KHULNA UNIVERSITY OF ENGINEERING & TECHNOLOGY

CSE 3200: System Development Project

Project Report
on
RADAR System with Real Time Object Detection

Developed by

Shahriar Parvej (1507109)

Abid Hasan Muin (1507119)

Md. Maruf Hasan (1507117)

Supervised by

Mr. Abdul Aziz

Lecturer

Department of Computer Science and Engineering

Khulna University of Engineering & Technology,

Khulna, Bangladesh

Signature of Supervisor

Acknowledgement

All praises go to Almighty ALLAH for his kindness & blessings. Without HIS desire we would not be here as we are today.

Thanks to our project supervisor ***Mr. Abdul Aziz, Lecturer***, Department of Computer Science and Engineering, KUET, for his untiring effort as well as strong support. He truly helped throughout the entire project with his correct decision & necessary advice. As a result, we are able to complete this **System Development Project** successfully.

Any constructive comments, suggestions, criticism from teachers as well as seniors will be highly appreciated and gratefully acknowledged.

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 1

Introduction

In this project we have designed a ‘**RADAR System with Real Time Object Detection’** using Arduino Board and the Processing Development Environment for measuring the distance while we used the SSD deep learning technique using Convolution Neural Networking method to identify the real time moving objects**.**

1.1 Statement of the Problem

We need to create a system where any obstacles can be detected by RADAR and using web camera we can detect/identify the object.

1.2 The Deployment

This application is developed as a 3rd year 2nd semester system development project, a course titled *“System Development Project”*. As a requisite we have gone through Arduino programming in Arduino Development platform to develop the RADAR which measures the distance of the object. We also need an Ultrasonic Sensor for detecting the objects, a small hobbyist Servo Motor for rotating the sensor and an Arduino Board for controlling them. We need to make a code and upload it to the Arduino Board that will enable the interaction between the Arduino and the Processing IDE to visualize the RADAR screen.

Apart from this, we also need to recognize the object which needs a Machine/Deep Learning Environment to deploy. In this case we used tensorflow API creating a virtual environment named tensorflow in anaconda. With the help of different packages and necessary python libraries we were able to do with the python code for our real time object detection. The webcam mounted used to take video as a input to be recognized all the objects in real time. This project is done with:

1. Arduino IDE
2. Processing IDE
3. Tensorflow API

1.3 Motivation

Security is one of the primary concerns in our life. In case of ground vehicles for self-driven mode we need both the RADAR system along with object detection and road condition observation. If we are able to do so, several accidents can also be handled. Taking it as a future concern, we have developed the project **"RADAR System with Real Time Object Detection".**

1.4 Objectives

- ✓ To deploy a program for measuring the distance of object with the help of sonar sensor.
- ✓ To create an on-screen radar display with the help of processing IDE which will visualize the RADAR to show the presence of the object in a finite range.
- ✓ To detect/identify the object with the help of webcam using machine learning technique.

CHAPTER 2

System Requirements

In this project we have worked on both hardware and software to develop the system. So the system is an integration of hardware and software. We used several hardware modules and worked on various software platforms using their library packages.

2.1 Software Requirements

Arduino IDE: The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, mac OS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino board.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Anaconda IDE: Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution is used by over 6 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and Mac OS.

Anaconda Navigator: Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, mac OS and Linux.

TensorFlow: TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

2.1 Hardware Requirements

Arduino Board: Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or bread boards (shields) and other circuits.

Sonar sensor: Sonar uses a sound transmitter and a receiver. When the two are in the same place it is monostatic operation. When the transmitter and receiver are separated it is bistatic operation. When more transmitters (or more receivers) are used, again spatially separated, it is multistatic operation. Most sonars are used mono-statically with the same array often being used for transmission and reception. Active sonobuoy fields may be operated multi-statically.

Servo motor: A servomotor is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors. Servomotors are not a specific class of motor although the term *servomotor* is often used to refer to a motor suitable for use in a closed-loop control system.

Bread Board: A breadboard is a construction base for prototyping of electronics. Originally it was literally a bread board, a polished piece of wood used for slicing bread. In the 1970s the solderless breadboard became available and nowadays the term "breadboard" is commonly used to refer to these.

Web camera: A web camera is a video camera that connects to a computer, and can let people see each other over the Internet. Most people that have webcams use them with an instant messenger to see each other at the same time. Webcams can also be used for recording videos and video blogs. The webcam can be part of a computer, mobile phone or it can be an independent device.

