Abstract

The project describes the design and implementation of an autonomous service robot which is capable of performing household chores. The main objective of this project is to build a robotic arm which fetches and delivers the objects required by the user through voice commands. By giving the voice instructions, the robot finds the object which was instructed and brings the object back to the master. The project also describes the integration of various mechanisms involved, such as Object detection and tracking using an open source deep learning neural network object classifier, navigating the robot based on the depth data, determining the exact position of the object using pose estimation (6 DOF pose) of ArUco marker detection in ROS and utilizing thisinformation to pick the object by performing inverse kinematics. The system is designed keeping in mind the self -governance of the robot. i.e. the entire system will be capable of running on a minicomputer such as Raspberry Pi and the intense operations are isolated from the robot and are run on a server.

Contents

List of Figures

Chapter 1

Overview

This chapter deals with the introduction, definition of problem statement, objectives, motivation and the expected outcome of the project.

1.1 Introduction

Robotics mainly deals with the engineering , construction and operation of robots. The field of robotics involves looking at how any technology which is constructed physically can perform a task or play a role in any interface or new technology. The field of robotics has contributed immensely in the advancement of technology. Robotics influences every aspect of work and home.From healthcare to military use and emergency response,as well as in manufacturing industries ,robotics is the key driver of competitiveness and flexibility.

There is an entirely new breed of robots some in humanoid form ,and others that take highly practical forms all their own.Presently the main focus in robotics is towards service robots.Robotic arm which is mechanical and programmable performs the same function similar to the human arm.It is required to accomplish complicated tasks, aiming to make life of human easier.For instance robotic arms are widely used in industries to perform tasks which generally cannot be performed by humans, also robotic arms are developed to aid people with disabilities.

Simple tasks like picking up the bottle, opening and closing doors are mundane activities that people do every day and often take for granted. But for individuals with physical disabilities or cognitive impairments, these can be daunting tasks.The main objective of this project is to build a robot which can fetch and deliver the objects in a given space thus,assisting the specially abled and old aged.

The project describes the design and implementation of an autonomous service robot which is capable of performing household chores. The main objective is to design and build a robot that can fetch and deliver the articles within a given space. The robot navigates to the target object, picks the object and then navigates back to the user.

1.2 Motivation

Nowadays, it is robotics and digital innovations that make life easier and ensure that all people who are differently abled – participate in the world.Robotic devices provide people with very severe disabilities — those that affect both the use of their arms and legs — the ability to perform tasks with minimal assistance or even independently.

The main motivation behind this project is to help out differently abled people and the old aged to unleash their own potential with robotic arm.This project will provide a new approach for physically disabled people to live with harmony so that they could do their work by themselves without relying on other people. Nowadays, it is robotics and digital innovations that make life easier and ensure that all people who are differently abled – participate in the world.Robotic devices provide people with very severe disabilities — those that affect both the use of their arms and legs — the ability to perform tasks with minimal assistance or even independently.

The main motivation behind this project is to help out differently abled people and the old aged to unleash their own potential with robotic arm.This project will provide a new approach for specially abled people to live with harmony so that they could do their work by themselves without relying on other people.

1.3 Problem Statement

Statement: Developing a hardware solution for autonomous service which serves the master’s command by fetching the object asked by him/her.

1.4 Objective

Robots to assist differently-abled people is a challenging area that include most aspects as demanding tasks, partly unstructured environment and to some extent autonomous behavior.

The main objective of this project is to build a robotic arm which fetches and delivers the objects required by the user through voice commands.During the completion of this project, the objectives to be achieved are:

- To understand the basic configuration of Dexter Er2 Robotic arm.
- To interface communication between PC to Raspberry Pi and Raspberry pi to Dexter.
- To recognize and decode the input voice signal using the Google API.
- To recognise the object by using the SSD object detection algorithm.
- To fetch the object which was asked.

1.5 Project Outcome and Mode of Demonstration

The designed project will be demonstrated in a real-time scenario.By giving the voice instructions, the robot finds the object which was instructed and brings the object back to the master.

1.6 Organization of the Report

The report for the proposed project was systematically and adeptly organised according to the instructions.Firstly,Introduction to the project is elaborated.Then,The basic purpose of us choosing this particular project is explained in the motivation part of the report,followed by the Problem statement and Objective of the project.In,the last section of the first chapter the Project outcome and mode of demonstration where the expected result and where abouts of demonstration of the hardware and software is elaborated.In,the second chapter of "Literary Survey",which contains two sections:First section contains the explanation of the previous researches which were done by others on this regarding topic.Second one contains the Literature survey,which explains the papers and publications we have used as reference. The next chapter is System overview consists of Proposed Block diagram and the basic software and hardware requirements such as dexter arm,Raspberry Pi,Camera,Motors etc. This chapter is followed by Design and Implementation which consists of the heuristic flow of the project,the complete working of the system of the project.The next chapter consists of Results:The outcomes expected in the earlier section is justified here.Lastly,the final section is Conclusion where the advantages,drawbacks,applications and the future scope has been elaborated.

Chapter 2

Literature Survey

2.1 Previous Research

Asif Shahriyar et al., [1] have proposed a Design of a Voice Controlled Robotic Gripper Arm in which using Neural Networks. The aim of their work is to propose a method to build an efficient Bangla voice controlled robotic gripper mechanism using Neural Networks.This robot consists of three modules: speech command recognition module, object classifier module and robotic gripper arm module. At first, the robot takes voice commands on which objects to grab and displace; then it finds the object using the object classifier module. And finally it grabs and displaces the object using the robotic gripper arm module. The speech recognition module(here, instead of traditional algorithms such as Hidden Markov Models(HMM) in which the input command is taken and then processed and verified against acoustic and language models , Recurrent Neural Networks (RNN) are used which is better at speech recognition.) and the object classifier module uses two distinct neural networks along with additional hardware to perform their tasks. This paper presents the design and fabrication process of the robot discussed so that robots can be made using this design that works under different situations.

Arshad Javeed et al., [2] designed an Autonomous service bot which is capable of performing household chores. The main objective is to design and build a robot that can fetch and deliver the articles within a given space. The robot navigates to the target object, picks the object and then navigates back to the user. The paper also describes the integration of various mechanisms involved, such as Object detection and tracking using an open source deep learning neural network object classifier such as yolo, navigating the robot based on the depth data, determining the exact position of the object using pose estimation (6 DOF pose) of ArUco marker detection in ROS and utilizing this information to pick the object by performing inverse kinematics.

Bandara et al., [3] have developed an interactive service robot arm for object manipulation.The neck mechanism is developed and it is integrated with the arm and the vision sensor. The robot is capable of identifying user hand gestures, identifying the objects according shape and color and understanding user voice commands. The robot is intended to perform pick and place operation according to the user voice and gesture command.Inverse kinematics analysis has been done using a geometric generating inverse kinematic solution for robot arm.

Chandra Mouli et al.,proposed Design and Implementation of Robot Arm Control Using LabVIEW and ARM Controller [4]. The projected work was focused to control the end effector of the robot arm to achieve any accessible point in an amorphous region using LabVIEW, ARM (Advanced RISC (Reduced Instruction Set Computing) Machine) microcontroller and Dexter ER2 robotic arm. LabVIEW uses analytical method to design the inverse kinematic model of the robot arm. The inverse kinematic model and robot arm control was implemented using LabVIEW and ARM microcontroller. LabVIEW uses the parallel communication to send the joint angles of the robot arm to the ARM.

Mehdi Mouad et al.,[5] have described an adaptive and flexible algorithm of planning and re-planning for mobile robots which mainly aims to improve navigation and obstacle avoidance.It is based on two algorithms one of which is based on path planning to avoid static obstacles and the second uses reactive avoidance algorithm to avoid unexpected obstacles and reach the target.Principle of limit-cycles is used for the reactive avoidance algorithm.According to the planning and re-planning algorithm, the robot generates an initial safe trajectory and anticipates the collision with unexpected obstacles according to local smooth trajectory modification.This algorithm is not efficient as there were problems encountered when unexpected objects were detected and the robot failed to move continuously.

Ayrton Oliver et al.,[6] have designed model which uses Kinect as a navigation sensor for mobile robotics. Kinect captures 3D point cloud data which is used in a 3D SLAM to create 3D models of the environment and localise the robot in the enironment.After projecting the 3D data into a 2D plane the Kinect sensor data is used for a 2D SLAM algorithm . The main objective of the research is to show how viable Kinect sensor is by comparing the performance of Kinect on 2D and 3D SLAM algorithm.One of the disadvantage of the prototype is the reduction in the Kinect's resolution (i.e 320x240) due to networks bandwidth limitation and large amount of data.However, the prototype offers significant advantages over the laser scanners.

Karthi Balasubramanian et al.,[7] have designed and implemented obstactle detection and avoidance robot.The key aspects of the design is obstacle detection, pattern recogntion and obstacle avoidance.The projected design uses a SONAR module that detects obstacles as well as calculates the distance of the obstacles.Machine vision is one of the components used in the design to detect the object of a particular colour by using Hough Transform.As come across one of the merits of this design is the faster execution of the algorithm which takes 01 sec for the real time opertion in Java.

