IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Real Time Object Detection Using Raspberry Pi

[1]R. Sai Sree, [2]P. Chandu, [3]B. Pranavi

[1]Student at SNIST, [2]Student at SNIST, [3]Student at SNIST

[1]Department pf Electronics and Communication Engineering

[1]Sreenidhi Institute of Science and Technology (SNIST), Hyderabad, India.

Abstract: With the hurried improvement of innovations, for example, advanced cells, autopilot vehicles, and automatons, always inserted gadgets have been competent with PC vision work. For example, Autopilot innovation need vehicles can keenness condition, break down scene and respond likewise; Artificial Intelligence (AI) redesigning and different capacities in the cell phone need to exactly find the article position in the picture; Object recognition may be the most mutual one that is embraced as an essential useful module for scene parsing in inserted applications, and consequently it has been the region of expanding interest. Continuous scene parsing through item recognition running on an inserted gadget is exceptionally testing, because of restricted memory and processing intensity of installed gadgets. To manage these difficulties, we overhaul a lightweight system without outstandingly diminishing recognition exactness. Hence the system is developed with hardware using Raspberry Pi! This redefines the object recognition system by a simple hardware technique using Raspberry Pi. This is finally done with a simple setup in the end with the hardware connection which gives the object detection for around a wide region of 2D-3D view. During most recent couple of years Object detection has gotten perhaps the most sizzling zone of PC vision, and numerous scientists are hustling to get the best object detection model. Because of that many best class models are a work in progress, for example, RCNN, RetinaNet, and Yolo and we've proposed it with the simple Raspberry Pi board

Index Terms **- Raspberry Pi3, Classification, localization, Detection, Segmentation, FPS, HDMI, Microcontroller**

I. INTRODUCTION

Object detection is applied in various perspectives, for instance, automated vehicle structures, development affirmation, an individual by walking acknowledgment, apply self-governance, robotized CCTV, object checking, and so forth. Starting late, object acknowledgment considering significant learning has developed altogether. Essential objective area systems are isolated into 2 species. They are acknowledgment approaches acceptable with the region recommendation and single-step pointer.

Regardless, the procedure for getting the recognizable proof model by means of setting up a tremendous number of tests is particularly directed by the immense number of tests. There are numerous strategies for recognizing an object, for example three-dimensional detection and computerized picture preparing. Besides, these systems don't achieve the perfect results similar to consistent execution. In this article, we gather a few models. So we got models with lighter objects and objects with a superior correlation with the foundation. People can select objects in our line of vision in a matter of milliseconds. Truth be told – simply check out you at this moment. You'll have taken in the environment, immediately recognized the objects present, and are currently seeing this article. To what extent did that take? That is constant object detection. How cool would be it in the event that we could get machines to do that? Presently we can! On account of the ongoing flood of discoveries in profound learning and PC vision, we can incline toward object detection calculations to identify objects in a picture – as well as to do that with the speed and precision of people.

