JSS MAHAVIDYAPEETHA
Sri Jayachamarajendra College Of Engineering, Mysore-570006
An Autonomous Institute Affiliated to
Visvesvaraya Technological University, Belgaum

"Robot Assistant For Elderly Person/Patient"

Report submitted in partial fulfilment of curriculum prescribed for the Award of the Degree of

Bachelor of Engineering
In
Computer Science And Engineering
By

Name	USN
ASIF.M	4JC10CS019
AKSHAY SHETTY	4JC10CS009
RAGHU D	4JC11CS414
PARASHIVA MURTHY C	4JC11CS408

Under the Guidance of
Smt. Trisiladevi C. Nagavi
Asst. Professor
Dept of CS&E,SJCE Mysore-06.

Sponsored by
TEQIP Phase II

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MAY - 2014

Jss Mahavidyapeetha
Sri Jayachamarajendra College Of Engineering, Mysore-570006
An Autonomous Institute Affiliated to
Visvesvaraya Technological University, Belgaum

Certificate

This is to certify that the work entitled **"Robot Assistant For Elderly Person/Patient"** is a bona fide work carried out by **Asif M, Akshay Shetty, Raghu D** and **Parashiva Murthy C.** in partial fulfilment of the award of the degree of **Bachelor of Engineering in Computer Science and Engineering of SJCE-Mysore** An Autonomous Institute Affiliated to Visvesvaraya Technological University Belgaum during the year 2014. It is certified that all corrections / suggestions indicated during CIE have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the Bachelor of Engineering degree.

Guide:
Smt. Trisiladevi C.Nagavi
Assistant Professor
Dept. of CS & E
SJCE - Mysore

Head of Department:
Dr. C.N.Ravikumar
Professor & H.O.D
Dept. of CS & E
SJCE - Mysore

Date:
Place: Mysore

Examiners:
1. . . . . .
2. . . . . .
3. . . . . .

Declaration

We, the students of 8th semester B.E Computer Science and Engineering, SJCE-Mysore do hereby declare that the project work entitled **"Robot Assistant For Elderly Person/Patient"** is a bona fide work carried out under the guidance of *Smt. Trisiladevi C.Nagavi*, Assistant Professor, Department of Computer Science and Engineering, SJCE-Mysore, submitted in partial fulfilment for the award of Bachelor of Engineering in Computer Science and Engineering of SJCE-Mysore An Autonomous Institute Affiliated to Visvesvaraya Technological University Belgaum during 2013-2014.

Further, we declare that we or any other person have not previously submitted this project report to any other institution/university for any other degree/diploma or any other person.

Asif M
Akshay Shetty
Raghu D
Parashiva Murthy C

Date:
Place: Mysore

Abstract

In the ever-changing world of technology, we are rapidly moving towards Automation of every Human activity. With the help of Robotics and Artificial Intelligence we are not only able to replicate the Human efforts but also provide the ability to perform the same activity for long duration of time and accuracy without being slowed-down by the Human disadvantages like Tiredness, Injury, Faulty decisions and other factors.

Robotics is branch of science and technology which involves interdisciplinary fields such as Electronics, Mechanical and Computer Science Engineering. Robotics is a field which touches every aspect of industries and Human life including Health care. Robots are used in performing critical surgeries and analysing the Human body to suggest professionals to take the some logical decisions depending on the results of Analysis.

The main objective of our project "Robot Assistant for Elderly Person / Patient" is to help the people with motor impairments to pick the everyday objects for their daily use using our simple, inexpensive robot in the absence of caretakers. Eventhough there are many such products in the industry which provide the above mentioned functionalities, what makes our project unique is that we have adopted Centralized Processing i.e., all the complex computations required by the robot is done by high-end processing devices such as laptops or PCs which are commonly available instead of using expensive embedded computation devices on Robot itself.

Acknowledgement

First and foremost, we would like to express our sincere gratitude to **Dr. Syed Shakeeb Ur Rahman**, Principal of Sri Jayachamarajendra College of Engineering for providing an excellent environment for our education and his encouragement throughout our stay in college. We extend our heartfelt regards to **Dr. C.N. Ravikumar**, HOD, Department Of Computer Science and Engineering for his constant support and encouragement.

We extend our deepest gratitude to our guide **Smt. Trisiladevi C.Nagavi**, who has supported us throughout our project with her patience and knowledge whilst allowing us the room to work in our own way. Our heartfelt gratitude to **Smt. Manimala S**, who helped us embark upon this project and whose timely suggestions helped in the betterment of the project.

Our personal special thanks to *Mr. Nagaraj (VIII Sem EC), Mr. Anushruth (VIII Sem EC)* and *Mr. Goutham Vijapur (VI Sem EC)* for sharing their precious time and knowledge about Electronics and Communication.

Finally, we thank our parents and friends for their constant support and encouragement.

Asif M
Akshay Shetty
Raghu D
Parashiva Murthy C

Contents

List of Figures

1 Introduction

Robotics[1] is the branch of technology that deals with the design, construction, operation, and application of robots as well as computer systems for their control, sensory feedback, and information processing. These technologies deal with automated machines that can take the place of humans in dangerous environments or manufacturing processes, or resemble humans in appearance, behavior, and/or cognition.

The type of robots that you will encounter most frequently are robots that do work that is too dangerous, boring, onerous, or just plain nasty. Most of the robots in the world are of this type. They can be found in auto, medical, manufacturing and space industries. In fact, there are over a million of these type of robots working for us today.

Today, robotics is a rapidly growing field, as technological advances continue, research, design, and building new robots serve various practical purposes, whether domestically, commercially, or militarily. Many robots do jobs that are hazardous to people such as defusing bombs, mines and exploring shipwrecks.

Computer vision[1] is the science and technology of machines that see. As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences and views from cameras.

In most practical computer vision applications, the computers are preprogrammed to solve a particular task, but methods based on learning are now becoming increasingly common.

Mechanical grippers One of the most common effectors is the gripper. In its simplest manifestation it consists of just two fingers which can open and close to pick up and let go of a range of small objects. Fingers can for example be made of a chain with a metal wire run through it. Hands that resemble and work more like a human hand include the *Shadow Hand* and the *Robonaut hand*. Friction jaws use all the force of the gripper to hold the object in place using friction.

Mobile Robot[2]**:** A mobile robot is an automatic machine that is capable of movement in any given environment. Mobile robots have the capability to

move around in their environment and are not fixed to one physical location. In contrast, industrial robots usually consist of a jointed arm (multi-linked manipulator) and gripper assembly (or end effector) that is attached to a fixed surface. Domestic robots are consumer products, including entertainment robots and those that perform certain household tasks such as vacuuming or gardening.

Mobile robot navigation There are many types of mobile robot navigation:

- **Manual Remote:** A manually teleoperated robot is totally under control of a driver with a joystick or other control device. The device may be plugged directly into the robot, may be a wireless joystick, or may be an accessory to a wireless computer or other controller.
- **Guarded Tele-op:** A guarded tele-op robot has the ability to sense and avoid obstacles but will otherwise navigate as driven, like a robot under manual tele-op. Few if any mobile robots offer only guarded tele-op.
- **Line-following Car:** Most of these robots operated a simple "keep the line in the center sensor" algorithm. They could not circumnavigate obstacles; they just stopped and waited when something blocked their path.
- **Autonomously Guided Robot:** An autonomously guided robot knows at least some information about where it is and how to reach various goals and or way-points along the way. "Localization" or knowledge of its current location, is calculated by one or more means, using sensors such motor encoders, vision.

Personal & **Domestic Robots:** Personal & domestic robots will help you with your household chores. They include robot vacuums, robot pet care, pool cleaners, lawn mowers and more.

Robotic vacuum cleaners and floor-washing robots that clean floors with sweeping and wet mopping functions. Some use Swiffer or other disposable cleaning cloths to dry-sweep, or reusable microfibre cloths to wet-mop.

Cat litter robots are automatic self-cleaning litter boxes that filter clumps out into a built-in waste receptacle that can be lined with an ordinary plastic bag.

Security robots which have a night-vision-capable wide-angle camera that detects movements and intruders. It can patrol places and shoot video of suspicious activities, too, and send alerts via email or text message; the stored history of past alerts and videos are accessible via the Web. The robot can also be configured to go into action at any time of the day.

The following are some of the Domestic and Industrial Robots that are available in the Robotic industry.

Figure 1: First generation Roomba vacuums the carpets in a domestic environment[3]

Figure 2: Robotic mapping can be used for a serving robot guide[3]

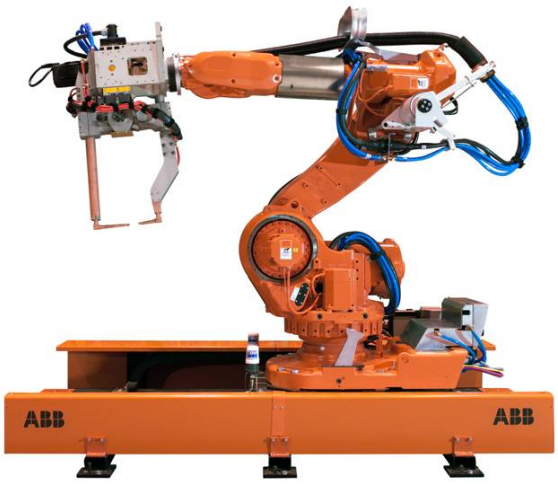

Figure 3: Industrial Spot Welding Robot[4]

Figure 4: Articulated industrial robot operating in a foundry[5]

Android Application Android is basically an operating system for smartphones. But we find now integrated into PDAs, touch pads or televisions, even cars (trip computer) or netbooks. The OS was created by the start-up of the same name, which is owned by Google since 2005.

- **Specifications:** This operating system is based on version 2.6 of Linux, so it has a monolithic system kernel, what means that all system functions and drivers are grouped into one block of code.

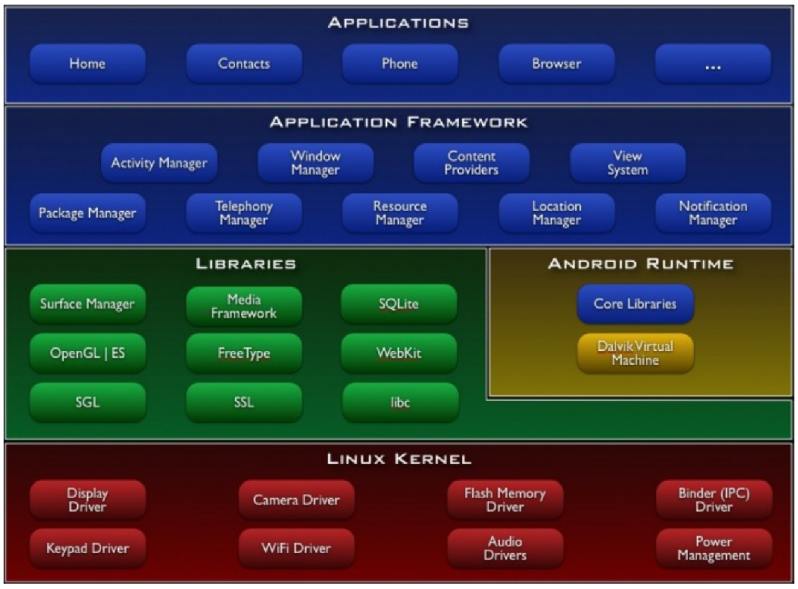

Figure 5: Architecture of Android

- **Architecture:**[10] Android consists of five layers:
  - The Linux kernel 2.6-which includes useful drivers that allow for example WiFi or Bluetooth.
  - The library written in C and C + + that provide higher level functionality such as an HTML engine, or a database(SQLite).
  - A runtime environment for applications based on a virtual machine, made for inefficient machines such as telephones. The aim is to translate JAVA in machine language understood by Android.
  - A JAVA framework that allows applications running on the virtual machine to organize and cooperate.
  - The user applications written in Java (Web browser, contact manager etc.)

- **Current Version:** Today android is in its 5th version, Android 2.1. Each version is designed to gradually correct the lack of APIs, to enhance the user interface and add functionality. The latest version adds such things as support in HTML5 in the browser, it allows multitouch

or it brings new Contact API, which defines a database for contact management.

- **Applications:**
  - **Google Applications:** Android includes most of the time many Google applications like Gmail, YouTube or Maps. These applications are delivered with the machine most of the time, except in certain cases, such as some phones running android on which the provider has replaced Google applications by its own applications.
  - **Widgets:** With android, it is possible to use widgets which are small tools that can most often get information. These widgets are directly visible on the main window.
  - **Android Market:** This is an online software store to buy applications. Developers who created applications can add them into the store, and these applications can be downloaded by users, they can be both free and paid.
  - **Multitasking:** Android allows multitasking in the sense that multiple applications can run simultaneously. With Task Manager it is possible view all running tasks and to switch from one to another easily.
- **SDK:**[11] A development kit has been put at disposal of everybody. Accordingly, any developer can create their own applications, or change the android platform. This kit contains a set of libraries, powerful tools for debugging and development, a phone emulator, thorough documentation, FAQs and tutorials.
- **Modifiability:** This allows everyone to use, improve or transform the functions of Android for example transform the interface in function of uses, to transform the platform in a real system embedded Linux.

People with motor impairments, Motor impairment[6] is the partial or total loss of function of a body part, usually a limb or limbs. This may result in muscle weakness, poor stamina, lack of muscle control, or total paralysis. Motor impairment is often evident in neurological conditions such a cerebral palsy, Parkinsons disease, stroke and multiple sclerosis.

An extreme form of motor impairment is locked-in syndrome, in which voluntary control of almost all muscles is lost, sometimes including the eyes, in an individual who retains cognitive function. The syndrome is caused by

damage to portions of the lower brain and brain-stem, from a stroke or other insult.

Assistive technologies [7] for people with motor impairments are,

- Mouth stick.
- Head wand.
- Single-switch access.
- Sip and puff switch.
- Oversized trackball mouse.
- Adaptive keyboard.
- Eye tracking.
- Voice recognition software.

Barriers to Access these technologies are,

- 40% of people with a motor impairment have difficulty using their hands.
- Users may not be able to use the mouse.
- Users may not be able to control the mouse or keyboard well.
- Users may be using voice-activated software.
- Users may become easily fatigued

Robot Assistant for Elderly Person or patient project concentrate on these type of people and also elder people with motor impairment due to lack of limb stamina. Users of this project are those people who are bed rested or motor impairment patient, these people needed to be nursed or taken care continuously and there should be someone who will atleast help these people to get some needy objects, This project is trying to adopt a Robot, which is able to understand the user requirement, recognize objects and pick the object for user, as a care taker for these people.