Z.Riaz et al.,[8]have proposed a complete SLAM solution for the indoor mobile robots,addressing feature extraction, autonomous exploration and navigation using the continuously updating map.The algorithm proposed uses Hough Transform for feature extraction such as lines and edges. By using the relative filter the localization was accomplised.The entire platform used for the prototype is Pioneer PeopleBot with SICK Laser Measurement System and odometry.The main objective is to make the the mobile robot completely autonomous by enabling it to build a map from which it localizes itself in real time applications.The results obtained proposed approach using SLAM was found to be very accurate which serves as the main advantage.

Vision guided robots have more ability, functionality and adaptivity in industrial assembly lines than normal robots[9].The concepts of computer vision and grasping applications are applied on Selective Compliant Assembly Robot Arm.Deep Convolutional Neural Network (CNN) model, called KSSnet, is developed for object detection based on CNN Alexnet using transfer learning approach. SCARA training dataset with 4000 images of two object categories associated with 20 different positions is created and labeled to train KSSnet model. The position of the detected object is included in prediction result at the output classification layer. The various object detection methods were compared by implementing on a circular object (to make the task simple) and observed A combination of Zerocross and Canny edge detectors has been used for position measurement of a circular object in undistorted images based on camera calibration parameters. This method has been successfully implemented to move robot to the desired position where the grasped object should be placed with positioning and 100 per cent of successful place detection. The various object detection methods implemented are quite complex and slow and hence are not quite efficient.

Robotics have largely been intelligent to accomplish complex and simple tasks.The paper[10] describes the design and development of an autonomous robotic arm with enhanced intelligence. It also explains the fabrication of the arm and the realization of the proposed algorithm. Here the servomotors are used as actuators which help in pick and place of the object. The communication has been established using a computer through a serial port interface using Flash Magic terminal controls the arm. They however do not use any object classification methods. The robots are designed to randomly pick and place objects whose distance is specified by the user and hence do not have great efficiency, although the arm works on a very high speed.They have used the embedded C language and Keil to deploy the serial communication between the arm and the computer.Adding user interface and intelligence to the robot for picking and placing of objects is done by addressing user interrupts instantly while operating in random pick and place mode.

A robotic arm reaching a particular position defined by a set of coordinates, pick up the object and place it at target location according to another set of coordinates, in a well defined hemispherical 3D space is described in the paper[11]. This robot arm in has 5 degree of freedom and micro servomotors are used as principle actuators. The arms control is done by an interface application developed in Rapid Application Development (RAD) software, using micro-C programming language. The user in the interface application enters the object position and destination co-ordinates and the object is grabbed by using gripper. The on-board micro-controller used is PIC 16F88 (8 bit micro-controller), which generates a PWM signal and establish a communication between computer and hardware. Visual Basic 6.0 serves as GUI software and is used to communicate with the micro-controller from the computer.

Assistive robotic arms are of huge advantage. The paper[12] describes one such robotic arm which is mounted on the wheel chair uses the computer vision algorithm to help pick objects.The object classification. he design utilizes a robotic arm with six degrees of freedom, an electric wheelchair, computer system and two vision sensors. One vision sensor detects the coarse position of the colored objects placed randomly on a shelf located in front of the wheelchair by using a computer vision algorithm. The other vision sensor provides fine localization by ensuring the object is correctly positioned in front of the gripper. The arm is then controlled automatically to pick up the object and return it to the user. The robot however usually experiences a delay of about 3.5 - 4seconds during each cycle.

Zhong-Qiu Zhao et al.[13] have provided with an exegesis of The Object detection with Deep learning.The overview of the paper is mainly focussed on the various methods of object detection and classification.The main focus Is laid on the Convolutional Neural Network(CNN) and types of CNN.A thorough Experimental analyses are also provided to compare various methods and conclusions are drawn on the basis of results obtained.The various methods of object detection such as R-CNN,Fast CNN,Faster CNN,YOLO and SSD have been described in detail along with their respective architectures.The two methods which were likely to implemented in this system were YOLO and SSD: through this review paper it was clear that YOLO has a difficulty in dealing with the small objects in groups,which is caused by strong spatial constraints imposed on bounding box predictions and struggles to generalize objects in new/unusual aspect ratios and configurations as such the answer to these problems were SSD. the SSD takes advantage of a set of default anchor boxes with different aspect ratios and scales to discretize the output space of bounding boxes. To handle objects with various sizes, the network fuses predictions from multiple feature maps with different resolutions.

Sergio Hernandez and others [14] have presented a thorough review on Design and Implementation of a Robotic using ROS and MoveIt.This paper deals with lesser degree of freedom with respect to the robotic arm where as the proposed system here deals with the six degrees of freedom. The importantance is laid on the ROS Dynamixel packages are available in a dynamixel motor stack of ROS. This stack contains packages that are used to interface with Robotis Dynamixel line of motors.The ouput file is the given to the URDF file: The URDF is an XML specification, which describe the robotic arm model. This file can calculate the 3D pose of arm robotic and detect potential collisions between the arm and its environment.This method of creating the URDF file has been adapted into the proposed system.

The paper by N.U.Alka et al.[15] have provided exegesis on the voice controlled pick and place vehicle using an android application. The programming language used is Arduino which is a simplified version and compatible with both C and C++programs. The Arduino project provides the Arduino Integrated development environment,which is a cross-platform application written in Java programming language. Applications include Ardupilot, Arduino me, Arduino phone, D.C. motor control etc.The proposed project also implements the pick and place feature including the object detection feature.The important parameters regarding the intended maximum weight,the angles and the maximum weight lift were analogous to the proposed project.

Sudarshan el at.[16] have laid down a detailed elucidation of manipulation of the robotic arm manipulation either simple or complex and the ways of maximizing the efficiency.The arm will be capable of picking and placing the objects and the trajectory planning is done with the help of Inverse Kinematics. Geometric approach is followed to obtain inverse kinematic solutions.The end result of the proposed system is the actuation of the arm to pick up the objects intelligently following the command and the method of inverse kinematics will also be implemented.

The paper introduces SSD, a fast single-shot object detector for multiple categories. A key feature of the model is the use of multi-scale convolutional bounding box outputs attached to multiple feature maps at the top of the network. This representation allows to efficiently model the space of possible box shapes.It experimentally validate that given appropriate training strategies, a larger number of carefully chosen default bounding boxes results in improved performance.It proposes a method to build SSD models with at least an order of magnitude more box predictions sampling location, scale, and aspect ratio, than existing methods. It also compares SSD with various other deeplearning objection detection methods and establish its advantages and prevalence over the other methods.

2.2 Summary of Literature Review

Paper [1] proves voice recognition using RNN and NLP is more efficient than conventional Hidden Markov chain. In 2019 March, Google has developed Voice to text API using RNN and NLP [20]. henceforth, voice recognition can be done efficiently using 2019th version Google API. For pick and place operations it is necessary to model a multi-DOF arm [2]. The problem of computing the joint variables through the position and orientation of the robot's end-effector is called inverse kinematics [3][4]. Robot navigation requires processing the depth data and making decisions based on its current location. [16][18] has described robot navigation using depth data and 3D point cloud and plan the feasible path.A simple pinhole camera is used to obtain the co-ordinates of the object. ArUco markers are used to track an object and determine its (x,y,z) coordinates in space.

Currently [17] shows that faster RCNN networks are the most accurate networks however they use a very high computational speed.Yolo network however has a very high computational speed making it one of the fastest deep learning networks in use however, they are quite low on efficiency.Hence SSD offers a balance between the efficiency and computation speed.Another major advantage of the SSD algorithm is that we are able to run it on a video and obtain sufficiently high accurate results.

Chapter 3

System Overview

This chapter deals with the overall architecture of the system, which includes the system block diagram, hardware and software constituents.

3.1 Proposed Block Diagram

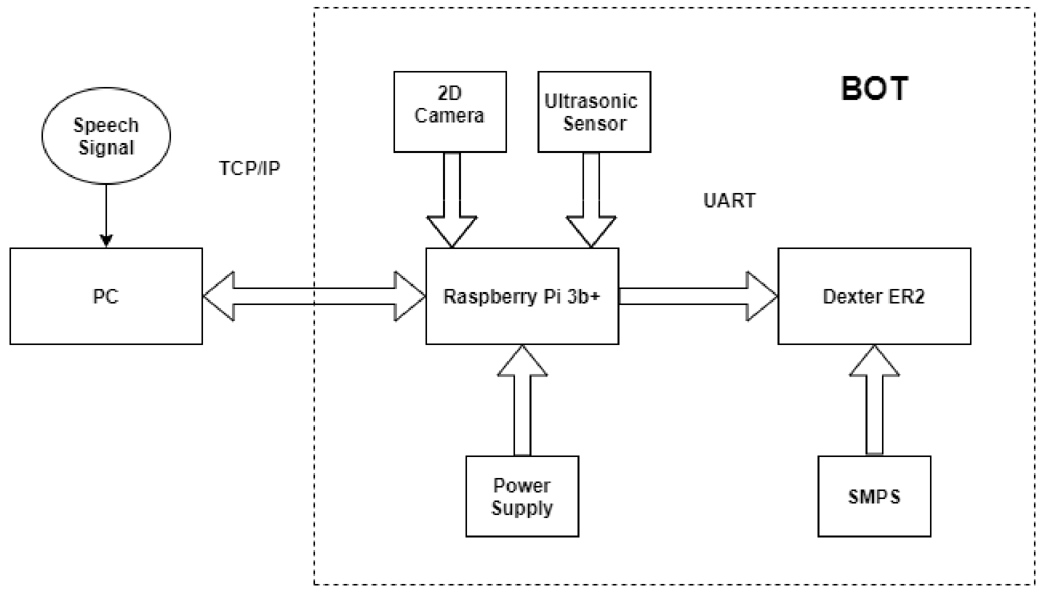


Figure 3.1: Block Diagram

Figure 3.1, depicts the block diagram of the proposed system.The block diagram as conspicuous contains three modules namely:the PC,the Raspberry pi and the Dexter arm.The Pi module and the Dexter arm can be integrated under the single module as Bot.The voice command is given by the user to the PC,then the PC communicates with the Raspberry pi module and after numerous calculations it communicates with the Dexter arm thus completing the desired action.