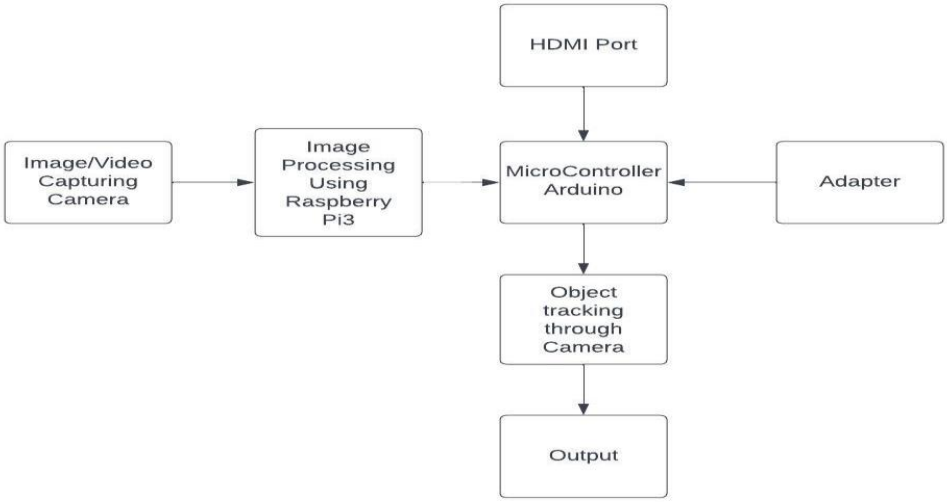

Figure 1. Block diagram of Object Detection using Raspberry Pi

Figure 1 depicts a block diagram of the working of the real time object detection model using Raspberry Pi3 connected to HDMI Port and adapter further it also figures up the final image through the camera

All the more as of late, a Raspberry Pi object detection approach was suggested that moderated the computational multifaceted nature issues related with R-CNN by defining the object detection issue as a solitary relapse issue, where bouncing box organizes and class probabilities are registered simultaneously. Despite everything stays trying for utilizing this methodology for ongoing object detection in video on implanted registering gadgets with restricted computational force and constrained memory. For instance, in various genuine applications, for example, constant surmising on PDAs and installed video observation, the accessible registering assets are constrained to a mix of low-power inserted GPUs or even simply implanted CPUs with restricted memory. In this way, continuous object detection in video on inserted gadgets stays a major test to handle. A considerable lot of object detection frameworks need to experience the picture more than one an opportunity to have the option to distinguish all the objects in the picture, or it needs to experience two phases to recognize the objects. Raspberry Pi cam doesn't have to experience these exhausting procedures. It just need to take a gander at the picture to identify all the objects and that is the reason they picked the name (Raspberry Pi3 Camera) and that is really the motivation behind why Raspberry Pi is an exceptionally quick model.During most recent couple of years Object detection has gotten perhaps the most sizzling zone of PC vision, and numerous scientists are hustling to get the best object detection model. Because of that many best class models are a work in progress.

II. LITERATURE SURVEY

In 2010, P. Angelov, P. Sadeghi-Tehran, R. Ramezani have suggested a Realtime Approach to Autonomous Novelty Detection and Object Tracking in Video Streams, International Journal of Intelligent Systems. The project's primary focus is on making color-based object detection and tracking a reality. As a visual-based project, the project will use video and image data that is continuously captured by a webcam connected to the Raspberry Pi as its input. By moving the camera in the direction of the object that it has detected, it will track that object.[1]

In the International Journal of Intelligent Systems in 2014, Samreen Amir and Bhawani Shankar Chowdhary suggested using the Raspberry Pi to implement the Line Tracking Algorithm. The proposed system worked as it was supposed to. The Raspberry Pi is smaller, but it is slower. Exactness of the two frameworks was comparative regardless of whether the FPS rate is totally different. Given the task it was created for, our algorithm can be used in almost any marine environment. Likewise around the same time 2014 Vijayalaxmi, K.Anjali, B. Srujana, P.Rohith Kumar have proposed Article location and following utilizing picture handling. The fundamental step in the detection process is to scan the image lattice and determine, at each location, whether Xs + W is an object or a background. This clearly requires a lot of computation and is typically carried out at multiple resolutions of the image pyramid in order to identify objects at various scales. There are a lot of ways to make it look older.[2]

Pushkar Protik Goswami and Dushyant Kumar Singh also proposed A Hybrid Approach for Real-Time Object Detection and Tracking in 2015. Underwater transmission lines and oil pipelines must be constantly monitored in today's world. We need an underwater vehicle rover that can follow these wires or pipelines and find the problem if it occurs for this purpose. We have created an intelligent quad-legged rover for this purpose. Image processing is used as a key for finding the problem or damage.[3]

