Index

List of Figures

List of Tables

List of Abbreviations

Abbreviations	Meaning
SI[1]	Swarm intelligence.
CPU [2]	Central processing unit.
LEDs[3]	Light emitting diode.
AVR[4]	Automatic voltage regulator.
RF [5]	Radio frequency.
DC[6]	Direct current.
AI[7]	Artificial intelligence.
SLAM[8]	Simultaneous localization and mapping.
AR[9]	Augmented reality.
3D[10]	3 dimensions.
ROS[11]	Robot operating system.
OS[12]	Operating system.
IOT [13]	Internet of things.
PSO[14]	Particle swarm optimization.
GNSS[15]	Global navigation satellite system.
GPS[16]	Global positioning system.
TCP/IP[17]	Transmission control protocol/internet protocol.
RFID[18]	Radio –frequency identification.
Wi-Fi[19]	Wireless fidelity.
WAN[20]	Wide area network.
AC[21]	Alternating current.
LTE-A[22]	Long term evolution-advanced.
BLE[23]	Bluetooth low-energy.
LAN[24]	Local area network.
NFC[25]	Near field communication.
LORa WAN[26]	Long range wide area network.
MQIT[27]	Message queue telemetry transport.
TSL/SSL[28]	Transport layer security/stands for secure sockets layer.
MCU[29]	Micro controller unit.
HVAC[30]	Heating, ventilation and air condition.
VR[31]	Virtual reality.
PI[32]	Pulse induction.
PCB[33]	Printed circuit board.
IR[34]	Infra-red.
GPR[35]	Ground penetrating radar.
AP[36]	Access point.
TGS[37]	Tesoro golden sabre.
TGSL[38]	Tesoro golden sabre light.
GB[39]	Giga byte.
DVM[40]	Digital voltmeter.

GM[41]	Ground mapping.
PC[42]	Personal computer.
IMU[43]	Inertial measurement units.
MEMS[44]	Micro electro mechanical system.
LIP[45]	Lithium poly.
GUI[46]	Graphical user interface.
UAV[47]	Unmanned aerial vehicle.
SAR[48]	Search and rescue.

Abstract

Swarm robotics is one of the most fascinating and new research areas of recent decades, and one of the grand challenges of robotics is the design of swarm robots that are self-sufficient. This can be crucial for robots exposed to environments that are unstructured or not easily accessible for a human operator, such as the inside of a blood vessel, a collapsed building, the deep sea, or the surface of another planet. The key factors in designing and building a group of swarm robots are cost and miniaturization with robustness, flexibility, and scalability. In robotics intelligence, self-assembly and self-reconfigurable are among the most important characteristics as they can add additional capabilities and functionality to swarm robots. Simulation and model design for swarm robotics is highly complex and expensive, especially when attempting to model the behavior of large swarm robot groups. Advantage of swarm robots to save time on task performance and reduce errors compared to using a single robot.

The main aim of the project discover landmines in an unknown environment and draw maps of their locations and try to get rid of it. Given the fact that there is no single sensor to determine the location of any type of landmines There are two possible solutions: one is the installation of all the necessary sensors on one vehicle, while the less conventional use is the use of several vehicles, each equipped with one sensor since the use of one sensor calls for a smaller, lighter and simpler robot, while completing the task requires cooperation between several units, which in turn poses issues of information exchange.

The value of our Project is Reduce human losses and benefit from the huge natural resources development revenues in those areas that are currently held hostage by mines, exploitation of mineral, petroleum resources, and a number of lands suitable for agricultural development and reclamation of arable land. And helping oil companies, real estate and tourism companies and national projects that want this land

1. Introduction of Swarm Robotics

Sometimes it is impossible to complete a task by a single person or it becomes quite difficult to that person to complete the work. In such cases, there is need of a team or group of members that can collaboratively work and make the work of the person or the user very much easy.

1.1. Swarm Definition

Swarm intelligence (SI) [(1)] is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems. The concept of SWARM ROBOTICS is based on this basis of grouping of multiple robots or devices and perform the desired task. Swarm robotics is a new approach to the coordination of multi-robot systems which consist of large numbers of mostly simple physical robots. This approach emerged on the field of artificial swarm intelligence, as well as the biological studies of insects, ants and other fields in nature, where swarm behavior occurs.

1.1.1 Advantages of SI Robotic

1) Scalable: The control architecture of each robot is same, no matter the number of robots.
2) Flexible: The robots may be inserted or deleted to/from the environment, no requirement for any change in the task operation.
3) Robust: Not only due to unit redundancy but also through minimalist unit design.

We introduce fundamental concepts of swarm robotics and get a little overview. Swarm robotics is a complex approach that requires an understanding of how to define swarm behavior, whether there is a minimum size of swarms, we define self-organization and develop an understanding of feedback systems. Swarms do not necessarily need to be homogeneous but can consist of different types of robots making them heterogeneous. We also discuss the interaction of robot swarms with human beings as a factor.

A wide range of networked systems exhibit emergent behavior. In nature, for example, flocks of birds, schools of fish, and swarms of bees all develop cohesive global behavior from purely local interactions. The goal of our research is to develop the tools necessary to design local control, communication, and estimation laws for individual agents that yield a desired group behavior.

1.1.2 The Function of Swarm Robotics

➢ **There are three main ways for swarm robots to function:**

1) Singular Intelligence

Robots are all controlled by one system. The robots themselves aren't intelligent independently, they are just responding to the requests of the controlling intelligence.

This is analogous to you remote controlling a robot. That robot has no intelligence past basic motion generation given some signal.

Despite this, you can tell it what to do to perform tasks. The swarm aspect of it is then just controlling multiple robots at the same time.

2) Robot-by-Robot Intelligence

Robots have independent intelligence and often have limited interaction with each other. These are quite analogous to ants. Ants have tasks that they desire to perform and they have limited communication with each other along with their own intelligence to perform this function.

3) A combination of the previous two

Robots have some general intelligence, perhaps a singular intelligence determining tasks and each individual robot takes that tasks and use their own intelligence and talking with each other to perform it.

1.1.3 Applications of Swarm Robotics

1) **Fire Fighting Robot**: a robot that can put the fire out of any fired places and this robot disguise by going in the hard places that people can't reach.
2) **Security swarm of robots:** This project has mount of robotics and every robot in a specific place and all of them are reached together by AI and draw a map for the place where they found in .they combine their all maps and every one of these robotics see the combination map. When there is a danger then it treat with that danger and get help of the rest.
3) **Fighter:** Robots attack certain targets alone and is estimated to work alone.
4) **Monitoring and tracking of cars:** monitoring traffic and tracking cars infringing or stolen or the time of crimes and a friend by processing pictures that they save the shape of the car and continue to move and see and at the same time connected to the police station.
5) **Ambulance Flying:** The words of the pilot in which all the medical devices to require what is left in the incident (to be located in the ambulance) uniforms and heart rate and heart rate and sugar, and in the first aid tools and arrived by a specialist doctor under the situation and people behave with the patient or I am exposed to the accident to the extent that the ambulance arrives.

1.2 Embedded System .

An embedded system is some combination of computer hardware and software, integrated to perform a particular function either fixed in capability or programmable, that is designed for a specific function or for specific functions within a larger system It uses a

Microcontroller/Microprocessor to perform a single job. It is a stand-alone device with no operating system.

1.2.1 Embedded Hardware

In Hardware is the fundamental resource of any embedded device and choosing a particular component depends on the requirement and specification of the designer.

The core of any embedded target is the electronic hardware – which resides on a Printed Circuit Board. The embedded development board is divided into five modules. They are Processor, Memory, Input devices, Output devices and Bus controllers.

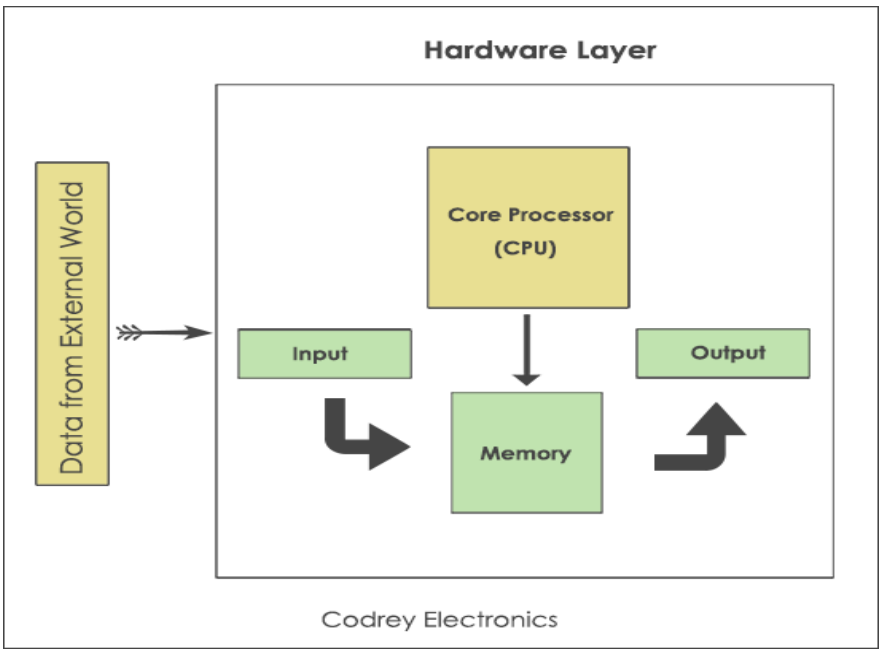


Figure 1.1 Hardware Components of an Embedded System

1.2.2 Embedded System Software

Software components are essential building blocks of embedded systems. Embedded C is the most widely used languages for embedded systems.

- **Why C for Embedded Systems?**
  - Performance wise better.
  - Easy to use.
  - More reliable.
  - Directly interacts with the hardware.

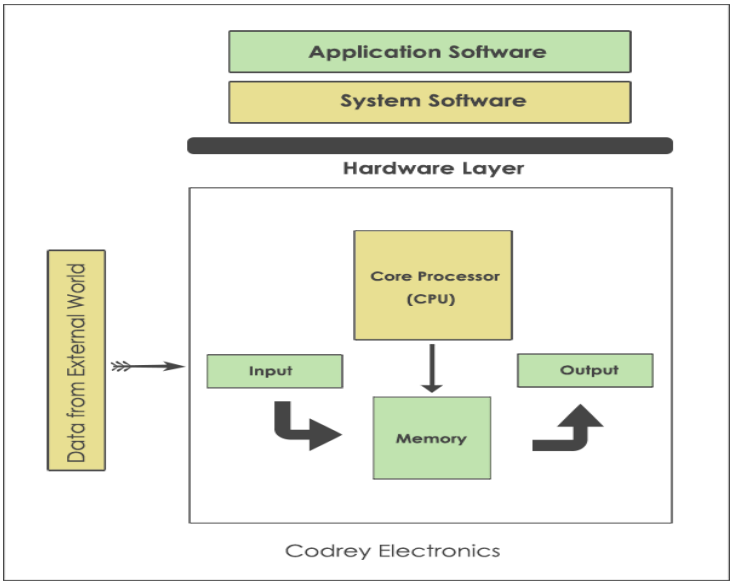


Figure 1.2 Software Components of an Embedded System

1.2.3 Steps to make your Embedded Product

Step1: Understand the requirements
First of all, you need to know and understand the end user specifications.

Step 2: Examine
Analyze the components (software and hardware) required to make the product.

Step 3: Design
It is the most critical phase of the development cycle. The developer needs to develop the embedded hardware and software individually and integrate both.

Step 4: Develop
The Programmer develops the Prototype using available hardware and software tools to match the customer specifications.

Step 5: Test
The developer examines the application by running test cases to prove the possible potential of the prototype.

Step 6: Deploy
After testing the product, the developer checks the result in a real environment to realize the Proof of Concept.

Step 7: Support and Upgrade
Depending on the user requirement, support and upgrade have to be provided to add new features at regular intervals.

1.2.4 Some Uses Embedded System in Project

We can control any circuit or machine by interfacing with 8051 microcontroller which is programmed with appropriate commands. Microcontroller has all or most of these features built-in to a single chip, so it doesn't need a motherboard and many components, LEDs [3] for example, can be connected directly to the AVR. If you tried this with a microprocessor.

1) Electrical and electronics project circuits or machines are developed using advanced technologies.
2) These circuits and machines are intended to control easily using very advanced programming techniques
3) The programming techniques are used to send appropriate control signals for machines or circuits in order to control
4) This can be achieved using various types of microprocessors and microcontrollers.
5) there are many types of microcontrollers are existing but, 8051 microcontroller is typically used because of its advantages

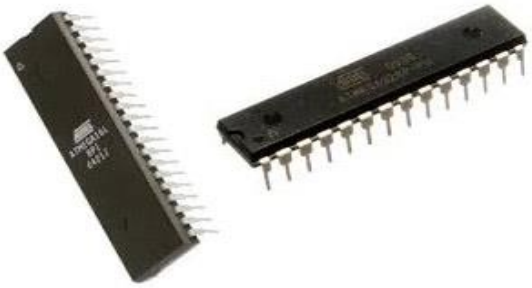


Figure 1.3 AVR Microcontroller

1.2.4.1 Metal Detector

The main aim of this Metal Detector project is to detect the metal in the surroundings and alert the siren when the metal is found. This Embedded System project has the application in mines. People used to dig the ground to locate the mines underneath the ground. Using this metal detector, the mines can be found without digging.

The metal detector contains the metal detecting circuit, microcontroller and the buzzer. This metal detecting circuit comprises of RF [5] oscillators and the operation of the circuit is based on a super heterodyning principle which is commonly used in superheat receivers. The output of the oscillator is a pulsating DC [6] which is passed through a low-pass filter. It is then passed to the amplifier for amplification of the signal.

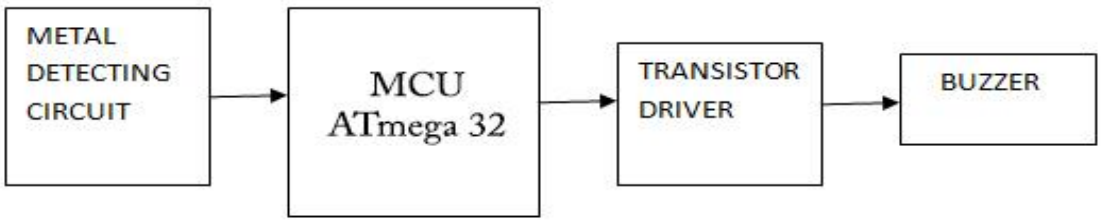


Figure 1.4 Simple Connection between Metal Detector and Microcontroller

1.2.4.2 H- bridge

AVR [4] is a quadruple H- bridge motor driver, as the name suggests it used to drive the DC motors. This IC works based on the concept of H- Bridge. H-bridge is a circuit which allows the voltage in either direction to control the motor direction.

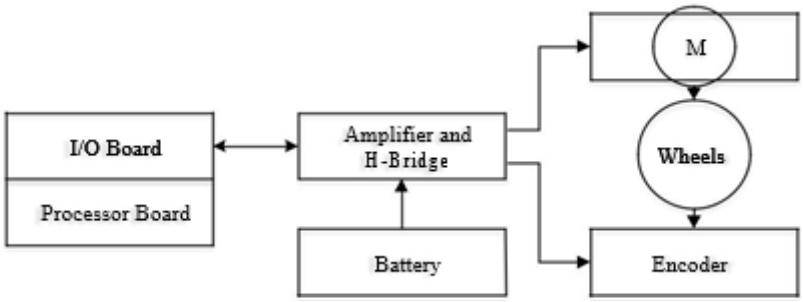


Figure 1.5 Motor control block diagram

1.2.4.3 Arm

The arm control by robotics is very popular in the world of robotics. The essential part of the robotic arm is a programmable micro controller based brick capable of driving basically three stepper motors to make the Robotic arm, having three stepper motors, to interface with the In-Development of a Microcontroller Based Robotic Arm Atmel 8051-basedmicro-controller. It provides more interfaces to the outside world and has larger Memory to store many programs.

1.3 Introduction of AI (ARTIFICIAL INTELLIGENCE)

One of the key feature that distinguish us, humans, from everything else in the world is intelligence. This ability to understand, apply knowledge and improve skills has played significant role in our evolution and establishing human civilization.

Artificial intelligence (AI) [7], sometimes called machine intelligence, in computer science AI research is defined as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. Colloquially, the term "artificial intelligence" is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving".

For instance, optical character recognition is frequently excluded from "artificial intelligence", having become a routine technology. Modern machine capabilities generally classified as AI include successfully understanding human speech, competing at the highest level in strategic game systems (such as chess and Go), autonomously operating cars, and intelligent routing in content delivery networks and military simulations.