1.1 Objective

The main objective of our project **"Robot Assistant for Elderly Person / Patient"** is to help the people with motor impairments to pick the everyday objects for their daily use using our Simple, Inexpensive Robot. The general objective can be broken down to four more specific objectives that would together achieve the overall goal of the project as follows:

1. To build a **Android Mobile Application**. This will be available to our user, to request the fetching of a particular object present in his/her room.

2. To study and develop a generic, automated and intelligent system which can process the user request command, initiate the required hardware and their functionalities and **detect the requested Object** from a current image of the room using the *Surface feature detection* algorithm.

3. To study and develop a smart system which can correlate the real time dimension of the room and represent the real time dimension into **"Digital Mapping System"**.

4. To build an **Inexpensive, Simple, Wireless Robot** which can receive the routing Instruction from the **Server System**s mentioned above. Robot should be able to **follow the path** specified by system and pick the **object requested** and then give the same to the requested user.

1.2 Existing Solution

Domestic Robot Assistant for the household:[8] The Robotics Research Institute at the University of Tokyo (Tokyo Universitys IRT) has developed a robot that can assist you in your daily domestic tasks.

The prototype measures 5 ft in height, weighs 287 lbs and has 32 degrees of freedom.

Currently, the robot can clean the floor, clear the table, collect things to put them in the garbage or even put them back where they go, collect your dirty laundry and put it in the washing machine.

It is equipped with 6 wheels, a battery that lasts up to 1 hour, 5 cameras on its head, a laser and ultrasonic sensors to detect obstacles.

The prototype is equipped with SLAM (simultaneous localization and mapping).

Figure 6: Existing Assistant Robot

Personal Robotic Arm: [9] In Cornell's Personal Robotics Lab, *Ashutosh Saxena*, assistant professor of computer science, is developing the technology to make such devices possible. They might appear first as assistants for the disabled and elderly and could cost less than $20,000 perhaps even as little as a few thousand dollars.

A household robot will have to adjust to constantly changing conditions: It will have to find the dishes on the table and find empty spaces in the dishwasher rack. It should be aware of where you are and what you're doing, so it can help if needed and not interrupt when you don't want it to.

The underlying technology is what computer scientists call "machine learning", in which a computer program takes note of events and in effect reprograms itself in response. Machine learning often works on the principle of "What I tell you many times is true, but not exactly". Show a computer

a lot of different cups and tell it that each one is a cup, and with the right programming, it will find the things that all the cups have in common and use that to identify cups in the future. Since sizes and shapes will never be exactly the same, the computer calculates the probability that a new object fits each of the models in its memory and chooses the one that scores highest. A similar process teaches the robot to find a cup's handle and grasp it correctly. This is not unlike what humans do in the first few months of life.

Placing objects is harder than picking them up, because there are many options. A cup should be right side up on a table but upside down in the dishwasher. A plate can lie flat on a table or slide vertically into a dish rack slot. So robots are programmed with different procedures for each type of object.

Figure 7: A robotic arm hands Ashutosh Saxena a cup

1.3 Proposed Solution

Our project **"Robot Assistant for Elderly Person / Patient"** contains, A Server system which processes the Tap button command given by a patient or a person and recognize the object they are requesting using a currently captured image of his/her room, that will be scanned for that Requested object. Its location is sent to a robot which follow the path given by the Server System and robot will pick the same object and give it to the patient.

- There are few Robot Which does similar operations but they have built in Processors, Camera, Microphone, Sensors and other required hardware installed in them and they cost in kilos of dollars but
- Our project is Remote-Server based system where all the complex and high processing operations are done by a laptop or Desktop and as a output set of routing data is sent to the Robot.
- Since most of them own a laptop or desktop the cost of Resources reduced and the Robot we are using need not to have higher Processor or RAM, all it just need few micro-controller which take the data from receiver and move the wheels towards object and pick the object and give it to user.
- The above idea and techniques reduces the cost of the Robot and can make available to the public easily.

1.3.1 Front-end Operations:

- A wireless User Interface installed near the patients bed or resting place which triggers a request to a needed object and sends that request to Server System.
- A wired/wireless Camera installed in his room, which is fixed in a particular location and helps the Server System to locate the objects.
- A Basic Modelled Robot with RF (Radio Frequency) receiver, which just accept the routing information from the Server System via RF transmitter. Information contains the route from robot to particular object and from there to patients location.
- Robot should be capable of holding the objects.

1.3.2 Back-end Operations:

- Server System can be any computational device which has the capable processor to process image and communication media to send information.
- Server can be a Laptop, Desktop with RF Transmitter.
- Command received from the Keypad is processed by Server System which recognizes the object requested by the user.
- System should enable the Camera and capture clear view image of the Room and process that image for the presence of that object.
- System should prepare the digital map of that environment and if the object is present, it should prepare a feasible route from robot to objects location and from there to patient.
- Server System will send the routing direction to the robot which follow the route and pick the object and give it to the user.

Overall operation of the project involve the proper synchronization of the elements of both back-end and front-end.

Problem domains are Image processing, Intelligent Digital Mapping and Robotics.

1.4 Time Schedule for Completion of the Project

Overall project activities are split into following phases:

No	Phases	Start Date	Expected Finish Date	Total No.of Days
1	Requirement Analysis	15 Sept 2013	30 Oct 2013	45
2	Planning Phase	01 Oct 2013	1 Jan 2014	92
3	Design Phase	01 Jan 2014	15 Feb 2014	45
4	Implementation	01 Feb 2014	30 March 2014	58
5	Testing	01 March 2014	01 May 2014	60
6	Presentation & Report	01 May 2014	31 May 2014	30

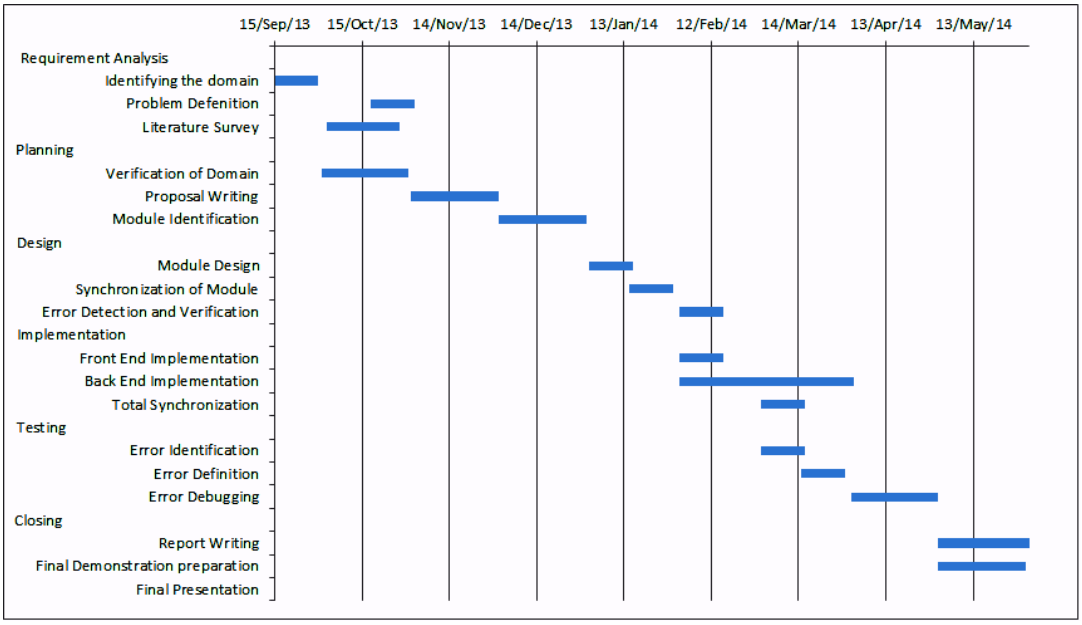

Figure 8: Gantt Chart of The Project

Some of the above mentioned project schedule are carried out in parallel manner. Design Phase, Implementation Phase and Testing Phase are carried out simultaneously to ensure the proper synchronization of the system.

1.5 Scope of the Project

1. This Project is meant to help the people with motor impairment in the absence of their caretaker.

2. Daily use objects are picked by Robot for User.

3. This project can be adopted in Hospitals, in house or in Elder Orphans Home to help weak people.

4. This Robot can also be used in bomb diffusing operations.

5. Robot can be adopted in restaurant for cleaning purpose on plain table.

6. Robot build in this project is a simple and inexpensive model and can be affordable to wide range of people.

7. Centralised processing help this Robot to update its software processing operations at any time and with no or less hardware change.

2 System Requirements and Analysis

2.1 Functional Requirements

The functional requirement of Complete System are divided into seven different phases. These are,

1. **Calibration Phase:**
    - Proper Calibration is done before the System is run online.
    - Calibration should be done only once when during the installation of the System.
    - Mention the effects in system calibrated operation due to the changes in the calibrated environment.
    - Results of the calibrations should be stored safely and made available for the reuse or backup operations.
    - Proper Guidelines and Manual should be given to the new operators for calibrating the system.
    - Mention if Failure in the Device or replacement of device need calibration or not.
2. **User Interaction:**
    - User Interaction with the system is very important phase so the Interface should to defined accurately.
    - Interface should not lead user into any kind of confusion in any sort of operation in the system.
    - Interface device should be portable.
    - UI is made available cheaply.
    - Ease of use and flexibility should be taken care.
3. **Request Processing:**
    - This phase should successfully accept the user command from the User interaction phase.
    - Queue system should be maintained for every sequence of requests.
    - Redundant requests should be warned.
    - Process the request so that unique object reference is made.

- Result of this phase should be unique and stored in the system data set.

4. **Camera Initialization** & **Image Capturing :**

   - This phase should activated as soon as the request command is received at the server system.
   - Should successfully initialize the Web camera in the room.
   - Capture and store the Image of the room environment.
   - Successful status of completion of these processes should be returned.
   - In case of any failure in the Hardware or in hardware synchronization system should report the error and message the caution to the user or operator.
   - For education or understanding purpose, every intermediate actions and operation should be displayed on screen.

5. **Image Processing:**

   - In this phase, system takes the input that indicate which object to detect.
   - Proper set of Object templates should be maintained in the system data set.
   - For a given template, system should be able to detect the object.
   - Object box should be created and the pixel coordinates should be sent to the result data set as soon as Object is Detected.
   - Any failure or exceptions in the modules should be reported and warned.
   - Proper error handling methods need to be suggested.
   - Status of the Module success should be sent and maintained.

6. **Route Calculation:**

   - Once the pixel coordinates of the detected object are available, this phase should calculate the width of the object, distance of the object from the robot and other route related distance measurements.
   - Measurements should be in some particular standard of measurement. preferably in centimetre "cm".

- Direction of the route should be mentioned.
- Direction can either in NEWS(Noth East West South) or FRLB(Forward Right Left Backward) format.
- Error Handling and Error Reporting
- Generate the Route file.
- Route file name should be specific and made available for the random access.
- Route file should contain directions and distances from robot to object and object to user.
- Return the success status of the Module to the main system.

7. **Robot Routing:**

- This Phase should access the route file generated in the previous phase and send the byte informations to the serial communication port.
- Successful write on serial communication port should return the acknowledgement.
- Information from the COM Port should be read by the USB to UART circuit board and information should be digitalized with the help of Micro-controller.
- Micro-controller should pass the digits bit to the RF Encoder.
- RF encoder should encode the digital information into analog frequencies and transmit to the receiver.
- At Robot the transmitted information should be received and decoded back to digits and according to set of binary data code the robot movements should be done.
- Robot Should follow the routing path transmitted and once reach the object point it should pick the object using its Robot arm.
- Once the object is picked robot should follow the return path mentioned in the route file and should drop the object near the user safely.
- Any Error and exceptions should be displayed on the server screen.
- Once the Robot Operation is Done, robot should return to its original calibrated place and rest.
- Even if some error or exception occur system should report the error and log the informations.

2.2 Non-functional Requirements

- System Should be facilitated with a well defined User Interface.
- Ease of use should be flexible.
- Overall System should be completely autonomous. i.e., System should be operated automatically.
- Required calibration has to be done before running the System online.
- Presentation should be good and easy to understand.
- Proper Error and Warning Messages should be displayed in exception cases.
- Any Missing of Hardware or failure of Hardware Devices should be mentioned before and while running the System online.
- Limitations of the overall System should be mentioned so that there is conflict in the user command or operations requested.
- Program should be written with the comments explaining the function properly.
- Documentation is done using IEEE standard with 12pt font size,oneside paper style and A4 Paper size.

2.3 Hardware and Software Requirements

Our project contain different sets of hardware requirements. One a Server System, which is a Computational Device and Software operated and should also have some additional hardware attached features for the communication purpose. Second a Robot, which is a embedded control system and should contain some additional supported hardware devices.

2.3.1 Hardware Requirements

- At **the Server-end System,** the hardware requirement are as follows:
  - **Processor:** Intel Pentium based System or higher processor.
  - **Processor speed:** 250MHz to 3.0GHz.
  - **RAM:** 512MB to 8GB.
  - **USB to UART Converter** Circuit.

  - **RF Transmitter device:** 4bit transmission at 435Mhz.
  - **RF Frequency encoder:** To encode the digital bits into analog frequencies.
  - **ATMEGA Controller:** To operate and Synchronize Transmitter.

- **Robot Specification:**
  - **ATMEGA Micro-controller Developer Board:** To receive the data from RF receiver and Control the movements of the Robot wheels and Robotic Arm.
  - **RF Receiver:** To receive the RF transmitter frequency at the same frequency of 435Mhz.
  - **RF Decoder**: To decode the RF received frequency into digital information for the output.
  - **DC Motor Driver Board:** H-Bridge Circuit to control the Robot Wheels movement and Manages the Voltage.
  - **Mechanical Robot Arm:** To Pick the requested Object.
  - **Servo Motor Controller:** To Control the Servo Motor Attached to the Robot Arm.

2.3.2 Software Requirements

- **Operating System:** Windows, Linux and Mac OS.
- **Matlab:** Matlab R2013b and above are supported.
- **HEX File Converting Software:** Compilers which can convert C/C++ embedded program into .HEX files.
- **ATMEGA Flash Burner:** To flash and Burn the .hex code into Controller.
- **USB to UART port Driver:** Additional software which identify the USB to UART converter circuit and communicate with the same.
- **Eclipse - Android:** To write, execute, simulate and debug the Android Application.
- **Eclipse - C embedded:** To write, simulate and debug the embedded programming for the controller.

