Human Tracker

A PROJECT REPORT

submitted by

R Sai Pranav (13BCE1108)

Sarveshwaran D K (13BCE1129)

Yash Chandak (13BCE1173)

in partial fulfillment for the PBL

of

CSE-322
Embedded Systems

School of Computing Science and Engineering

VIT® UNIVERSITY (Estd. u/s 3 of UGC Act 1956) VELLORE ▪ CHENNAI www.vit.ac.in

May – 2016

School of Computing Science and Engineering

CERTIFICATE

The project report entitled "**Human Tracker**" is prepared and submitted by **R Sai Pranav (Register No: 13BCE1108), Sarveshwaran D K (Register No: 13BCE1129) and Yash Chandak (Register No: 13BCE1173).** It has been found satisfactory in terms of scope, quality and presentation as partial fulfillment of the requirements for the PBL of **CSE322 – Embedded Systems** in VIT University, Chennai, India.

(Prof. Pradeep Kumar T S)

Abstract

With the advancement in CCTV cameras and web monitoring, it has become crucial for the video analyst to automatically identify the objects in the field of view that may be relevant for inspection. Most of the current systems are based on simple frame subtraction to figure out the moving objects in the environment and subsequently report it on the screen. This technique though manages to select all important moving objects; it also ends up selecting false positives which are of no use. In this project we plan on adding an secondary layer of classifier to categorize the moving objects and report only the ones important to the analyst, (in our case: moving humans)

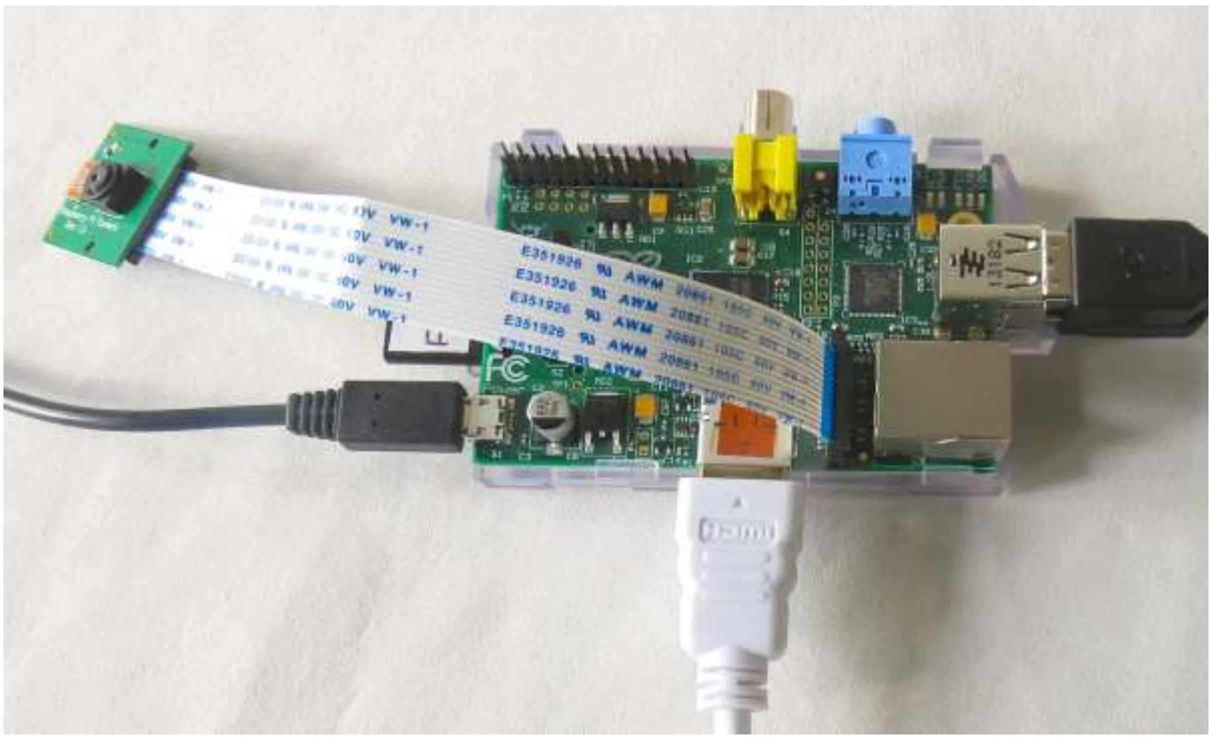
The entire system is developed on Raspberry Pi with the RasPi camera to work in real time scenarios where humans need to be detected. We use the neural network classifier for categorising the moving objects. The entire code was developed on Python and provides an easy interface for usage. The program works on the assumption that the camera is held static at the location such that the background remains constant. This helps in frame subtraction to quickly get the possible moving objects in the scene.