CHAPTER 3

Workflow

3.1 Workflow Diagram

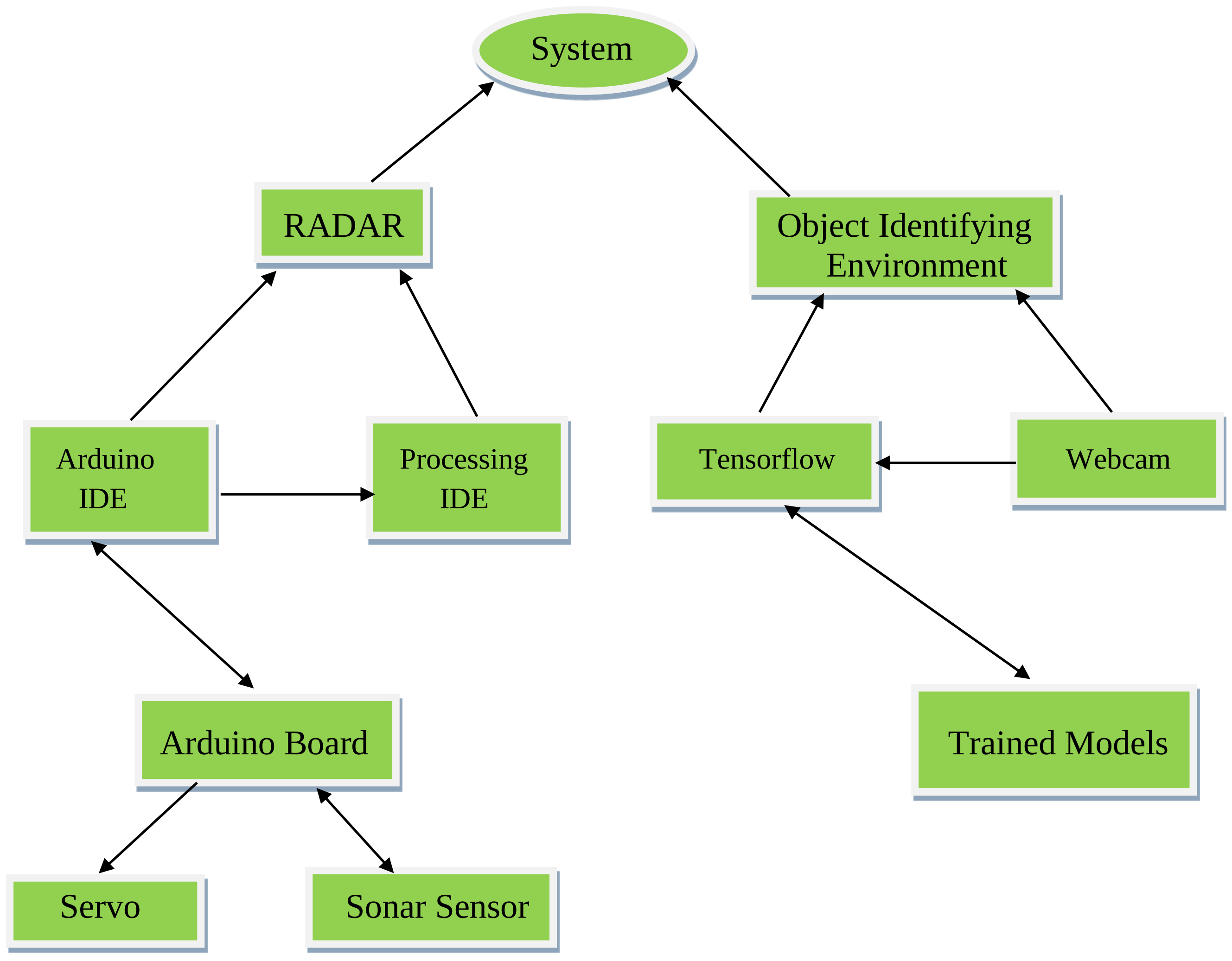


Fig 3.1.1: Workflow Diagram for RADAR System with Real Time Object Detection

3.2 Precise Description

The workflow diagram represents the entire project with a graphical view. According to the representation the system is a combination of RADAR and an object detecting (real time) environment.

The arduino board controls the servo for a particular range rotation, sonar is used to measure the distance. The Processing IDE takes the data as input and creates an environment according to a code written in Java which represents the area of range with a particular geometrical view that helps to detect position and the distance within range alongside the data which is also shown below that view.

