybrid approach

Also, we would like to explore the possibility of running SIFT detector only inside the bounding box of the detected region of the object and see if it produces a postive impact on the speed on our model.

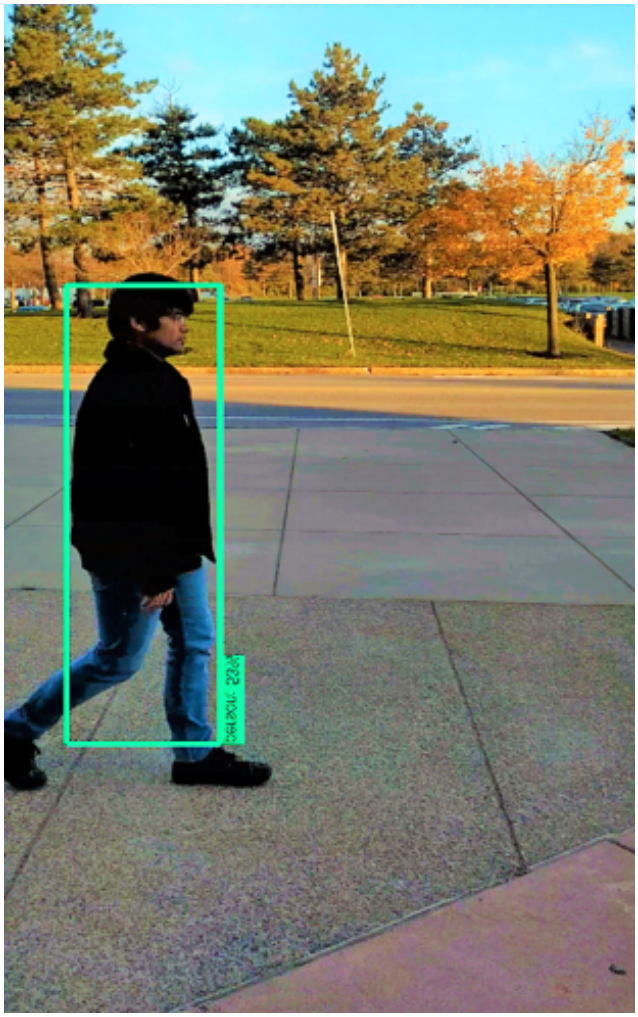

Figure 2. Objects detected in test videos

5. References

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your paper. When referenced in the text, enclose the citation number in square brackets, for example [**?**]. Where appropriate, include the name(s) of editors of referenced books.