- **AVR Studio:** To compile and Build the Microcontroller program (ATMEGA8A)
- **Serial Communication Software:** Any Software or program that can transmit the informations to serial communication port of the System such as COMP Ports(USB Ports).
- **Win-Avr 2010+:** To support embedded C/C++ header files and libraries.
- **Keil - μvision:** To compile and Build embedded program for 8051+ microcontroller.
- **PDF Reader:** To view the report.
- **LaTeX:** To prepare the documentation.
- **SmartDraw:** Tool for designing activity diagram, flow diagram and other Software engineering related diagrams.

3 Tools and Technology Used

The following are the list of tools used in this project.

3.1 MATLAB

MATLAB[12] is a multi-paradigm numerical computing environment and language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces and interfacing with programs written in other programs written in other languages including C, C++, Java and Fortan.

An additional package Simulink, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems.

3.1.1 Computer Vision:

Matlab provides Computer Vision Toolbox which provides algorithms, functions, and applications for the design and simulation of computer vision and video processing systems. You can perform object detection and tracking, feature detection and extraction, feature detection and extraction, feature matching, stereo vision, camera calibration, and motion detection

tasks. The systems toolbox also provides tools for video processing, including video file I/O, video display, object annotation drawing graphics and compositing. Algorithms are available as MATLAB functions, System objects and Simulink blocks.

Feature Detection, Extraction and Matching

Computer Vision System Toolbox provides a suite of feature detectors and descriptors. Additionally the system toolbox provides the functionality to match two sets of feature vectors and visualize the results. When combined into a single window, feature detections, extraction and matching can be used to solve many computer vision design challenges such as image registration, stereo vision, object detection and tracking.

A feature is an interesting part of an image such as a corner, blob, edge or line. Feature extraction enables you to derive a set of feature vectors also called descriptors from a set of detected features. Computer Vision System Toolbox offers capabilities for feature detection and extraction that includes BRISK, MSER and SURF detection for blobs and regions.

Feature Matching is the comparison of two sets of feature descriptors obtained from different images to provide point correspondences between images. Computer Vision offers functionality for feature matching includes configurable matching metrics including SAD, SSD and normalized cross-correlation, Hamming distance for binary features etc.

Object Detection and Recognition

Object detection and recognition are used to locate, identity and categorize objects in images and video. Computer Vision System Toolbox provides a comprehensive suite algorithms and tools for object detection and recognition.

Object classification allows you to detect or recognize an object in an image by training an object classifiers based on training data from different object classes .The classifier accepts image data and assigns the appropriate object or class label.

Feature points are used for object detection by detecting a set of features image,extracting feature descriptors and matching features between the

reference image and an input. This method of object detection can detect reference objects despite scale and orientation changes and is robust to partial occlusions.

Training is the process of creating an object detector or classifier to detect or recognize a specific object of interest.The training process utilizes positive images of the object of interest at different scales and orientations, negative images of backgrounds typically associated with the object of interest and non objects similar appearance to the object of interest.

Camera Calibration

Camera calibration is the estimation of a cameras intrinsic, extrinsic and lens-distortion parameters. Typical uses of a calibrated camera are correction of optical distortion artefacts, estimating distance of an object from a camera, measuring the size of objects in an image, and constructing 3D views for augmented reality systems. Computer Vision System provides an application and functions to perform all the essential tasks in the camera calibration work-flow such as fully automatic detection and location of checkerboard calibration pattern including corner detection with sub-pixel accuracy, estimation of all intrinsic and extrinsic parameters including axis skew, and calculation of radial and tangential lens distortion coefficients.

3.2 Eclipse for Android Application Development

3.2.1 Android Developer Tools

ADT (Android Developer Tools) is a plug-in for Eclipse that provides a suite of tools that are integrated with the Eclipse IDE. It offers too many features that help us to develop Android applications quickly. ADT provides GUI access to many of the command line SDK tools as well as a UI design tool for rapid prototyping, designing, and building of our application's user interface.

3.2.2 ADT Version

ADT 22.3.0 Dependencies:

- Java 1.6 or higher is required.
- Eclipse Helios (Version 3.6.2) or higher is required.

- This version of ADT is designed for use with SDK Tools r22.3. If you haven't already installed SDK Tools r22.3 into your SDK, use the Android SDK Manager to do so.
- Added support for Android 4.4 (API level 19).
- Fixed problem with parsing view hierarchies containing classes in the java.* name space.
- Fixed problem importing Android projects that have the same name as an existing project.

3.2.3 Specifications to Run the Apps In Mobile Device

- **OS:** Android OS, v4.0 (Ice Cream Sandwich) and above.
- **Screen Resolution:** 480x800 pixels and above.
- **Processor:** 1GHz and above.
- **Bluetooth Support:** Yes, v 3.00 and above.

3.2.4 Testing Tools

Android Emulator Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in the Java programming language using the Android Software Development Kit. An emulator is an application that imitates a real device. Emulators enable developers to try out their applications on a virtual device without needing the actual device. For example, Android has an emulator that imitates mobile devices. The emulator can be used to represent a specific mobile phone to test Android applications on. Android Support Library, revision 19.0.1

To Download the SDK Tool: http://developer.android.com/sdk/index.html#download

3.3 Eclipse for Embedded Programming

We are using the Eclipse IDE for creating embedded programs required to run the robot. Eclipse allows us to select Atmega16 as microcontroller option and C/C++ build option to develop and run embedded programs. The embedded program is built and run using C/C++.However Atmega

processor requires the executable to be in the form of a hex file. Eclipse provides us with an option of converting the binary executable file to hex file which is then uploaded to your micro-controller unit.

List of additional software required for Eclipse embedded C programming and flashing purposes are mentioned below

- *WinAVR-20100010-install :* Provide the library supported needed for Eclipse for supporting embedded programming like conversion from binary executable to hex files.
- *JRE-7u9-windows-i586:* Provides Java runtime environment to run Java functionalities.
- *USBASP driver:* Required to provide Serial Communication.

3.4 AVR Studio Compiler

Atmel$^{(R)}$ Studio [13] is the integrated development platform (IDP) for developing and debugging Atmel ARM$^{(R)}$ Cortex-M processor-based and Atmel AVR$^{(R)}$ microcontroller (MCU) applications. The Atmel Studio 6 IDP gives you a seamless and easy-to-use environment to write, build and debug your applications written in C/C++ or assembly code. Atmel Studio 6 supports all 8- and 32-bit AVR MCUs, the new SoC wireless family, and SAM3, SAM4 and SAM D20 MCUs. It also connects seamlessly to Atmel debuggers and development kits.

Key Features:

- Integrated C/C++ compiler
  - Provides one seamless environment with everything the designer needs to write, build, program and debug C/C++ and assembly code.
  - Customers can choose the coding style that best fits their project and generates the most optimal code for their designs.
- Advanced debugging features
  - Support for complex data breakpoints
  - Nonintrusive trace support for SAM3 and SAM4 family of devices, including:

    - Statistical code profiling
    - Interrupt trace / monitoring
    - Data trace

- Integrated editor with visual assist: Write code faster with visual assist code completion tools.
- In-system programming and debugging: Provides a seamless interface to all Atmel in-circuit programmers and debuggers.
- Full debug views: Creates a transparent view into CPU and peripherals, enabling easy code development and debugging.
- Full chip simulation: Delivers an accurate model of CPU, interrupts and peripherals.

List of additional software required for AVR Studio and for Flash Boot loader are mentioned below

- *WinAVR-20100010-install :* Provide the library supported needed for supporting embedded programming like conversion from binary executable to hex files.
- *USBASP driver:* Required to provide Serial Communication.

3.5 Other Software Tools:

3.5.1 WinAVR-20100010

WinAVR$^{(TM)}$ is a suite of executable, open source software development tools for the Atmel AVR series of RISC microprocessors hosted on the Windows platform. Includes the GNU GCC compiler for C and C++.

3.5.2 LaTex

LaTex is a document preparation system and document markup language. It is widely used for the communication and publication of scientific documents in many fields, including mathematics, physics, and computer science. It also has a prominent role in the preparation and publication of books and articles that contain complex multilingual materials. LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language. LaTeX is not the name of a particular editing program, but refers to the encoding or tagging conventions that are used in LaTeX documents.

4 Literature Survey

In his paper on ***"Assistive Robotics"***[14] , David Jaffe defines robot as "A robot is a mechanical or virtual intelligent agent that can perform tasks Automatically or with guidance, typically by remote control. In practice a robot is usually an electro-mechanical machine that is guided by computer and electronic programming. Robots can be autonomous, semi-autonomous, or remotely controlled. Robots range from humanoids such as ASIMO and TOPIO to nano robots, swarm robots, industrial robots, military robots, mobile, and servicing robots. By mimicking a lifelike appearance or automating movements, a robot may convey a sense that it has intent or agency of its own."

Specifically the VDL definition of an Assistive robotics states as "An Assistive robot performs a physical task for the wellbeing of a person with a disability. The task is embedded in the context of normal human activities of daily living (ADLs) and would otherwise have to be performed by an attendant. The person with the disability controls the functioning of the robot". Our project basically deals with creating an assistive robot that is autonomous and does the fetch and carry operations.

In another paper "EI-E: An Assistive Robot that fetches Objects from flat surfaces", the authors talk about an assistive robot that is explicitly designed to take advantage of the human environment where objects are found in flat surfaces that are orthogonal to gravity such as floors, tables and shelves in order to retrieve unmodeled, everyday objects for people with motor impairments. In their platform El-E(Elevated-Engagement), a robot is designed around two key innovations.First El-E is equipped with a laser pointer interface that detects when a user illuminates a location with a green laser pointer and estimates the 3D location selected by this "point and click". This enables a user to unambiguously communicate a 3D location to the robot with modest effort which provides a direct way to tell the robot which object to manipulate or where to go .Second El-E is able to translate its manipulator and associated sensors to different heights which enables it to grasp objects on variety of surfaces such as the floor and table-slab, using the same perception and manipulation strategies.

The motivation of autonomous robots with manipulation capabilities offer the potential to dramatically improve the quality of life for people with motor impairments. Moreover the elderly population worldwide is increasing

substantially as a percentage of overall population. This aging population creates a real need for affordable, robust robotic assistance as elder persons are have been shown to require assistance in activities in everyday living.

Currently this assistance is most often provided by a human caregiver, such as spouse or nurse, which reduces privacy and independence and often places a heavy burden on a loved one or entails high costs. Highly trained animals such as service dogs or helper monkeys can also provide physical assistance but they come with a host of other complications including high costs and training.

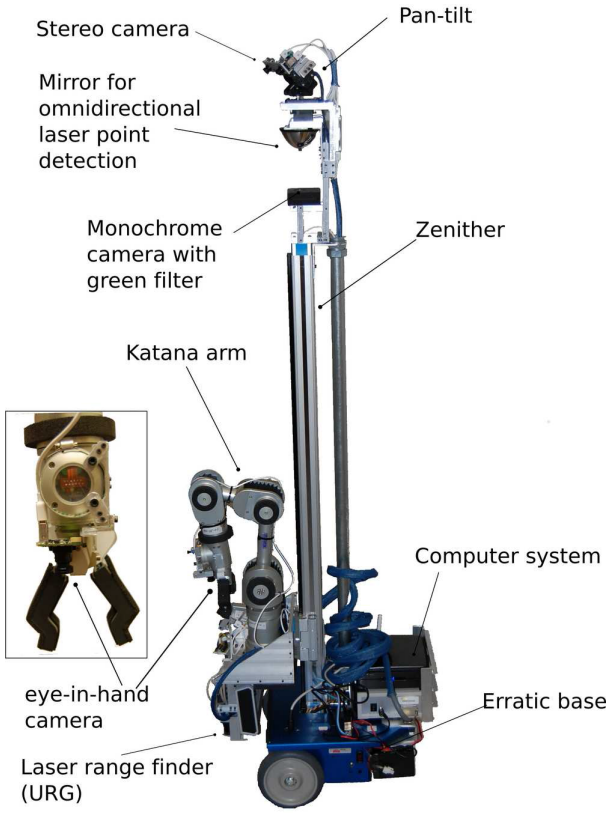

Figure 9: Image of entire mobile manipulator with integrated embedded system

The robot is primarily constructed from the off-the-shelf components as shown in figure 1.Its mobile base is an ER-RATIC platform from Videre De-

sign which includes on board computer with a Core Duo processor and 1 GB of memory with which the robot performs all its computations. The computer runs Ubuntu GNU/Linux and we have written most of our software with Python and occasionally C++.The robot has an eye-in-hand camera with a green filter and a catadioptric mirror mounted on top of the zenither. The robots weight was intentionally concentrated in its base to assure stability even in the case where the arm is stretched out.

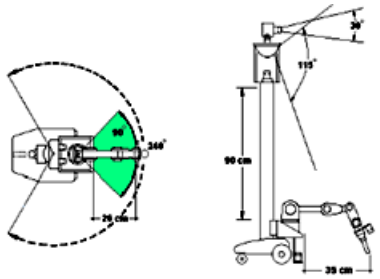

Figure : Left, the manipulator workspace shown in green along with the catadioptric camera's field of view. Right, vertical field of view of the stereo and catadioptric camera.

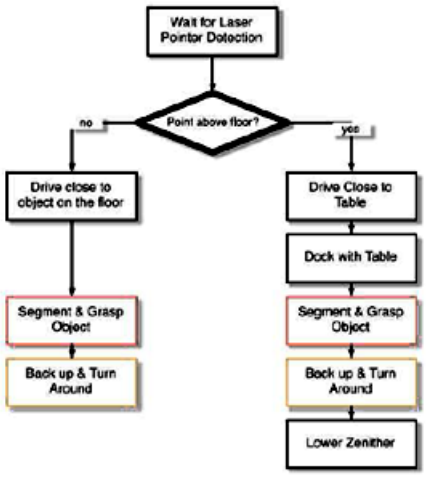

Figure : Object Retrieval Process

Figure 10: Object Retrieval Process and Manipulator Workspace

The steps involve are

1. 3D Estimation for the laser Pointer Interface
   (a) Detect the laser spot using the omnidirectional camera.
   (b) Look at the laser spot using the stereo pair.
   (c) Estimate the spots 3D location by detecting corresponding points in the stereo pair.

2. Moving the Object:

   For navigation, we use a two step reactive control technique used traditionally for controlling mobile robot bases. Use a combination of repulsive potential fields induced by obstacles perceived by the laser range finder and an attractive potential field centred at the laser point to get close to the location indicated by the user. After a point is detected, the robot enters the two step process.

3. Surface Detection

   The system starts out by using the Zenither to move the URG laser scanner from a height. A 3D scan is obtained by registering the 2D URG scans with corresponding zenither height. After obtaining the 3D scan, we calculate a function whose maximum should give us the height of the planer surface. In the first scan the robot calculates an initial estimate of surface height,h1.In the second scan, the zenither is move slowly over the interval [h1-5cm,h1+5cm] to calculate a more accurate estimate h2.