The object identifying environment is based on tensorflow using the trained model and the data from the webcam input where the convolution neural network method is used to identify object.

CHAPTER 4

Project Implementation

4.1 Development of the RADAR

- ✓ First of all we made a cardboard stand for connecting the Ultrasonic sensor to the Servo motor. We folded it, glued it and secured to the servo motor using a screw.
- ✓ Then, we connected the Ultrasonic Sensor HC-SR04 to the pins number 10 and 11 and the servo motor to the pin number 12 on the Arduino Board.
- ✓ After then, we made a code and uploaded it to the Arduino Board that will enable the interaction between the Arduino and the Processing IDE.

After uploading a radar display will be activated which is created in Processing IDE, using the data from the Ultrasonic Sensor HC-SR04, and interaction established between that data with the Processing IDE it will show the informations on the radar display.

4.2 Object Detection (Real Time)

- ✓ At first, Anaconda latest version[1] (currently 5.3.1) for Python 3.7 was installed.
- ✓ In the following step, TensorFlow GPU was installed. But before proceeding some prerequisites were fulfilled. Such as –
    1. Nvidia GPU (GTX 650 or newer)
    2. CUDA Toolkit v9.0[2]
    3. CuDNN v7.0.5[3][4]

4. Anaconda with Python 3.7 (Optional)

- ✓ Then "Environmental Variables" were set in the system variable path.
- ✓ The GPU driver was updated.[5]
- ✓ A new Conda Virtual Environment was created with the name tensorflow_gpu. Then the environment was activated using "activate tensorflow_gpu" in Anaconda / Command Prompt.
- ✓ TensorFlow GPU was installed for Python using "pip install –ignore-installed –upgrade tensorflow-gpu" command.
- ✓ A new Python interpreter session was started using "python" command in the Anaconda / Command Prompt.
- ✓ Then to verify that the tensorflow_gpu was successfully installed following command was inserted in Python interpreter "import tensorflow as tf".
- ✓ Before installing "TensorFlow Models" some prerequisite models had to be installed. They are as such –
    1. Package name – pillow
    2. Package name - lxml
    3. Package name - jupyter
    4. Package name - matplotlib
    5. Package name – opencv

  The command for the installation process is "conda install <package-name>"
- ✓ Next step includes downloading tensorflow model[6] and linking it with the virtual environment.
- ✓ The TensorFlow Object Detection API uses Protobufs to configure model and training parameters. But before the framework can be used, the Protobuf libraries must be downloaded and compiled.
- ✓ After everything is done correctly, to ensure the valid installing the following command was inserted "jupyter notebook" from

"TensorFlow/models/research/object_detection" and thus this should start a new "jupyter notebook" server on the machine.

- ✓ For the "levelImg installation" a new Conda Virtual environment was created using "conda create –n labelImg pyqt=4" and then activated it using "activate labelImg".
- ✓ Under "TensorFlow" folder a new folder named "addons" was created and "labelImg" was downloaded[8] (TensorFlow/addons/labelImg).
- ✓ Now the dependencies and compiling packages were installed by opening Anaconda / Command Prompt then "cd" into "TensorFlow/addons/labelImg" then the following commands were used –
    1. conda install pyqt=4
    2. conda install lxml
    3. pyrcc –py3 –o resources.py resources.qrc
- ✓ To test the successful installation "python labelImg.py" was inserted in Anaconda Command Prompt under path "TensorFlow/addons/labelImg".