Sarthak Kaingade, Vikrant More, Dhiraj Dhule, Pradeep Gaidhani, and Nitin Gupta proposed using a multirotor unmanned aerial vehicle for vision-based object detection and tracking in a 2016 journal. Previous research on UAV-based object detection and tracking can be broken down into a number of subfields in this paper. Implementation of the "Follow Me" mode, in which the UAV follows a person, has been the focus of some researchers. The person's computer at the ground control station sends the flying UAV its GPS location.[4]

M. Karthikeyan, M. Kudalingam, P. Natrajan, K. Palaniappan, and A. Madhan Prabhu also proposed the Tracking Robot by Using Raspberry PI with Open Computer Vision (CV) in 2016. At the conclusion of this project's automatic mode, the robot follows the picked object, examines its color, and then places the object in the appropriate colored container. At the end of manual mode, the robot moves and performs the task as desired by the user in accordance with the application's commands.[5]

III. PROPOSED METHODOLOGY

One of the important applications of profound learning is object acknowledgment. It often involves using a variety of techniques, such as a pre-prepared model that uses CNN, move learning, or starting from scratch with n datasets to understand the article with more ages to increase the accuracy of the result.

In this project, an accuracy of more than 95% is achieved in object recognition using the pre-trained model MobileNet. The model has thousands of photos ready to help it understand the object. a physical object, such as a chair, TV, monitor, or a person. This programme connects the Raspberry Pi to a USB camera. Using an IP Webcam or a Pi camera should also be possible.

In the proposed framework, profound learning is utilized. Among that Convolutional Brain Organization is utilized with pre-prepared model MobileNet to perceive the article with more exactness at ongoing. Keras, Tensorflow backend, numpy, and other libraries are installed when the Raspberry Pi is booted from the SD Card. The Raspberry Pi is integrated with a USB camera to serve as the application for real-time object recognition.

Hardware Required: Raspberry Pi, Adapter, HDMI port cable, USB Camera, SD card

Software Required: Raspbian OS with libraries Installed, SD card formatter, Etcher.

The scripts that make up this project's software are written in Python, one of the most popular programming languages for problems involving machine learning. Among the open-source deep learning frameworks that may be used for object detection are Caffe, Darknet, and Tensorflow. For this project, the work has been given to the open-source computer vision library OpenCV. A DNN module in OpenCV's vast collection of computations for continuous PC vision allows for the usage of pre-made models to infer information from recently referenced structures. It has been demonstrated that the models in this module's computer chip execution perform better than the ones in the systems to which they were transferred. OpenCV was built with NEON and FPV3 compatibility so that it could take full use of the hardware enhancements of the ARM CPU.The detector is run by this script with any of the models specified. It loads the network as soon as it starts. Frames are taken from the camera module when the network is loaded. Prior to being passed in to the organization, the pictures should be preprocessed and changed over into a mass (Double Huge Item). Normalization, resizing, and scaling are all part of this preprocess. To accommodate the selected network input, Images are scaled and resized. To conduct the normalisation, the RGB values of the photos are subtracted from the mean RGB values of the images in the training set. This is done to prevent the lighting of the input photos from shifting. When the blob is prepared, a forward pass is performed, during which the predictions are computed and the blob is dispersed over the network. The duration of this procedure is measured in inference time. Before presenting the final predictions, the detector filters out false positives twice. In the first stage, predictions with scores below the confidence threshold value are disqualified. False detections will be made if this value is set too low. Given that this value is 0.3, any detection with a confidence score below 30% is not taken into account as a detection. The second step of separating is non greatest concealment (NMS) that is utilized to guarantee an item is just recognized once, eliminating more modest covering bouncing boxes. While the separating is finished, bouncing boxes are drawn around the distinguished items along with a mark of the anticipated classes and their certainty scores.