1.3.1 Types of AI

1) Reactive machines.
2) Limited memory.
3) Theory of mind.
4) Self-awareness.

1.3.2 Goals of AI

- **To Create Expert Systems** − the systems which exhibit intelligent behavior, learn, demonstrate, explain, and advice its users.
- **To Implement Human Intelligence in Machines** − Creating systems that understand, think, learn, and behave like humans.

1.3.3 Programming Without and With AI

Table 1.1 the programming without and with AI is different in following ways

Programming Without AI	**Programming With AI**
A computer program without AI can answer the **specific** questions it is meant to solve.	A computer program with AI can answer the **generic** questions it is meant to solve.
Modification in the program leads to change in its structure.	AI programs can absorb new modifications by putting highly independent pieces of information together. Hence you can modify even a minute piece of information of program without affecting its structure.
Modification is not quick and easy. It may lead to affecting the program adversely.	Quick and Easy program modification

1.3.4 Components of Intelligence

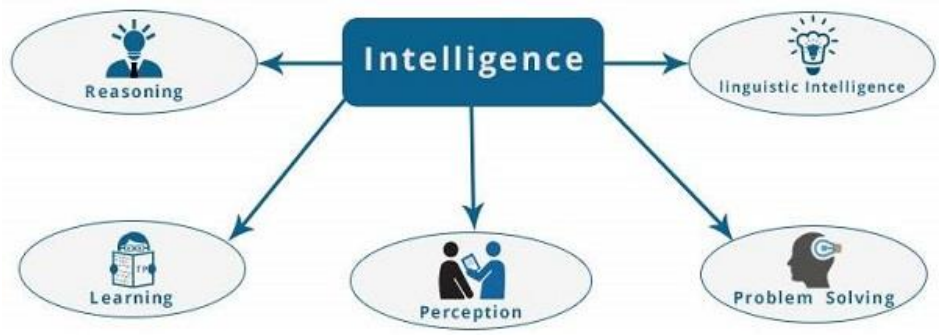


Figure 1.6 Components of Intelligence

1.3.4.1 Reasoning

It is the set of processes that enables us to provide basis for judgment, making decisions, and prediction.

1.3.4.2 Learning

It is the activity of gaining knowledge or skill by studying, practicing, being taught, or experiencing something. Learning enhances the awareness of the subjects of the study. The ability of learning is possessed by humans, some animals, and AI-enabled systems.

- **Auditory Learning**

  It is learning by listening and hearing. For example, students listening to recorded audio lectures.

- **Episodic Learning**

  To learn by remembering sequences of events that one has witnessed or experienced. This is linear and orderly.

- **Observational Learning**

  To learn by watching and imitating others. For example, child tries to learn by mimicking her parent.

- **Perceptual Learning**

  It is learning to recognize stimuli that one has seen before. For example, identifying and classifying objects and situations.

- **Spatial Learning**

  It is learning through visual stimuli such as images, colors, maps, etc. For Example, A person can create roadmap in mind before actually following the road.

- **Stimulus-Response Learning**

  It is learning to perform a particular behavior when a certain stimulus is present. For example, a dog raises its ear on hearing doorbell.

1.3.4.3 Problem Solving

It is the process in which one perceives and tries to arrive at a desired solution from a present situation by taking some path, which is blocked by known or unknown hurdles. Problem solving also includes decision making, which is the process of selecting the best suitable alternative out of multiple alternatives to reach the desired goal are available.

1.3.4.4 Perception

It is the process of acquiring, interpreting, selecting, and organizing sensory information. Perception presumes sensing. In humans, perception is aided by sensory organs. In the domain of AI, perception mechanism puts the data acquired by the sensors together in a meaningful manner.

1.3.4.5 Linguistic Intelligence

It is one's ability to use, comprehend, speak, and write the verbal and written language. It is important in interpersonal communication.

1.3.5 AI in our project

Intelligent Robots − Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment.

- **Image processing**

1.3.5.1 Image Definition

Signal processing is a discipline in electrical engineering and in mathematics that deals with analysis and processing of analog and digital signals, and deals with storing, filtering, and other operations on signals. These signals include transmission signals, sound or voice signals, image signals, and other signals etc.

Out of all these signals, the field that deals with the type of signals for which the input is an image and the output is also an image is done in image processing. As it name suggests, it deals with the processing on images.

It can be further divided into analog image processing and digital image processing. We used image processing to detect the metal detector objects by its dimensions (x, y) to give orders for other components to perform.

❖ SLAM Technique:-

1.3.5.2 SLAM Definition

SLAM [8] (Simultaneous Localization and Mapping) is a technology which understands the physical world through points. This makes it possible for AR [9] applications to recognize 3D [10] Objects & Scenes, as well to instantly Track the world, and to overlay digital interactive augmentations.

Some of which do not include a camera at all. SLAM Visual is a specific type of SLAM system that takes advantage of 3D vision to perform location and mapping functions when neither the environment nor the location of the sensor are known. Visual SLAM comes in different formats, but the general concept works the same in all SLAM optical systems.

❖ ROS System:-

1.3.5.3 ROS Definition

The Robot Operating System (ROS) [11] is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

ROS is a fully open source robotics development environment. The ROS is a shortcut to the Android operating system, a Meta operating system or a framework that works on many other operating systems such as Linux. ROS is in fact a meta-operating system, something between an operating system and middleware.

1.3.5.4 Programming with ROS

ROS is language-independent. At this time, three main libraries have been defined for ROS, making it possible to program ROS in Python, Lisp or C++. In addition to these three libraries, two experimental libraries are offered, making it possible to program ROS in Java or Lua.

1.3.5.5 The General Organization of ROS

ROS philosophy can be summarized in the following five main principles:

- Peer-to-Peer
- Tools-based (microkernel)
- Multi-language

- Thin
- Free and open source.

1.3.5.6 Basic Notions in ROS

The basic principle of a robot operating system is to run a great number of executable in parallel that need to be able to exchange data synchronously or asynchronously.
For example, a robotics OS [12] needs to query robot sensors at a set frequency (Ultrasound or infra-red distance sensor, pressure sensor, temperature sensor, gyroscope, accelerometer, cameras, microphone, etc.), retrieve this data, process it (Carry out what is known as a data merge), pass it to processing algorithms (speech processing, artificial vision, SLAM – simultaneous localization and mapping, etc.) and lastly control the motors in return.
This whole process is carried out continuously and in parallel. Moreover, the robotics OS needs to manage contention to ensure efficient access to robot resource.

1.4 Project organization

And here we will discuss the project organization for each chapter

Chapter 2 discusses artificial intelligence and uses of each technique that we have used in our project, and how we used these techniques as (Image processing, SLAM, ROS).

Chapter 3 discusses swarm robotics idea and its uses and how we used it in our project, advantage and disadvantage of swarm robotics, and IOT explanation.

Chapter 4 discusses metal detector and mining detection, whole idea of project, represents the hardware and software of project.

Chapter 5 discusses features that we used in project and future work that can be done with project.

2.1 Definition of image

An image is defined as a two-dimensional function(x, y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x, y) is called the intensity of that image at that point. When x, y, and amplitude values of F are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns.

Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location. These elements are referred to as picture elements, image elements, and pixels. A Pixel is most widely used to denote the elements of a Digital Image.

2.1.1 Types of image

1) **BINARY IMAGE**– The binary image as its name suggests, contain only two pixel elements i.e. 0 & 1, where 0 refers to black and 1 refers to white. This image is also known as Monochrome.

2) **BLACK AND WHITE IMAGE**– The image which consist of only black and white color is called BLACK AND WHITE IMAGE.

3) **8 bit COLOR FORMAT**– It is the most famous image format. It has 256 different shades of colors in it and commonly known as Grayscale Image. In this format, 0 stands for Black, and 255 stands for white, and 127 stands for gray.

4) **16 bit COLOR FORMAT**– It is a color image format. It has 65,536 different colors in it. It is also known as High Color Format. In this format the distribution of color is not as same as Grayscale image. A 16 bit format is actually divided into three further formats which are Red, Green and Blue. That famous RGB format.

2.1.2 Image as a matrix

As we know, images are represented in rows and columns we have the following syntax in which images are represented

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

Figure 2.1 Image as matrix

2.2 Image processing

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image.

- **Image processing basically includes the following three steps:**

1) Importing the image with optical scanner or by digital photography.
2) Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.
3) Output is the last stage in which result can be altered image or report that is based on image analysis.

- **There are two types of methods used for image processing namely:**

1) **Analog image processing:** can be used for the hard copies like printouts and photographs.
2) **Digital image processing:** techniques help in manipulation of the digital images by using computers.

The three general phases that all types of data have to undergo while using digital technique are Pre-processing, Enhancement, Display and Information Extraction.

2.3 Applications of image processing

1) **GUI**: how to show image and to fix it.
2) **Core operation**: to make enhancement to the picture to get high resolution, any picture has a filter.
3) **Image processing**: its mathematical operations illustrate how to sum two pictures or to difference two pictures.
4) **Machine learning**: how to make the machine take an action according to input database.
5) **Object tracking**: is to follow an object in its track.
6) **Feature detection**: it made description to the input database.

2.4 The best programming languages for image processing

1) Python
2) Java
3) C/C#/C++
4) Matlab

2.5 Python Language

2.5.1 Python Language Introduction

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Python 3.0 was the next version and was released in December of 2008 (the latest version of Python is 3.6.4). Although Python 2 and 3 are similar there are subtle differences. Perhaps most noticeably is the way the print statement works, as in Python 3.0 the print statement has been replaced with a print () function.

2.5.1.1 Applications for Python

- **Python in Artificial Intelligence (AI)**

  Python is fast, scalable, robust and platform agnostic. These advantages make Python a perfect fit for AI. Using Python, you can replicate every idea with a few lines of code which is not possible with other languages. It provides libraries such as '*Keras'* and '*Tensor Flow'* that bring out machine learning functionalities. Also, the libraries provided by Python like – 'Scik it learn' is highly used in AI algorithms. Scikit is a free machine learning library featuring various regression, classification and clustering algorithms. Above all, Python is a free Open-Source language with a good community support. All these reasons combined makes learning Python an easy choice over other languages for AI applications.

- **Python in Big Data**

  Python is extensively used for analyzing huge chunk of data and extracting useful insights to drive businesses. Apart from its simplicity which is a great boon, Python also has an exhaustive set of Data Processing libraries. This makes Python a no-brainer for any organization looking to work with data. Python is fast and highly scalable.

These features help Python in generating insights in real-time environments and making it one of the preferred languages for Big Data.

- **Python in Data Science**

Learning Python is of great importance to Data Science professionals. Ever since the introduction of numerical engines of Python like 'Pandas' and 'NumPy', academic scholars and researchers have switched to Python from MATLAB.

- **Python in Testing Frameworks**

Testing is another industry that is seeing an increase use of Python language. Python is great for validating ideas or products for established companies. Python has many built-in testing frameworks that handles debugging and offers fast workflows and execution. Testing tools like 'Unit test' and 'Nose test' makes testing easier and hassle-free. It supports cross-platform & cross-browser testing with different frameworks such as '*PyTest'* and '*Robot'*. Testing can be a challenging task for organization. With Python the process is made a lot simpler.

- **Python in Web Development**

The simple fact that Python lets you build a lot more with a fewer lines of code makes it stand out. This helps in building prototypes efficiently and during debugging these codes. The provided by Python is a boon for all developers as it can be used to create dynamic and highly secure web apps. By learning Python, you can also perform web scraping, allowing you to fetch details from other websites. Apps such as Instagram, Bit Bucket, and Interest are built on frameworks like these.

2.6 Library of Python

While The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

2.6.1 NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases. NumPy is licensed under the BSD license, enabling reuse with few restrictions.

2.6.2 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. OpenCV introduces a new set of tutorials which will guide you through various functions available in OpenCV-Python. This guide is mainly focused on OpenCV 3.x version.

2.7 GUI Features in OpenCV

(Graphic user interface) is used to learn to load an image, display it and save it back, Learn to play videos, capture videos from Camera and write it as a video.

2.7.1 Getting Started with Images

1) Here, you will learn how to read an image, how to display it.
2) how to save it back
3) How to learn these functions : cv2.imread() ,cv2.imshow(), cv2.imwrite()
4) Optionally, you will learn how to display images

➢ Read an image Using OpenCV:

1) cv2.IMREAD_COLOR: Loads a color image. Any transparency of image.
2) Will be neglected. It is the default flag.
3) cv2.IMREAD_GRAYSCALE: Loads image in grayscale mode.
4) cv2.IMREAD_UNCHANGED : Loads image as such including alpha channel

```
import numpy as np
import cv2

# Load an color image in grayscale
img = cv2.imread('messi5.jpg',0)
```

➢ Display an image:

Use the functioncv2.imshow () to display an image in a window. The window automatically fits to the image size

```
cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

➢ Write an image:

Use the functioncv2.imwrite () to save an image. First argument is the file name, second argument is the image you want to save.

```
cv2.imwrite('messigray.png',img)
```

This program loads an image in gray scale, displays it, save the image if you press ‘s’ and exit, or simply exit without saving if you press ESC key.

```
import numpy as np
import cv2

img = cv2.imread('messi5.jpg',0)
cv2.imshow('image',img)
k = cv2.waitKey(0)
if k == 27:         # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'): # wait for 's' key to save and exit
    cv2.imwrite('messigray.png',img)
    cv2.destroyAllWindows()
```

2.7.2 Getting Started with Videos

1) Learn to read video, display video and save video.
2) Learn to capture from Camera and display it.
3) You will learn these functions : cv2.VideoCapture(),cv2.VideoWriter()

➢ Capture Video from Camera:

Often, we have to capture live stream with camera. OpenCV provides a very simple interface to this. Let’s capture a video from the camera (I am using the in-built webcam of my laptop), convert it into grayscale video and display it. Just a simple task to get started.

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

Figure 2.2 Output Capture Video from Camera

➢ Saving a Video :

So we capture a video, process it frame-by-frame and we want to save that video. For images, it is very simple, just use cv2.imwrite ()

Below code capture from a Camera, flip every frame in vertical direction and saves it.

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('output.avi',fourcc, 20.0, (640,480))

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)

        cv2.imshow('frame',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

2.7.3 Object Tracking

To make object tracking we need to know the difference between the back ground and the current frame

1) **The back ground:** is a picture which contain all details except the object that I need to detect.
2) **The current frame:** it is the back ground + the object that I want to detect.
3) Make subtraction between the back ground and the current frame.
4) Use suitable filter.

2.7.3.1 Blob Detection

It is a technique which make scanning on all the pixels of the image after passing the filter, when it find color changing it detect the object and give it a color , so it can calculate 'containing rectangle'.

2.7.3.2 Object detection

It is a group of blocks which make weak description to details of the image until it describe the object perfectly.

➢ **Edge features:**

1) Scanning on pixels and if the color changed it recognize that it is the beginning of an edge.
2) If we want to check if there is an edge or not , we use “the Mask”
3) The mask can be line vertical or horizontal line or curve.

2.8 Core Operation

Means applying filters—an *image filter* is a piece of software that examines an input image pixel by pixel and algorithmically applies some effect in order to create an output image.

2.8.1 Basic operation on image

➢ **Basic operation**

Learn to:

1) Access pixel values and modify them.
2) Access image properties.
3) Setting Region of Image (ROI).
4) Splitting and merging images.

2.8.1.1 Arithmetic Operations on Images

➢ **Goal**

1) Learn several arithmetic operations on images like addition, subtraction, bitwise operations etc.
2) You will learn these functions: cv2.add (), cv2.addWeighted () etc.

➢ **Image Addition**

You can add two images by OpenCV function, cv2.add () or simply by numpy operation, res = img1 + img2.Both images should be of same depth and type, or second image can just be a scalar value.

2.8.1.2 Image Blending

This is also image addition, but different weights are given to images so that it gives a feeling of blending or transparency.
Images are added as per the equation below:

$$(x) = (1 - \alpha)\ 0(x) + \alpha f1\ (x)$$ [2]

By varying α from 0 → 1, you can perform a cool transition between one images to another.
Here I took two images to blend them together. First image is given a weight of 0.7 and second image is given 0.3.
cv2.addWeighted () applies following equation on the image.

$$dst = \alpha \cdot img1 + \beta \cdot img2 + \gamma$$ [3]

Here γ is taken as zero.

```
img1 = cv2.imread('ml.png')
img2 = cv2.imread('opencv_logo.jpg')

dst = cv2.addWeighted(img1,0.7,img2,0.3,0)

cv2.imshow('dst',dst)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Figure 2.3 Output of image blending

2.8.2 Mathematical tools in OpenCV

2.8.2.1 Image Thresholding

Here, the matter is straight forward. If pixel value is greater than a threshold value, it is assigned one value (may be white), else it is assigned another value (may be black). The function used is cv2.threshold. First argument is the source image, which should be a grayscale image. Second argument is the threshold value which is used to classify the pixel values. Third argument is the max Val which represents the value to be given if pixel value is more than (Sometimes less than) the threshold value.
OpenCV provides different styles of thresholding and it is decided by the fourth parameter of the function.