Though the application developed deals with classifying humans, it can be trained to classify anything, whatsoever. For example it can be used to classify animals, and the system can be deployed in zoos to raise a warning when animal by chance moves out of the cage. It can also be used to take wildlife photography such that the camera only takes a snap when the required animal/bird is in the field of view.

Implementation

Components required:

1. Raspberry Pi
2. Power source (power bank)
3. Display monitor (to view real time human tagging)
4. RaspiCam to capture videos
5. SD card to boot the Pi and store the executing code and results



The models feature a Broadcom system on a chip (SOC), which includes an ARM compatible CPU and an on chip graphics processing unit GPU (a VideoCore IV). CPU speed ranges from 700 MHz to 1.2 GHz for the Pi and on board memory is 1 GB RAM. Secure Digital SD card is used to store the operating system and program memory in either the SDHC or MicroSDHC sizes. The board has four USB slots, HDMI and composite video output, and a 3.5 mm phono jack for audio. Lower level output is provided by a number of GPIO pins which support common protocols like I2CThe model also has an RJ45 Ethernet port.

It has a Level 1 cache of 16 KB and a Level 2 cache of 128 KB. The Level 2 cache is used primarily by the GPU. The SoC is stacked underneath the RAM chip, so only its edge is visible.

Algorithm

Training Phase:

1. Collect positive and negative samples for the target
2. Augment the dataset with morphed and manipulated samples to make the system more robust
3. Set the parameters of the neural network as desired
4. Divide the dataset in training and validation set
5. Acquire sample from training set
6. Forward pass and make the class prediction
7. Compute the error in prediction
8. Backpropagate the error to learn the correct parameters
9. Compute the accuracy on validation set
10. Repeat till Error converges
11. Save the parameters