One of the most challenging problems in this area of computer vision is object detection. Object detection's goal is to limit the range of items in a scene and assign names to the objects' jumping boxes. The most popular solution to this problem is to re-justify already-prepared classifiers such that names are assigned to bouncing boxes in a scene. For instance, a standard sliding window approach can be utilized where a classifier decides the presence of an object and its related mark for every single imaginable window in the scene. Be that as it may, this sort of approach has noteworthy impediments as far as high computational intricacy as well as high detection mistake rate. Starting late, significant neural frameworks (DNNs) have shown prevalent execution in an extent of different applications, with object detection being one of the key regions where DNNs have inside and out defeated existing systems. In particular, different convolution neural framework (CNN)- based techniques have been appeared to achieve forefront object detection execution. For example, in the Region-CNN (R-CNN) approach, a CNN configuration is used to create hopping encase suggestions an image as opposed to a sliding window approach, and thusly a classifier simply perform gathering on ricocheting box recommendation. In spite of the way that R-CNN can make top tier precision, the whole ace procedure is moderate and difficult to improve since each fragment must be arranged freely

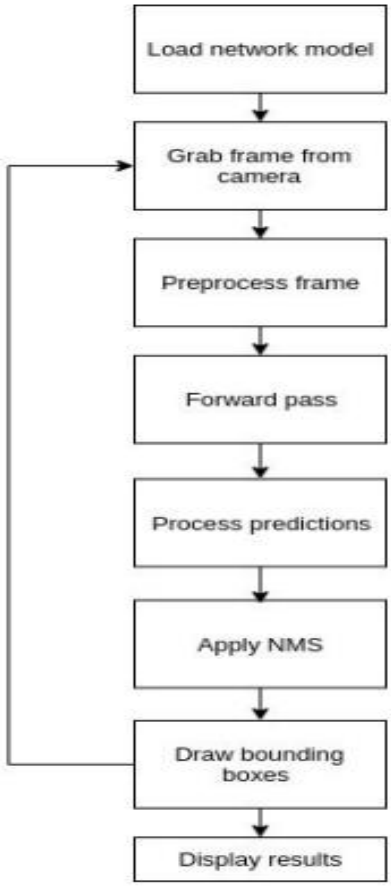

Figure 2: System Flow

Figure 2 indicates the system flow in which the output frames are designed, it initially involves loading the model grabing frame from the camera, processing the frame and further processing it for prediction and then applying NMS to find the frames and final results are obtained from it.

The objective of this experiment was to determine the running speed at which the models could capture live frames on a Raspberry Pi. They were only tested to see how fast they could go at different input sizes in this test; in experiment 2, they were tested to see how well they could find objects. 100 edges were caught and handled multiple times per model at various information sizes: 224 by 224, 160 by 160, and 96 by 96, respectively, and each frame's overall processing and inference times were noted. The Raspberry Pi 3 Model B+ computer's CPU is routinely timed at 1.4GHz. Even having the thermal throttle threshold set to 70°C, it was determined during setup testing that the detector couldn't operate for very long without overheating and throttling the CPU. After that, the temperature and clock speed were both set to 1.2 GHz. frequency remained constant. The memory divide gave the GPU access to 70 megabytes of memory, which is the absolute

minimum needed to run the desktop environment and the camera. This gave the central processor as much RAM as was logically necessary for executing the models. Another point to consider is how speed is impacted by GUI size. Using a resolution that is too high might result in a noticeably decreased throughput. Although 320 x 240 is still a high resolution for a GUI, it doesn't noticeably slow down the system throughout the testing.sufficient to see everything in the picture.

Figure 3: SSD detecting a person at three meters distance with image an input size of 160x160 pixels

In Figure 3 The experiment's goal was to evaluate the models' object detecting skills. It was done by calculating the detector's average confidence level across 80 frames of video that were taken from four different distances from the camera. Security in the house is one use for item identification. If object detection was used in a surveillance system, automatic intruder detection would be achievable. So, in this experiment, a human was chosen as the item to be detected.

IV. IMPLEMENTATION

Implementing real-time object detection using a Raspberry Pi involves several steps, including setting up the hardware, installing the necessary software libraries, and writing the code. Here's a general outline of the process:

1. Hardware setup: Connect a compatible camera module to your Raspberry Pi. Make sure it's properly connected and recognized by the system.