- **Different types are:**

1) cv2.THRESH_BINARY
2) cv2.THRESH_BINARY_INV
3) cv2.THRESH_TRUNC
4) cv2.THRESH_TOZERO
5) cv2.THRESH_TOZERO_INV

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('gradient.png',0)
ret,thresh1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)
ret,thresh3 = cv2.threshold(img,127,255,cv2.THRESH_TRUNC)
ret,thresh4 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO)
ret,thresh5 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO_INV)

titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]

for i in xrange(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()
```

t is given below :

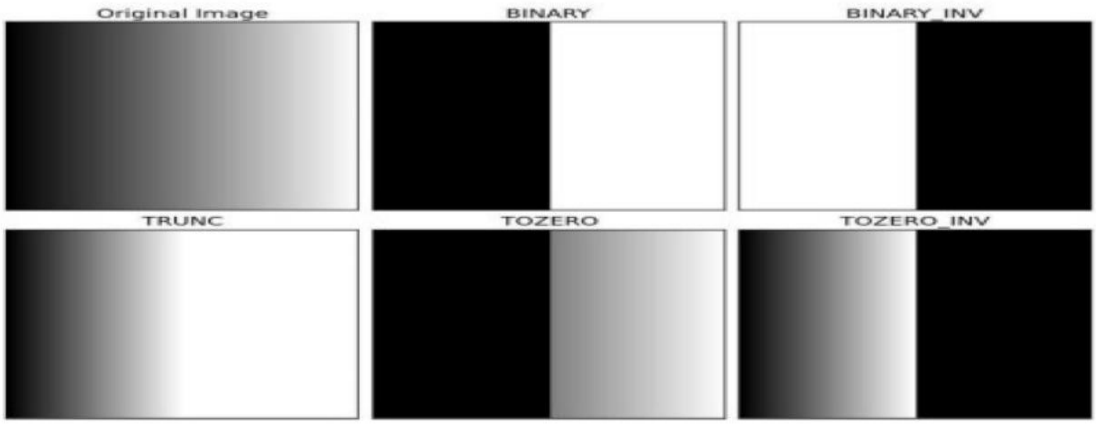


Figure 2.4 Image thresholding filter

2.8.2.2 Image thresholding with differentiation

```
import cv2
import numpy as np

class AVG: ##for average background##
        def __init__(self,Capture):
            self.Capture=Capture

        def Initial_Frame(self):
            self._, self.Frame=self.Capture.read()
            self.AVG=np.float32(self.Frame)

        def Get_AVG_IMAGE(self,ALPHA):
            self._,self.Frame=self.Capture.read()
            cv2.accumulateWeighted(self.Frame,self.AVG,ALPHA)
            self.Res=cv2.convertScaleAbs(self.AVG)
            return self.Frame ,self.Res


        def Get_Cap(self):
            return self.Capture



class Main:   ##create track bar we avg background
        def __init__(self):
            self.Background=AVG(cv2.VideoCapture(0))
            self.Cap=self.Background.Get_Cap()


        def nothing(self,args):
            pass
```

```
    def Image_Processing(self):
        alpha=cv2.getTrackbarPos('Alpha','Frame Image')
        alpha /=100000.0
        Frame,Avg=self.Background.Get_AVG_IMAGE(alpha)
        f_gray=cv2.cvtColor(Frame,cv2.COLOR_BGR2GRAY)
        Avg_gray=cv2.cvtColor(Avg,cv2.COLOR_BGR2GRAY)
        Diff=cv2.absdiff( f_gray, Avg_gray)
        ret,thresh=cv2.threshold(Diff ,50, 255, 0)
        return Avg,Frame,thresh

    def Release_GUI(self):
        cv2.destroyAllWindows()
        self.Cap.release

    def Show_GUI(self):
        Avg,Frame,thresh =self.Image_Processing()
        cv2.imshow("Threshold",thresh)
        cv2.imshow("Frame Image",Frame)
        cv2.imshow("Average Background",Avg)

    def Initial_GUI(self):
        cv2.namedWindow('Frame Image',flags=0)
        cv2.createTrackbar('Alpha','Frame Image',0,1000,self.nothing)

        self.Background.Initial_Frame()

    def Worker(self): ##betshaghl kol el functions elly katbnaha fo2
        self.Initial_GUI()
        while True:
            self.Show_GUI()
            key=cv2.waitKey(20)
            if key==27:
                break
        self.Release_GUI()

Program=Main()
Program.Worker()
```

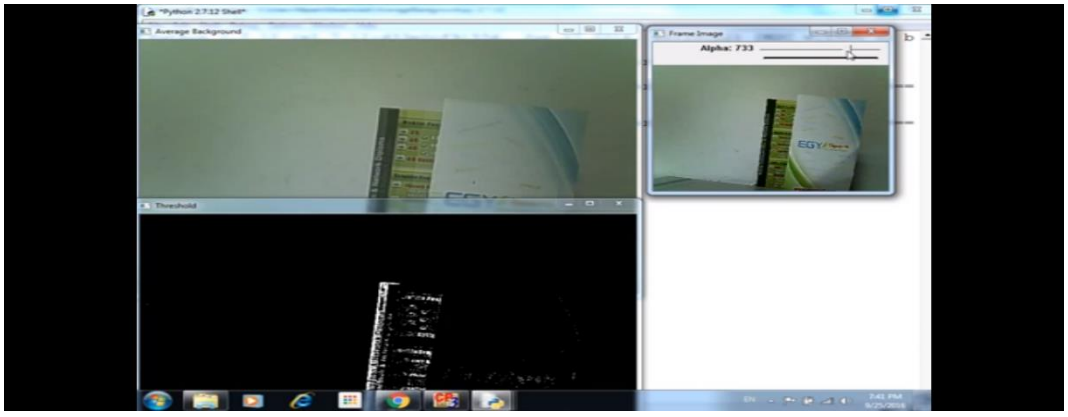


Figure 2.5 Output Image Thresholding with Differentiation

2.9 Object Detection

Swarm intelligence (SI) is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems. The concept of SWARM ROBOTICS is based on this basis of grouping of multiple robots or devices and perform the desired task. Swarm robotics is a new approach to the coordination of multi-robot systems which consist of large numbers of mostly simple physical robots. This approach emerged on the field of artificial swarm intelligence, as well as the biological studies of insects, ants and other fields in nature, where swarm behavior occurs.

2.9.1 Face detection using haar-cascades

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes
Etc. you can use OpenCV to create one. Its full details are given here: Cascade Classifier Training.
Here we will deal with detection. OpenCV already contains many pre-trained classifiers for face, eyes, smile etc.
Those XML files are stored in OpenCV/data/haarcascades/ folder. Let's create face and eye detector with
OpenCV.

```
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

img = cv2.imread('sachin.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

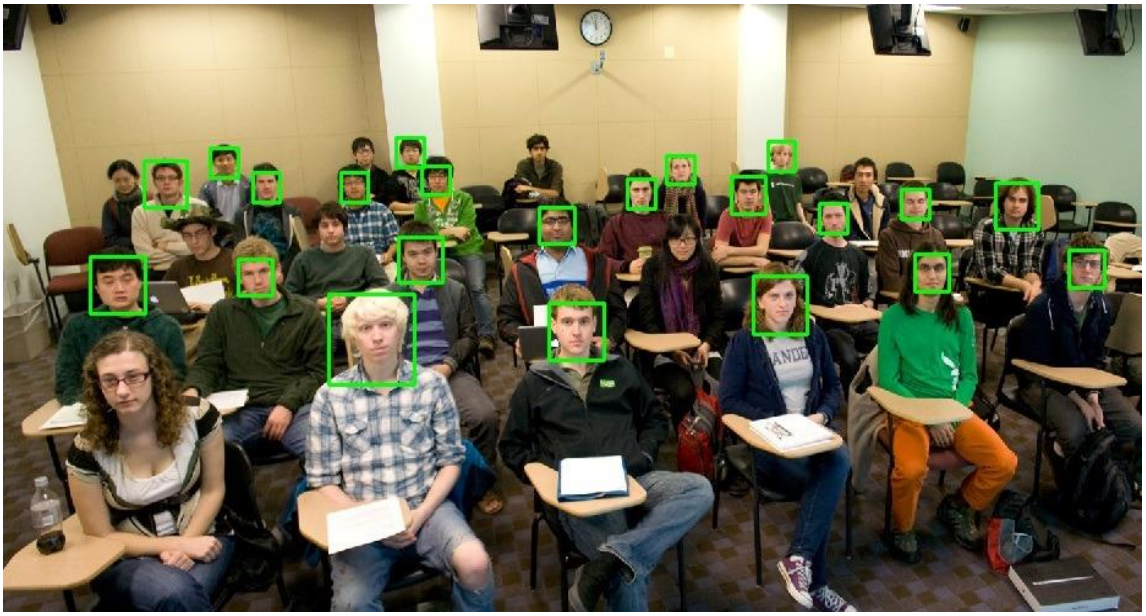


Figure 2.6 Output Face detection

2.9.2 Object detection using Kinect

Object detection is the process of finding instances of real-world objects such as faces, bicycles, and buildings in images or videos. Object detection algorithms typically use extracted features and learning algorithms to recognize instances of an object category. It is commonly used in applications such as image retrieval, security, surveillance, and automated vehicle parking systems.

It is one of the fundamental problems in computer vision and a lot of techniques have come up to solve it, including Feature-based object detection, Viola-Jones object detection, SVM classification with histograms of oriented gradients (HOG) features and Image segmentation and blob analysis. Other methods for detecting objects with computer vision include using gradient-based, derivative-based, and template matching approaches.

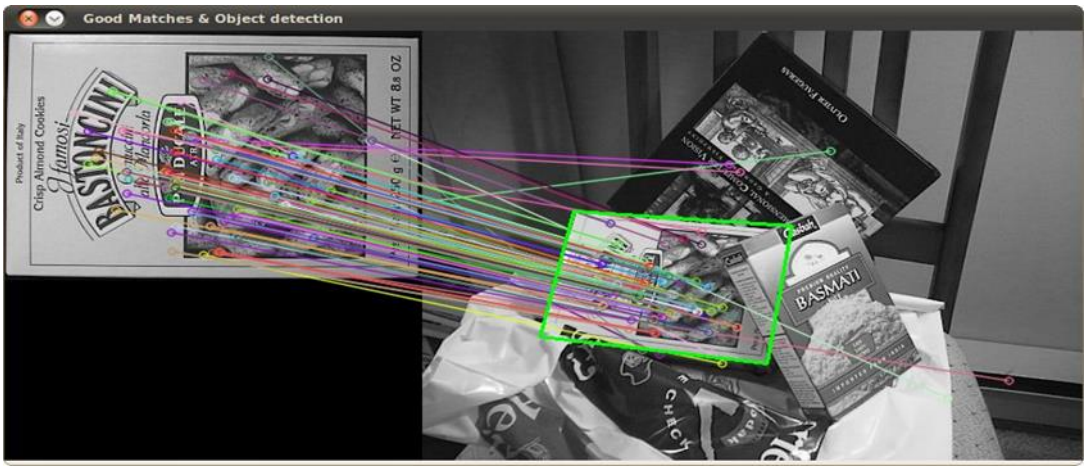


Figure 2.7 Detecting a reference object (left) cluttered scenes (right)

2.9.2.1 Kinect

The Kinect is a tool that provides full 3D images comprised of 640x480 RGB plus depth [27]. It is made of a RGB camera, an infra-red depth sensor, a multi-array microphone and a 3 axis (XYZ) accelerometer

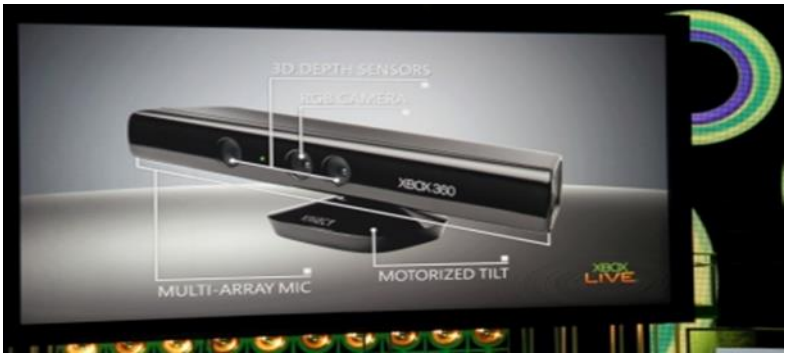


Figure 2.8 the Kinect

2.9.2.2 SURF algorithm

SURF stands for Speeded up Robust Features and is an algorithm which extracts some unique key points and descriptors from an image. Details on the algorithm can be found here and on its OpenCV implementation here and here. A set of SURF key points and descriptors can be extracted from an image and then used later to detect the same image. SURF uses an intermediate image representation called Integral Image, which is computed from the input image and is used to speed up the calculations in any rectangular area. It is formed by summing up the pixel values of the x, y co-ordinates from origin to the end of the image. This makes computation time invariant to change in size and is particularly useful while encountering large images. The SURF detector is based on the determinant of the Hessian matrix. The SURF descriptor describes how pixel intensities are distributed within a scale dependent neighborhood of each interest point detected by Fast Hessian.

Object detection using SURF is scale and rotation invariant which makes it very powerful. Also it doesn't require long and tedious training as in case of using cascaded Haar classifier based detection. The detection time of SURF is a little longer than Haar, but it doesn't make much problem in most situations if takes some tens of millisecond more for detection. Since this method is rotation invariant, it is possible to successfully detect objects in any orientation, case where the Haar classifier fails miserably.

Figure 2.9 Feature key points are represented by the circle centers

➢ **Short description of what SURF does:**

- Find interest points in the image using Hessian matrices
- Determine the orientation of these points
- Use basic Haar wavelets in a suitably oriented square region around the interest points to find intensity gradients in the X and Y directions. As the square region is divided into 16 squares for this, and each such sub-square yields 4 features, the SURF descriptor for every interest point is 64 dimensional.
- For a description of the 4 features, please refer the paper – they are basically sums of gradient changes.

Surf is a non-free algorithm. Other feature detector algorithms implemented in OpenCV include SIFT, FAST, BRIEF, ORB as listed here.

2.9.2.3 Fast Library for Approximate Nearest Neighbors

Feature Matching with FLANN (Fast Library for Approximate Nearest Neighbors)

After unique key points and descriptors are extracted, from both images object and scene, a matching must be done. OpenCV has two implemented matching strategies, the Brute-Force Matcher and FLANN Matcher as explained here and here.

The Brute-Force matcher is simple and it takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

The FLANN library contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features, and it works faster than Brute-Force Matcher for large datasets.

2.9.2.4 The Point Cloud Library

The Point Cloud Library (PCL) is a library developed by Rusu et al. (2010) who define this library as "one of our most recent initiatives in the areas of point cloud perception" and they say that the goal of their work is to "provide support for all the common 3D building blocks that applications need".

2.9.2.5 ROS and SLAM

They will be explained in detail in this chapter.

2.9.2.6 OpenCV

To get the code running with OpenCV 3.1 these libraries must be included in the path.