4. Finding Flat Surfaces

   After having detected the top edge of the surface supporting the object indicated by the user. Our robot drives forward with its laser range set exactly h2*, stopping when the surface is within, an empirically determined, 12 cm of laser scanner. Together the 3D scan acquisition and the servoing to the surfaces edge compose the docking behavior as seen in figure.

5. Manipulating on Flat Surface

   The robot grasps objects by aligning the gripper above the object with the fingers in suitable orientations an then lowering the gripper onto the object. The position and orientation of the object is determined by visual segmentation using eye-in-hand camera.

6. Delivery

   After a successful grasp has been confirmed, the system will back away from the table 40cm and turn 180 degrees. The robot waits for laser

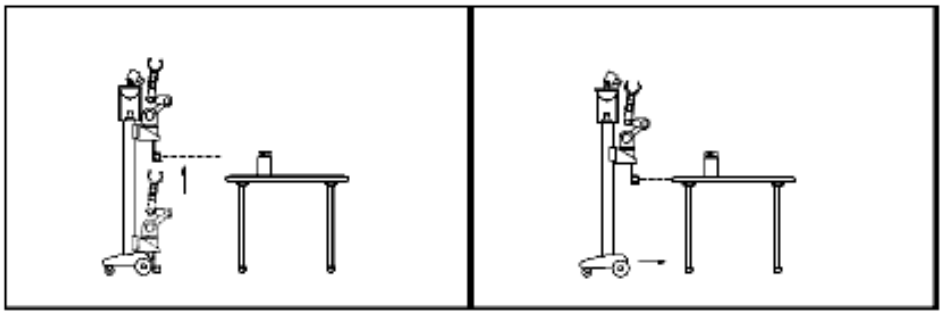

Figure 11: Using Zenither Robot to Scan The Table

pointer detection. The users laser command shows the robot where to look and the robot confirms the users presence. To do this, the robot orients its stereo camera in the direction of of the laser detection and uses a Viola-Jones frontal face detector as implemented in OpenCV in the manner similar to determine whether or not a person is near the laser designated location.

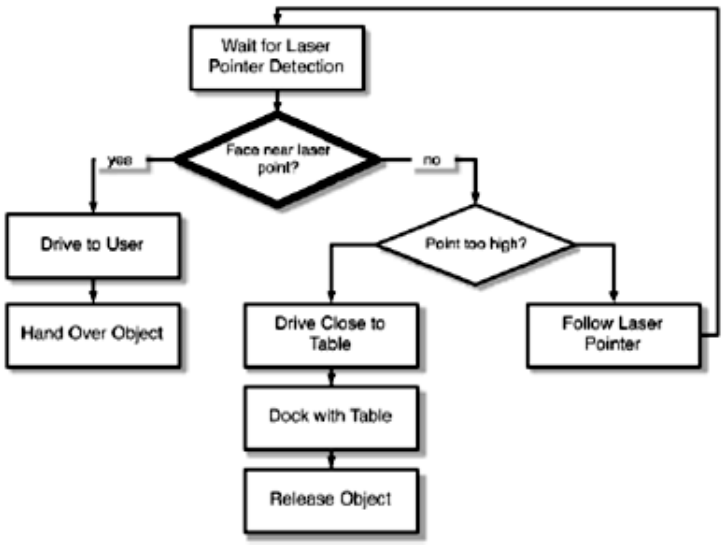

Figure 12: Object Delivery Process

The paper has showcased a prototype of robot that could fetch common objects such as football, vitamin bottles, a medicine and a bottle of soy milk. However due to inbuilt processing systems involved in the robot, the cost of the robot is very high and might be affordable to only rich people. This constraint is a major inspiration for us to develop inexpensive robots that would perform the similar tasks but affordable to all .This can be accomplished by allowing the complex computations required by the robot to be done by external systems such as laptops or personal computers which are commonly available.

5 System Design

Design of the Overall System involve three interdisciplinary area such as **Computer Science**, **Electronics** and **Mechanical Design**. Where **Software Design** involve the Design of the majority of the phases like System calibration, User interface, Request processing, Image capturing, Object Detection, Mapping and Routing. Whereas **Electronics design** involve the designing of the additional RF transmitter and Receiver circuit board and its operations, design of the Robot and its controlling and other intercommunication activities like encoding, decoding and bit-code mapping and **Mechanical Design** involve the designing of the Robot and Robotic Arm.

These three Interdisciplinary Designs are explained in following Subsections.

5.1 Software Design

In our project we have used many Programs and Applications at different points of the overall System such as

1. A Android Application
2. Request Processing Program
3. Template Extraction Program
4. Object Detection using Feature Detection Method
5. Perspective Dimension Evaluation Program
6. Routing Distance Calculation Program

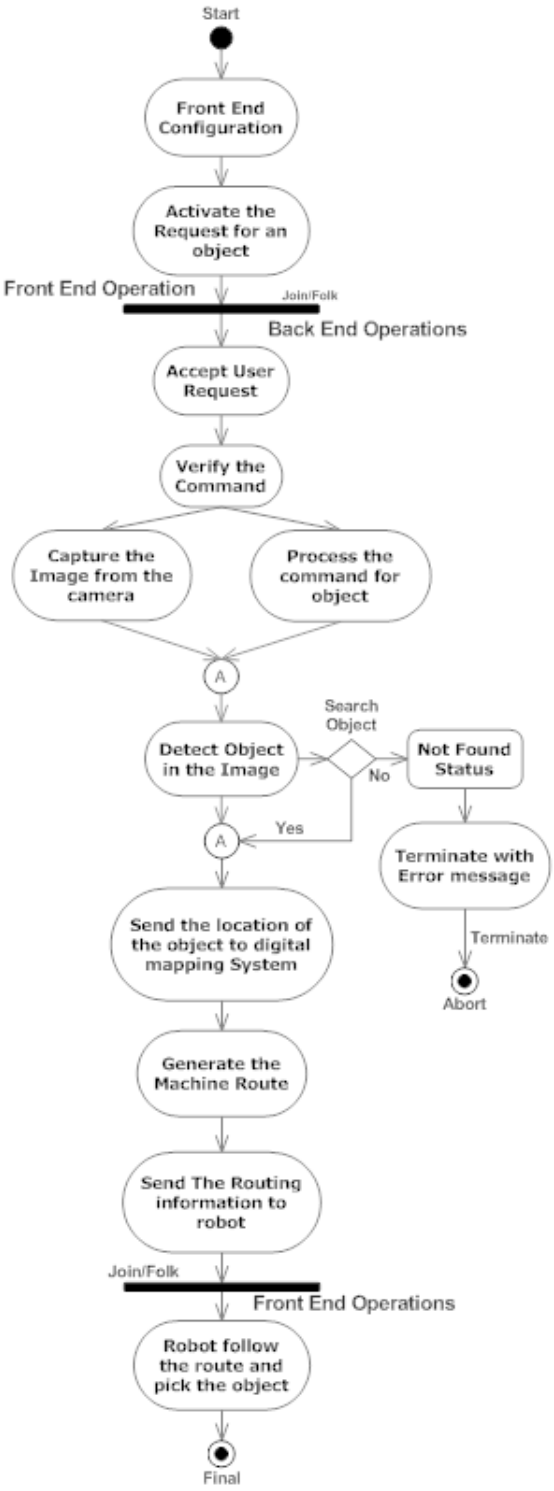

Figure 13: Activity Diagram of the Overall System

7. Generating Routing Map File Program
8. Send Route Bytes to Serial Communication Port Program

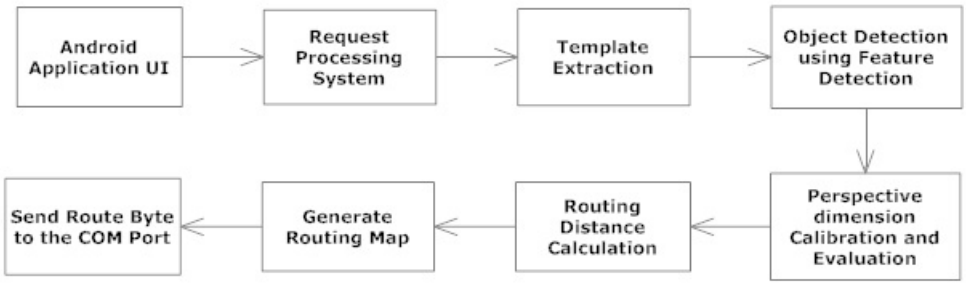

Figure 14: Block Diagram of Software System

5.1.1 A Android Application

To Activate the request for an object user need to have some device or medium which is capable of sending information or command to the Server system and also is easily available to the user and easy to operate. The device or medium should be portable and dedicated for Request Sending Operations, so we are developing an android application for this soul purpose. Now a days smart-phones are almost used by everyone and installing an dedicated App. in the Smart-phone is flexible and optimal compared to having a Dedicated Physical Device like touch-pad or Key-pad for the Same Usage.

This Android Application should display certain objects on its launch screen and when user wanted our robot to pick anyone of these objects then all he need to do is to press a icon button that resemble that object on his screen. Once the object request button is pressed the android application should connect to the Server System using *Bluetooth* as communication media and should send a file to the server system. This file should contain the information related the request made for Example, If user presses an Icon of **water bottle** then a automated file should generate and should be sent the Server System, this file should contain some information regarding object like **"WaterBottleRequested"**. This File further used in other phase of System.

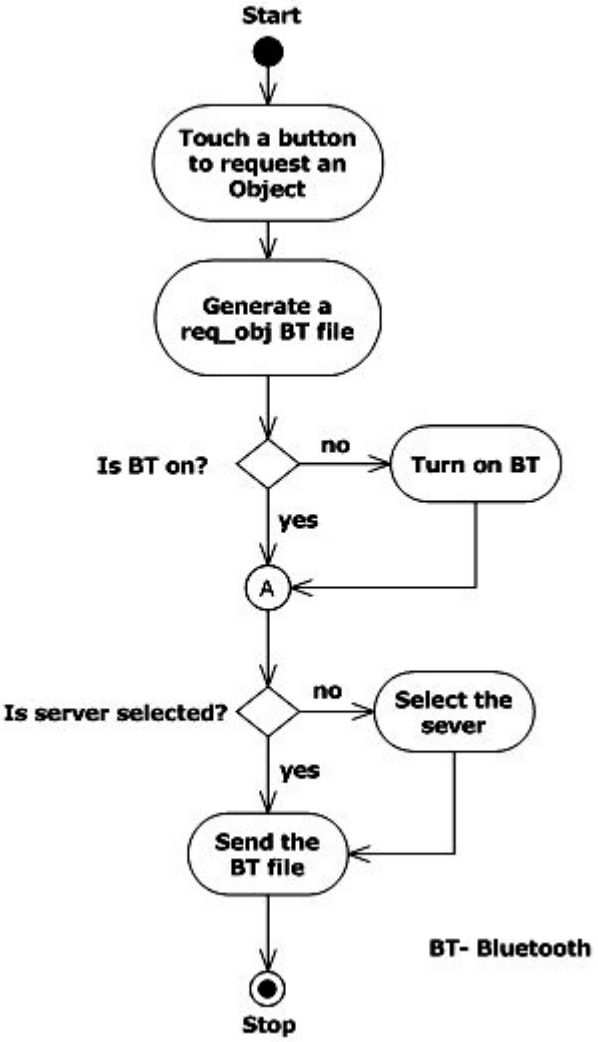

Figure 15: Activity Diagram for Android App. UI

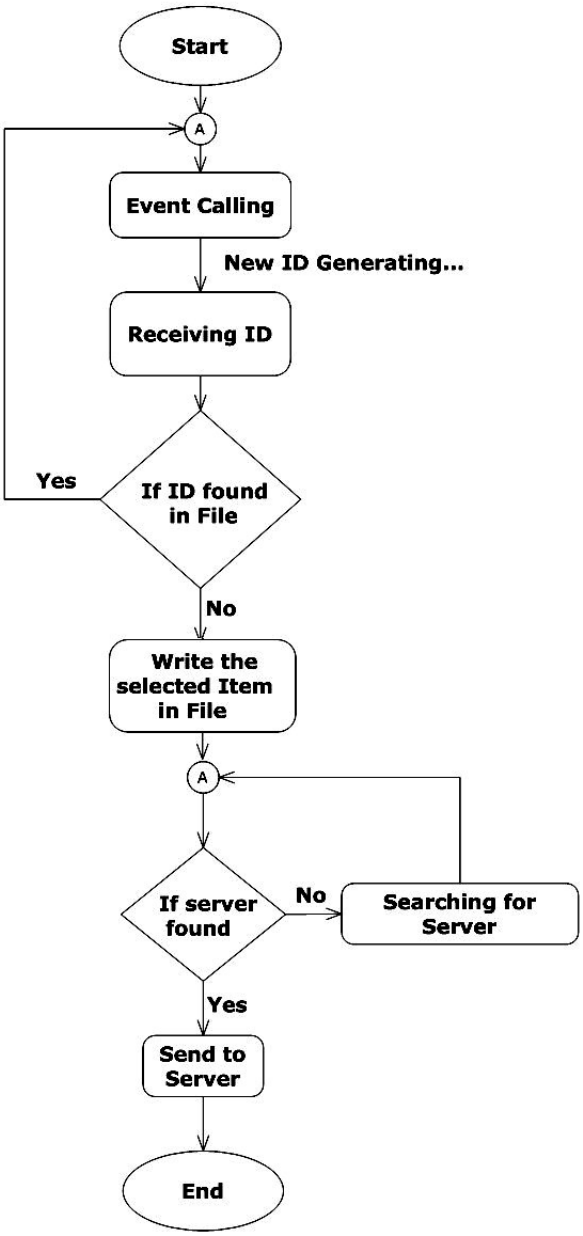

Figure 16: Flow Chart of Event Handling for Android App UI

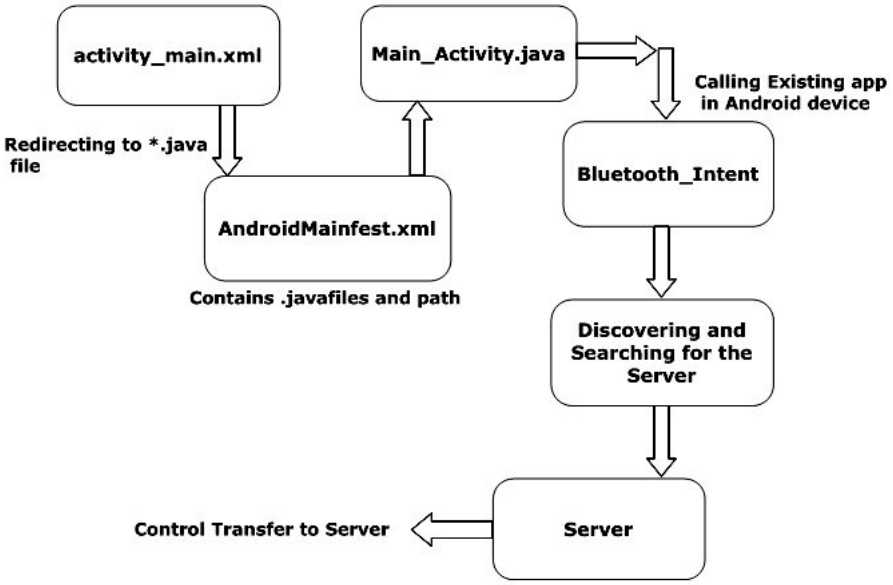

Figure 17: Control Flow of Android Application System

Figure-15 shows the control flow of the android application system and various activity of the user interface is explained in Figure-16 Activity Diagram for Android Application User Interface. Programmatic Event Handling flow chart of the android application is shown in Figure-17.