2. Software setup: Install the Raspberry Pi operating system (Raspbian or Raspberry Pi OS) on your Raspberry Pi
Update the system packages by running the following commands in the terminal:

```
sudo apt-get update
sudo apt-get upgrade
```

Install OpenCV, a popular computer vision library, by following the official OpenCV installation instructions for Raspberry Pi.

3. Download a pre-trained object detection model: Choose a pre-trained object detection model that suits your requirements. Popular models include SSD, YOLO, and Faster R-CNN.Download the pre-trained model weights and configuration files from the respective model's official repository.

4. Write the Python code: Create a new Python file and import the necessary libraries, including OpenCV

and the deep learning framework of your choice (such as TensorFlow or PyTorch).Load the pre-trained model using the downloaded weights and configuration files.Initialize the camera module to capture video frames.Preprocess the captured frames for input to the object detection model (e.g., resizing, normalization).Pass the preprocessed frames through the object detection model and obtain the bounding box

coordinates and class labels for detected objects.Overlay the bounding boxes and class labels on the frames.Display the annotated frames in real-time using OpenCV's display functions.

5. Run the code: Save the Python code on your Raspberry Pi.Open a terminal, navigate to the directory where you saved the code, and run the Python script using the command: `python your_script_name.py.Remember to refer to the specific documentation of the object detection model you choose and the corresponding deep learning framework for more detailed instructions on loading and using the model.

Note: Real-time object detection on a Raspberry Pi can be computationally intensive, so the inference speed may vary depending on the model and Raspberry Pi version. Optimize the code, consider using hardware acceleration (e.g., OpenCL, VPU), or explore lighter-weight models for better performance if needed.

V. RESULTS AND DISCUSSION

This section presents various results obtained by the Raspberry Pi 3B+ board which involves python programming embedded with the control of Raspberry Pi kit. The total agenda of object detection using tensor flow on both raspberry pi and windows is done and the results are shown:

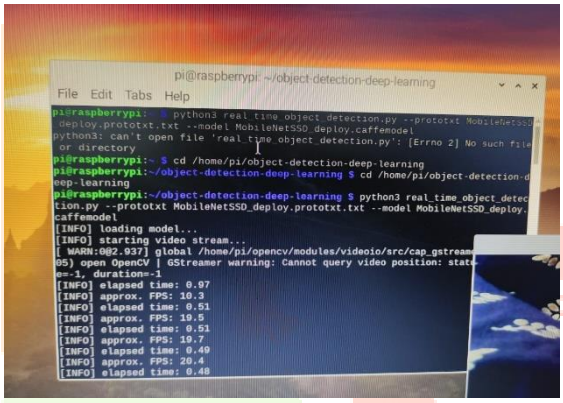

Figure 4: Object detection implementation using Raspberry Pi 3 B+

In Figure 4 it is depicted that real time object detection using Raspberry Pi3 using B model is implemented using the API like using the interface of raspberry Pi and detecting the objects surrounded by the Raspberry Pi CAM

Figure 5: Implementation of the complete connection using Raspberry Pi 3 B+ Model board

In the above Figure 5, the entire program need to run using the complete Raspberry Pi connection ports which form an API in the windows and run it like a Pro on any Software monitor Application.

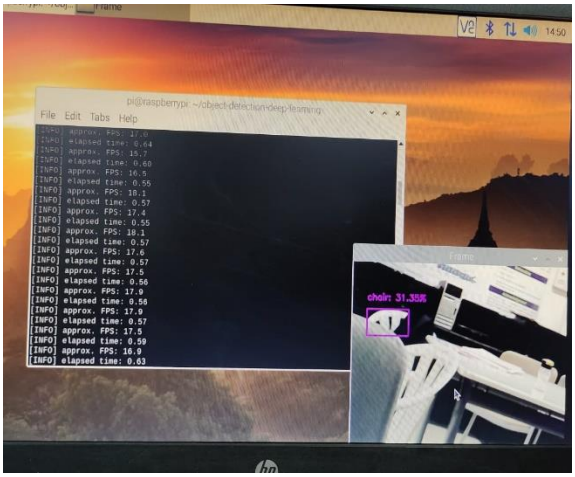

Figure 6: Pi camera interfacing and enabling, the captured image[bottle] can be seen in the monitor.