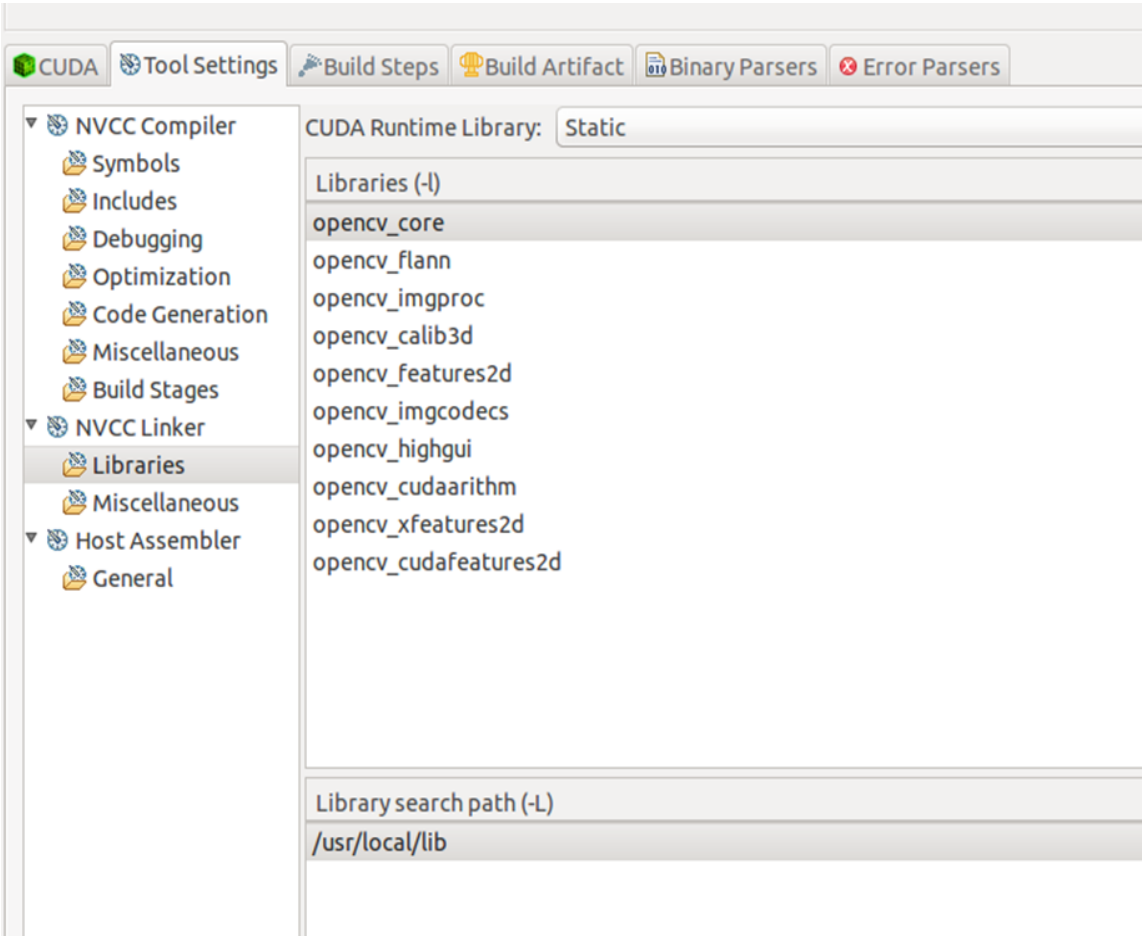


Figure 2.10 OpeanCV libraries

- **OpenCV CPU**

This first implementation is using only CPU [2] only methods of OpenCV 3.X.

- **OpenCV GPU**

This second implementation is using both GPU and CPU methods of OpenCV 3.X.

- **Localization**

Localization is done after key points and feature descriptors are extracted from both object and scene images. It searches for the right position, orientation and scale of the object in the scene based on the good matches. This method also draws lines between matching good key points and a square where and if the object was detected. For finding the best translation matrix Random sample consensus (RANSAC) is used.

- **Execution times of the 2 implementations**

Only the time spent extracting the key points and feature descriptors and the time for extracting the matches are taken in account. The time for loading images into CPU and GPU memories is ignored.

In the console can be seen that the GPU version finishes in about 20 ms for images not bigger than 800×600 which means the GPU implementation can run in real-time. The CPU version finishes in about 200 ms and could process only a few frames per second.

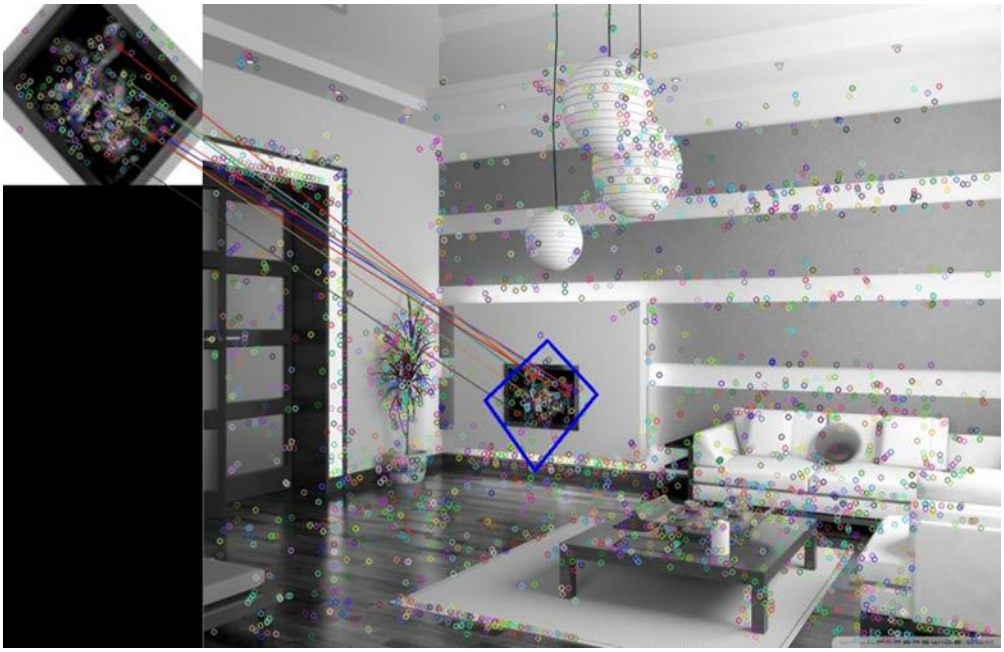


Figure 2.11 Correctly found object

2.9.3 Object detection using Stereo Vision Camera

Stereo vision camera is widely used in depth detection and 3D reconstruction on a varies of places, such as UAVs and other robotics. In this project, I build a stereo vision camera system using two USB cameras which work with v4l2 standards. The project contains

the hardware built procedure, camera optical axis alignment method, image rectification procedure, and the real time depth map generation method.

2.9.3.1 Theory of stereo vision

If the images of an object are captured from two different points of view, the 3D information can be derived from the disparity of the two images. We use a pinhole camera model to illustrate the theory of stereovision. Figure one is a stereovision system with two parallel cameras. The object point P is projected on the image plane of the two cameras. For left and right cameras, the position of image points of P is different. The xl is the distance between left camera axis and image of P on left camera image plane, and xr is the distance between the optical axis of right camera and image point of P on right camera image plane. f is the focal length of the two cameras. The distance between two optical axes is called baseline (b). It is notable that the length of xl and xr are different. The boxes equations are obtained from similar triangles. From these relations, the location of the 3D points in object space can be derived.

1) $z = (f \cdot b) / (xl - xr) = (f \cdot b) / d$
2) $x = x\,l \cdot (z/f$
3) $y = y\,l \cdot (z/f)$

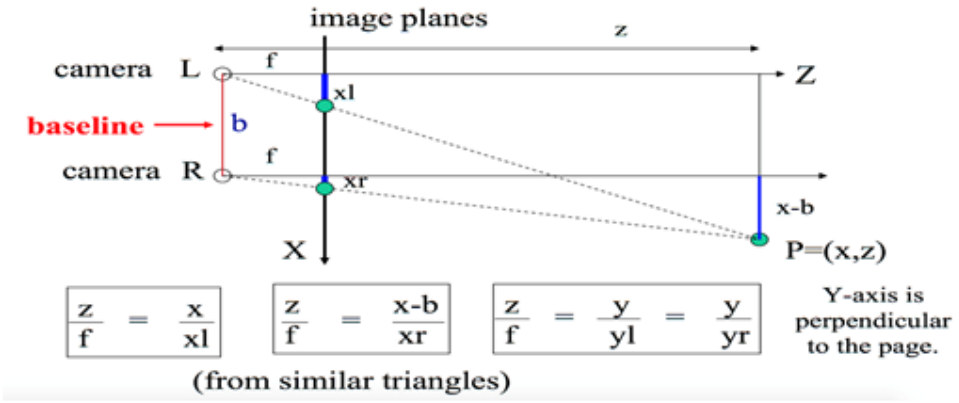


Figure 2.12 Optic axes of 2 cameras are parallel

The difference between xr and xl is called disparity. Eqn. (1) shows that the depth z is inversely proportional to disparity. This method of determining depth z from disparity d is called triangulation.

2.9.3.2 Hardware configuration

If the images of an object are captured from two different points of view, the 3D information can be derived from the disparity of the two images. We use a pinhole camera model to illustrate the theory of stereovision. Figure one is a stereovision system with two

parallel cameras. The object point P is projected on the image plane of the two cameras. For left and

1) Stereo Vision Camera

It designed a stereo vision camera mount to install two camera modules on it, roll of the board camera module are adjustable. The plate connecting two cameras is made of carbon fiber, which is a rigid material resist.

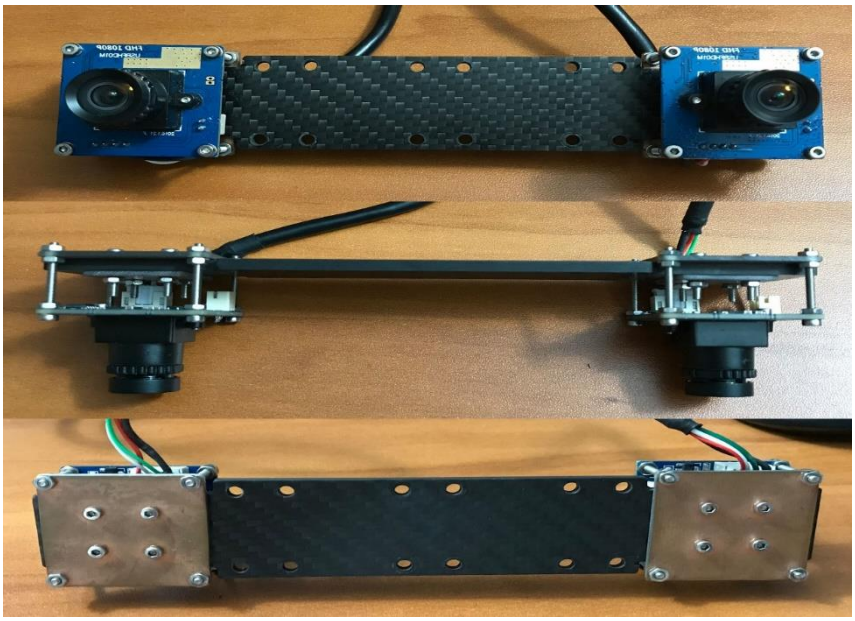


Figure 2.13 stereo vision camera

2) Board Camera Module

ELP-USBFHD01M-FV board camera.

3) Baseline

Baseline = 120mmThe baseline is the distance between two camera's optical axes. A short baseline stereovision system has larger overlapped visual area, which is able to detect the objects close to the camera. But the disparities at larger distance are too small to be distinguished, Besides that, the larger baseline system has to searching the larger range when execute the stereo matching algorithm, which needs more time and hardware resource, meanwhile, the mismatching probability will increase. It is important to choose a proper baseline to meet a specific requirement.

4) Focal Length

Focal Length = 4.35mm low distortion lens .The focal length is the distance between the lens and the image sensor when the subject is in focus. The short focal length lens has a large view angle, but will produce large distortion. The long focal length lens has a narrow

view angle, but the distortion is small. The distortion will drastically reduce the accuracy of the depth map, at the same time it require an accurate depth map to get position information.

2.9.3.3 Software implementation

This project uses two method to build the software of stereo vision camera system. One method is using the built-in functions in opencv to control the camera hardware. However, upon testing, some camera control functions is not well implemented in opencv. It developed a new software stack working around opencv. I use v4l2 API to control camera hardware directly. The code in folder stereo_cam_OpenCV is developed using opencv functions in python. The code in folder stereo_cam_v4l2_c is developed using v4l2 API in c. The c code is a early phase test for v4l2 API, and I only implemented some functions to control camera parameters, no video recording function included. The code in folder stereo_cam_v4l2_python is a fully developed package based on v4l2 API. It has camera control, video recording, and stereo vision depth map generation functions.

2.9.3.4 Steps to use the Stereo Vision camera icon to create a depth map

Before using the stereo vision camera code to create depth map, you need to do camera hardware alignment and software calibration.

1) Camera Hardware Alignment.

It developed a cross laser calibration method to calibrate the roll, please refer to STEREO VISION CAMERA HARDWARE ALIGNMENT for more detail. The following figure

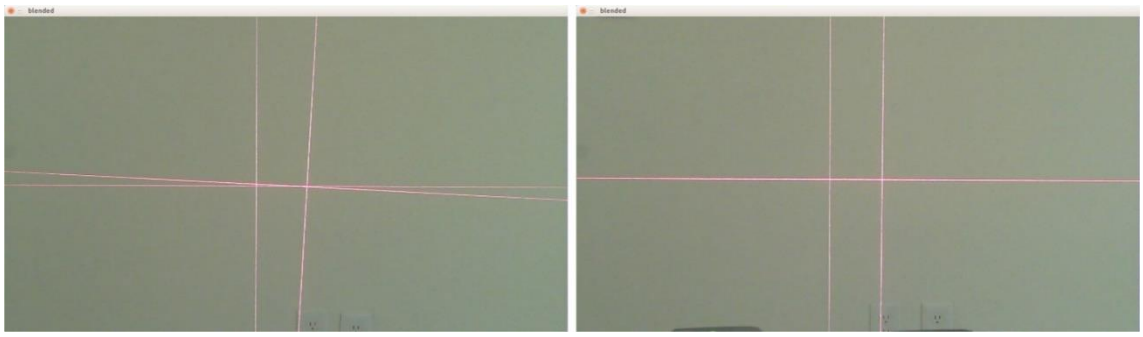


Figure 2.14 Shows the misaligned (left) and in-aligned optical (right) axes.

2) Stereo Vision Calibration.

It developed a stereo vision calibration workflow based on opencv functions. Please refer to **STEREO CAMERA CALIBRATION** for detailed calibration procedure.

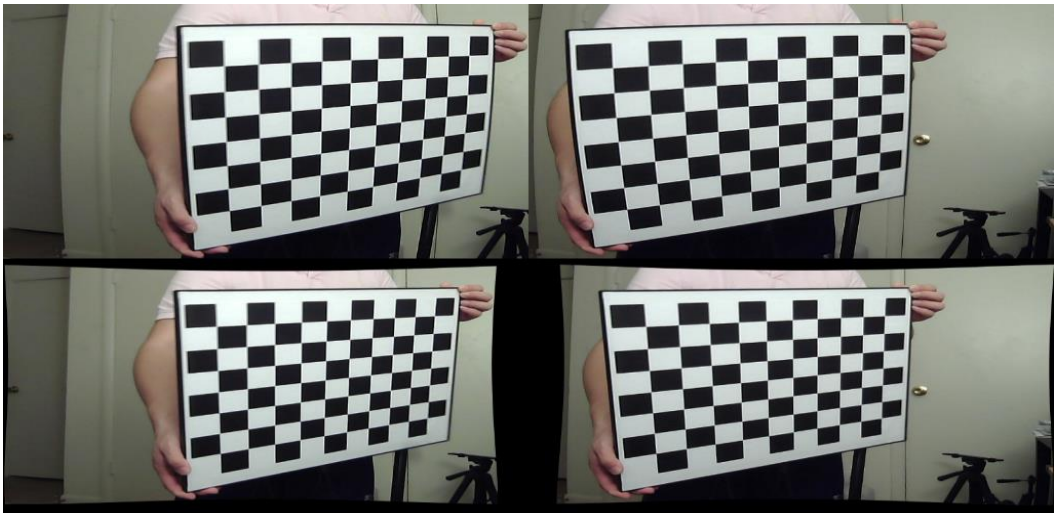


Figure 2.15 shows the original frame pairs captured by stereo camera

3) Capture Frames and Calculate Depth Map.

You can either use package stereo_cam_v4l2_python or package stereo_cam_opencv to capture stereo vision frame pairs and calculate depth map. Please refer to the comment in code for instructions on parameter setting. Following are captured frame pair and the corresponding depth map. In the depth map, the red color means a distance near to camera, and the blue color mean a distance far from the camera. With depth map, we are able to reconstruct the 3D map.

Figure 2.16 Stereo Vision Camera Frame Pair

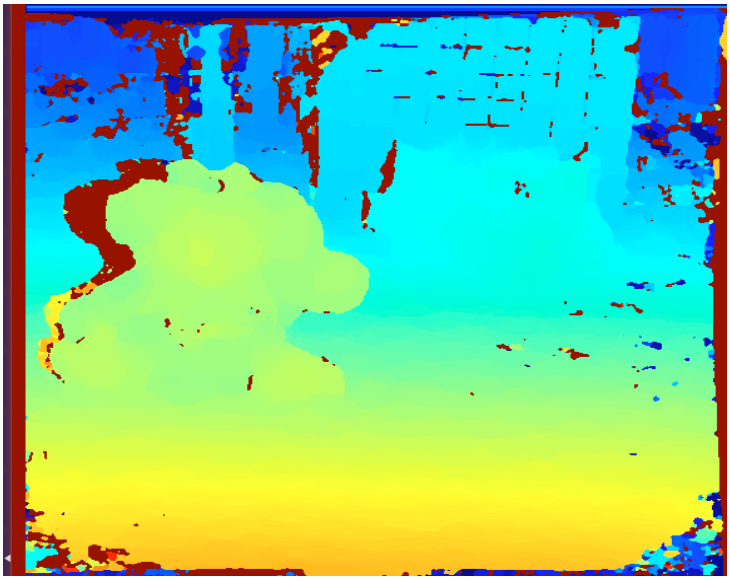


Figure 2.17 Stereo Vision Camera Depth Map

2.9.3.5 Stereo Vision depth map

Stereo vision involves extraction of depth information from two different views of a scene obtained by two different cameras.