3.2 Software Requirements

The Project uses wide range of software applications for various applications such as deep learning used object recognition and classification, Google voice assistant API which is used to recognise and process the speech signals, Raspbian OS which is necessary to run the system along with the python coding language and its various coding modules helps in integrating the whole project.

- **Raspbian OS:** Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware. An operating system is the set of basic programs and utilities that make your Raspberry Pi run. However, Raspbian provides more than a pure OS: it comes with over 35,000 packages, pre-compiled software bundled in a nice format for easy installation on the Raspberry Pi.
- **Python Coding Language**: Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability.

  Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. The python language consists of various libraries which are used in the project such as :

  - **Tensorflow:** An open source machine learning library for research and production.Tensorflow architecture works in three parts:
    Preprocessing the data,Build the model Train and estimate the model.In this model the input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output. It includes a wide range of efficient functions which help in implementing the deep learning models.
  - **OpenCV:** OpenCV is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together

to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

- **Google Voice Assistant API:** Google Cloud Speech-to-Text enables developers to convert audio to text by applying powerful neural network models in an easy-to-use API. The API recognizes 120 languages and variants to support your global user base.It is one of the most popular python library which can enable voice command-and-control, transcribe audio from call centers, and more. It can process real-time streaming or prerecorded audio.The recent advancement in the Google API presents an all Recurrent Neural Network Transducers model making it much faster and efficient.
- **Recurrent Neural Network Transducers:** RNN-Ts are a form of sequence-to-sequence models that do not employ attention mechanisms. Unlike most sequence-to-sequence models, which typically need to process the entire input sequence (the waveform in our case) to produce an output (the sentence), the RNN-T continuously processes input samples and streams output symbols, a property that is welcome for speech dictation. In the implementation, the output symbols are the characters of the alphabet.The RNN-T recognizer outputs characters one-by-one, as you speak, with white spaces where appropriate. It does this with a feedback loop that feeds symbols predicted by the model back into it to predict the next symbols, as described in the figure below which gives the representation of an RNN-T, with the input audio samples, x, and the predicted symbols y. The predicted symbols (outputs of the Softmax layer) are fed back into the model through the Prediction network, as yu-1, ensuring that the predictions are conditioned both on the audio samples so far and on past outputs. The Prediction and Encoder Networks are LSTM RNNs, the Joint model is a feedforward network (paper). The Prediction Network comprises 2 layers of 2048 units, with a 640-dimensional projection layer. The Encoder Network comprises 8 such layers.

- **Natural Language Processing With spaCy in Python:** spaCy is a free and open-source library for Natural Language Processing (NLP) in Python with a lot of in-built capabilities.NLP is a subfield of Artificial Intelligence and is concerned with interactions between computers and human languages. NLP is the process of analyzing, understanding, and deriving meaning from human

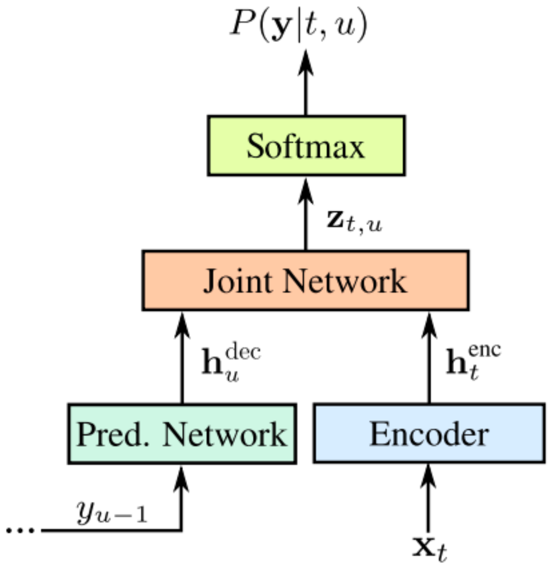


Figure 3.2: RNN-T

languages for computers.The important thing is to derieve insights from unstructured data and for that the input text must be coverted to machine level , for which NLP is used.

- **IKPy:** IKPy is a Python Inverse Kinematics library, designed to be simple to use and extend. It provides Forward and Inverse kinematics functionality, bundled with helper tools such as 3D plotting of the kinematics chains. Being written entirely in Python, IKPy is lightweight and is based on numpy and scipy for fast optimization. IKPy is compatible with many robots, by automatically parsing URDF files. It also supports other (such as DH-parameters) and custom representations. Moreover, it provides a framework to easily implement new Inverse Kinematics strategies. It can also be used as a standalone library.It is executed as described below

  * Compute the Inverse Kinematics of every existing robot.
  * Define the kinematic chain using arbitrary representations: DH (Denavit–Hartenberg), URDF standard
  * Automaticly import a kinematic chain from a URDF file.
  * IKPy is precise (up to 7 digits): the only limitation being your underlying model's precision, and fast: from 7 ms to 50 ms (depending on your precision) for a complete IK computation.
  * Plot your kinematic chain: no need to use a real robot (or a simulator) to test the algorithms

• Object Identification Algorithm : Single Shot Multi-box Detector is a method for detecting objects in images using a single deep neural network. Our approach, named SSD, discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location. At prediction time, the network generates scores for the presence of each object category in each default box and produces adjustments to the box to better match the object shape. Additionally, the network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes. The SSD model is simple relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature resampling stage and encapsulates all computation in a single network. This makes SSD easy to train and straightforward to integrate into systems that require a detection component.

3.3 Hardware Requirements

The implementation of the projet requires the use of various hardware components which includes the Dexter Er-2 robotic arm, various sesnsors, power supply, raspberry pi 3b+, a computer/laptop which behaves as the server.

- **Raspberry pi 3b+:** The raspberry pi as shown in figure 3.2 is small on borad computers which provide a platform to perform a lot of computations faster than a regular computer. The below are the specifications of raspberry 3b+ system. Specifications:
  - Quad core 64-bit processor clocked at 1.4GHz.
  - 1GB LPDDR2 SRAM.
  - Dual-band 2.4GHz and 5GHz wireless LAN.
  - Bluetooth 4.2 / BLE.
  - Higher speed ethernet up to 300Mbps.
  - Power-over-Ethernet capability (via a separate PoE HAT)

Figure 3.3: Raspberry pi 3 b+

- **Ultrasonic Distance Sensor (HC-SR04):** The HC-SR04 Ultrasonic Distance Sensor as shown in figure 3.3 is a 4 pin module used for detecting the distance to an object using sonar. The HC-SR04 uses non-contact ultrasound sonar to measure the distance to an object, and consists of two ultrasonic transmitters (basically speakers), a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce off any nearby solid objects, and the receiver listens for any return echo. That echo is then processed by the control circuit to calculate the time difference between the signal being transmitted and received. Specifications:
  - Operating voltage: +5V.
  - Practical Measuring Distance: 2cm to 80cm.
  - Accuracy: 3mm.
  - Measuring angle covered: <15°.

- Operating Current: <15mA.
- Operating Frequency: 40Hz.
- Number used : 8

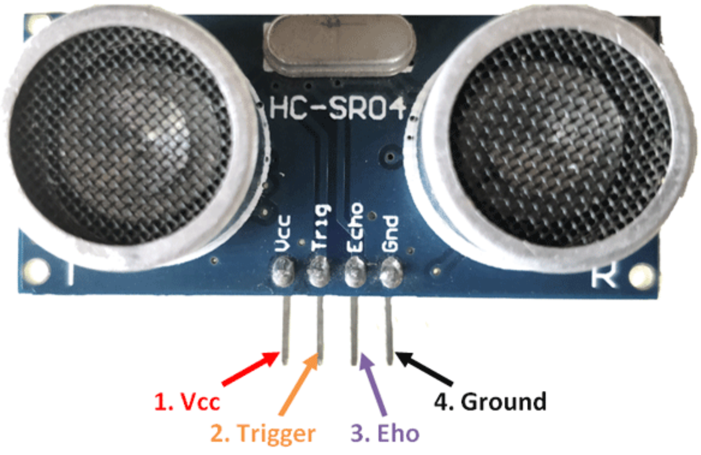


Figure 3.4: HC-SR04 Ultrasonic Sensor

- **USB Camera:** The See3CAM-CU55 is a 5-MP USB3.1 Gen1 UVC Color Camera as shown in the figure-3.It is a two-board solution containing camera sensor board as shown in figure 3.4 based on 1/2.5" AR0521 image sensor from on Semiconductors and USB3.1 Gen 1 Interface board. The powerful on-board Image Signal Processor (ISP) brings out the best image quality of this 2.2micron pixel 5MP AR0521 CMOS image sensor making it ideal for next generation of high resolution image applications.

