Fast and Economical Object Tracking using Raspberry Pi 3.0

Purnima Prakas, T.J Nagalakshmi

ABSTRACT--- This paper proposes a way to construct a financially cheap and fast object tracking using Raspberry Pi3. Multiple object detection is an important step in any computer vision application. Since the number of cameras included is more these gadgets are compelled by expense per hub, control utilization and handling power. We propose a tracking system with low power consumption. The framework is completely designed with python and OpenCV. The tracking quality and accuracy is measured using publicly available datasets.

Keywords: Cameras, Computer vision, Python, Raspberry Pi, Pi camera, Tracking.

I. INTRODUCTION

Multiple Object detection and recognition is an important topic when it comes to computer vision. It is widely used in applications like automatic traffic control systems and for driver assistance system [1]. On the other hand, these systems are suffering badly with power consumption and cost. The cost of these devices increase with every node. Visual Object tracking is a system which faces various challenges like illumination and pose. At many cases the number of objects in the field of detection may vary overtime. To solve these problems there are several methods and algorithms are developed [1] [2]. However, these systems are designed to achieve the projected accuracy and processing speed on a personal computer. These personal computers has required computing capacity to drive the algorithm designed but when these algorithms are deployed in to an embedded system their accuracy and computing power does not meet the expectations. We used Raspberry pi3 which has a quad core processor. It's has all functions of a basic computer and operates on a Linux OS. Raspberry pi operates as a heart of the proposed system as it holds the tracking algorithm and the camera input.

The Kernalized correlation Filters (KCF) tracker is used in the proposed system. KCF tracker is designed using OpenCV python. This tracker is designed for individual objects firstly and at the end all the individual trackers are combined with frame to frame detection. Traditional KCF trackers suffered tracking errors when the size of the object gradually changed. Hence we took advantage over size of information for tracking. Re- assignment and back tracking is based on the measurement of the predicted object and the last appearance position. OpenCV is used to manipulate and acquire input runtime. We used OpenCV python to reduced complexity and increase robustness of the algorithm In the further sections we will discuss about the hardware set up and tracking efficiency.

II. FUNCTIONAL DESCRIPTION

The Functional description of various components are given below:

A) Raspberry Pi :

The Raspberry Pi [3] is a low cost, credit card sized single board with a considerable computing power. It provides various features like Ethernet port, GPIO pins for connecting sensors and various devices, Camera connector (Raspberry pi camera). USB ports to connect various devices like Keyboard, mouse, Monitor etc. The raspberry pi does not have any internal mass storage unit or built in operating system. It requires a version of an SD card which will serve as mass storage and holds Linux OS. The system proposed here uses Raspberry Pi3 as shown in the figure 1. This model board is a micro controller kit with ARM11 processor provided with internet / Ethernet connectivity, 512 MB memory, dual USB connectivity and supports Pidora, Raspbmc, Raspbian

Figure 1: Raspberry Pi3

The raspberry pi operates as the heart of the proposed system as it holds the Raspberry Pi camera and the proposed algorithm. It operates at 5V 2 A. It supports Wi-Fi connectivity and hence this device can be controlled anywhere from the world via internet.

B) Raspberry Pi Camera:

The Raspberry Pi camera is a 8 mega pixel camera which uses Sony IMX219 sensor designed specifically to fit an on board device like raspberry pi. It can capture 3280x2464 static images and also extends its support to 640x480p90, 1080p, 720p60 video. It is attached to the raspberry pi through a dedicated CSI interface. The

Revised Manuscript Received on August 14, 2019.

Purnima Prakas, Department of ECE, Saveetha School of Engineering, Chennai, Tamil Nadu, India.

T.J Nagalakshmi, Department of ECE, Saveetha School of Engineering, Chennai, Tamil Nadu, India.

dimension of camera is 25mm x 23mm x 9mm. Since the size and weight of the camera is less, it is flexible to be used in various applications.

Figure 2: Pi camera

Raspberry pi camera should be connected to the camera port of raspberry.Pi camera should be configured by opening the preferences option from the menu.It requires a reboot to have the system settings saved.

C) KCF Tracker:

Kernel Correlation Filter (KCF) is a version of correlation filter. In a correlation filter two data points are taken and if the correlation between these two points are high then the correlation value is high. The same principle is adopted in KCF. The correlation between the previous patch in the frame and the future frame is compared. In a traditional correlation filter, the tracker performance degrades when the object size changes. In a KCF tracker the object being tracked is updated using a linear regression model. Kernel functions like Gaussian functions are used in calculation dot product of high dimensional spaces without calculating higher dimensional vectors.The regression weights are represented by

$$\hat{\alpha} = \frac{\hat{z}}{\hat{k}^{aa} + \lambda} \quad (1)$$

Where Kaa is a kernal function calculated on auto-correlation of training samples a. The regression function has to be calculated on several patches. The candid patches forms circular matrix. Training and testing stages completely rely on computation of kernal function . For Gaussian systems the kernal can be calculated as

$$k^{xy} = exp(\frac{-1}{\sigma^2}(\|x\|^2 + \|y\|^2 - 2\mathcal{F}^{-1}(\hat{x} \odot \hat{y}))) \quad (2)$$

The regression weights given by equation 1 is updated by linear interpolation.