After calibration, and then rectification (which projects images back to a common plane) corresponding image points can be found by minimizing cost function, which in simplest case summaries differences in pixels intensities in neighborhood of two analyzed points, so by comparing these two images, disparity map can be computed, for this stereo pair contains differences in horizontal coordinates of corresponding points, values in disparity map are inversely proportional to the scene depth.

Stereo vision is similar to human binocular vision. Objects which are closer to our eyes have larger relative shift, then the ones further away

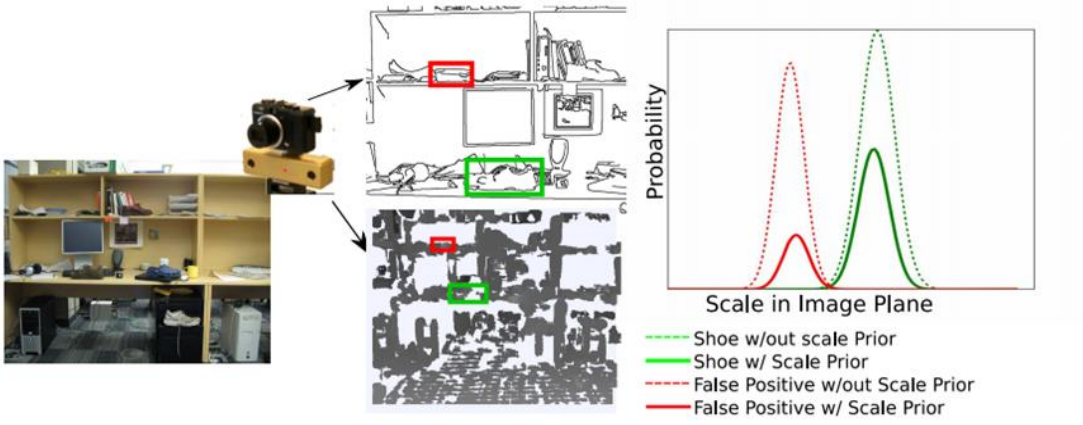


Figure 2.18 Stereo Vision depth map

2.9.3.6 Image processing using stereo vision

After calibration, and then rectification (which projects images back to a common plane) Human has the ability to roughly estimate the distance and size of an object because of the stereo vision of human's eyes. In this project, we proposed to utilize stereo vision system to accurately measure the distance and size (height and width) of object in view. Object size identification is very useful in building systems or applications especially in autonomous system navigation. Many recent works have started to use multiple vision sensors or cameras for different type of application such as 3D image constructions, occlusion detection and etc.

Multiple cameras system has becoming more popular since cameras are now very cheap and easy to deploy and utilize. The proposed measurement system consists of object detection on the stereo images and blob extraction and distance and size calculation and object identification. The system also employs a fast algorithm so that the measurement can be done in real-time. The object measurement using stereo camera is better than object detection using a single camera that was proposed in many previous research works. It is much easier to calibrate and can produce a more accurate results.

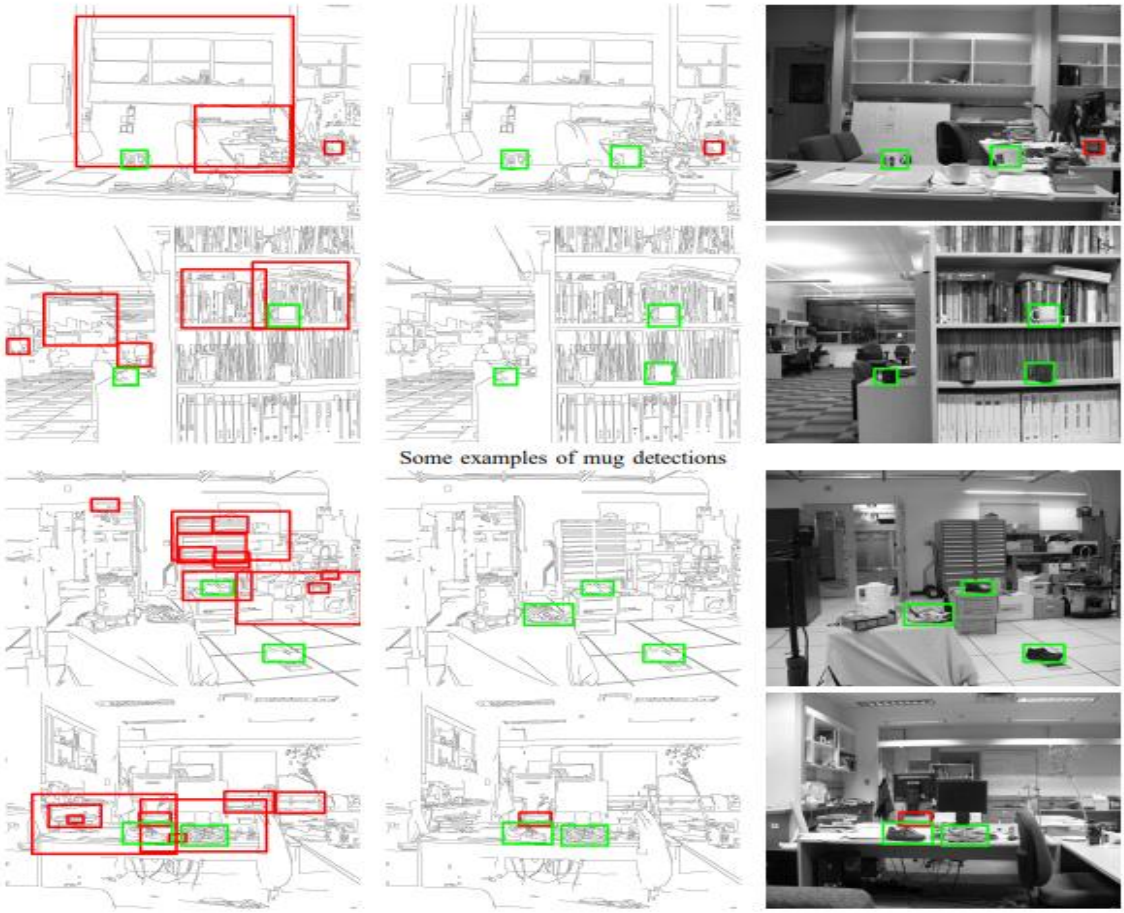


Figure 2.19 Examples of object detections.

2.10 Types of Communication

We have two kinds of communication divided into **Parallel communication** and **Serial communication.**

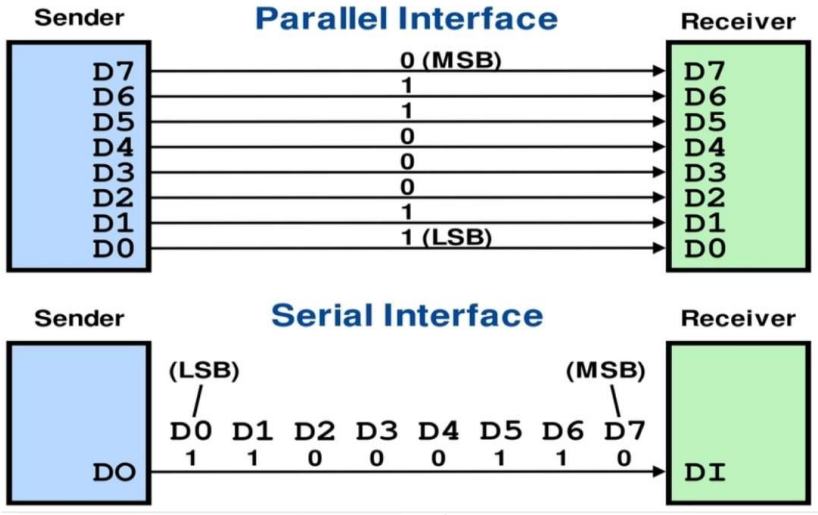


Figure 2.20 communication of the two Types

In this case the receiver device didn't know how the character coming from sender (LSB or MSB) so we have two techniques to know by which way its coming.

Known it translate the character as zeros and ones binary.

2.10.1 Parallel communication

It is the oldest one and traditional but we should have ports or socket to get the ability of communication between devices, it have the ability to send eight bit at the same time and every bit have wire.

- **Advantage:** It have the ability to send all bits at the same time.
- **Disadvantage:** Complexity (cause every bit have wire), Low speed.
- **Uses:** Printer, Scanner.

2.10.2 Serial communication

Is the process of sending data one bit at a time, sequentially, over a communication channel, this is in contrast to parallel communication, where several bits are sent as a whole.

- **The two types of Techniques**

1) Synchronous
2) Asynchronous

Hint: The two kinds for keep away from wrong reading.

2.10.2.1 Synchronous

Its function is should prepared, must have a clock for starting the communication, should have pulse to send the data and clock for end, the data is doesn't over lapped to each other cause it have waiting time to receive.

2.10.2.2 Asynchronous

It's unprepared function, in case of sender device have data it will transfer directly, and other device receive it, beside its faster and cheaper than synchronous.

Logically, as the Asynchronous is best way to be chosen.

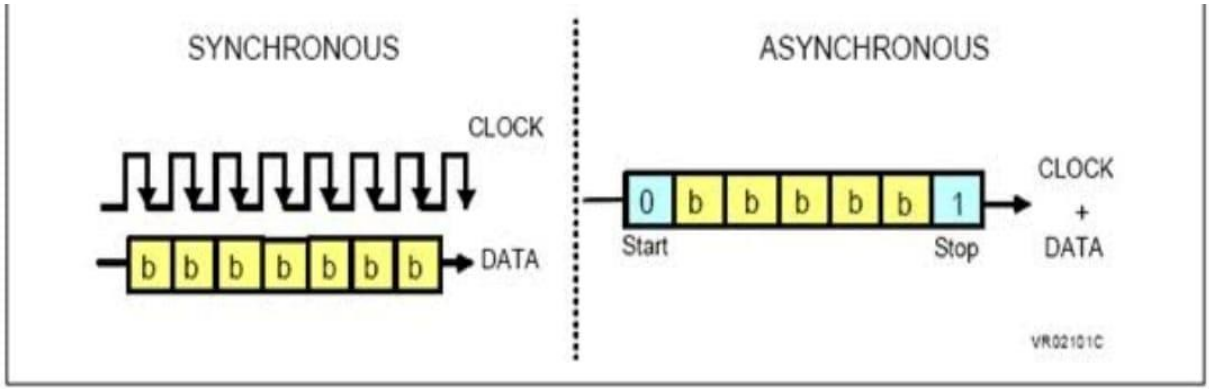


Figure 2.21 show the two techniques clearly

SO, If we have two devices (A, B) sender and receiver respectively, we should know the ability of the processor of device A for send and the ability of processor of device B for receive, this ability is calculated by bit/sec and depends on number of character of sender have to transmit.

For example: If device A has fifty characters to send and by known we have ten bits of UART so it's ability five hundred bps.

Python should know the libraries of UART protocol, firstly download the libraries of UART to communicate easily between them, secondly the python is have ability to receive and send by UART protocol.

➢ UART Protocol

This is program having the ability to send and receive data directly, e.g. (Bluetooth of mobile), Known it translate the send character as zeros and ones binary.

Figure 2.22 UART frame

Start bit: is one bit just for make the receiver knows the sender will send data.

Data bit: is seven bits for data sending.

Parity bit: is one bit it the protocol for error check (Optional).

Stop bit: is one bit for ending data send.

The total number of bits at UART frame is Ten bits.

SO, If we have two devices (A, B) sender and receiver respectively, we should know the ability of the processor of device A to send and the ability of processor device B for receive, this ability is calculated by(bit/sec) and depends on number of character that sender have.

So, if device A has fifty characters to send and have ten bits of UART so it's ability five hundred bps.

Ability= number of character* number of bits [1]

Known the UART protocol is not known by python so we need to download the libraries that support the UART to communicate between them. After did that, the python is have ability to receive and send by UART protocol.

2.11 Robotics operation system

2.11.1 Introduction to ROS

The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

ROS is an open-source, meta-operating system for your robot. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers. ROS is similar in some respects to 'robot frameworks. The ROS ecosystem now consists of tens of thousands of user's worldwide, working in domains ranging from tabletop hobby projects to large industrial automation systems.

ROS is sometimes called a Meta operating system because it performs many functions of an operating system, but it requires a computer's operating system such as Linux. One of its main purposes is to provide communication between the user, the computer's operating system, and equipment external to the computer. This equipment can include sensors, cameras, as well as robots. As with any operating system.

The benefit of ROS is the hardware abstraction and its ability to control a robot without the user having to know all of the details of the robot. For example, to move a robot's arms, a ROS command is issued or scripts in Python or C++ written by the robot designers cause the robot to respond as commanded. The scripts can, in turn, call various control programs

that cause the actual motion of the robot's arms. It is also possible to design and simulate your own robot using ROS. These subjects and many others will be considered in this book. In this book, you will learn a set of concepts, software, and tools that apply to an ever-increasing and diverse army of robots. For example, the navigation software of one mobile robot can be used, with a few changes, to work in another mobile robot. The flight navigation of an aerial robot is similar to that of the ground robot and so on. All across the broad spectrum of robotics, system interfaces are standardized or upgraded to support increased complexity. There are readily available libraries for commonly used robotics functions. ROS not only applies to the central processing of robotics but also to sensors and other subsystems. ROS hardware abstraction combined with low-level device control speeds the upgrade toward the latest technology.

ROS is an open source robotic software system that can be used without licensing fees by universities, government agencies, and commercial companies. The advantages of open source software are that the source code for the system is available and can be modified according to a user's needs. More importantly for some users, the software can be used in a commercial product as long as the appropriate licenses are cited. The software can be improved and modules can be added by users and companies.

2.11.1.1 Foundation controls ROS

A ROS distribution is a set of ROS software packages that can be downloaded to your computer. These packages are supported by the Open Source Robotics Foundation (OSRF), a nonprofit organization. The distributions are updated periodically and given different names by the ROS organization. More details about the ROS organization are

2.11.1.2 The General Organization of ROS

1) Peer-to-Peer
2) Tools-based (microkernel)
3) Multi-language

- **Peer to Peer:**

A sufficiently complex robot comprises several onboard computers or boards connected via Ethernet, plus sometimes off board computers for intensive computation tasks. A peer-to-peer architecture coupled to a buffering system and a lookup system (a name service called ‘master’ in ROS), enables each component to dialogue directly with any other, synchronously or asynchronously as required.

- **Tools-based:**

Rather than a monolithic runtime environment, ROS adopted a microkernel design, which uses a large number of small tools to build and run the various ROS components. As you cover the ROS tutorials, you will learn to use several commands used to manipulate nodes and messages. Each command is in fact an executable. The advantage of this system is that a problem with one executable does not affect the others.

➢ Multi-languages:

ROS is language-neutral, and can be programmed in various languages. The ROS specification works at the messaging layer. Peer-to-peer connections are negotiated in XML-RPC, which exists in a great number of languages. To support a new language, either C++ classes are re-wrapped (which was done for the Octave client, for example) or classes are written enabling messages to be generated. These messages are described in IDL (Interface Definition Language).000000

2.11.1.3 Explanation for Commands

If you are communicating with ROS via the terminal window, it is possible to issue commands to ROS to explore or control nodes in a package from the command prompt, as listed in the following table.

Table 2.1 ROS commands

Command	Action	Example usage and subcommand examples
`roscore`	Starts the Master	`$ roscore`
`rosrun`	Runs an executable program and creates nodes	`$ rosrun [package name] [executable name]`

`rosnode`	Shows information about nodes and lists the active nodes	`$ rosnode info [node name]`
		`$ rosnode<subcommand>`
		Subcommand: `list`

`rostopic`	Shows information about ROS topics	`$ rostopic<subcommand><topic name>` Subcommands: `echo`, `info`, and `type`
`rosmsg`	Shows information about the message types	`$ rosmsg<subcommand> [package name]/ [message type]` Subcommands: `show`, `type`, and `list`

2.11.1.4 Tools in ROS

As we said earlier, ROS is a collection of **Tools** and **Algorithms**. Some are much-used during programming, simulation or executing robot tasks. Some of the tools and algorithms that the ROS programmer might often encounter are:

Gazebo: a 3D simulator

Rviz: A 3D visualization system (unlike Gazebo, does not include a physical motor)

rqt_graph: displays a visual graph of the processes running in ROS and their connections.