Figure 3.5: USB Camera Module

- **Dexter ER2:**Dexter ER2 is heavy Duty Robotic Arm is a 6 Axis robotic Arm + Servo Gripper. It uses seven metal gear servo motors with 15Kg/cm torque and two servo motors with 7Kg/cm torque. Robot Arm has 6 degrees of freedom as shown in figure 3.5 which includes Base rotation, Shoulder rotation, Elbow rotation, Wrist pitch and roll. Out of which Shoulder rotation, Elbow rotation, Wrist pitch has two 15Kg/cm torque servo motors in parallel for enhancing the performance of the robotic arm.

Specifications:

- Number of Axes:5
- Axis Movement
  - Axis 1: Base rotation 180°
  - Axis 2: Shoulder rotation*180°
  - Axis 3: Elbow rotation 180°
  - Axis 4: Wrist pitch 180°
  - Axis 5: Wrist roll 180°
- Maximum Operating Radius : 320mm
- End Effecter: DC servo motor based
- gripper with Parallel
- finger motion
- Maximum Gripper Opening : 55mm
- Actuators :5V DC servo motors
- Maximum Payload: 50 grams
- Weight: 1.5Kg

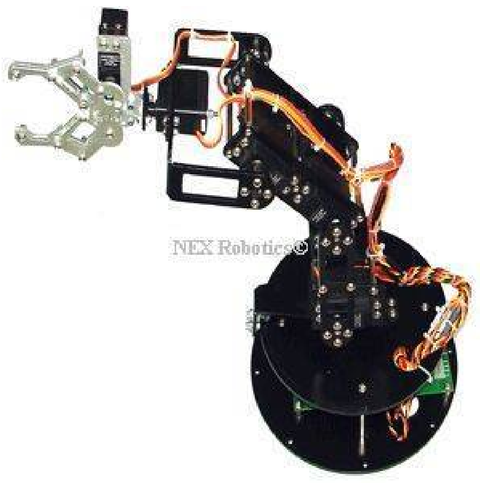


Figure 3.6: Dexter ER2

- **Laptop/Computer(Server):** The server is responsible for performing the intensive tasks that the Robot (Raspberry Pi 3B+) cannot perform on its own. The operations performed by the Server include processing the voice signal, Object classification – to identify all the objects in any given image sent by the Robot and determine the action that needs to be performed.

- **Power Supply:** A switched-mode power supply (SMPS) is an electronic circuit as shown in figure 3.6 that converts power using switching devices that are turned on and off at high frequencies, and storage components such as inductors or capacitors to supply power when the switching device is in its non-conduction state. Like other power supplies, an SMPS transfers power from a DC or AC source (often mains power) to DC loads, such as a personal computer, while converting voltage and current characteristics. Unlike a linear power supply, the pass transistor of a switching-mode supply continually switches between low-dissipation, full-on and full-off states, and spends very little time in the high dissipation transitions, which minimizes wasted energy. A hypothetical ideal switched-mode power supply dissipates no power.

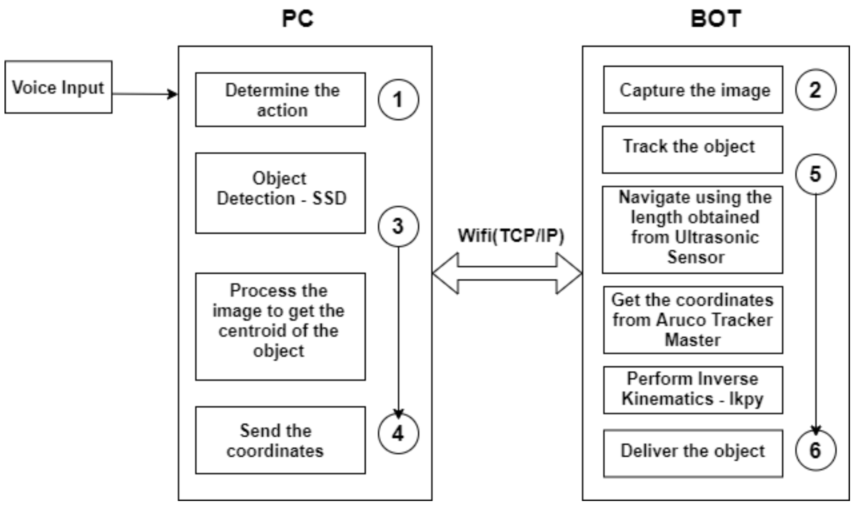
Figure 3.7: Switched Mode Power Supply

Chapter 4

Design and Implementation

This chapter is dedicated for detailed explanation of the architecture of the system and flow of program code used for execution, along with description of working.

4.1 Flowchart



PC
BOT
Voice Input
Determine the action
1
Object Detection - SSD
3
Process the image to get the centroid of the object
Send the coordinates
4
Wifi(TCP/IP)
Capture the image
2
Track the object
5
Navigate using the length obtained from Ultrasonic Sensor
Get the coordinates from Aruco Tracker Master
Perform Inverse Kinematics - Ikpy
Deliver the object
6

Figure 4.1: Flow diagram

Figure 4.1,depicts the flow diagram of the proposed system.The flow diagram is the chronological order in which the actions take place inside the system.The first main module is the PC:The operations such as action determination,object detection and sending the coordinates take place.The second module is the Bot:The operations such as capturing,tracking,navigating and delivering the object take place. The two modules communicate with each other in TCP/IP configuration.

4.2 Working

The project is divided into 6 parts:
1. Robot assembly
2. Speech recognition
3. PC and Raspberry pi communication
4. Object recognition
5. Robot Navigation
6. Raspberry pi to Dexter communication

4.2.1 Robot assembly

The Robotic system was driven by two DC geared motors, along with 2 castor wheels for support. The Dexter ER2 Robotic arm was mounted on the chassis, the Kinect sensor and a secondary webcam were fixed to the chassis at a vertical height of 70 cm from the base. IR modules are placed at the front region of the bot for obstacle detection in the range of 0-20cms. The bot is powered with 5V-12A SMPs.

4.2.2 Speech recognition

The speech signal, recorded using the system's microphone is converted to text format by using Google's Speech Recognition API in Python. Input voice signal is hence recognized using Google API which uses RNN-T and converted to text.
Once the entire signal has been decoded, the sentence is analyzed to determine the action that needs to be performed by processing the text (Natural Language Text Processing). Here, the given sentence is split into tokens. A token, in this case, is a group of words that are interrelated in the sentence. The generated tokens are then categorized or assigned as values to different parts of speech using Natural language toolkit spaCy, POS tags in python. After tagging the words in the sentence, the words are Figure above shows the

algorithm applied to one of the commands given to the Robot, "Dexter, Bring the water bottle". If either the action, that is determined from the verb or the object is not recognized or the sentence doesn't start from Dexter, then the instruction is rejected. The accuracy of analyzing the text can be improved by having a large data set, against which the tagged words are compared. The set of actions and objects that the robot can identify are as

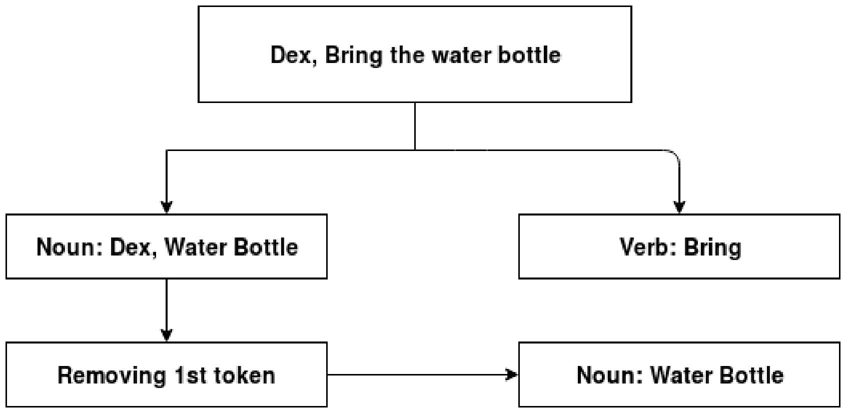


Figure 4.2: Natural language processing

follows:
Actions: {get, fetch, bring, want, need, ...}
Objects: {bottle, can, flask ...}
These sets consist of words that are synonyms of a particular object class. Hence, including more number of words would help to better interpret the given command.

4.2.3 PC and Raspberry pi communication

Once the voice command is decoded, the information about Action and Object will be sent to Raspberry Pi using TCP/IP protocol. With help of client server programming in python, PC being server and Raspberry pi being client while Action-object information and Coordinates of the image transfer to Raspberry pi and PC being client, Raspberry pi being server while transferring images.