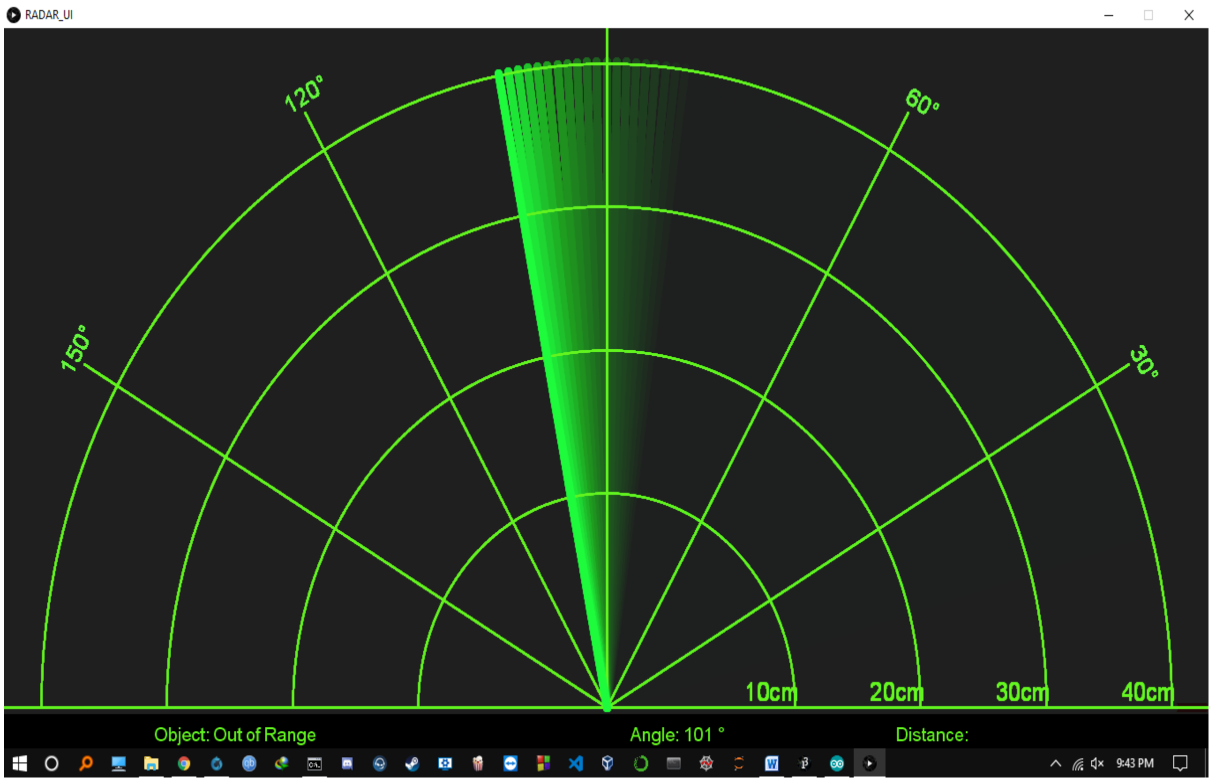
Lastly, after successfully installing all the required packages, downloading models for object detection and running the python code using necessary python libraries, real time objects were successfully detected from the webcam video stream as input.

CHAPTER 5

Experimental Result & Limitations

5.1 Measuring Distance and Position

(a) Object out of Range: When the object is out of range the radar monitor is clean as there's no interrupt in its way. It finds no objects and so the radar informs as no object is present in its range.