2.11.2 Basis in ROS

Here we will explore some of the main components of ROS. One of the primary purposes of ROS is to facilitate communication between the ROS nodes. These nodes represent the executable code. The code can reside entirely on one computer, or nodes can be distributed between computers or between computers and robots. The advantage of this distributed structure is that each node can control one aspect of a system.

For example, one node can capture the images from a camera and send the images to another node for processing. After processing the image, the second node can send a control signal to a third node for controlling a robotic manipulator in response to the camera view. The main mechanism used by ROS nodes to communicate is by sending and receiving messages. The messages are organized into specific categories called topics. Nodes may publish messages on a particular topic or subscribe to a topic to receive information.

Figure 2.23 ROS File system level

- **The explanations of each block in the file system:**

1) **Packages:** The ROS packages are the most basic unit of the ROS software. It contains the ROS runtime process (nodes), libraries, configuration files, and so on, which are organized together as a single unit. Packages are the atomic build item and release item in the ROS software.

2) **Package Manifest:** A ROS *manifest* (manifest.xml) is a minimal specification about a ROS package and supports a wide variety of tools, from compilation to documentation to distribution

3) **Meta packages:** are specialized Packages in ROS (and catkin). They do not install files (other than their package.xml manifest) and they do not contain any tests, code, files, or other items usually found in packages

4) **Messages:** The ROS messages are a type of information that is sent from one ROS process to the other. We can define a custom message inside the msg folder inside a package. The extension of the message file is .msg.

5) **Services:** The ROS service is a kind of request/reply interaction between processes. The reply and request data types can be defined inside the srv folder inside the package.

6) **Repositories:** Most of the ROS packages are maintained using a Version Control System (VCS) such as Git, subversion (svn), mercurial (hg), and so on. The collection of packages that share a common VCS can be called repositories. The package in the repositories can be released using a catkin release automation tool called bloom.

2.11.2.1 ROS Master

One responsibility of the Master is to keep track of nodes when new nodes are executed and come into the system. Thus, the Master provides a dynamic allocation of connections. The nodes cannot communicate however until the Master notifies the nodes of each other's existence.

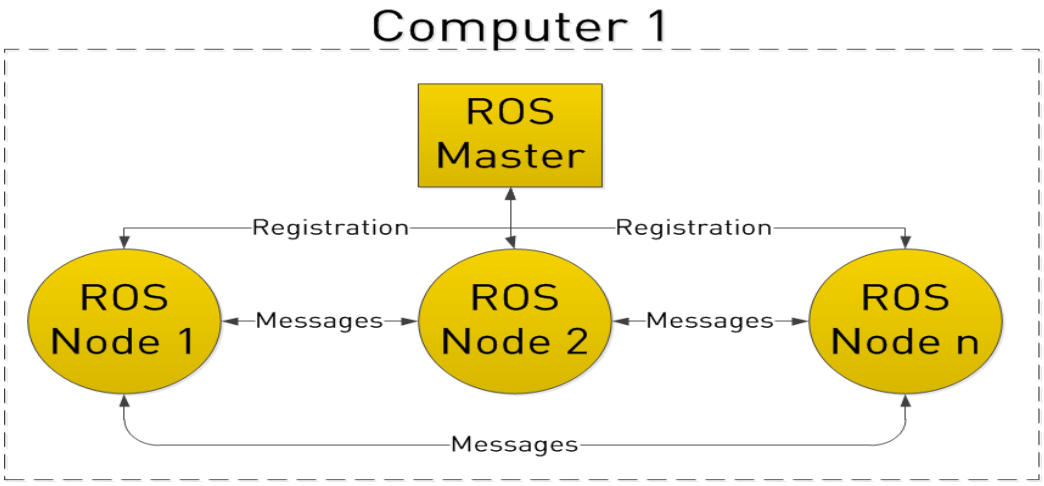


Figure 2.24 Communication between nodes

2.11.2.2 ROS Nodes

Basically, nodes are processes that perform some computation or task. The nodes themselves are really software processes but with the capability to register with the ROS Master node and communicate with other nodes in the system. The ROS design idea is that each node is independent and interacts with other nodes using the ROS communication capability. The Master node is described in the ROS Master section to follow.

A node can independently execute code to perform its task but can also communicate with other nodes by sending or receiving messages. The messages can consist of data, commands, or other information necessary for the application.

2.11.2.3 ROS Topics

Some nodes provide information for other nodes, as a camera feed would do, for example. Such a node is said to publish information that can be received by other nodes. The information in ROS is called a **topic**. A topic defines the types of messages that will be sent concerning that topic (Channel)

2.11.2.4 ROS Messages

A message is a compound data structure. A message comprises a combination of primitive types (character strings, Booleans, integers, floating point, etc.) and messages (a message is a recursive structure).

2.11.2.5 ROS Services

A topic is an asynchronous communication method used for many-to-many communication. A service meets a different kind of need; that for synchronous communication between two nodes.

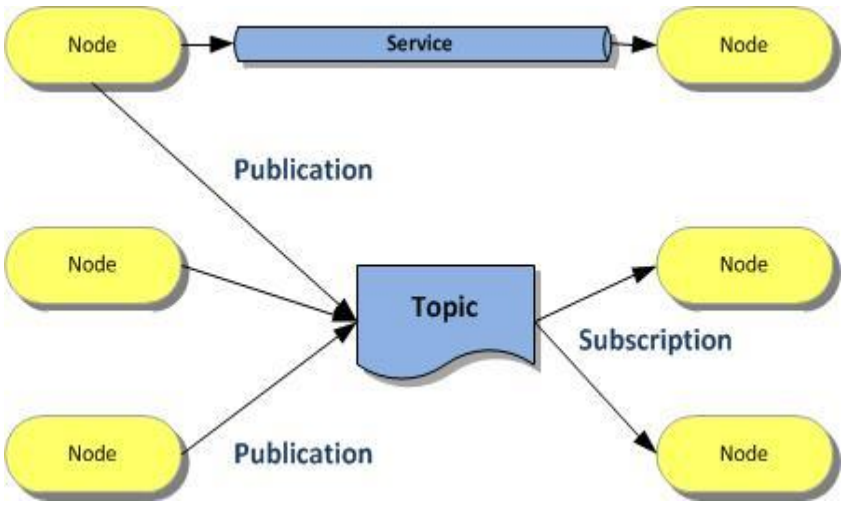


Figure 2.25 Publish and subscribe block diagram

2.11.3 Understanding the Community Level

These are ROS resources that enable a new community for ROS to exchange software and knowledge. The various resources in these communities are as follows:

1) **Distributions:** Similar to the Linux distribution, ROS distributions are a collection of versioned Meta packages that we can install. The ROS distribution enables easier installation and collection of the ROS software. The ROS distributions maintain consistent versions across a set of software.
2) **Repositories:** ROS relies on a federated network of code repositories, where different institutions can develop and release their own robot software components.

3) **The ROS Wiki:** The ROS community Wiki is the main forum for documenting information about ROS. Anyone can sign up for an account and contribute their own documentation, provide corrections or updates, write tutorials, and more.
4) **Bug ticket system:** If we find a bug in the existing software or need to add a new feature, we can use this resource.
5) **Mailing lists:** The ROS-users mailing list is the primary communication channel about new updates to ROS, as well as a forum to ask questions about the ROS software.
6) **ROS Answers:** This website resource helps to ask questions related to ROS. If we post our doubts on this site, other ROS users can see this and give solutions.
7) **Blog:** The ROS blog updates with news, photos, and videos related to the ROS community (http://www.ros.org/news).

2.11.3.1 The prerequisites to start with ROS

Recently ROS was not used in on windows as operating system but there is a Trial version for working on windows operating system but for now the Agreed operating system for working in ROS is Linux.

Before getting started with ROS and trying the code in this book, the following prerequisites should be met:

1) **Linux:** we have to use Ubuntu as the operating system for installing ROS.
2) **ROS desktop full installation:** Install the full desktop installation of ROS.

2.11.3.2 Running ROS Master

Before running any ROS nodes, we should start ROS Master and the ROS parameter server. We can start ROS Master and the ROS parameter server using a single command called roscore, which will start the following programs:

1) ROS Master
2) ROS parameter server
3) rosout logging nodes

The rosout node will collect log messages from other ROS nodes and store them in a log file, and will also rebroadcast the collected log message to another topic. The topic /rosout is published by ROS nodes working using ROS client libraries such as roscpp and rospy and this topic is subscribed by the rosout node which rebroadcasts the message in another topic called /rosout_agg. This topic has an aggregate stream of log messages. The command roscore is a prerequisite before running any ROS node.

2.11.3.3 Creating ROS Package

The ROS packages are the basic unit of the ROS system. We can create the ROS package, build it and release it to the public. The current distribution of ROS we are using is Jade/Indigo. We are using the catkin build system to build ROS packages. A build system is responsible for generating 'targets'(executable/libraries) from a raw source code that can be used by an end user. In older distributions, such as Electric and Fuerte, rosbuild was the build system. Because of the various flaws of rosbuild, catkin came into existence, which is basically based on CMake (Cross Platform Make). This has lot of advantages such as porting the package into other operating system, such as Windows. If an OS supports CMake and Python, catkin based packages can be easily ported into it.

The first requirement in creating ROS packages is to create a ROS catkin workspace. Here is the procedure to build a catkin workspace.

Build a workspace folder in the home directory and create a src folder inside the workspace folder :

$ mkdir ~/catkin_ws/src

Switch to the source folder,The packages are created inside this package :

$cd ~/catkin_ws/src

Initialize a new catkin workspace :

$ catkin_init_workspace

We can build the workspace even if there are no packages :

$ cd ~/catkin_ws

2.11.3.4 The dependencies in the packages are as follows

"roscpp": This is the C++ implementation of ROS. It is a ROS client library which provides APIs to C++ developers to make ROS nodes with ROS topics, services, parameters, and so on. We are including this dependency because we are going to write a ROS C++ node. Any ROS package which uses the C++ node must add this dependency.

"std_msgs": This package contains basic ROS primitive data types such as integer, float, string, array, and so on. We can directly use these data types in our nodes without defining a new ROS message.

1) After creating this package, build the package without adding any nodes using the catkin_make command. This command must be executed from the catkin workspace path. The following command shows you how to build our empty ROS package :

~/catkin_ws$ catkin_make

2) After a successful build, we can start adding nodes to the src folder of this package. The build folder in the CMake build files mainly contains executables of the nodes that are placed inside the catkin workspace src folder. The devel folder contains bash script,

header files, and executables in different folders generated during the build process. We can see how to make ROS nodes and build using catkin_make.

2.11.4 Mine Mapping

Mine Mapping can be done on activating the metal detector sensor. For keeping the update on the coils traced area, a scan_map is updated with the location of the coils while the robot is navigating. In ROS, messages can be passed easily by the help of publishing and subscribing the topics from the ROS nodes. ROS nodes are just the C++ code or python script. The C++ code subscribes to the coil Messages and transform from the coil frame of reference to the minefield frame using the transform listener node of ROS which uses the robots model transform (TF) tree. After the transformation the coils location is recorded into the image according to the resolution of the map and then a scan_map in the form of the occupancy grid is published for viewing purpose.

Similarly on detection of the mine signals, we have a mine_map C++ node to keep a track of the positions of the mines. It subscribes to the set_mine message is published from the mine detector node which states the location of the mine in the minefield metric frame. The location is inserted to an image specific to the resolution of the mine_map and published. Here also the static map service is updated with the mine location to consider it as an obstacle for the navigation stack which is used for the navigation around the field with obstacles.

2.11.5 Mine Identification

The existence of a metal in the field is known once the detection alert value crosses a threshold. However it is important to distinguish between the mines and the metals in the field.

2.11.6 Obstacle and Mine Avoidance

ROS includes a navigation package which supports in the moving the mobile base around the field. This navigation stack takes the sensor data, odometry data and the goal point as the inputs and outputs safe velocity commands to the robot for its navigation from its location to the final goal point. The sensor data can be received from the camera, laser, ultrasonic, infrared, etc. In this case, we are using the laser which supports building a map and robots localization. The navigation stack accepts the sensor message in the form of LaserScan or PointCloud; sent over ROS with tf frame and time dependent information. Laser Scan message is a 2D scan data with the capture of intensities along its scan points. Point Cloud message stores information about the number or points in the world.

According to the scan points, the obstacle detection is carried out updating the map server with the obstacle points. The path planning uses this map server as an input map to move from initial point to the goal point. This path is traced using the navigation stack corresponding to the ROS trajectory planner or the DWA (Dynamic Window Approach) planner with selective Dijkstra's and A* algorithm this becoming the global planner. The

local planner is the velocity commands provided to the robot to follow the global plan trajectory.

2.11.7 Gazebo

2.11.7.1 Introduction

The gazebo simulator is used to have the algorithm tested in the real scenario setup. Gazebo is well equipped with the libraries for the sensors like laser scanner, cameras, IMU, GPS and many more, which help implement an test the algorithms on the minefield

2.11.7.2 Gazebo Supports

1) Designing of robot models.
2) Rapid prototyping and testing of algorithms.
3) Regression testing using realistic scenarios.
4) Simulation of indoor and outdoor environments.
5) Simulation of sensor data for laser range finders, 2D/3D cameras, Kinect style sensors, contact sensors, force-torque, and more.
6) Advanced 3D objects and environments utilizing Object-Oriented Graphics Rendering Engine (OGRE).

2.11.7.3 Usage in our project

Gazebo in our project for Robot moving and operation system, but not as traditional functions, so we used for robot application:

1) Mapping.
2) Simulation.
3) Localization.
4) Transformation system.

2.11.7.4 Simulating using Gazebo

Before starting with Gazebo and ROS, we should install the following packages to work with Gazebo and ROS

1) **gazebo_ros_pkgs:** This contains wrappers and tools for interfacing ROS with Gazebo.
2) **Gazebo-msgs:** This contains messages and service data structures for interfacing with Gazebo from ROS.
3) **Gazebo-plugins:** This contains Gazebo plugins for sensors, actuators, and so on.
4) **Gazebo-ros-control:** This contains standard controllers to communicate between ROS and Gazebo.

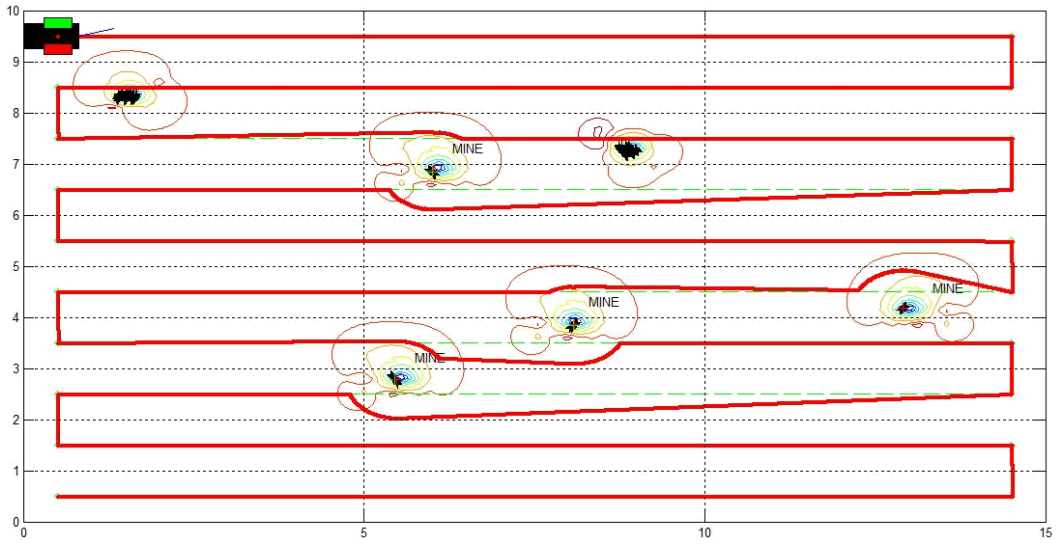


Figure 2.26 Raster scanning with mine avoidance using Potential field path Planning

2.11.7.5 Trajectory planning simulation

In the Figure 2.26, the simulation of the robot is carried out. The robot is attached with an end-effector for visualization of a metal detector like situation which records its position and reads the field data which consists of the mine regions. Here the contour plots describe the metal regions with mine related signals highlighted. The end-effector acts like a metal detector which keeps sweeping the arena and if its locations reading go above the threshold level, then the data is recorded for its analysis.