4.2.4 Object recognition

For identifying the real-time image sent by the bot to the PC, Object detection algorithm - SSD (Single Shot Multibox Detector)an open source Fully Convolutional Neural Network is used. SSD is designed for object detection in real-time. Faster R-CNN uses a region proposal network to create boundary boxes and utilizes those boxes to classify objects. While it is considered the start-of-the-art in accuracy, the whole process runs at 7 frames per second. Far below what a real-time processing needs. SSD speeds up the process by eliminating the need of the region proposal network. To recover the drop in accuracy, SSD applies a few improvements including multi-scale features and default boxes. These improvements allow SSD to match the Faster R-CNN's accuracy using lower resolution images, which further pushes the speed higher. SSD uses VGG16 to extract feature maps. Then it detects objects using the Conv4-3 layer. For illustration, we draw the Conv4-3 to be 8×8 spatially (it should be 38×38). For each cell (also called location), it makes 4 object predictions.Each prediction composes of a boundary box and 21 scores for each class (one extra class for no object), and we pick the highest score as the class for the bounded object. Conv4-3 makes a total of 38 × 38 × 4 predictions: four predictions per cell regardless of the depth of the feature maps. As expected, many predictions contain no object. SSD reserves a class “0” to indicate it has no objects.It returns the boundary of the object if the object is detected.

4.2.5 Robot Navigation

For the Robot to navigate to the desired location, the Robot has to identify the objects present around it. It also has to determine how far these objects are with respect to it so that it can have an estimate of the distance to be covered in order to reach a specific object this can be obtained from SSD algorithm but the distance at which the arm has to actuate will be determined by calculating depth of the object using the length obtained with the help of the Ultra Sonic Sensor.

Determining the movement of the bot:

The locations of the objects were obtained in form of a bounding box.The (xmin, ymin) were actually the coordinate of the top-left most edge of the rectangular bounding box and the (width,height) were actually the width in x direction and height in y direction. Figure below shows the boundary of the object(here, bottle) detected. Hence, centroid of

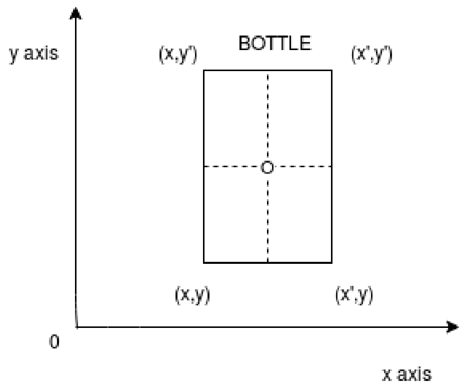


Figure 4.3: Calculating centroid

the object can be determined using the following formula:

$$X_o = \frac{x' - x}{2}$$

$$Y_o = \frac{y' - y}{2}$$

where: (Xo,Yo) is the centroid of the object.

Converting from pixels to centimeters:

(xo,yo) = 0.0264583333(Xo,Yo)

where 0.026458333 is the conversion factor to obtain cm values from pixels.

The bot will start navigating according to the obtained centroid.

Navigation using Vector Field Histogram method:

The VFH method uses a two-dimensional Cartesian histogram grid as a world model. This world model is updated continuously with range data sampled by on-board range sensors.The idea of VFH is based on VFF (Virtual Force Filed) method. As the name indicates its a field, so obstacles detected at a certain distance from vehicle will apply repulsive force on vehicle to move away from obstacles and to draw the vehicle toward goal point an attractive force is used. VFH uses a certainty grid-like radar screen, where obstacles found by sensor will count up sum certainty value at the corresponding coordinates in the certainty grid. Which means, higher the certainty value reveals that the real object is detected by sensor range. In real time, the grid is continuously updated at every instant hence this method is suitable for sparse moving objects. The VFH method subsequently employs a two-stage data-reduction process in order to compute the desired control commands for the vehicle. In the first stage the histogram grid is reduced to a onedimensional polar histogram that is constructed around the robot's momentary location. Each sector in the polar histogram contains a value representing the polar obstacle density in that direction. In the second stage, the algorithm selects the most suitable sector from among all polar histogram sectors with a low polar obstacle density, and the steering of the robot is aligned with that direction. After analysing the polar function and determining where the obstacles are present , the robot makes the decision using a cost function to determine which path it must travel on. Path chosen by the cost function G have two weighing parameters: G = a*(target-direction)+b*(previous-direction)

target-direction = difference between new direction and goal direction

previous-direction = difference between previously selected direction and new direction

Thus using the cost function we determine the path traversed by the robot.

ArUco Marker Detection and Pose Estimation:

An ArUco marker is a synthetic square marker composed by a wide black border and a inner binary matrix which determines its identifier (id). The black border facilitates its fast detection in the image and the binary codification allows its identification and the application of error detection and correction techniques. The marker size determines the size of the internal matrix. For instance a marker size of 4x4 is composed by 16 bits. Some examples of ArUco markers: It must be noted that a marker can be found rotated in the

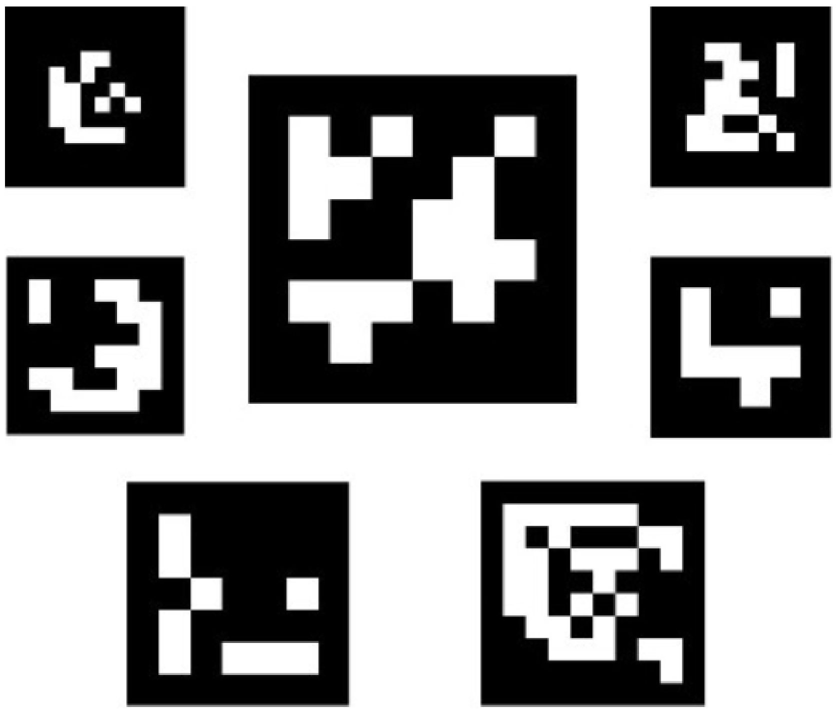


Figure 4.4: Example of markers images

environment, however, the detection process needs to be able to determine its original rotation, so that each corner is identified unequivocally. This is also done based on the binary codification. The marker detection process involves the following steps:

ArUco Marker Detection:

- The first step is to analyze the given image and find square shapes that are candidates of the ArUco marker. Here adaptive thresholding is used to convert the image to a black and white image, and then contours are extracted from the image. In this process, the convex that are not squares are discarded.
- The next step is to check if the detected squares form an ArUco marker. Perspective transformation is applied to the image in order to obtain the canonical form, then the image is divided into 16 cells based on the border size. Each of the cells is decoded as a '1' or '0' depending on whether it is a white or a black square. This results in a 16-bit sequence which can be decoded to obtain the ID of the ArUco marker.

POSE estimation:

- Once the marker is detected, the next step is to determine its pose for that we need to know the calibration parameters of the camera. This is the camera matrix and distortion coefficients.

- **Camera Calibration**:

  Some cameras introduce significant distortion to images. Two major kinds of distortion are radial distortion and tangential distortion. Radial distortion causes straight lines to appear curved. Radial distortion becomes larger the farther points are from the center of the image. Radial distortion can be represented as follows:

$$x_d = X(distorted)$$

$$y_d = Y(distorted)$$

$$x_d = x(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

$$y_d = y(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

  Similarly, tangential distortion occurs because the image-taking lens is not aligned perfectly parallel to the imaging plane. So, some areas in the image may look nearer than expected. The amount of tangential distortion can be represented as below:

$$x_d = X(distorted)$$

$$y_d = Y(distorted)$$

$$x_d = x + [2p_1xy + p_2(r^2 + 2x^2)]$$

$$y_d = y + [p_1(r^2 + 2y^2) + 2p_2xy]$$

  In short, we need to find five parameters, known as distortion coefficients given by:

$$Distortioncoefficients = k_1k_2p_1p_2k_3$$

  In addition to this, we need to some other information, like the intrinsic and extrinsic parameters of the camera. Intrinsic parameters are specific to a camera. They include information like focal length (fx,fy) and optical centers (cx,cy). The focal length and optical centers can be used to create a camera matrix, which can be used to remove distortion due to the lenses of a specific camera. The camera matrix is unique to a specific camera, so once calculated, it can be reused on other images taken by the same camera. It is expressed as a 3x3 matrix: camera matrix=

$$\begin{vmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{vmatrix}$$

For the USB camera used in our project the camera matrix is as shown below:

$$\begin{vmatrix} 3.6 & 0 & 0.014 \\ 0 & 3.6 & 0.014 \\ 0 & 0 & 1 \end{vmatrix}$$

After the calibration we get the camera matrix: a matrix of 3x3 elements with the focal distances and the camera center coordinates (a.k.a intrinsic parameters), and the distortion coefficients: a vector of 5 elements or more that models the distortion produced by the camera.