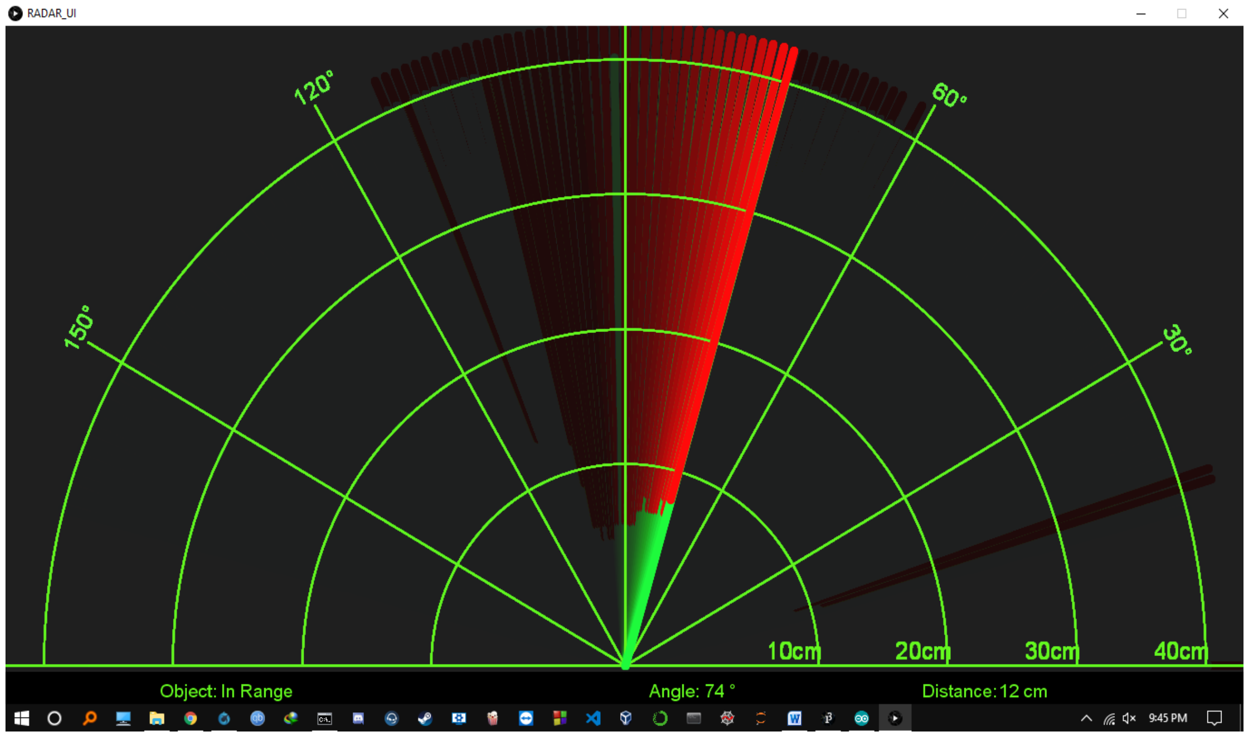
Object: Out of Range

Angle: 101 °

Distance:

Fig 5.1.1: No Presence of Objects within Range

(b) Object within Range: When the object comes within range the radar detects it with a measurement of distance and the position of the obstacle. This also gives us the idea about the size of the obstacle or object.



Object: In Range

Angle: 74 °

Distance: 12 cm

Fig 5.1.2: Presence of Objects within Range

5.2 Detecting Real Time Objects

When any pre-trained object appears in the frame it uses SSD to identify that real time object which was the video stream input of the webcam.