In Figure 6 it is displayed that the image captured by the Raspberry Pi camera can be found with percentage accuracy around 98% in the display interface. Here bottle is captured.

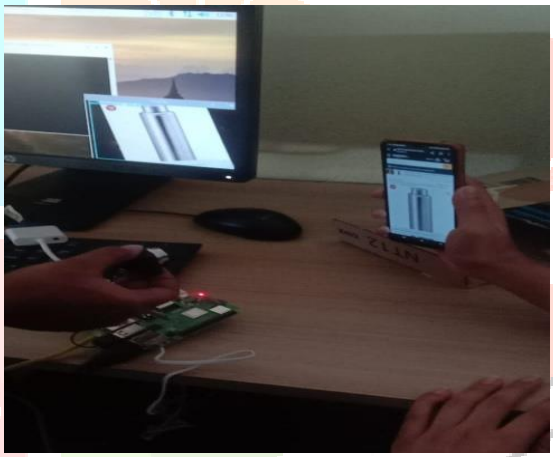

Figure 7: Chair is detected

In Figure 7 chair was placed and it is also detected with utmost accuracy and capture of the image is also segmented here when other images are detected with the camera and its utmost distance is found with 31% in the camera module.

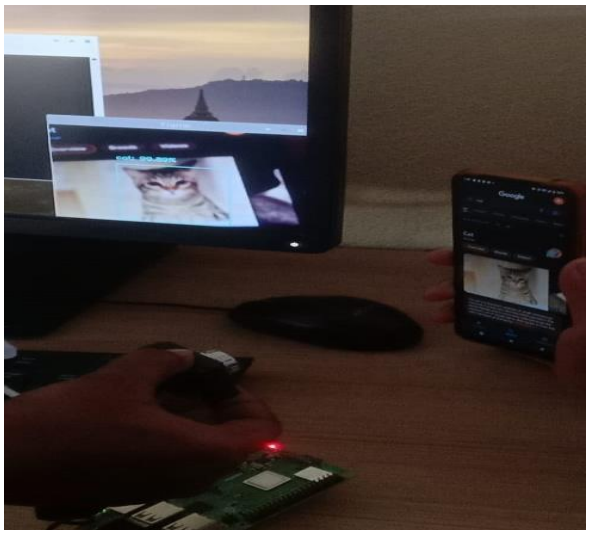

Figure 8: Cat's image is detected

In Figure 8 When cat's image is placed in interfacing with the Raspberry Pi camera the object is detected and its atmost accuracy is mentioned with the frame.

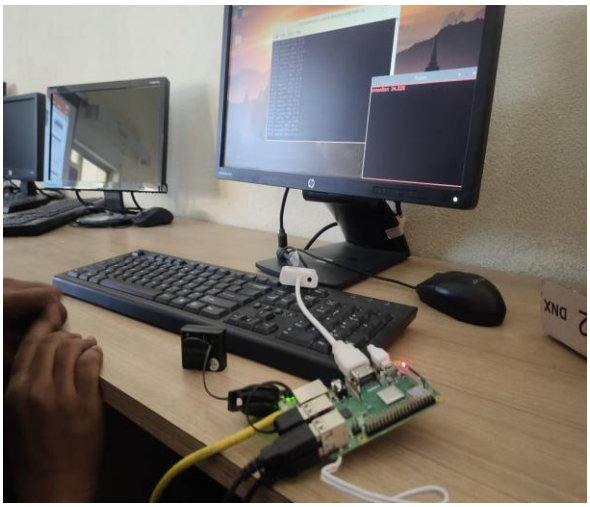

Figure 9: Complete Set up using PC interfacing of Raspberry Pi in an API

Figure 9 displays the complete connection established in frame with the PC and final output is obtained as Frames in it and detected with percentage accuracy.