When we estimate the pose with ArUco markers, we estimate the pose of each marker individually. The camera pose respect to a marker is the 3d transformation from the marker coordinate system to the camera coordinate system. It is specified by a rotation and a translation vector

- The aurco module provides provides a function to estimate the poses of all the detected markers. The pose estimated using the above process depends on the orientation of the marker, hence the real world coordinates (x, y, z) obtained may vary with the orientation of the marker.

The figure 4.5 indicates the transformation of 2D camera matrix to 3D real world coordinates.

This may cause inaccuracy in the measurement and in-turn to be a problem when utilizing the coordinates while picking the object. Hence, it is necessary to apply corrections on the coordinates by utilizing the Euler angles. This is done by rotating the point (x, y, z) in the opposite direction.

Rotation in opposite direction:

Rx = Rotation matrix along X axis
Ry = Rotation matrix along Y axis
Rz = Rotation matrix along Z axis

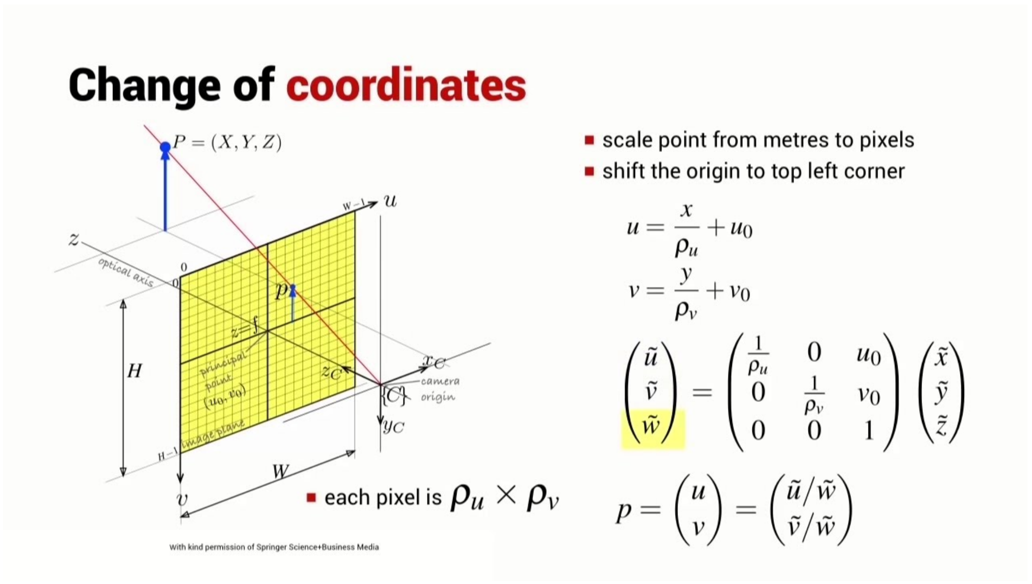

Change of coordinates
$P = (X, Y, Z)$
scale point from metres to pixels
shift the origin to top left corner
$u = \frac{x}{\rho_u} + u_0$
$v = \frac{y}{\rho_v} + v_0$
$\begin{pmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{pmatrix} = \begin{pmatrix} \frac{1}{\rho_u} & 0 & u_0 \\ 0 & \frac{1}{\rho_v} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{pmatrix}$
$p = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \tilde{u}/\tilde{w} \\ \tilde{v}/\tilde{w} \end{pmatrix}$
each pixel is $\rho_u \times \rho_v$
optical axis
principal point (u_0, v_0)
image plane
camera origin
$z = f$
x_C
y_C
z_C
$\{C\}$
H
W
u
v
z
p
With kind permission of Springer Science+Business Media

Figure 4.5: 2D camera matrix to 3D real world matrix

Rx($-\alpha$) =

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & cos(-\alpha) & -sin(-\alpha) \\ 0 & sin(-\alpha) & cos(-\alpha) \end{vmatrix}$$

Ry($-\alpha$) =

$$\begin{vmatrix} cos(-\beta) & 0 & sin(-\beta) \\ 0 & 1 & 0 \\ -sin(-\beta) & 0 & cos(-\beta) \end{vmatrix}$$

Rz($-\gamma$) =

$$\begin{vmatrix} cos(-\gamma) & -sin(-\gamma) & 0 \\ sin(-\gamma) & cos(-\gamma) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Let d be the measured translation matrix

$$d = [x, y, z]$$

Let d' be the corrected translation matrix

$$d' = R_z(-\gamma)R_y(-\beta)R_x(-\alpha) \times d^T$$

Finding Joint angle:

Calculation of joint angles requires an arm model consisting of representation of links and joints. The physical dimensions of Dexter ER-2 for each joint were measured and URDF

(Unified Robot Description Format) format in XML was created. Chains (list of links with initial joint states) from URDF file are fed to IKPy solver in Python. The Homogeneous transnational matrix (4x4 orientation and translation) are fed to the inverse kinematics method to generate arm joint angles.

- **Creating URDF files**
  The physical dimensions such as length, breadth and height of each links and rotation, pitch and yaw details are mentioned in a file with xml format in $.urdf$ extension. A $catkinworkspace$ is created in ROS middleware. A $rospackage$ is developed in it and the URDF file was tested. The figure below gives the output of created URDF file.
- **Finding Inverse kinematics**
  Taking the physical dimension from the urdf file and the 3D coordinates of the objects, -ik_py an inverse kinematics solver library in python calculates the joint angles.

$$\theta_i = \arctan(\frac{y}{x})$$

$$\theta_{3h} = \theta_i + \frac{\pi}{2}\sqrt{\frac{x^2 + y^2}{2}}$$

$$y_p = y - l_3 sin(\theta_{3h})$$

$$c = \frac{x_p^2 + y_p^2 - l_1^2 - l_2^2}{2l_1l_2}$$

$$\alpha = \arccos(-c)$$

$$\theta_2 = \alpha + \pi$$

$$\theta_1 = \arctan(\frac{y_p}{x_p})$$

$$\theta_3 = \theta_{3h} - \theta_1 - \theta_2$$

Where, (x, y) is the position of the end effector θ_i are the joint angles

4.2.6 Raspberry pi to Dexter communication

Using serial communication between Dexter arm and Raspberry pi, the obtained joint angles are fed to each of the joints (servo motors) in order to pick the object. The format of the data transfer is as follows:

# 2 <time(msec) to execute an instruction $>$ # 1 $<$ joint angle of 1st joint $>$ $<$ speed (degrees per second) $>$ # 2 $<$ joint angle of 2nd joint $>$ $<$ speed (degrees per

second)>****

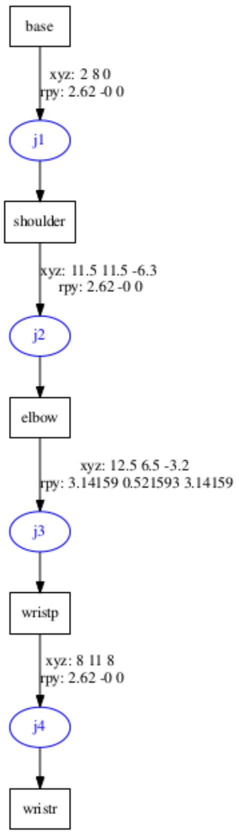


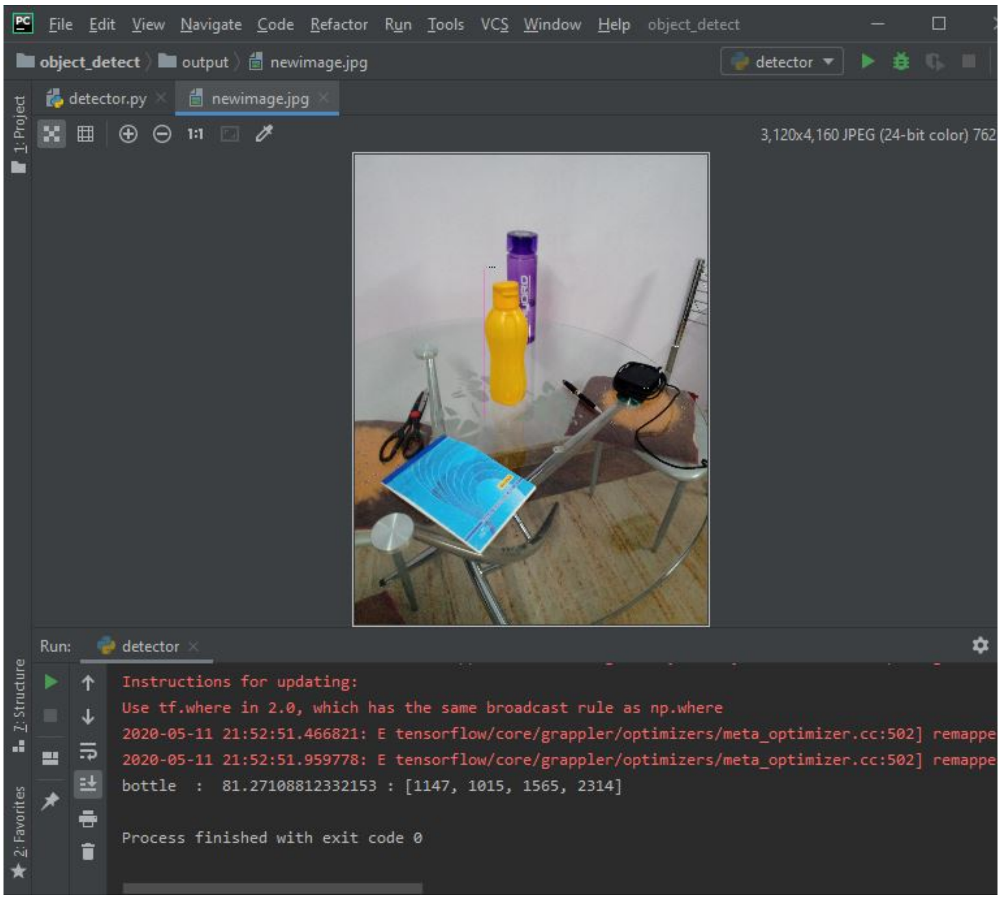
Figure 4.6: Output of URDF file

Chapter 5

Results and Discussion

The following are the snippets of the results obtained.