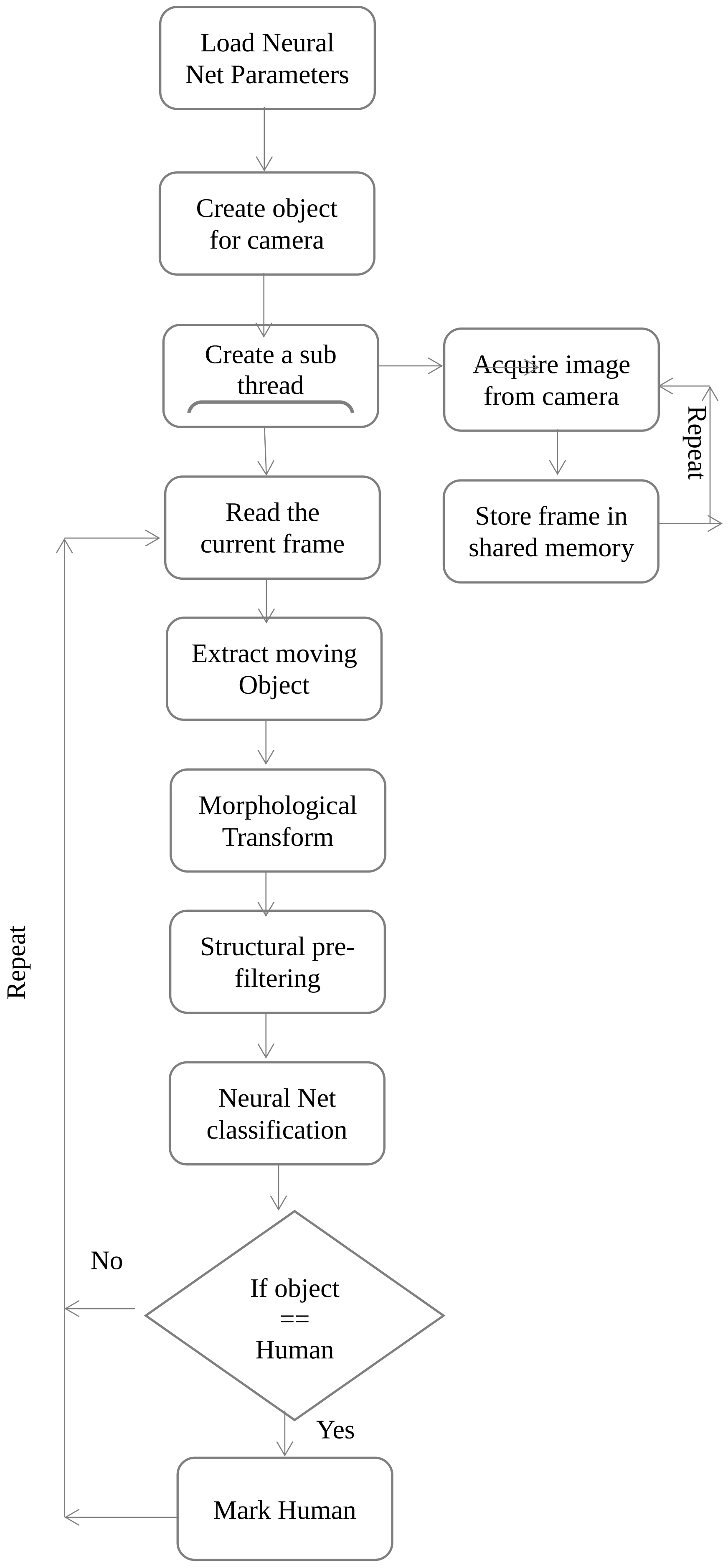
Execution Phase:

1. Create object of the videoCapture class
2. Load the saved neural net parameters
3. Capture photos indefinitely in a child thread to remove IO Latency
4. In the main thread:
    a. Use the frame difference to extract moving objects
    b. Apply morphological transform to make objects more prominent
    c. Based on structural property filter out unlikely objects
    d. Classify the filtered moving objects with only single forward pass of neural net
    e. Mark it if it is the target vehicle
5. Keep repeating

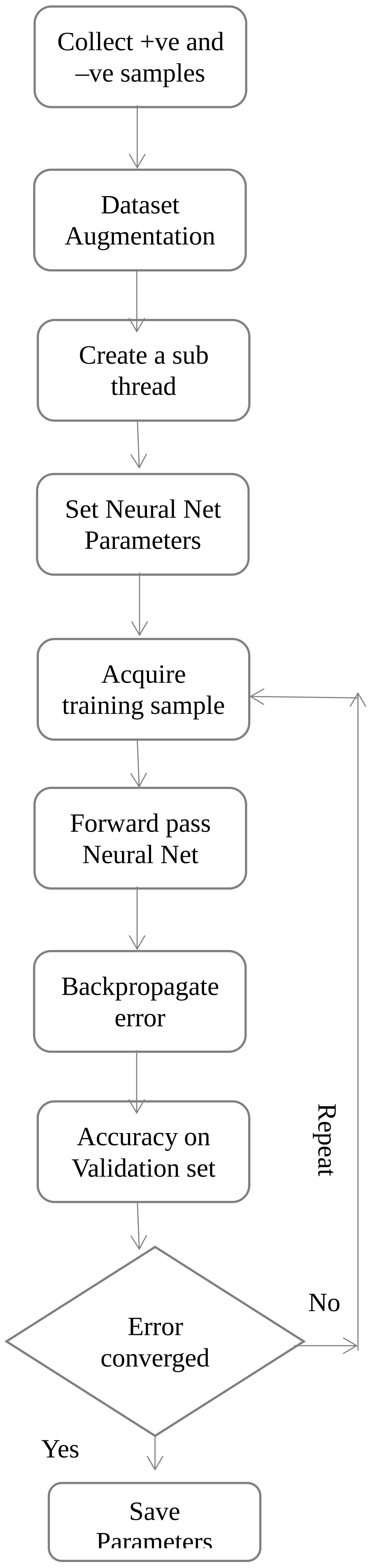
Code:

<Attached externally with this report document>

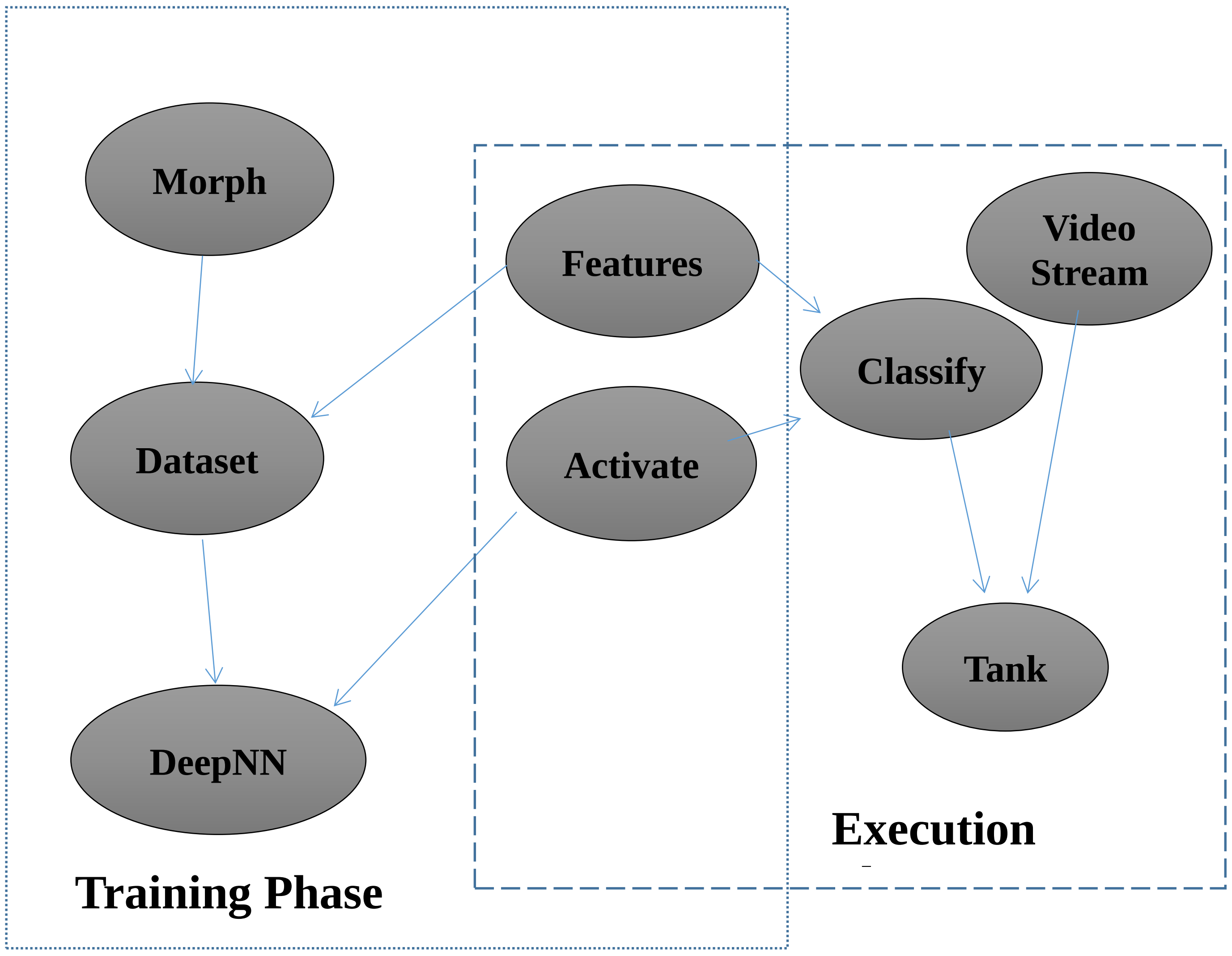
Execution Flowchart



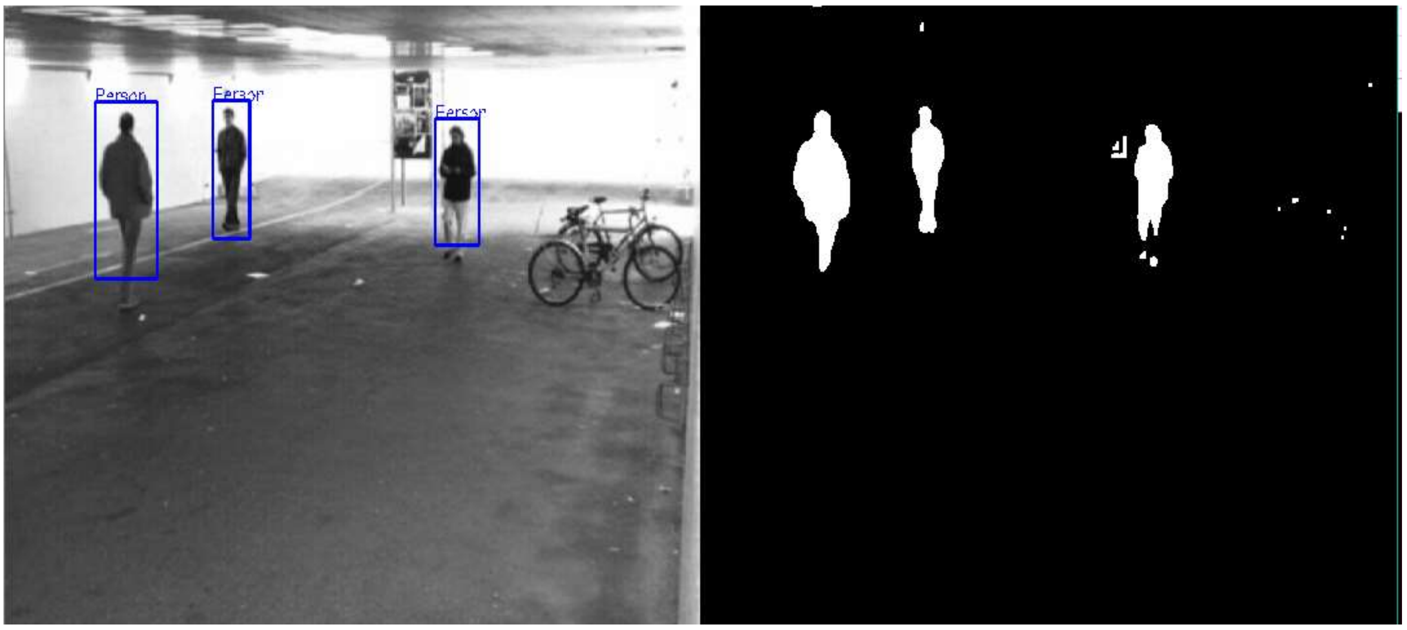
Training Flowchart



Schematic Dependency Diagram



Results



Full Video for result:

https://drive.google.com/open?id=0B7-8xSA7hiAiMWlaSGpZOEVrWHc