VI. CONCLUSION

Raspberry Pi3! is quick, has at standard exactness with best twophase locators (on 0.5 IOU) and this makes it an incredible object detection model. Uses of Object Detection in areas like media, retail, fabricating, apply autonomy, and so forth need the models to be very fast (a little trade off on precision is alright) yet Raspberry Pi is additionally exact. This makes it the best model to pick in these sorts of uses where speed is significant either in light of the fact that the items should be ongoing or the information is simply too large. Some different applications like Security or Autonomous driving require the exactness of the model to be extremely high due to the delicate idea of the space, you don't need individuals passing on right? Perhaps we can't utilize Raspberry pi in such touchy spaces. Visual saliency detection, one of the most significant and testing assignments in PC vision, expects to feature the most predominant object districts in a picture. Various applications consolidate the visual saliency to improve their exhibition, for example, picture editing and division, picture recovery and object detection. Despite fast turn of events and accomplished promising advancement of object detection, there are as yet many open issues for future work.

On a tiny, specialised board, live object identification using TensorFlow is carried out for the object detection of the live video feed. It will be applied in a real-time manner. One of the key uses for embedded systems will be this. the most recent iteration of the artificial intelligence technology employed in tensorflow. Convolutional neural network is the core idea of tensor flow. This technology will also work with artificial intelligence, which may be applied to or used in several sectors.

REFERENCES

[1] Yong-ik Yoon, Jee-ae Chun, Tracking System for mobile user Based on CCTV. Information Networking (ICOIN), 2014 International Conference on, Phuket, 10-12 Feb. 2014, pp. 374-378.
[2] Viren Pereira, Vandyk Amsdem Fernandes, Junieta Sequeira, Low Cost Object Sorting Robotic Arm using Raspberry Pi. Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS), 2014 IEEE, Trivandrum, 26-27 Sept. 2014, pp. 1-6.
[3] Yimamuaishan. Abudoulikemu, Yuanming Huang, Changing, A Scalable Intelligent Service Model for Video Surveillance System Based on RTCP . Signal Processing Systems (ICSPS), 2010 2nd International Conference on (Volume:3), Dalian, 5-7 July 2010, V3-346 - V3-349.
[4] C. Bahlmann, Y. Zhu, Y. Ramesh, M. Pellkofer, T. Koehle, A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. IEEE Intelligent Vehicles Symposium, Proceedings,2005, pp. 255-260.
[5] Adrienne Heinrich, Dmitry Znamenskiy, Jelte Peter Vink, Robust and Sensitive Video Motion Detection for Sleep Analysis. Biomedical and Health Informatics, IEEE

[6] Journal of (Volume:18 , Issue: 3) , 2168-2194, 20 September 2013, pp. 790-798.
[7] Y. Amit and P. Felzenszwalb, "Object Detection", Computer Vision, pp. 537-542, 2014.
[8] J. Redmon, "Darknet: Opensource Neural Networks in C", Pjreddie.com, 2013-2016.
[9] P. Angelov, P. Sadeghi-Tehran, R. Ramezani, A Real-time Approach to Autonomous Novelty Detection and Object Tracking in Video Streams, International Journal of Intelligent Systems, ISSN 0884-8173, 2010, invited paper.
[10] P. Angelov, R. Ramezani,X. Zhou, Autonomous Novelty Detection and Object Tracking in Video Streams using Evolving Clustering and Takagi-Sugeno type Neuro-Fuzzy System, 2008 IEEE International Joint Conference on Neural Networks within the IEEE World Congress on Computational Intelligence, Hong Kong, June 1-6, 2008, pp.1457-1464, ISBN 978-1-4244-1821- 3/08.