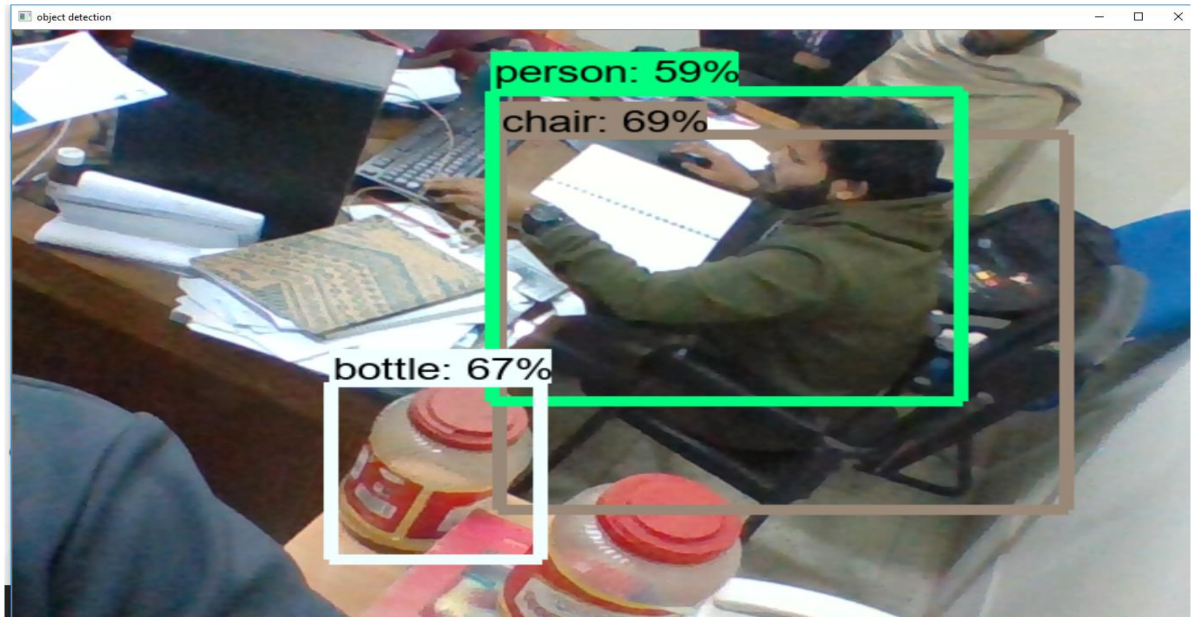


Fig 5.2.1: Detecting Person, Chair and Bottle

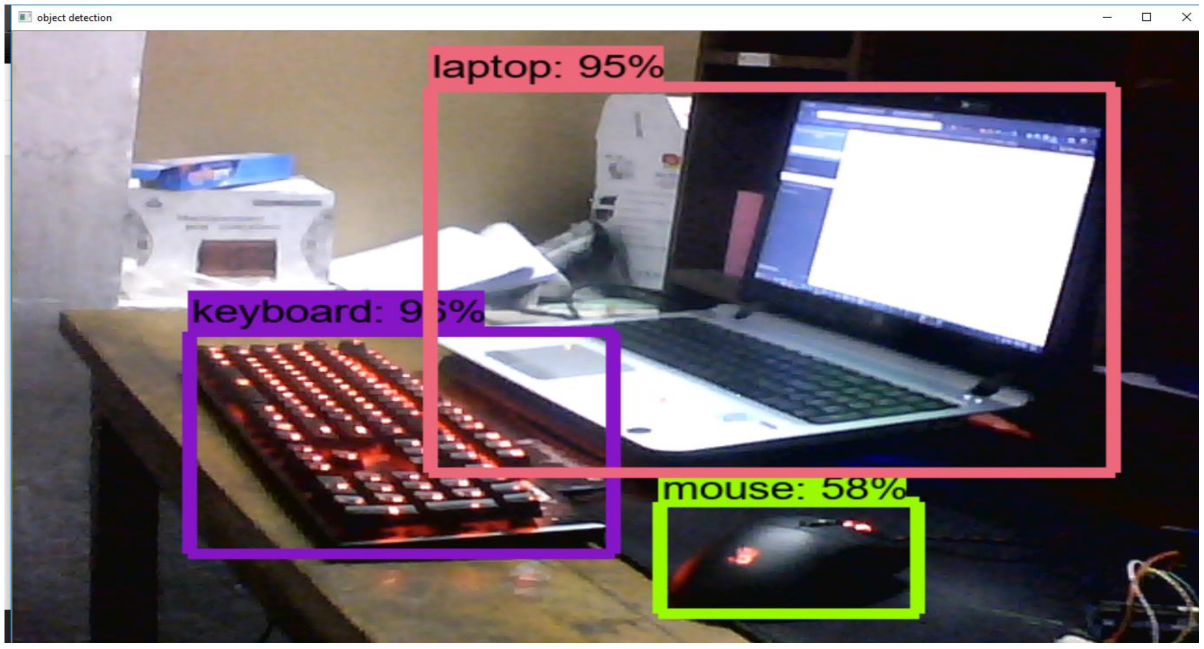


Fig 5.2.2: Detecting Laptop, Mouse and Keyboard

5.3 Features and Operations

- It can visualize obstacles within a finite range.
- It can detect and identify trained objects.
- Security issues can be handled by improving these features.

5.4 Limitations

- As we used sonar sensor we have a very short range to visualize obstacles.
- We used tensorflow API in object detection. As, it is already (machine learning) trained, so we can only identify ninety of the model objects that are initially trained for the system.
- Measuring speed of the moving object was not possible due to lack of hardware facilities and excessive cost.

CHAPTER 6

Future Works

- Autonomous vehicles which will include following things –
  - Self-driving mode
  - Accident prevention
  - Preemptive measures if system finds any anomaly (e.g.: Driver may fall into sleep or may be a sudden victim to Heart-Attack or Stroke)

- Smart assistant system for the blinds that guides –
  - Movement avoiding different obstacles
  - Make aware of damaged roads
  - Awares of watery or slippery conditions of roads processing with the trained image

CHAPTER 7

Conclusion

The **“RADAR System with Real Time Object Detection”** is successfully designed and developed using various software environments and hardware tools. The system can be used for further research on automation on various aspects where human interaction is a must nowadays. It can be embedded in a central system of vehicles that will reduce the possibilities of unwanted accidents. It can also be used in the smart assistance system for the blinds. But the main concern throughout the project was focusing on future development of a system that will prevent accidents most of the cases turning on the auto control mode when the vehicle is about to beyond control due to heart-failure/stroke/sleepy condition of the driver. It also will monitor and apply emergency braking system automatically if needed.

References

1. https://www.anaconda.com/download/
2. https://developer.nvidia.com/rdp/cuda-download
3. https://developer.nvidia.com/rdp/cudnn-download#a-collapse705-9
4. https://developer.nvidia.com/compute/machine-learning/cudnn/secure/v7.0.5/prod/9.0_20171129/cudnn-9.0-windows10-x64-v7
5. https://www.nvidia.com/Download/index.aspx
6. https://github.com/tensorflow/models
7. https://www.youtube.com/watch?v=COlbP62-B-U&feature=youtu.be&t=7m23s
8. https://github.com/tzutalin/labelImg