Figure 5.1 shows the object recognition output which is obtained using ssd by training 10 different objects with 1024 datasets each. The trained model file is fed in the SSD algo-


File Edit View Navigate Code Refactor Run Tools VCS Window Help object_detect
object_detect output newimage.jpg
detector
detector.py
newimage.jpg
3,120x4,160 JPEG (24-bit color) 762
1: Project
Run:
detector
Z: Structure
2: Favorites

```
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
2020-05-11 21:52:51.466821: E tensorflow/core/grappler/optimizers/meta_optimizer.cc:502] remappe
2020-05-11 21:52:51.959778: E tensorflow/core/grappler/optimizers/meta_optimizer.cc:502] remappe
bottle  :  81.27108812332153 : [1147, 1015, 1565, 2314]

Process finished with exit code 0
```

Figure 5.1: Object recognition module

rithm it gives the coordinates of the detected object as shown in the figure. In the master program using the obtained points we found the centroid which gives the 2D translation vector in camera coordinate.i.e.

$$[(1147 - 1015)/2, (2314 - 1565)/2] = [66, 374.5]$$

Figure 5.2 shows the actuation of the robotic arm using the obtained translation vector from Aruco marker pose estimation module and URDF file which is being fed to ikpy solver. The joint angles being obtained is thus fed to the bot serially and resulted successful picking up of the object. The obtained join angles were: 60, 50, 50, 80, 90, 95 degrees.

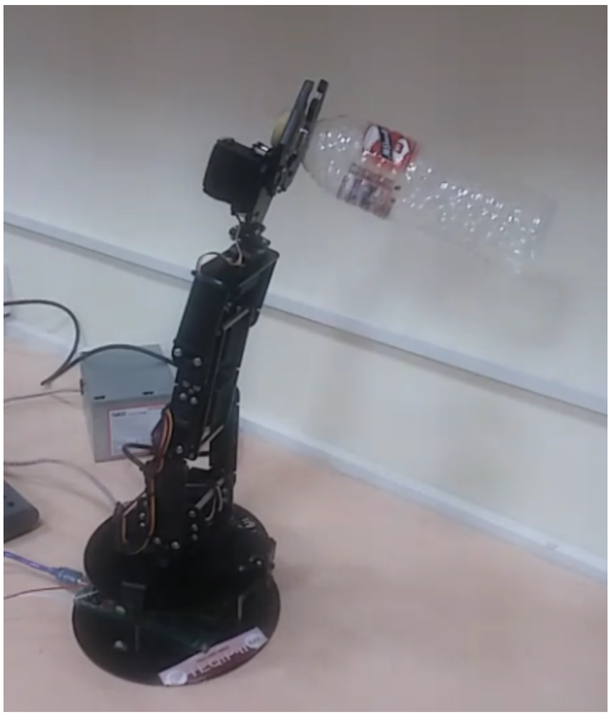
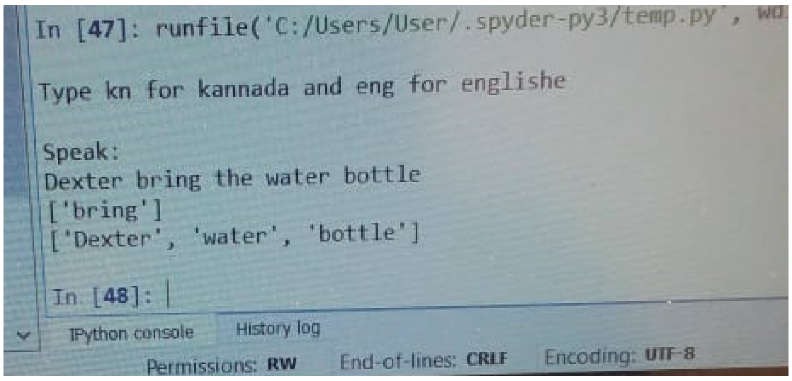


Figure 5.2: Picking action of the arm

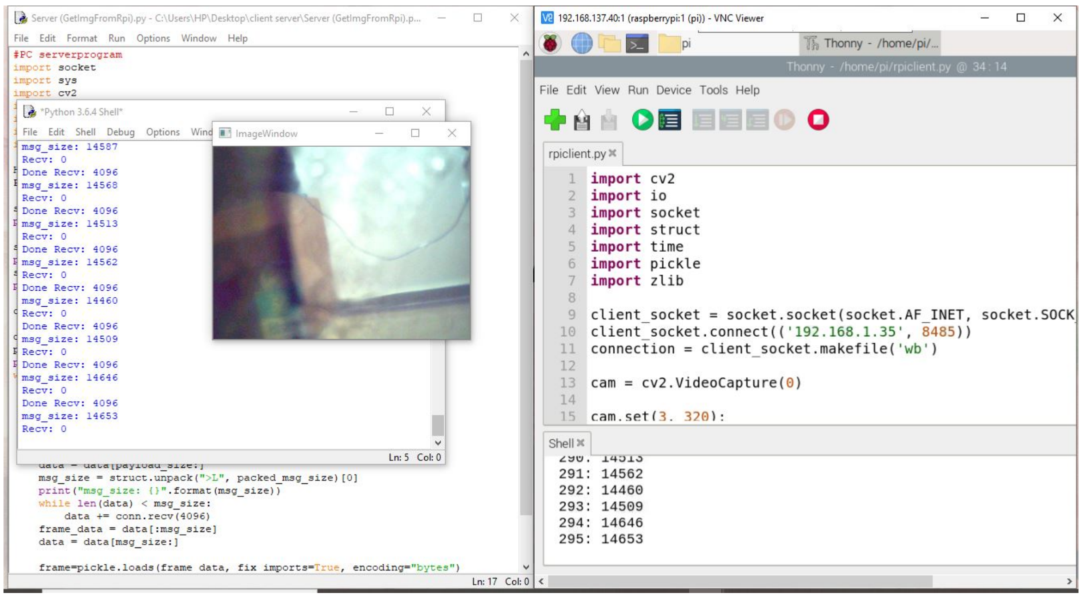
Figure 5.3 shows the output of natural language processing. User can enter Kn for Kannada, by default it takes English commands. The command shall be valid if the first token is Dexter. If the command is valid, then the action verbs are thus extracted.



```
In [47]: runfile('C:/Users/User/.spyder-py3/temp.py', wd

Type kn for kannada and eng for englishe

Speak:
Dexter bring the water bottle
['bring']
['Dexter', 'water', 'bottle']

In [48]: |
```

Figure 5.3: Output of natural language processing

Figure 5.4 shows the image exchange between server (the PC) and client (Raspberry pi) using static IP address.