Once the sufficient data above the threshold is obtained, then the mine location is estimated and then followed by its classification as a mine or metal. Having the localization of the mine with respect to the robot field frame, the mine is considered to be an obstacle by adding its location onto the obstacle map. Due to this, a repulsive field is created around the mine forcing the robot to move away from the mine only if it is in close proximity. Or else this can force the robot to take unnecessary paths increasing the cost of movement even if the force is small coming from the mine. The path of the robot is traced depicting the robots avoidance to the mines and obstacles following a raster scan pattern for area coverage.

2.11.7.6 Detection Simulation

Was used for avoiding the mines in which the mines were updated on the map server as an obstacle having a repulsive potential around it since the navigation stack uses the potential field technique to have an inflation radius around the obstacle and the robots move around the minefield.

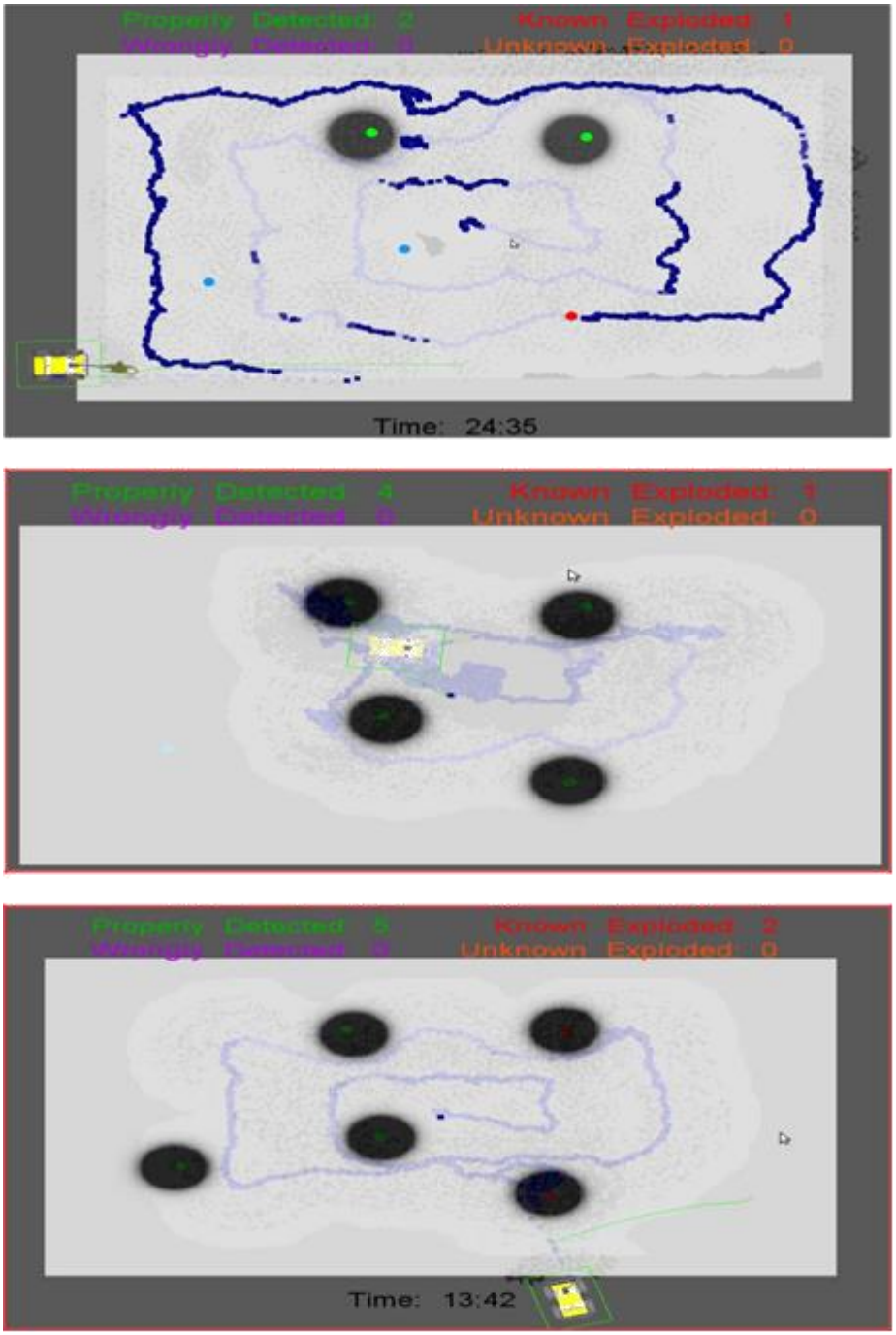


Figure 2.27 Gazebo Simulation Results

2.12 SLAM

The SLAM (simultaneous localization and mapping) GMapping uses the odometry data for its pose and the laser scan data to understand the environment from which the map prediction is carried out. It keeps checking the previous map logged into the map server and updates it if there are new data points captured based on the given conditions for recording the data.

And there is many techniques that can be used for Mapping and localization that will be discussed for each one.

Figure 2.28 Mapping using SLAM GMapping

2.12.1 Localization and Mapping

- **Localization**

Where am I ?

The process of estimating the position and orientation of the robot whether in an indoor or outdoor environment.

This can be done using different techniques

The focus will be on outdoor localization.

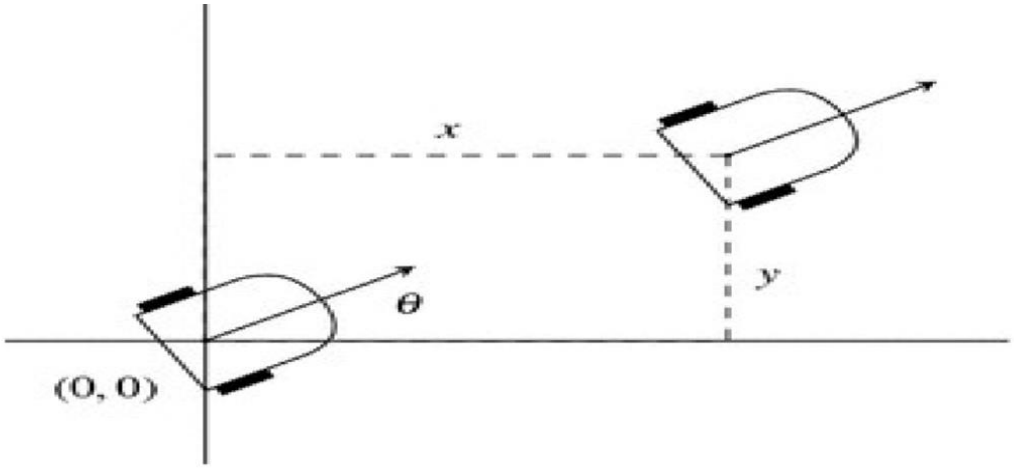


Figure 2.29 Localization technique

2.12.1.1 Localization Techniques and Protocols

- **Local technique:** Indoor localization the process of accurately estimate the position and orientation of the robot moving inside building's environment.

Protocols:

1) Optical encoders.
2) Visual odometry.
3) Hall-effect encoders.
4) INS (inertial navigation system).
5) Scan matching.

- **Global Technique:** Compared to indoor localization, outdoor localization is a more challenging process than indoor localization due to the lack of the ability to control the environment nor to predict it.

Protocols:

1) GPS (Traditional/High precision)
2) Landmark recognition

- **Hybrid technique**

Protocol:

1) (IMU+GPS)

- **Local (Optical encoders)**

Using optical encoders on the robot wheels you can get an estimate of the position and orientation of the robot at any moment.

Disadvantage:

1) The wheels can slip and give false readings for the robot position.
2) Position error grows with time.

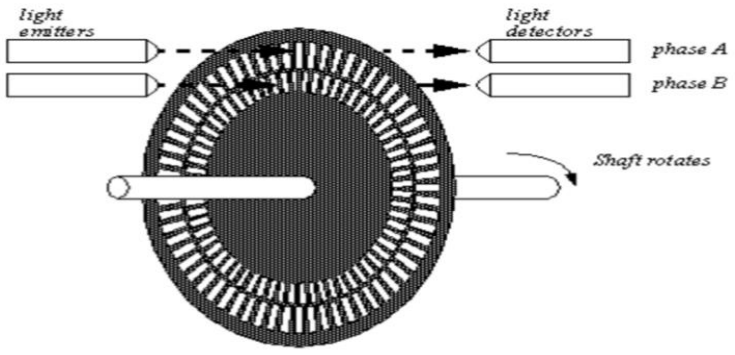


Figure 2.30 Optical Encoder on the wheel

➢ Local (Odometry)

Visual Odometry is the process of determining the position and orientation of a robot by analyzing the associated camera images and looking for changes that the motion induced in the image.

1) **Advantages:** Self-contained, provides a good estimate of the position and orientation of the robot.
2) **Disadvantages:** Position error grows with time.

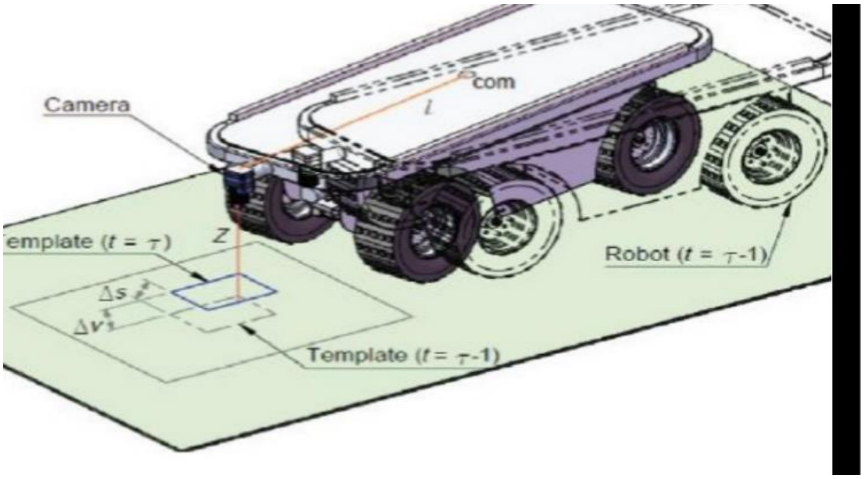


Figure 2.31 Odometry on the wheel

➢ Local (INS)

Inertial Navigation Systems use gyroscopes and accelerometers data to estimate the position and orientation over time by integrating acceleration to get position and angle rates to get orientation. (Rockets and planes)

1) **Advantages:** Self-contained, provides a good estimate of the position and orientation of the robot.
2) **Disadvantages:** Shifts over time as any small error gets integrated twice (e^2).

➢ Local (Scan Matching)

Scan matching is the process of detecting the transformation between consecutive scans.

1) **Advantages:** Self-contained, provides a good estimate of the position and orientation of the robot.

   Known: Works on indoor environments perfectly.

2) **Disadvantages:** Works poorly in outdoor environment especially free of obstacles regions.

Figure 2.32 Scan matching over land

➢ Global (GPS)

GPS stands for global positioning system which is is a satellite-based radio navigation system.

1) **Advantages:** Self-contained, provides an estimate of the position and orientation of the robot.
2) **Disadvantages:** Works only on outdoor environments. Position readings have a big variance.

➢ Laser scan matching

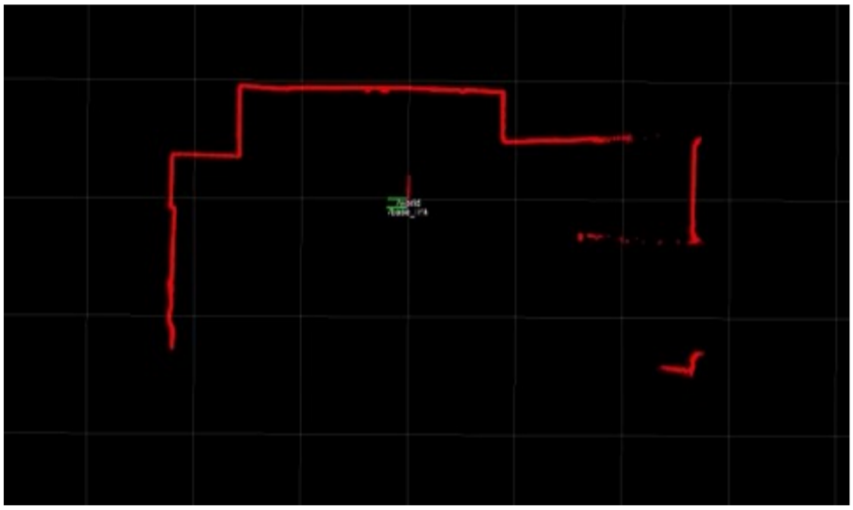


Figure 2.33 Laser scanning on land

2.12.1.2 2D Mapping

➢ 2D mapping Occupancy Grid-based Mapping

In occupancy grid-based mapping, Bayesian estimation (conditional probability) procedure is used to determine the occupancy grid cell state probabilities.

Given a current estimate of the state of the cell and based on an observation, you can estimate the new cell state.

When the cell is revisited what happens is that the cell will be updated to give a more accurate estimate of the state.

GMapping package in ROS uses almost the same concept but in a different way using more optimization and scoring algorithm to get scores for each grid cell, can be used easily on any kind of robot configuration you have but with some condition.

➢ 2D Mapping GMapping – Parameters

1) **throttle_scans (int, default: 1):** Process X out of every this many scans.
2) **base_frame:** (string, default: "base_link")
3) **3map_frame:** (string, default: "map")
4) **odom_frame:** (string, default: "odom")
5) **map_update_interval (float, default: 5.0):** Update the map every 5 seconds.
6) **maxUrange (float, default: 80.0):** The maximum usable range of the laser.
7) **iterations (int, default: 5):** The number of iterations of the scan matcher.
8) **linearUpdate (float, default: 1.0):** Process a scan each time the robot translates this far.
9) **angularUpdate (float, default: 0.5):** Process a scan each time the robot rotates this far.
10) **xmin (float, default: -100.0), xmax (float, default: 100.0)**
11) **ymin (float, default: -100.0), ymax (float, default: 100.0):** Initial map size (in metres)
12) **delta (float, default: 0.05):** Resolution of the map.
13) **maxRange (float):** The maximum range of the sensor. If regions with no obstacles within the range of the sensor should appear as free space in the map, set maxUrange < maximum range of the real sensor <= maxRange.

2.12.2 Application of SLAM

➢ Motion Planning

Motion planning is a navigation problem.

It determines a future course of actions that drives the robot from an initial state to a goal state.

The actions should drive the robot in a collision free path to reach its goal.

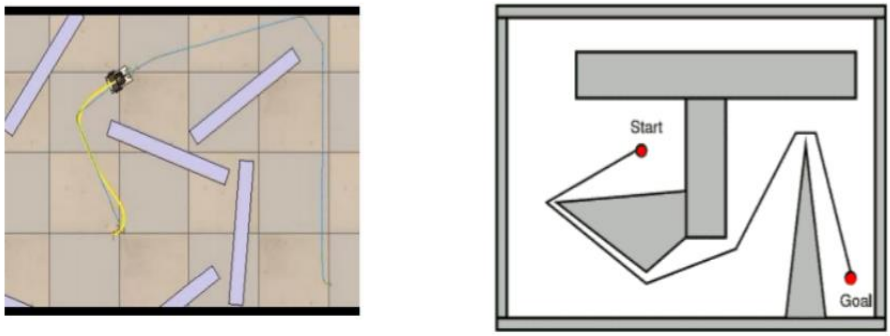


Figure 2.34 the direction of robot with obstacles

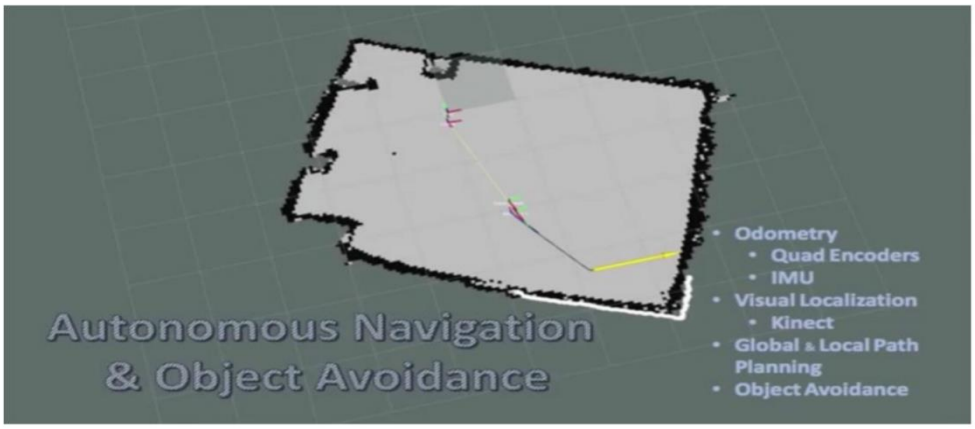


Figure 2.35 Object Avoidance

➢ Motion Planning Concepts – Plan

A path that will bring the robot from an initial position to a goal position.