3

$$\hat{\alpha} = ((1 - interpf) \times \hat{\alpha}) + (interpf \times \hat{\alpha}_{new}) \quad (3)$$

$$a = ((1 - interpf) \times a) + (interpf \times a_{new}) \quad (4)$$

The KCF tracker can be dissected in to pieces which can handle training, detection, model update, appearance representation.

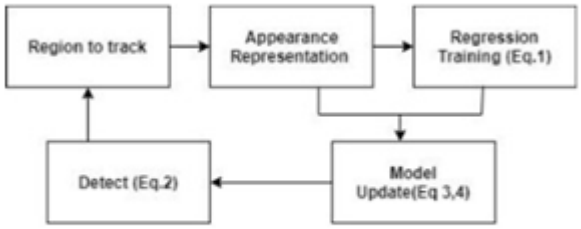

Figure 3: KCF block diagram

There are various versions of KCF available, they are CTF [5], SCT [5], DPT [5], SMACF [5], MAD [5], sKCF [4], KCF2 [4], SME [4], LDP [4].

III. IMPLEMENTATION

KCF tracker algorithm is implemented using OpenCV version 4.0 which extends its support for deep learning and various computer vision algorithms. Since Raspberry pi 3 supports Python for control of its ports [6] and GPIO its very flexible and robust to go with python. When the program is executed the user has to use there mouse and select the object to be tracked. We have not implemented any object detection block since it limits the object to be tracked. After selecting the region of interest (ROI) to be tracked the algorithm tracks the object seamlessly.

Figure 3: Selecting ROI

Figure 4: Tracking-video

When the object is moved away from the line of sight of the camera , it remembers the object and when it appears again it detects the object. The frames per second of the video is kept at a minimum level to boost the accuracy. Since Raspberry is an IOT (Internet Of Things) device , live streaming can be done via connecting it to a server.

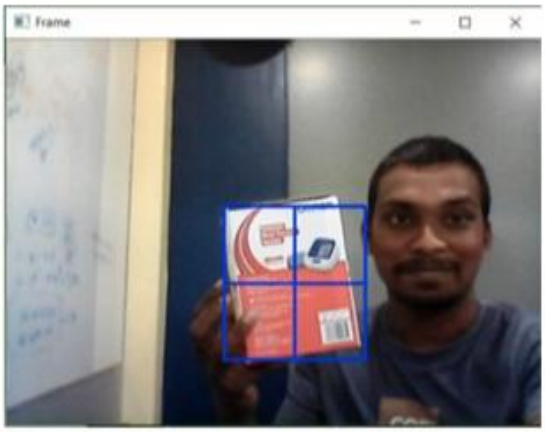

Figure 5: ROI selection for Live tracking

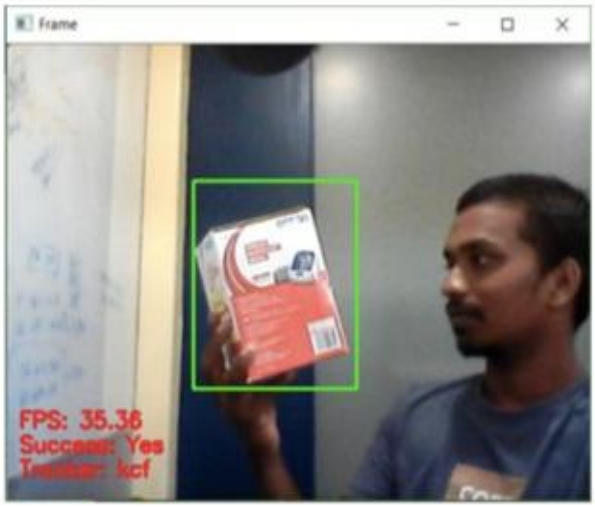

Figure 6: Live Tracking

IV. RESULT

An efficient, robust and low cost tracking method is proposed. Since this system occupies less space, it can be implemented in robots, drones, Traffic signals etc. This system can track both video and live feed input. We have optimised the Frames per second to boost the tracker efficiency.

V. ACKNOWLEDGEMENT

The authors would like to thanks Mrs.TJ. Nagalakshmi for her support and guidance through the development of this project and for her advice in making it possible

REFERENCES

1. N. Majer, K. Schindler, and B. Schiele, “New features and insights for pedestrian detection,” in Computer vision and pattern recognition (CVPR), 2010 IEEE conference on. IEEE, 2010, pp. 1030–1037.
2. Possegger, T. Mauthner, P. Roth, and H. Bischof,“Occlusion geodesics for online multi- object tracking,”in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1306–1313.
3. Matt Richardson and Shawn Wallace, Getting Started with Raspberry Pi. United States of America: O’Reilly Media, 2013. [4]AM. Kristan et. al. (2015) “The Visual Object Tracking VOT2015 challenge results.” IEEE International Conference on Computer Vision Workshop(ICCVW) 2015.
4. Kristan et. al. (2016) “The Visual Object Tracking VOT2016 challenge results.” European Conference on Computer Vision Workshop(ECCV).
5. Raspberry Pi Foundation, [Online], http://www.rasp-berrypi. org/ help/what- is-a- raspberry-p