5.1.2 Request Processing Program

It is the part of the main software system, where system continuously waits for new user request. User request from the previous module **'Android User Interface'** is sent in the form of a text file, this text file contains the request message(say *request_BT_file*). While system waits for new user request it iteratively checks for the exists of *request_BT_file* in a particular file path defined, if such file is not exist in that particular file path then System wait else if the file exists then program extract the modification time from the *request_BT_file* and compute the modification time difference between the previous *request_BT_file* and new *request_BT_file*.

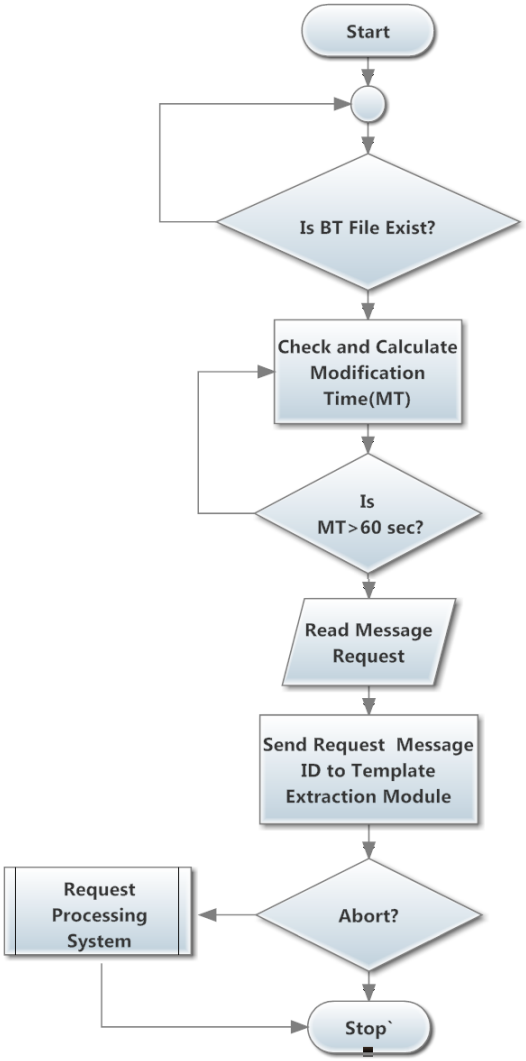

Figure 18: Flow Diagram for Request Processing System

For proper synchronization of all the modules of our system **"Robot Assistant for Elderly Person/Patient"** the time delay for completion of the overall task should be defined or assumed. Time delay is defined in according to the time taken by the Complete System to Execute the Operations flawlessly. We have to ensure that user is not giving redundant request and for this purpose we have to specify the time difference for each request.

In **Request Processing** module we have time difference constraints so that system should get enough time to complete its previous task requested by the User. Once we have Modification Time Difference of *request_BT_files* we check the difference for minimum difference of 60 seconds i.e., **MTD > 60sec** if this condition satisfies then program will read the *request_BT_file's* content and refine it to a particular data structure and send the Request message data to the next module. If **MTD** is less than 60 second system repeat to check for newly modified file.

Request Processing module start the above mentioned functionalities as soon as the overall main system is online and continues to execute the functionalities till Server operator manually abort the system operations or till the server goes offline.

Figure-18 shows the flow diagram of the Request Processing System and Request Processing Program is a sub-program of entire main System Program and Operations from this module to other modules are processed sequentially.

5.1.3 Template Extraction

Once the request object ID is received at Template Extraction Module, it will check for the match for the object ID. In our project we have defined the constraints on the set of objects that can be requested by the user. List of Object displayed on the Android Application Screen are fixed objects and for each object we have defined unique Object ID. For each defined objects on the list we have to maintain the object templates, these templates are the exact replica of the object presented on the user room and these templates should have same features as the objects need to be detected. Set of Templates should be stored in the System Dataset so that these templates are always available to system for its computation and operations.

Templates store are in the form of image preferably in *.png* or *.jpg* format. When there is a match in the object ID template extraction module search for that particular template image file in the system dataset if found the template Extraction Module will return the path of the template image to next module of the overall software system that is **Object Detection Module**.

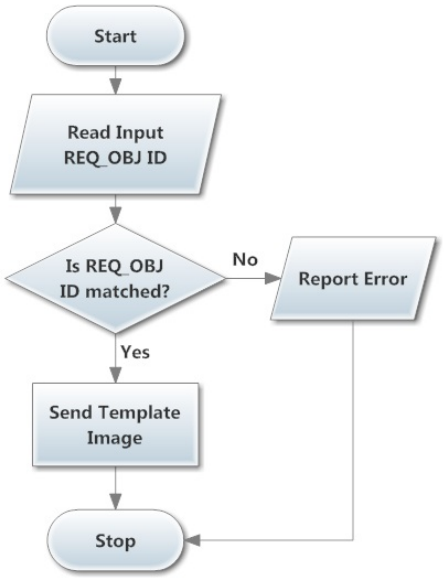

Figure 19: Flow Diagram for Template Extraction Module

If the object ID is not matched with any of the pre-defined object template list then Module should report the error and log the error informations. If object ID is a match and there is no template image stored in the system then system should report the Missing file error and abort current request for fetch that object and ask the system operator to calibrate the object and create a template file for the reported missing template file. The name of the Template file Images should be static and pre-defined.

Further for the accuracy of the object detection we can use the another set of different object templates which are the possible matches for the requested object Id, for example, for the first object template file if the object is not detected then template extraction module will return a second template file for the same object but with different in structure, this increases the accuracy of the object detection in the overall software system.

5.1.4 Object Detection using Feature Extraction Method

Feature extraction[15] for compact representation of image data in computer vision. Feature extraction a type of dimensionally reduction that efficiently represents interesting parts of an image as compact feature vector. This approach is useful when image sizes are large and a reduced feature representation is required to quickly complete tasks such as image matching and retrieval.

Feature detection, feature extraction and matching are often combined to solve common computer vision problems such as **object detection** and **recognition**, **content based image retrieval**, **face detection** and **recognition** and **texture classification**. Common feature extraction techniques include the *Histogram of Oriented Gradients*(HOG), *Speeded Up robust Features*(SURF), *Local Binary Patterns*(LBP), *Haar Wavelets and color histograms*.

We are using the latest image detection scheme called **SURF** which stands for Speeded-Up Robust Features. SURF detector focuses its attention on blob like structures in the image. The structures can be found at corners of the object but also at locations where the reflection light on specular surfaces is maximal(i.e light speckles).

The surf detector algorithm can be summarized in the following steps

1. Form the scale-space response by convolving the source image using DoH filters with different .
2. Search for local maxima across neighbouring pixels and adjacent scales within different octaves.
3. Interpolate the location of each local maxima found
4. For each point of interest, return x, y, σ, the DoH magnitude and the Laplacians sign.

Object Detection is the main objective of the feature detection, object can be detected by comparing the matched features between two images where first image represent as template image of the object that need to be detected, second image is the Scene image (Image of the environment from where the object need to be detected). The below is the activity diagram

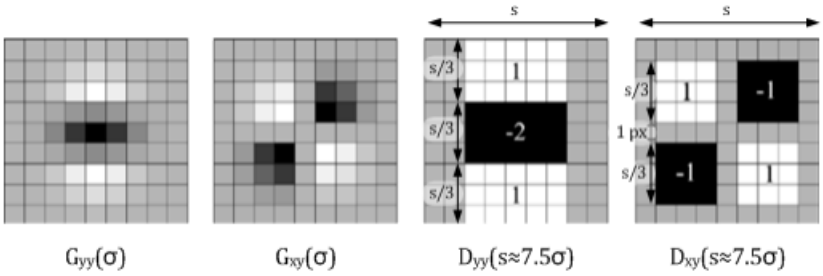

Figure 20: Gaussian second Derivatives

$$DoH(x,y,\sigma) \approx \frac{D_{xx}(x,y,\sigma)\cdot D_{yy}(x,y,\sigma) - \left[\beta \cdot D_{xy}(x,y,\sigma)\right]^2}{\sigma^2}$$

$$where\ \beta = \frac{\left\|G_{xy}(x,y,\sigma)\right\|_F \cdot \left\|D_{yy}(x,y,\sigma)\right\|_F}{\left\|G_{yy}(x,y,\sigma)\right\|_F \cdot \left\|D_{xy}(x,y,\sigma)\right\|_F} \approx 0.9$$

for the Object detection method used in this project where two images are taken and corresponding methods are applied as described in the figure.

In object detection process we consider two color images, one the template image received by the **Template Extraction** module and other image that is captured by the camera which will capture the recent image of the environment from where object need to be detected. These images are converted to *gray* image formate for the feature detection purpose and we should make sure that the images are in the proper formate before passing to next method of feature detection. We use **SURF Feature Detection** method to detect the features from both images and in next flow We extract the Feature from both images for further comparison purpose. Using the extracted features from both images we match these features are check whether features are matched or not, if features are matched then we should make sure that is there enough number of features are matched so that object can be identified those matched features. If there are no enough number of features are matched or if No features are matched then we report the error and terminate the process. The limit for the number of feature should be matched is predefined by the *matchfeatures* method, this method will give an exception if there are no enough matched features reporting about the same. We use this exception to decide the error factor.

Once there are enough features are matched by the previous module we use these matched features to **Estimate the Geometric Transfer**

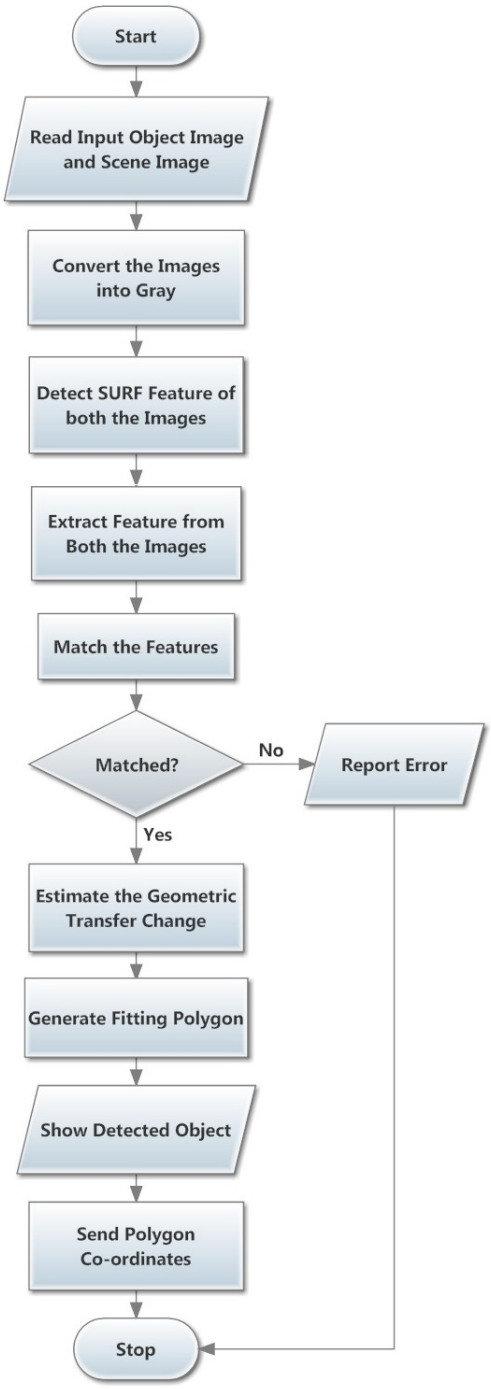

Figure 21: Flow Diagram for Object Detection Module

Change, this process make sure that whether the detected object is moved Geometrically i.e., it checks whether the feature matched are transformed geometrically or not, if transformed then it calculate the transformed angle and ratio of transformation.

Using the Template image of the object received from **Template Extraction Module** we Generate a fitting polygon and then we multiply the Geometric transformation factor to this fitting polygon this ensure that the resultant fitting polygon also shows the Geometric transformation and increase the accuracy of the object detected. The fitting object is then displayed on the scene image representing the detection of the object. The fitting polygon in then is passed to next module of the project called ***Perspective Dimension Evaluation*** where coordinates of the fitting polygon is used to find the actual dimension of the detected object and for other calculation needs.

5.1.5 Perspective Dimension Evaluation

Real-world objects can be viewed at a range of distances and thus can be experienced at a range of visual angles within the visual field. Given the large amount of visual size variation possible when observing objects, we examined how internal object representations represent visual size information. In a series of experiments which required observers to access existing object knowledge, we observed that real-world objects have a consistent visual size at which they are drawn, imagined, and preferentially viewed. Importantly, this visual size is proportional to the logarithm of the assumed size of the object in the world, and is best characterized not as a fixed visual angle, but by the ratio of the object and the frame of space around it. Akin to the previous literature on canonical perspective, we term this consistent visual size information the canonical visual size.