Server (GetImgFromRpi).py - C:\Users\HP\Desktop\client server\Server (GetImgFromRpi).p...
File Edit Format Run Options Window Help
Python 3.6.4 Shell
File Edit Shell Debug Options Wind
ImageWindow
Ln: 5 Col: 0
Ln: 17 Col: 0

```
#PC serverprogram
import socket
import sys
import cv2
```

```
msg_size: 14587
Recv: 0
Done Recv: 4096
msg_size: 14568
Recv: 0
Done Recv: 4096
msg_size: 14513
Recv: 0
Done Recv: 4096
msg_size: 14562
Recv: 0
Done Recv: 4096
msg_size: 14460
Recv: 0
Done Recv: 4096
msg_size: 14509
Recv: 0
Done Recv: 4096
msg_size: 14646
Recv: 0
Done Recv: 4096
msg_size: 14653
Recv: 0
```

```
    data = data[payload_size:]
    msg_size = struct.unpack(">L", packed_msg_size)[0]
    print("msg_size: {}".format(msg_size))
    while len(data) < msg_size:
        data += conn.recv(4096)
    frame_data = data[:msg_size]
    data = data[msg_size:]

    frame=pickle.loads(frame data, fix imports=True, encoding="bytes")
```

192.168.137.40:1 (raspberrypi:1 (pi)) - VNC Viewer
pi
Thonny - /home/pi/...
Thonny - /home/pi/rpiclient.py @ 34 : 14
File Edit View Run Device Tools Help
rpiclient.py
Shell

```
import cv2
import io
import socket
import struct
import time
import pickle
import zlib

client_socket = socket.socket(socket.AF_INET, socket.SOCK
client_socket.connect(('192.168.1.35', 8485))
connection = client_socket.makefile('wb')

cam = cv2.VideoCapture(0)

cam.set(3, 320);
```

```
290: 14513
291: 14562
292: 14460
293: 14509
294: 14646
295: 14653
```

Figure 5.4: Image exchange between server and client

Chapter 6

Conclusion

6.1 Project Summary

Many specially-abled people can do their chores without relying on others using the designed model. The model can detect the speech signal signals and extract the verbs and noun. The bot shall obey the commands only if the speech is initiated with the noun Dexter. Dexter was chosen because applying it is a unique work which is not used in normal sentences hence it will reduce the turning on and off of the bot unnecessarily. When the speech signal contains action verbs like bring, fetch this indicates the bot to move forward and start capturing pictures using 2D camera. The bot sends the picture file to the server. Server determines existence of the object and gets the coordinates of the image using ssd. Server sends the image coordinates to the bot and bot moves accordingly. the ultrasonic sensor will be integrated with the bot if the distance between bot and the object and the object is found then the bot will stop and actuation of the arm takes place.Now the bot recognises the arucomarker using Aruco-Tracker[20] python API. Using pinhole camera model, 3D coordinates of the 2D image shall be determined. This 3D coordinates shall be used as a transnational vector.Taking the physical dimension from the URDF enum and the 3D coordinates of the objects, -ik_py an inverse kinematics solver library in python calculates the joint angles and this angles are transferred to the Dexter arm to pick the bottle.

This system is very user friendly. The bot understands the native language- Kannada which helps the specially abled people who understand only Kannada.

6.2 Advantages

- Our system can be very helpful for the people with disabilities to walk. It makes them more independent.
- Robotic arm is also beneficial for the elderly which will assist to perform daily chores such as to fetch and deliver objects.
- The bot understands the native language- Kannada which helps the specially abled people who understand only Kannada.
- Since it is not mechanically driven system it doesn't require any strength or effort to operate
- The proposed project can also be implemented in the day to day lives in the shops and stores.The robot can autonomously pick and place the needed object for the customer thus,eliminating the human intervention to an extent.
- The bot can be trusted more than any other human labours

6.3 Drawbacks

- More electric power is used to drive the system to actuate arm as well as wheels.
- It requires good internet connection. DHCP can not be used. So it works with static IP address. i.e. The system works only with mobile data hotspot not with WiFi.
- The process is bit slow.
- Initial cost is very high but at the long run it will be cheaper than getting a labour.
- The use of sonar sensors during path selection to deploy the Virtual Field Histogram it may not avoid an obstacle that comes into the frame immediately and very close.

6.4 Applications

Applications of an autonomous service robot for pick and place are extensive and diverse. They are used from industries to hospitals and can be potentially used from restaurants to battle fields.

- The proposed project will be in great aid for the differently abled.The pick and place feature of the system will come in great help for the needed.
- Robotic arm is also beneficial for the elderly which will assist to perform daily chores such as to fetch and deliver objects.

- Defense It can be used in the field of defence to pick up harmful objects like bombs and diffuse them safely.
- Robotic arm is widely used in industries to pick up the required parts and place it in correct position to complete the machinery fixture. It can be also used to pick and place objects on the conveyor belt as well as pick up defective products from the conveyor belt.
- The proposed project can also be implemented in the day to day lives in the shops and stores.The robot can autonomously pick and place the needed object for the customer thus,eliminating the human intervention to an extent.

6.5 Future Scope

Future works for the proposed system involves:

- Applying localization for the bot which then makes the process faster.
- Applying network security. The proposed system can be hacked hence implementing security is very essential.
- Designing an efficient power source for the bot as the designed system draws significant amount of power.

References

[1] **F**ariha Musharrat Haque, Asif Shahriyar Sushmit, M. A. Rashid Sarkar, *"Design of a Voice Controlled Robotic Gripper Arm using Neural Networks"*, International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS-2017)

[2] **A**rshad Javeed, Varun Ganjigunte Prakash, Dr. Sudarshan Patil Kulkarini, *"Autonomous Service Robot"*, AIR '19 02-06 July 2019 Chennai, India

[3] **H**.M.R.T. Bandara, M.M.S.N. Edirisighe, B.L.P.M. Balasooriya, A.G.B.P Jayasekara, *"Development of an Interactive Service Robot Arm for Object Manipulation"*, Robotic and Control Laboratory Department of Electrical Engineering University of Moratuwa Moratuwa, Sri Lanka, 2017.

[4] **M**r. C. Chandra Mouli 1 , Ms. P. Jyothi 1 , Prof. K. Nagabhushan Raju,"Design and Implementation of Robot Arm Control Using LabVIEW and ARM Controller" ,IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE) e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 6, Issue 5 (Jul. - Aug. 2013), PP 80-84 www.iosrjournals.org

[5] **M**ehdi Mouad, Lounis Adouane, Djamel Khadraoui, Philippe Martinet,*"Mobile Robot Navigation and Obstacles Avoidance based on Planning and RePlanning Algorithm"*, IFAC Symposium on Robot Control International Federation of Automatic Control September 5-7, 2012. Dubrovnik, Croatia.

[6] **A**yrton Oliver,Steven Kang,Burkhard C. Wünsche,Bruce MacDonald,*"Using the Kinect as a Navigation Sensor for Mobile Robotics"*,IVCNZ '12, November 26 - 28 2012, Dunedin, New Zealand.

[7] **K**arthi Balasubramanian, Arunkumar R, Jinu Jayachandran, Vishnu Jayapal, Bibin A Chundatt, Joshua D Freeman,*"Object Recognition and Obstacle Avoidance Robot"*,2009 IEEE Chinese Control and Decision Conference (CCDC 2009).

[8] **Z**. Riaz , A. Pervez, M. Ahmer, J. Iqbal ,"A Fully Autonomous Indoor Mobile Robot using SLAM",2010 IEEE.

[9] **M**ohannad Farag, Abdul Nasir Abd Ghafar, Mohammed Hayyan ALSIBAI,*"Real-Time Robotic Grasping and Localization Using Deep Learning-Based Object Detection Technique"*, Automatic Control and Intelligent Systems (I2CACIS) 2019 IEEE International Conference on, pp. 139-144, 2019.

[10] **S**anju S, Nippun Kumaar A A, Sudarshar TSB, *"Autonomous Robotic Arm with Enhanced Intelligence"*, International Journal of Advanced Computational Engineering and Networking, ISSN: 2320-2106, Volume-2, Issue-9, Sept.-2014

[11] **H**imanshu K. Patel, Prabuddh Verma, Shreya Ranka,*"Design and development of co-ordinate based autonomous robotic arm"*, IEEE International Conference INSPEC Accession Number: 12571843, at Institute of technology, Nirma University, Dec,2011

[12] **P**riyanka Karuppiah, Hem Metalia and Kiran George,*"Automation of a Wheelchair Mounted Robotic Arm using Computer Vision Interface "*, 2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC),17917136, May-2018.

[13] **Z**hong-Qiu Zhao, Member, IEEE, Peng Zheng,Shou-tao Xu, and Xindong Wu, Fellow, IEEE *"Object Detection with Deep Learning: A Review"*,IEEE Conference Ahemedabad,2019

[14] **S**ergio Hernandez-Mendez,Carolina Maldonado-Mendez,Antonio Marin-Hernandez and Homero Vladimir Rios-Figueroa *"Design and Implementation of a Robotic Arm using ROS and MoveIt!"*, IEEE Conference ATU College , New Delhi,Sept-2011

[15] **N**. U. Alka, A. A. Salihu, Y. S. Haruna and I. A. Dalyop *"A Voice Controlled Pick and Place Robotic Arm Vehicle Using Android Application"* Department of Electrical and Electronics Engineering, A. T. B. U. Bauchi Corresponding Author: N. U. Alka

[16] **S**anju S, Nippun Kumar A A,Sudarshan TSB *"Autonomous Robotic arm with enhanced intelligence "* Amrita Vishwa Vidyapeetham-University; Department of Computer Science Engineering, Amrita School of Engineering,

[17] **W**ei Liu1, Dragomir Anguelov2, Dumitru Erhan3, Christian Szegedy,Scott Reed, Cheng-Yang Fu1, Alexander C. Berg,*"SSD: Single Shot MultiBox Detector"*,University of Michigan, Ann-Arbor

[18] N.A. ainuddin, Y.M. Mustafah, Y.A.M. Shawgi, N.K.A.M. Rashid, *"Autonomous Navigation of Mobile Robot Using Kinect Sensor"*, 2014 International Conference on Computer and Communication Engineering.

[19] Benjamin Langmann, Klaus Hartmann and Otmar Loffeld ZESS *"Depth of camera technology comparision and performance"*– Center for Sensor Systems, University of Siegen, Paul-Bonatz-Strasse 9-11, Siegen, Germany

[20] https://ai.googleblog.com/2015/08/the-neural-networks-behind-google-voice.html