In the real world, the particular view of an object (i.e., its projected retinal image) depends on where the observer is standing with respect to that object. This fact is implicitly understood by observers choosing where to sit in a cinema theatre, where to stand in an art gallery, or where to move to get a better view of an item of interest. When observers walk around an object, changing the viewing angle of an object without changing its distance, this image transformation is called a perspective change. Similarly, when observers approach or back away from an object to change its retinal size within their visual field without changing the viewing angle, the image

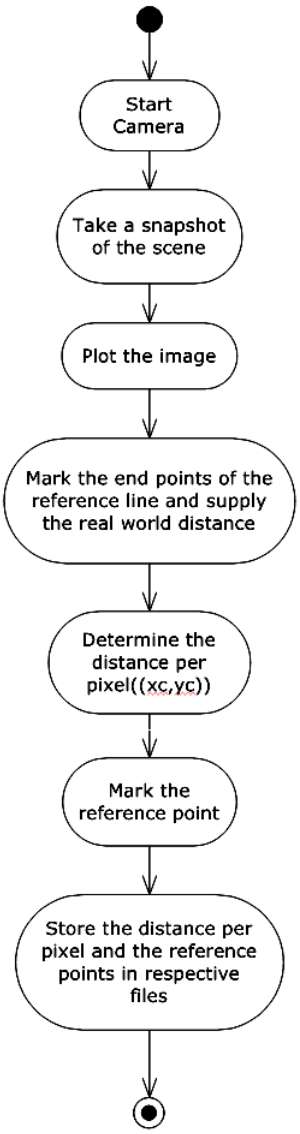

Figure 22: Activity Diagram for the Calibration System of Perspective Dimension Evaluation Module

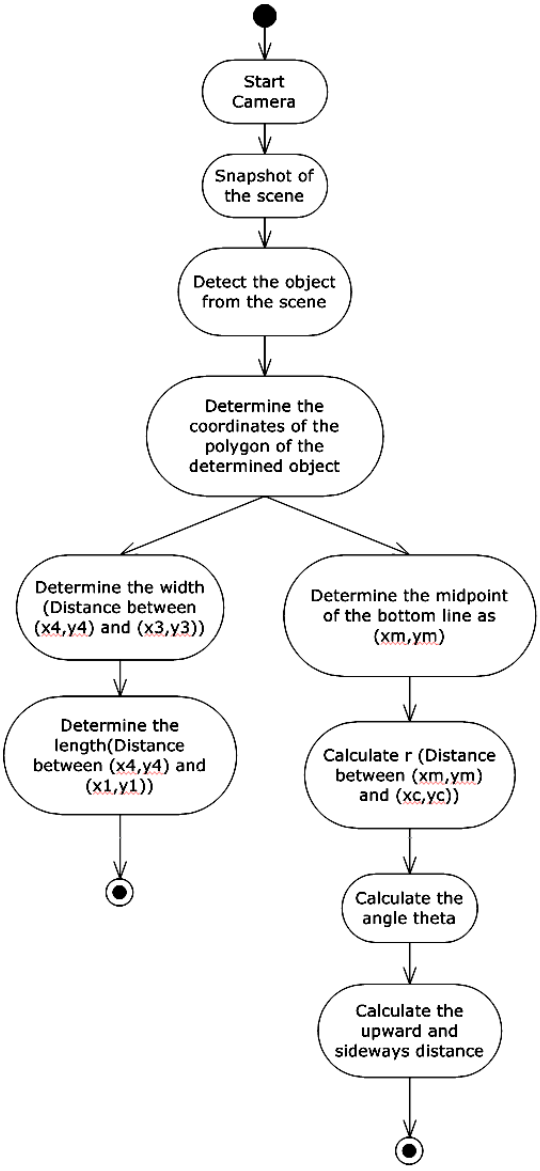

Figure 23: Activity Diagram for Perspective Dimension Evaluation Module

transformation is called a visual size change. Given the many possible object views that can be experienced by an observer, what information about perspective and size is present in object representations.

Perspective is determined by the physical orientation of the object relative to the direction of gaze of the observer. Similarly, visual size is determined by the physical size of the object relative to the distance of the observer to the object.

We might expect knowledge about the real-world size of the object to matter for an objects canonical size. Intuitively, smaller objects in the world subtend smaller visual angles on average than larger objects in the world. For example, a typically sized car would subtend about 30 degrees visual angle at a typical viewing distance of 9 m. For a penny to subtend that same visual angle it would have to be held only 3 cm away from one eye; at a more typical arms-length viewing distance, it subtends 3.5 degrees. Thus, natural experience with objects might predict a systematic relationship between real-world size and canonical visual size. Alternatively, maximizing the available object information could determine canonical size, for example, if the object is centered in the high-acuity foveal or para-foveal region of the visual field. Such an account might predict that all objects would have the same canonical visual size that is related to acuity falloff with eccentricity, possibly modulated by the internal complexity of the surfaces features of the object.

Spatial Calibration to obtain the dimensions and location of the object. We cannot obtains the dimensions of the object in the image using only one reference point. However if are able to calibrate the system using an object of known dimensions we can find the dimensions of any object in the system. The concept has been used successfully in Mars Curiosity rover where its Mars Hand Lens Images uses a penny to calibrate its own system. The MAHL then accurately determines the size of any rock in the image captured.

We are using a system where we are calibrating our system using a reference line of known length . Calibration involves finding the distance per pixel in known units and using that value to accurately determine the dimensions of any object captured by our camera. In order to find the distance of the object from the reference point , its important that we set the reference point during the calibration.

Distance between any two pixels in the image = $\sqrt{((x2-x1)^2 - (y2-y1)^2)}$
Where (x1,y1) and (x2,y2) are the locations of the two pixels.

Using the distance per pixel, we calculate the distance as
Distance = (Distance in pixels) * (Distance per pixel)

These distances and the values are stored in a data structure that can be used for further processes and these distance are also used to find the direction of the objects from the calibrated landmark position and results are attached to the data structure.

5.1.6 Routing Distance Calculation

This method is where we do the computation and generation of the route informations that are need to be sent the robot for its travel towards the object. We have used few assumptions and constraints for this purpose. The *Constraints* are as follows

- Robot and User position are fixed for a calibrated environment.
- Object are placed on a platform which is focussed by the camera.
- Object can be moved but should always be focussed to camera.
- We use a landmark on the platform which is used to for route calculation purpose and it is fixed always.
- Distance of the Object are measure from the Landmark specified.

The below figure shows the flow diagram of the **Routing Distance Calculation Module** and this module consists of few manual calibration part that should be done before setting the system online. The calibrations are used to find the *Pixel location* and *Actual Location* of the landmark on the platform and Robot actual coordinates with respect to the specified landmark. These are the calibration data that is required to Calculate the Route and distance.

We have to make sure that the Calibration Results are fed before the system goes online otherwise exception error need to be faced. **Routing Distance Calculation Module** uses one more data which which is the matrix data structure from the previous module *Perspective Dimension Evaluation.* Using all these data are used calculate the route and direction from the

robot to object. The next process of calculation of Route in accord to Robot parameters are explain in detail in the next subsection.

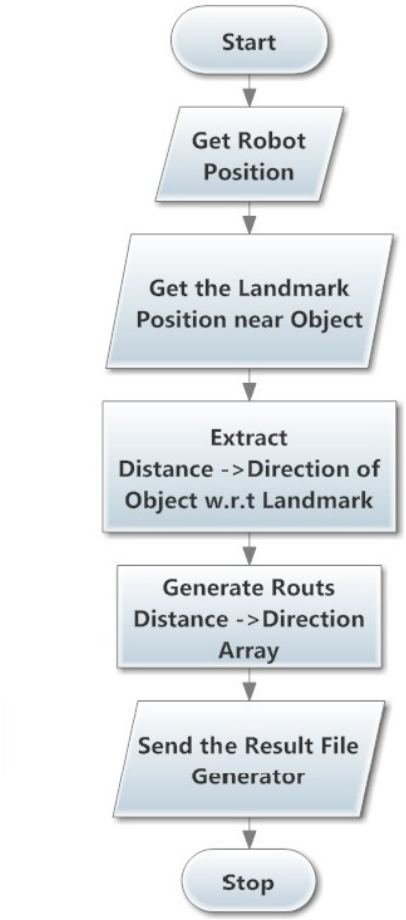

Figure 24: Flow Diagram for Routing Distance Calculation Module

5.1.7 Generating Routing Map File Program

The Dynamic Routing Mechanism is a technique to find an optimal path. And in here we come across how the robot reaches its destination. We can find plenty of ways that the robot can reach destination but all we need is an optimal path. To find this optimal path at least we need the distance and

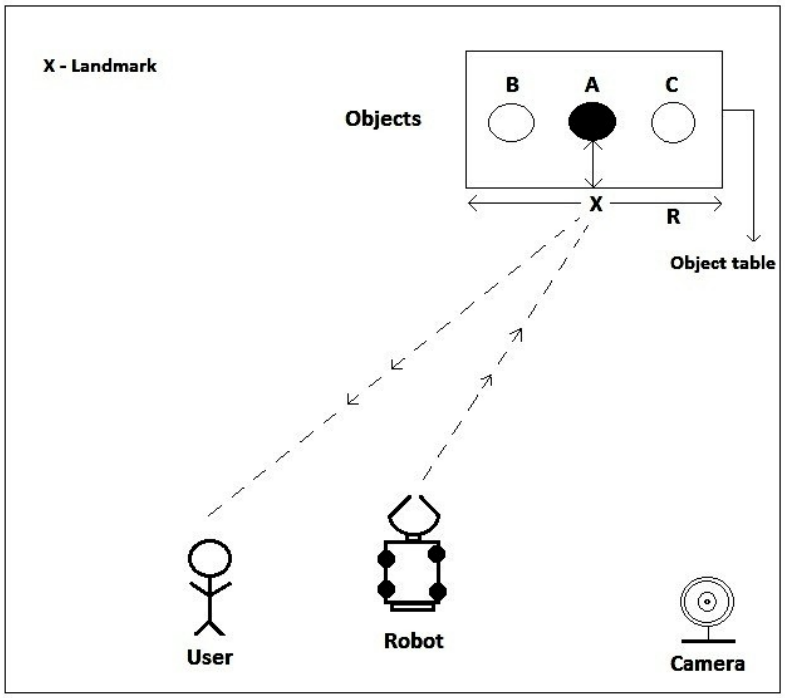

Figure 25: Schematic Diagram of Distance calculation Module

the direction as two parameters. Like so, here in this process we read two parameters such as the direction and the distance. The robot must move in any direction to reach the destination. For that we have set default four direction along with the rotation that is nothing but Turn. Initially, the robot will move in a forward direction, maybe we can set this as default, and also you can give your own direction that the robot can move, like Backward to move the robot back or it may be Left or Right to navigate to the any of the direction. Once you set the direction for a robot it can find the path. But the matter is how far the robot should move. For this we have to calculate the distance between the object and the robot. Basically, the distance is mainly depends on the Circumference of the wheel. The distance can be finding out by the number of rotations of the wheel.

Once these two parameters are satisfied we can go for further process. Here we store these two parameters in an array. Let us say this array as Distance→Direction array. This array consists of the direction that the robot should move and the distance, how far the robot should move from the origin. The landmark is set for the robot to reach form the starting point. Once the

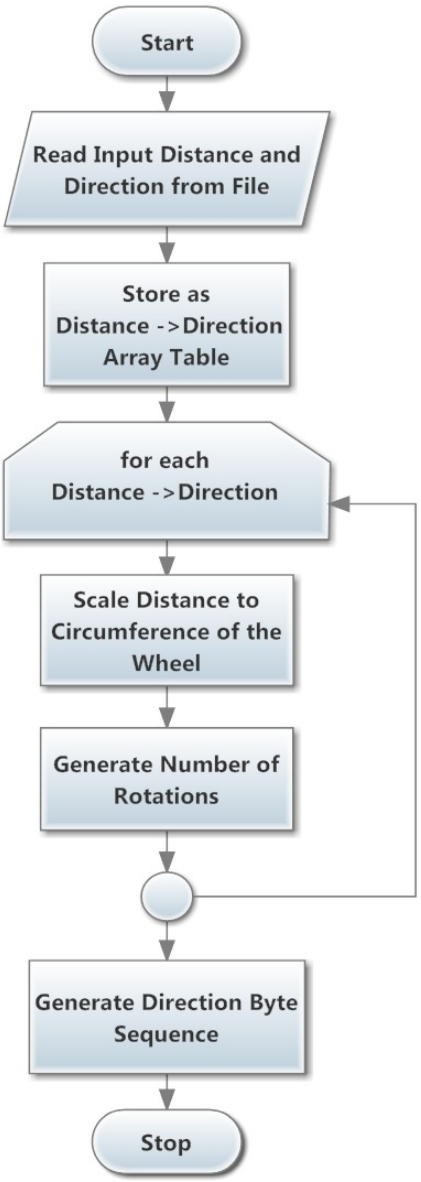

Figure 26: Flow Diagram for Generating Routing Map File Program

landmark is reached the robot will pick up the particular object which has been detected and lead it to the patient or an elderly person.

This process is continuous, the robot should move on until it finds its destination. Mean while it has to calculate its distance recursively. Once the destination is found, then only the process ends until that we should calculate the number of rotations the robot is made. Now scale distance to circumference of the wheel. Consider that the while has a diameter of 25, and if the wheel finishes its one rotation it is nothing but it has covered the 25 cm in any of the direction.

We should generate the number of rotation of the wheel by calculating the circumference of the wheel. The circumference can be calculated by the diameter of the robot wheel. For example, the diameter of the wheel is fifty and the circumference becomes fifty cm. The number of rotation made is now one. Similarly if it is 100 then the rotation is 2 and so on. These values will be stored in an array.

Once we get these Rotation numbers we can generate a byte sequence for these values. This can be done by the parameters that we have mentioned above. It will display a byte sequence in such way that the robot is moved. It is also based on the direction and the distance of the robot. For example if the robot moves forward and its wheel diameter is 50 and it rotates 10 times it will display the output as FFFFFFFFFF. Similarly for all other directions it will display the byte sequence.

Finally, based on all above discussions we can get the byte sequence as output in this phase and it is obtained by the robot direction and the distance between the object and the robot. Here we conclude that the generation of the byte code sequence will bring you an optimal path for the robot to find its destination.

5.1.8 Sending Route Bytes to Serial Communication Port

After the Generation of the Route file we need to transmit the file information to Robot using wireless technology. The Wireless technology we are using here is Radio Frequency(**RF**) Transmission technology. About the transmission technology and other communication details are explain the **Electronics Design** Section.

Before sending the Route file information to our Robot using the RF technology we have to send the Route file information to the ***System Serial Communication Port*** also referred as **COM-PORT**. Once the RF Device is attached to the Server System then we have to find the **COM-PORT** number of the device and should store and fed to the system in this module.

Since we are using **Matlab** platform serial communication is easier than the *embedded C programming*. Matlab allow as access the serial communication port as a file and we can use the common file access functions and method for serial communication also. In this module we have to take care of the many error and exceptions. The following are the important Exceptions to be handled carefully with proper error recovery methods.

- Existence of the Route File should be assured.
- Detection of RF Transmission Module on the specified COM PORT number should successful.
- Null Information in the Route file should be not be sent.
- Close the Route file and importantly COMPORT access after the use otherwise it cannot be used in next use unless it is closed manually.
- Manual Interruption while sending the information.

In case of the manual interruption of this module the program is terminated without closing any files and COM PORT access of that program hence user will not be able to reuse the COM PORT access unless it is close. Since we are using the functions for every module we will not be able to use the access variable once the program is terminated for closing the COM PORT access manually and for reuse of the Module we need to restart the MATLAB program so that it can return all its system sources to the System and can access all the resources refreshed, this make sure that even COM PORT access are revoked and reassigned by the server system to the our module.

Informations are need to be sent character-wise i.e., one character at a time because the RF transmission module is programmed in a such a way that it read a character from the COM PORT and transmit the bit codes to the robot and also one character represent certain distance that should be covered by the Robot. The COM PORT informations are very volatile and they are erased within miliseconds so we have to make sure that we

should transmit the same information for a period of time so that RF Transmission module gets enough time to retrieve the character information from the COM PORT and encoded it and transmit to the Robot so that robot should move defined distance. The Delay is need to calculated by considering the parameters like Baud Rate[16] of the COM PORT Transmission and RF Transmission module program, Robot Receiver module reception delay rate and the distance that can be moved by the robot for the duration it receive the information.

5.2 Electronics Design and Implementation

Our project ***"Robot Assistance for Elderly Person / Patient"*** is a embedded system, it mainly contain software system for majority of the operations such as object detection, routing and also embedded software for communication purpose but it also include the electronics devices and systems for completing the projects objective. In our project we have used the electronics devices mainly for wireless communication purpose and Controlling purpose. Controlling electronic device are explain in detail in the next section of Mechanical Design. For communication purpose our project use mainly two major device sets. they are,

- RF Transmission Device
- RF Receiver Device

RF Stands for Radio Frequency

5.2.1 RF Transmission Device

RF transmission Device consists of the following devices which support and complete the transmission of the received data from the COM PORT, about the serial communication of COM PORT is explain in the previous section of this article. The devices used in the Transmission Device are

- AT-8051 Microcontroller
- USB 2 UART converter
- RF transmitter Device and Encoder

The main objective of the transmitter device is to receive the COM PORT data from the server system, encode the received data which in the form of character to Binary codes and transmit the binary codes as radio frequencies to the receiver attached to the Movable Robot.

USB to UART converter

Description: The USB provides an easy method of connecting TTL to RS232 signals on PC via a USB port.Ideal for microcontroller circuits. The design uses Future Technology Devices Intl Ltd's (FTDI)FT232BM/BL chip. The FTDI website provides the USB drivers for a virtual COM PORT as well as Windows DLLs for interfacing with VisualBasic and Visual C, plus other applications. USB to UART converter is used to covert the USB COM PORT data into the binary clocked data.[**?**]

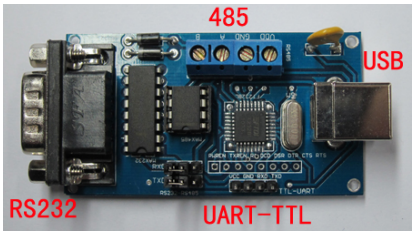

Figure 27: Picture of USB to UART Device

USB to UART TTL: USB to TTL UART Converter, connect circuit:(MCU TX/RX connect PIC/ATMEL/AVR chip UART TTL pin, isn't connect MAX232 TX/RX), only use four lines can work(+5v,GND,TX,RX). When you want to use the FTDI board you have to connect the RX to the TX of the device, and the TX of the FTDI to the RX of the device. The only way to make the MAX works, the lines between the device and the MAX must be crossed.

AT-8051 Microcontroller

Description:

The AT89C51RC is a low-power, high-performance CMOS 8-bit microcontroller with 32K bytes of Flash programmable read only memory and 512 bytes of RAM. The device is manufactured using Atmels high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 and 80C52 instruction set and pinout. The on-chip Flash allows the program memory to be user programmed by a conventional non-volatile memory programmer. A total of 512 bytes of internal RAM are available in the AT89C51RC. The 256-byte expanded internal RAM is accessed via MOVX instructions after clearing bit 1 in the SFR located at address 8EH. The other 256-byte RAM segment is accessed the same way as the Atmel AT89-series

and other 8052-compatible products. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51RC is a powerful microcomputer which provides a highly-flexible and cost effective solution to many embedded control applications.[17]

Pin Configurations The below figure shows the Pin configuration of AT89C51RC microcontroller.

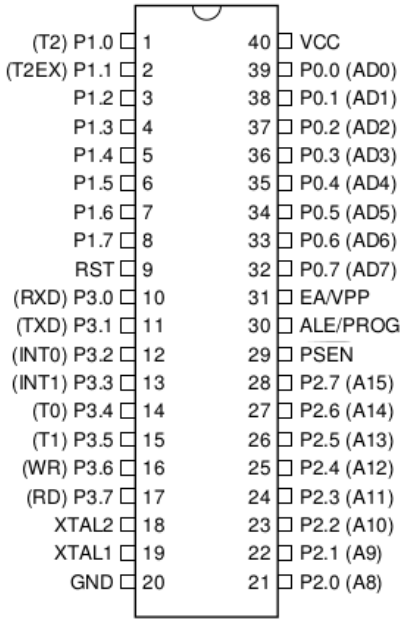

Figure 28: Pin configuration of AT89C51RC microcontroller

Pin Description

- **VCC:** Supply voltage.
- **GND:** Ground.
- **Port 0:** Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high impedance inputs. Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups. Port 0 also receives the code bytes

during Flash programming and outputs the code bytes during program verification. External pull-ups are required during program verification.

- **Port 1:**Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, Port 1 also receives the low-order address bytes during Flash programming and verification.

- **PORT 2:**Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups.

- **Port 3** Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pull-ups. Port 3 receives some control signals for Flash programming and verification. Port 3 also serves the functions of various special features of the AT89C51RC, as shown in the following table.

Port Pin	Alternative Function
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	WR (external data memory write strobe)
P3.7	RD (external data memory read strobe)

Purpose

The main purpose of the AT89C51RC microcontroller is to read the information sent from the server system to COM PORT and extract the COM

PORT data which is in the form of character. These read character are encoded to binary unique codes and send these binary codes to the RF Transmitter encoder to transmit the corresponding frequencies from the RF transmitter.

RF Transmitter Device and encoder

Description

A wireless radio frequency (RF) transmitter and receiver can be easily made using HT12D Decoder, HT12E Encoder and ASK RF Module. Wireless transmission can be done by using 433Mhz or 315MHz ASK RF Transmitter and Receiver modules. In these modules digital data is represented by different amplitudes of the carrier wave, hence this modulation is known as Amplitude Shift Keying (ASK).[19] Radio Frequency (RF) transmission is more strong and reliable than Infra-red (IR) transmission due to following reasons:

- Radio Frequency signals can travel larger distances than Infra-red.
- Only line of sight communication is possible through Infra-red while radio frequency signals can be transmitted even when there is obstacles.
- Infra-red signals will get interfered by other IR sources but signals on one frequency band in RF will not interfered by other frequency RF signals.

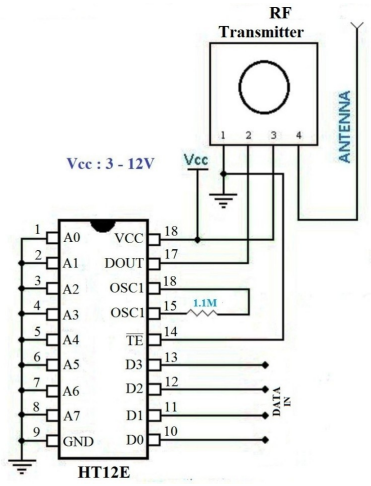

Figure 29: Circuit Diagram of RF transmitter

HT12E Encoder IC will convert the 4 bit parallel data given to pins D0 D3 to serial data and will be available at DOUT. This output serial data is given to ASK RF Transmitter. Address inputs A0 A7 can be used to provide data security and can be connected to GND (Logic ZERO) or left open (Logic ONE). Status of these Address pins should match with status of address pins in the receiver for the transmission of the data. Data will be transmitted only when the Transmit Enable pin (TE) is LOW. 1.1M resistor will provide the necessary external resistance for the operation of the internal oscillator of HT12E.

Figure 30: Circuit Device of RF transmitter

5.2.2 RF Receiver Device

RF Receiver device is attached to the Robot and it consists of the RF Receiver Module device, ATMEGA 32bit microcontroller and motor driving circuit. These devices receives the RF transmitted data, encode the data to binary code and control the movements of the Robot.

- RF Receiver Module Device
- ATMEGA 32bit Microcontroller

RF Receiver Module Device

RF Receiver receives the data transmitted using RF Transmitter. HT12D decoder will convert the received serial data to 4 bit parallel data D0 D3. The status of these address pins A0-A7 should match with status of address

pin in the HT12E at the transmitter for the transmission of data. The LED connected to the above circuit glows when valid data transmission occurs from transmitter to receiver. 51K resistor will provide the necessary resistance required for the internal oscillator of the HT12D.[19]

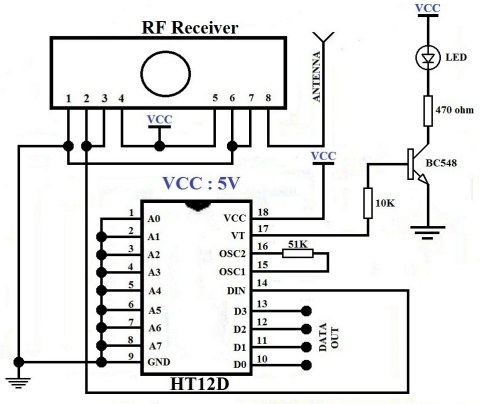

Figure 31: Circuit Diagram of RF receiver

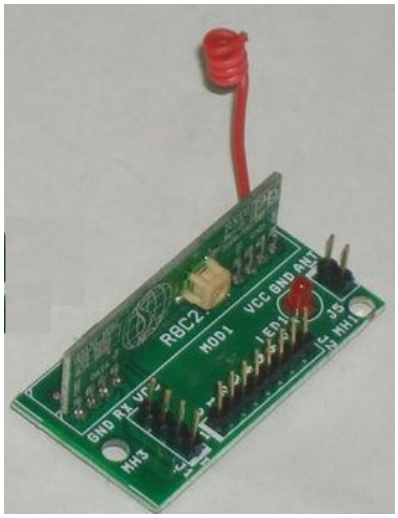

Figure 32: Circuit Device Module of RF receiver

ATMEGA 32bit Microcontroller

Overview

The AtmelAVR ATmega32 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

The AtmelAVRAVR core combines a rich instruction set with 32 general purpose working reg- isters. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

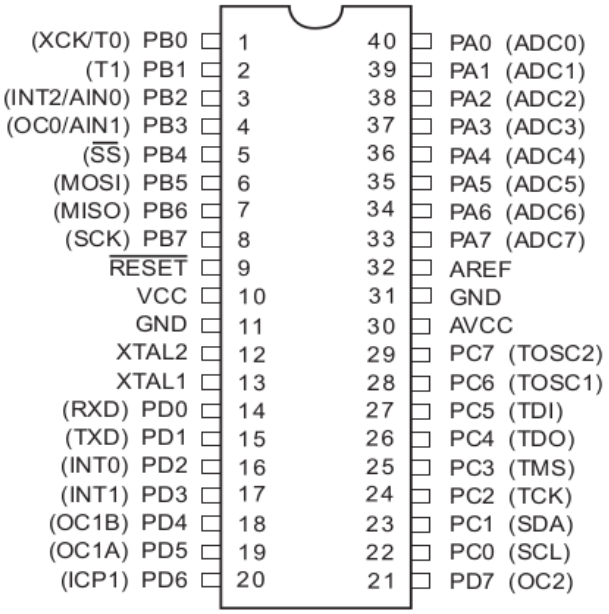

Figure 33: Pin Configurations of ATMEGA 32

Pin Descriptions

- **VCC** Digital supply voltage.

- **GND** Ground.
- **Port A** (PA7..PA0) Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.
-

5.3 Mechanical Design

5.3.1 Robot Design

–Need to write

Platform and Moving Mechanism

–Need to write

Controlling of Operations

–Need to write

5.3.2 Robot Arm Clip Design

–Need to write

6 System Implementation

6.1 Request Processing System

6.1.1 Algorithm

1. The request from the Android app is registered in Bluetooth_Req_Data.txt.

2. If the file exists go to step 3.Otherwise go to step 1.

3. Get the current time and compare it with the modified time of the file. If the request is generated less than 20 seconds ago, the requested is accepted and go to step 4. Othewise **goto** step 1.

4. Read the file and get the requested object name

5. Get the scene image by using the StartCamera **module**.

6. Get the template of the requested object from the database using the FindTemplet() **module**.

7. If the requested object is present in the scene, get the polygon coordinates of the object in the scene using the FeatureObj **module**.
   [PolygonPoints,˜,˜]=FeatureObj(ObjectTemplet, SceneImage)

8. Get the real world width, height, sideways distance and upward distance of the detected object from the reference point.
   [width, height, Distance, Side, Up] =
   FindDimension(PolygonPoints)

9. Generate a path matrix
   RtMx=PathMatrix(distanceRightways,distanceUp)

10. Generate the route file
    Filename=GenerateRoute(RtMx);

11. Send the file to the robot using the SendRoute **module** using RF transmission

12. Get the status of the transmission.

13. Ask the user **if** he wants to **continue** .If yes go to step 1

14. End

6.1.2 Data Structures

1. PolygonPoints is a matrix which contains the rectangular coordinates of the detected object in the scene.

2. Width and height are scalar variables containing the real world height and width of the object.

3. Distance, distanceUp, distanceSideways are the scalar variables which store the real world values of the distances of the object from the reference point

4. RtMx is a 2 dimensional matrix which contains the pairs (distance,direction).

6.1.3 Error Handling

1. If the Bluetooth_Req_Data.txt doesnt exist , the program loops.

2. In case the object is not detected, return a message and program goes into loop.

6.2 Template Extraction

6.2.1 Algorithm

1. Get the requested object name from the
   Bluetooth_Req_Data.txt.

2. Switch (Object name)
{
Case 'WATERBOTTLE REQUESTED':
   Get the water bottle template image
   from the database

10

Case 'MOBILE_REQUESTED' :
   Get the mobile template image from the database

Case 'SPEC_REQUESTED':
   Get the spectacles template image
   from the database.

Case 'SYRUPBOTTLE_REQUESTED': Get the syrup bottle

template image from the database.

Case 'Glass Requested' : `Get the glass template image from the database`

`Case 'Oil Requested'` : Get the oil bottle template from the database.
}
3. End

6.2.2 Data Structures

1. ObjectImage contains the template image of the requested object.

6.2.3 Error Handling

1. In case Bluetooth_Req_Data.txt is missing, a corresponding error message is displayed.
2. In case the requested image template doesnt exist, print an error message stating that the requested template is missing.

6.3 Object Detection using Feature Extraction

6.3.1 Algorithm

1. Read the requested object template image as boxImage.
2. Convert boxImage from RGB to Gray.
3. Convert the Scene image from RGB to Gray.
4. Detect the SURF features the boxImage .
5. Detect the SURF features of the SceneImage.
6. Extract the features of the boxImage.
7. Extract the features of the SceneImage.
8. Match the features of the boxImage and SceneImage.
9. Show the matched features of the boxImage and SceneImage.

10. Estimate the Geometric transform of the matchedBoxPoints and MatchedScenePoints.

11. Get match of boxImage in the SceneImage.

12. Display the images of the boxImage and SceneImage

13. Display the polygon coordinates of the matched boxImage in the SceneImage and display the polygon

6.3.2 Data Structures

1. 2d matrix SceneImage to hold the gray image of the Scene

2. 2d matrix BoxImage to hold the gray image of the template image of the object

3. 2d Matrix BoxPOints to hold the surf features of the object image.

4. 2d Matrix ScenePoints to hold the surf features of the scene object.

5. 2d Matrix showMatchedFeatures to hold the matched features of the boxImage and the SceneImage.

6.3.3 Error Handling

1. In case the boxImage doesnt have a match in the SceneImage, print a message stating that the requested object was not found.

6.4 Finding the real world dimensions of the detected object

6.4.1 Algorithm

1. Determine the coordinates vertices of the rectangular polygon coordinates.

   Bottom Left = (x1,y1)
   Bottom Right= (x2,y2)
   Top Left = (x3,y3)
   Top Right = (x4,y4)

2. Find the width of the object in pixels
   Width of the object in pixel

= sqrt((x1−x2)^2+(y1−y2)^2)

3. Find the height of the object in pixels
   Height of the object in pixel
   = sqrt((x1−x3)^2+(y1−y2)^2)

4. Find the real world measures of width and height
   Real world width = Width in pixels * Distance Per Pixel
   Real world height = Height in pixels * Distance Per Pixel

5. Find the coordinates of the midpoint of the bottom edge.
   xmid = (x1+x2)/2
   ymid = (y1+y2)/2

6.Find the distance from (xmid,ymid) to the reference point (xref,yref)
   Distance in Pixels =
   sqrt ((xref−xmid)^2+(yref−ymid)^2)

7. Find the real world measure of the distance
   Distance =
   Distance in Pixel * Distance Per Pixel

8. Find the slope between the (xmid,ymid) and (xref,yref)

9. **if**(yref < ymid)
   Theta = atan((ymid−yref)/(xmid−xref))
   else
   Theta = atan((ymid−yreft)/(xmid−xref))

10. Find Sidewise distance of the object from the
    Reference point (xref,yref)
    Sidewise distance =
    Distance in pixel * cos(theta) * distance Per Pixel

11. Find the upward distance of the object from the reference
    point (xref,yref)
    Upward distance =
    Distance in pixel * sin(theta) * Distance Per Pixel

6.4.2 Data Structures

2d matrix PolygonCoordinates which holds the coordinates of the Polygon.

6.4.3 Error Handling

In case of numerical operation exception , display a message stating the occurred error.

6.5 Generating the Path Matrix

6.5.1 Algorithm

1. Define routePath as a matrix of order 7X2

2. The first pair in the matrix defines the initial distance the robot has to travel from its resting place in the forward direction.
   <Distance=200cm, Direction=Forward>

3. The second pair specifies that the robot should rotate to left.
   <Distance=0, Direction=Left>

4. The third pair specifies the distance the robot has to move in the forward direction.
   <Distance=100, Direction=Forward>

5. The fourth pair defines the direction in which the robot has to rotate sidewise.

   if(Sidewise distance > 0)
   <Distance=0,Direction=Right>
   else if (sidewise Distance <0)
   <Distance=0,Direction=Left>
   else
   <Distance=0,Direction=Forward>

6. The fifth pair defines the distance the robot has to move in the sidewise direction.
   <Distance=SidewiseDistance,Direction=Forward>

7. The sixth pair defines the direction the robot has to rotate next.

if(Sidewise distance > 0)
<Distance=0,Direction=Left>
else if (Sidewise Distance <0)
<Distance=0,Direction=Right>
else
<Distance=0,Direction=Forward>

8. The seventh pair defines the distance the robot has to move in the forward direction.
<Distance=Upward Distance , Direction=Forward>

9. End

6.5.2 Error Handling

In case the sidewise distance is negative (for left direction) ,convert to absolute value before defining the corresponding pair.

6.5.3 Data Structure

Matlab 2 dimensional path_matrix to store the path route and directions.

6.6 Generating the route file for the robot using the PathMatrix

6.6.1 Algorithm

1. Open a file in the write mode

2. Find the number of rows in the PathMatrix
Length=Size(RtMx,1)

3. **for** i=1 to Length
if (PathMatrix[1,2]=Forward)
if(distance < 25)
Number of rotations = 1
else
Number of rotations =
round(Distance/Circumference of the robot wheel)
endIf

```
    for k=1 to Num
       Write to the file (F"0)

   else if(PathMatrix[1,2]=Right)
    if(distance < 25 )
       Number of Rotations = 1
    else
       Number of rotations =
         round(Distance/Circumference of the robot wheel)
    endIf

    for k=1 to Rot
       Write to the file (R"0)

   else if(PathMatrix[1,2]=Left)
    if(distance < 25 )
       Number of Rotations = 1
    else
       Number of rotations =
         round(Distance/Circumference of the robot wheel)
    endIf

    for k=1 to Rot
       Write to the file (L"0)

   else if(PathMatrix[1,2]=Backward
    if(distance < 25 )
       Number of Rotations = 1
    else
       Number of rotations =
         round(Distance/Circumference of the robot wheel)
    endIf

    for k=1 to Num
       Write to the file (B"0)
   endIf
endfor
```

4. End

6.6.2 Error Handling

In case the else-if conditions do not match the Pathmatrix values , a message must be displayed stating that the value found was incorrect.

6.6.3 Data Structures

PathMatrix is a two dimensional matrix with each row having the pair ¡Distance,Direction¿

6.7 Sending the Route File to the Robot

6.7.1 Algorithm

1. Open **for** serial communication using the COM PORT
   of the robot available from the device driver.

   St=serial(PortID)

2. Open the route file in the read mode

3. While (true)
   3.1 Scan the character from the file
   3.2 **if** (character is not `'X'` and character is not empty)
   Write to the serial port
   Display the character
   Provide a delay of 0.75ms
   else
   Display `'Terminating the character'`
   Close the serial port
   Close the route file
   endIf
   endWhile
4. End

6.7.2 Error Handling

In case the manual interruption, no existence of the route and COM PORT file and serial port is not available, terminate the transmission and display a message for the exception.

6.7.3 Data Structures

1. A file pointer to the serial port
2. A file pointer to the route file.

6.8 Calibration of the Camera

6.8.1 Algorithm

1. Gets the image of the Scene using the StartCamera module.
2. Find the number of rows and columns and color brands of the scene image.
3. First we find the distance per pixel in the horizontal direction .
4. Request the user to left click on end point of the horizontal line and then right click on the other end point.
5. Enter the distance and the unit of measure.
6. Find the distance in pixels as
   Distance in pixels = $\sqrt{((x2-x1)^2 - (y2-y1)^2)}$
7. Find the distance per pixel as
   DistancePerPixel= distance calculated/distance in pixels
8. Next we need to find the distance per pixel in the vertical direction.
9. Request the user to left click on the endpoint of the vertical line and then right click on the other end point.
10. Enter the distance and unit of measure.
11. Find the distance in pixels as
    Distance in Pixels = $\sqrt{((x2-x1)^2 - (y2-y1)^2)}$
12. Find the distance per pixel in the vertical direction as
    DistancePerPixelVertical= distance calculated/distance in pixels.
13. Next we need to set the reference point
14. Request the user to click on the reference point location.
15. Save the distancePerPixel, distancePerPixelVertical and the reference points coordinates in 3 files.
16. End

6.8.2 Error Handling

In case while finding the distance per pixel, the user accidentally clicks or performs the steps in incorrect way , display a message giving info to the user on how to perform the steps in a right way.

6.8.3 Data Structures

1. DistaneInPixels is a scalar quantity that specifies the distance between two pixels
2. DistancePerPixel is a scalar quantity that specifies the distance per pixel in a particular unit.

6.9 Start Camera

6.9.1 Algorithm

1. Search for a camera for taking video input.
2. Activate the camera for video input as V=videoinput(winvideo,1,RGB24_1280X960);
3. Get the status of the camera
4. If unsuccessful, goto step
5. Pause for 5 seconds
6. Take a snapshot
7. Save the image.
8. End

6.9.2 Error Handling

In case of trouble to connecting to the camera, an error message should display the connection failure and ask the user to try again.

6.9.3 Data Structures

1. A videoinput pointer to reference the camera object
2. A rgb image matrix to hold the snapshot of the scene.

6.10 Sending Request from the Android application

6.10.1 Algorithm

1. Open the Robollamador app in an android phone
2. Select the item to be fetched by clicking on the corresponding icon.
3. Switch on the Bluetooth of the device and ask the user to select the laptop which runs our main application.
4. Connect to the laptop.
5. Delete and then recreate the file Bluetooth_Req_Data.txt
6. Switch(selected item in the android app)

   Case WATERBOTTLE : Write WATERBOTTLE_REQUESTED to the file
   Case MOBILE : Write MOBILE_REQUESTED to the file.
   Case TOWEL : Write TOWEL_REQUESTED to the file.
   Case SPECTACLES : Write SPEC_REQUESTED to the file.
   Case SYRUP : Write SYRUP_REQUESTED to the file.
   Case GLASS : Write GLASS_REQUESTED to the file.
   Case OIL : Write OIL_REQUESTED to the file.

   endSwitch
7. End

6.10.2 Data Structures

ArrayList to store requested object details and for validation purpose.

6.10.3 Error Handling

1. In case the android phone is unable to connect to the laptop via the Bluetooth, display the message showing connection failure and request the user to try again.
2. In case the Bluetooth_Req_Data.txt is not created, display an error message stating that the action failed.

7 System Testing and Results

7.1 Testing

7.2 Result

7.3 Analysis of The Result

–Need To write

8 Applications

1. This Project helps the motor impairments patients in the absence of Nurse.
2. Help the Elderly person in his Room to pick and give him some object to his bed when there are no caretaker from him/her.
3. High processor system can be installed in a hospital to control tens of Robots at a time each in different room serving different patient.
4. Robot can be improved by its mechanical strength to move heavy objects like furniture on demand of user.
5. Since this Robot is completely Controlled by the Server Any Updates in the software operation and computational optimization can be updated easily without any additional expenses.

9 Conclusion and Future Work

The 'Robot Assistant for Elderly Person/Patient' is a highly ambitious project combining all major fields such as computer science, electronics, mechanical engineering and medical science. The project has succeeded in its primary aim of creating an efficient but inexpensive robot prototype that would cater to the needs of the old and sick people. The robot works efficiently for the all the subtasks that make up the entire system working .The Robollamador app works on even basic android phones that most people

Figure 34: Output

possess .The app works in conjunction with the laptop to form an effective request-response model. In our system, the laptop works as the central processing system for all major computations. Camera is fixed and is connected to the laptop. The camera takes snapshots of the place where the objects are kept .The server continuously runs in the background and then starts the main application in case of a new request from the app. The Feature detection module detects the object in the scene .The Find Dimension successfully the finds the real world measures of the width and height of the object and also the distance of the object from the reference point. The system then generates a file that specifies the route directions for the robot and then sends it to the robot. The robot performs the steps as specified, fetches the object and finally gives it to the requester. The overall system works in a progressive manner that is, every task requires previous operation to be successful. The project has successfully integrated image processing applications with electronics to create a highly functional hybrid of both. The robot is inexpensive to build and the server can be setup on any laptop or personal computers.

The future of this project is very promising and some of them are

- The robot can be evolved to take advantage of better hardware and electronic components.
- The image processing tasks of the project such as the object detection, finding the real world dimensions etc can be improvements to take advantage of the booming image processing techniques available .This would highly increase the success rate of the system and the system may withstand any severe constraint to the environment of operation.
- The processor of the robot could be replaced by a high end processor. This would cater to the needs high image processing computations.
- The software can be developed into a effective API toolkit that can help even common people to easily install the system. This can lead to a widespread use of our project.
- The robot hardware can be developed to incorporate simplicity while setting up the system.
- Instead of a central computer, the processor in the robot itself can do the complex computations. This would make the system autonomous.

References

[1] Robotics - Wikipedia - http://en.wikipedia.org/wiki/Robotics#Autonomy_levels

[2] Mobile Robots - Wikipedia - http://en.wikipedia.org/wiki/Mobile_robot

[3] Domestic Robot - Wikipedia - http://en.wikipedia.org/wiki/Domestic_robot

[4] LINK-SIC - Welding Robot - http://www.linksic.isy.liu.se/?page=industrialrobotics-2

[5] Industrial Robot - Wikipedia - http://en.wikipedia.org/wiki/Industrial_robot

[6] Motor Impairment - International Neuromodulation Society -http://www.neuromodulation.com/motor-impairment

[7] Motor Impairments - Web Accessibility - The University of Melbourne - http://www.unimelb.edu.au/accessibility/training/types/motor.html

[8] Existing Assistant Robot - RoboShop - http://www.robotshop.com/blog/en/domestic-robot-assistant-196

[9] Future Robotic Arm - EZRA Magazine - http://ezramagazine.cornell.edu/WINTER12/ResearchSpotlight.html

[10] Android - http://developer.android.com/

[11] Android - http://www.vogella.com/tutorials/Android/article.html/

[12] MATLAB - Mathwork - http://www.mathworks.in/products/computer-vision/

[13] ATMEL STUDIO - http://www.atmel.in/tools/atmelstudio.aspx

[14] Assistive Robot - http://students.washington.edu/zhexu/files/hri2008_workshop.pdf

[15] SURFFeature - Mathwork -

[16] BaudRate - electronic design - http://electronicdesign.com/communications/what-s-difference-between-bit-rate-and-baud-rate#4

[17] Atmel - Datasheet - www.keil.com/dd/docs/datashts/atmel/at89c51rc_ds.pdf

[18] USB to UART - website - http://www.nbglin.com/485.htm

[19] electroSome - RF Transmitter and Receiver - http://electrosome.com/wireless-transmitter-and-receiver-using-ask-rf-module/

Appendix