Notes: The path must be collision-free and minimum distance.

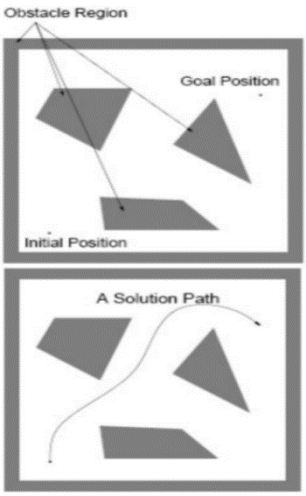


Figure 2.36 Short distance path

➢ Motion Planning Concepts – Cell decomposition

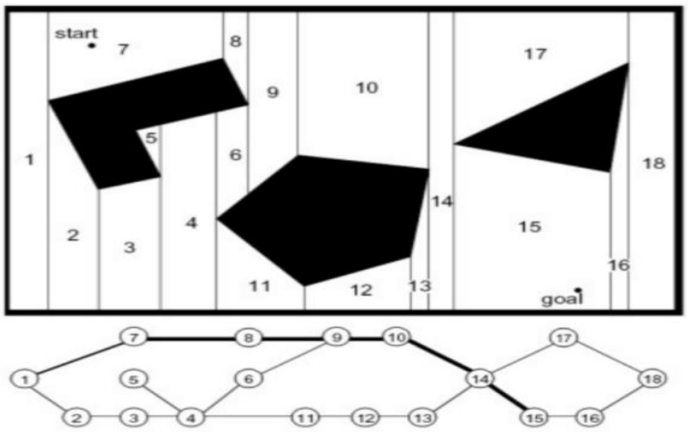


Figure 2.37 Cell decomposition

➢ Motion Planning Concepts – Configuration space

Algorithms assumes that the robot is a single point while the robot of course has dimensions.

The configuration space is the ordinary map that you got from scanning the environment but with expanding obstacles with the robot dimensions.

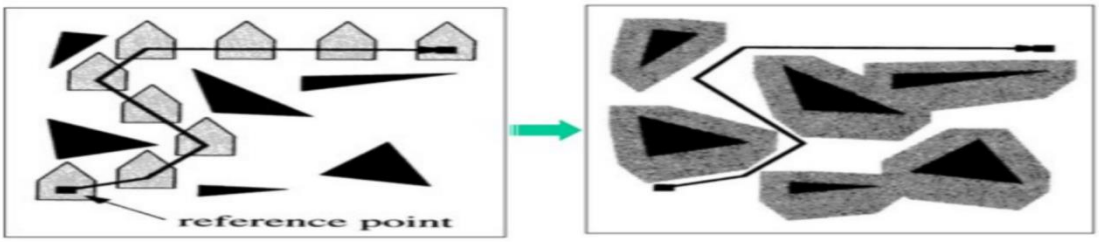


Figure 2.38 Configuration space

Figure 2.39 Costmap configuration space

➢ Motion Planning methods

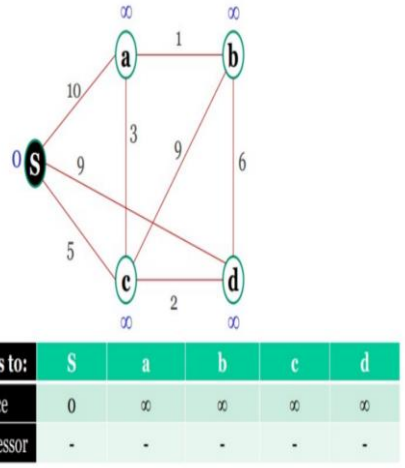
There are plenty of planning methods (Discrete planning, combinatorial planning, etc.)

In the following slides, we will discuss two famous algorithms in motion planning.

1) Motion Planning Methods - Dijkstra

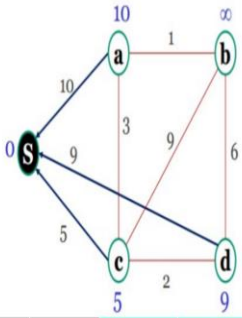
Dijkstra is a graph search algorithm that solves the shortest path problem for a fully connected graph.

Figure 2.40 Points at initial state in dijkstra method



From s to:	S	a	b	c	d
Distance	0	∞	∞	∞	∞
Predecessor	-	-	-	-	-

Figure 2.41 (a) Initial state Dijkstra method



From s to:	S	a	b	c	d
Distance	0	10	∞	5	9
Predecessor	-	s	-	s	s

Figure 2.41 (b) Dijkstra method on point

2) Motion Planning Methods - A*

Similar to Dijkstra, it's a graph search algorithm but instead of exploring all the nodes to get the best path, it explores nearest nodes to the goal.

Figure 2.42 Points at initial state in A* method

2.12.3 Block diagram for Move Base

➢ ROS Package- move_base

A 2D navigation stack that takes in information from odometry, sensor streams, and a goal pose and outputs safe velocity commands that are sent to a mobile base.

It has several components to achieve the task of motion planning.

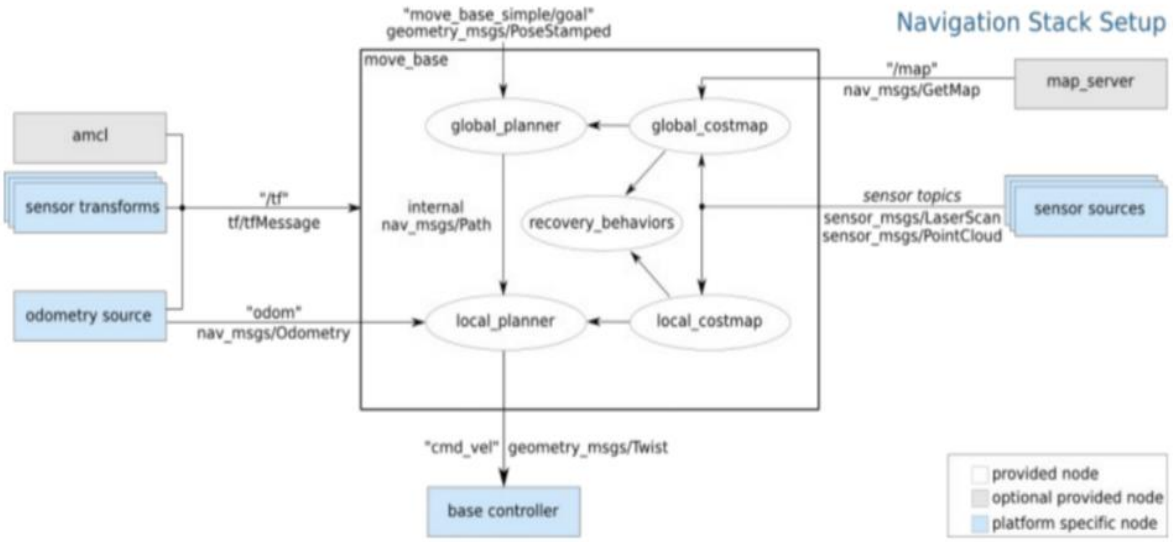


Figure 2.43 Navigation stack block diagram

2.12.3.1 Transformation System

- **Transformation Matrix**

1) Basic the specific application of matrices.
2) The matrix describes a rotation and translation around some axes.
3) We can imagine a point again but observed from another observer who's rotated by two angles.

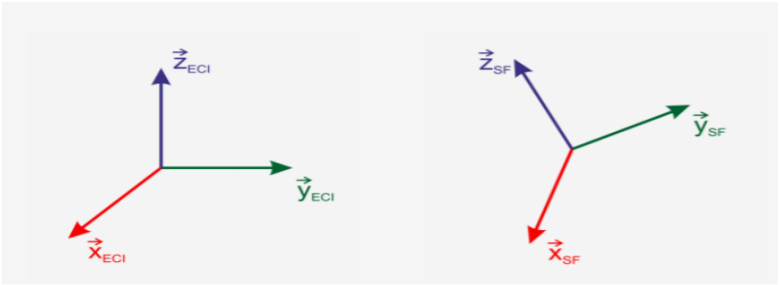


Figure 2.44 Transition matrix

- **Translation and rotation**

Imagine the case where a robot is moving in an environment and we need to know how to map scan points that the robot see to the coordinates of the map.

Figure 2.45 Scan points on map

- **Transformation System**

1) Tool for keeping track of coordinate frames over time.
2) Maintains relationship between coordinate frames in a tree structure buffered in time.
3) Let the user transform points, vectors, etc., between coordinate frames at desired time.
4) Implemented as published/subscriber model on the topics.

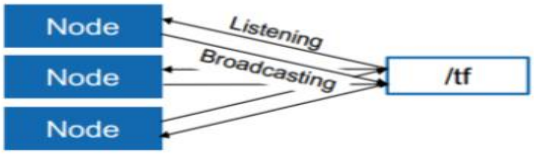


Figure 2.46 Transformation system block diagram

2.12.3.2 Costmap

It takes the map that the robot created before and construct a cost map which is an extension of obstacles in the map by the robot dimensions.

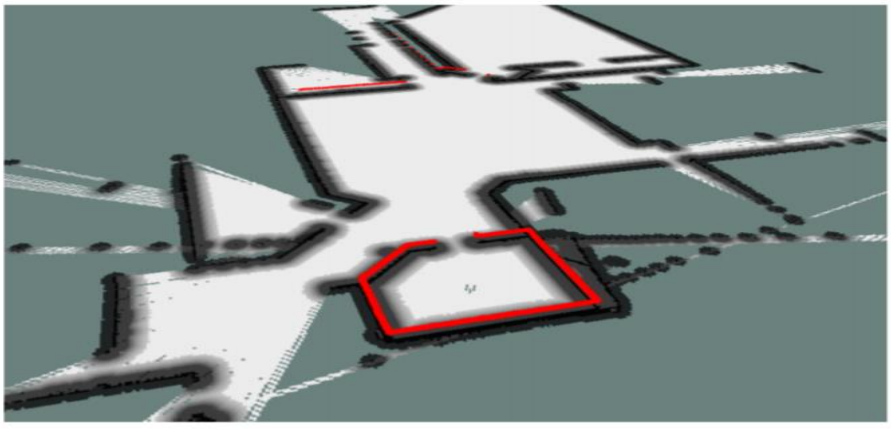


Figure 2.47 Costmap construction

2.12.3.3 Global planner

This package provides an implementation of a fast, interpolated global planner for navigation.

It takes the goal location and the cost map to construct a path from initial position to goal.

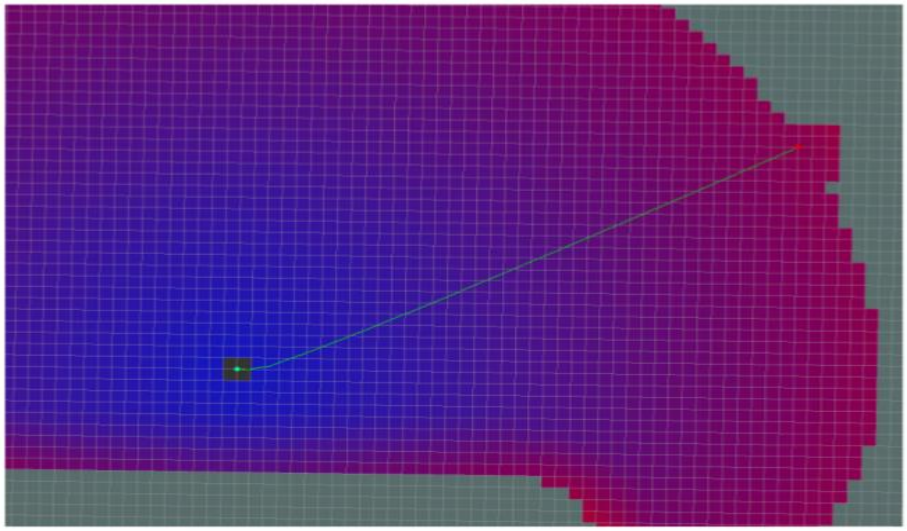


Figure 2.48 Global Planner on map

2.12.3.4 DWA_local_planner

This package provides an implementation of the Dynamic Window Approach to local robot navigation on a plane.

Given a global plan to follow and a costmap, the local planner produces velocity commands to send to a mobile base.

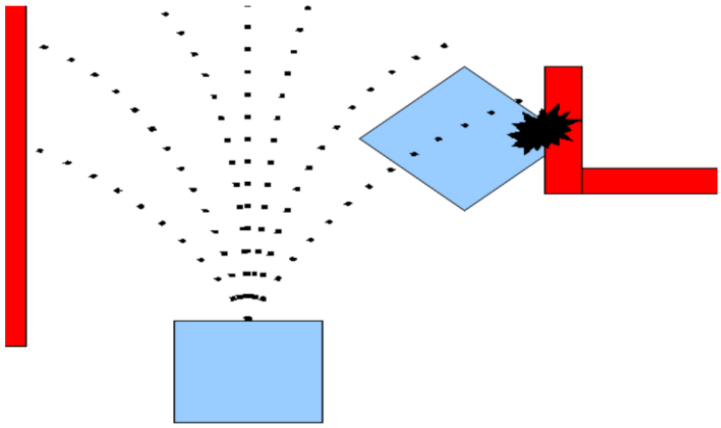


Figure 2.49 DWA local Planner

➢ ROS interface ROS serial-python

The ROS serial-python package contains a Python implementation of the host-side ROS serial connection. It automatically handles setup, publishing, and subscribing for a connected ROS serial-enabled device.

If you have a sensor that works on an Arduino or any microcontroller, you can initialize a ROS node to subscribe or publish messages on it using ROS serial-python package.

2.13 Understanding the robot URDF

2.13.1 Robot Modeling: URDF

- **URDF:** stands for Unified Robot Description Format.

Defines an XML format for representing a robot model.

1) Kinematic and dynamic description.
2) Visual representation.
3) Collision model.
4) Description consists of a set of link elements and a set of joint elements.

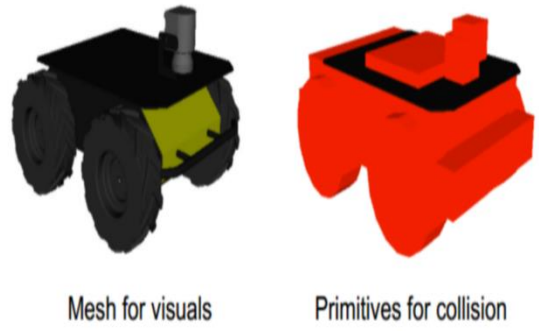


Figure 2.50 (a) Robot visuals

robot.urdf

```
<robot name="robot">
  <link> ... </link>
  <link> ... </link>
  <link> ... </link>

  <joint> .... </joint>
  <joint> .... </joint>
  <joint> .... </joint>
</robot>
```

figure 2.50 (b) Robot URDF

2.13.1.1 URDF Link

A rigid body by its physical properties (dimensions, position of its origin, color, and so on). Links are connected together by **joint** components.

```
<?xml version='1.0'?>
<robot name="trim_robot">
    <!-- Base Link -->
    <link name="base_link">
        <visual>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <box size="0.5 0.5 0.25"/>
            </geometry>
        </visual>
    </link>
</robot>
```

Figure 2.51(a) URDF links

```
<?xml version='1.0'?>
<robot name="trim_robot">
    <!-- Base Link -->
    <link name="base_link">
        <visual>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <box size="0.5 0.5 0.25"/>
            </geometry>
        </visual>
        <collision>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <box size="0.5 0.5 0.25"/>
            </geometry>
        </collision>
    </link>
</robot>
```

Figure 2.51 (b) URDF links

2.13.1.2 URDF Link Elements

- **Visual**

Displays the visual model of the link

Ability to add your own mesh file or using tags such as box, circle, rectangle …etc.

- **Collision**

Gives the simulator the information about the link model

Then the simulator computes the interaction between the link and different simulation objects upon the given information.

Ability to add your own mesh file or using tags such as box, circle, rectangle …etc.

- **Inertial**

Gives the simulator the inertial information about the link model.

Then the simulator computes the motion of the link upon the given inertial information.

Mass is the resistance to move in a linear motion.

Inertia is the resistance to move in a rotational motion.

```
<?xml version='1.0'?>
<robot name="trim_robot">
    <!-- Base Link -->
    <link name="base_link">
        <visual>
            <origin xyz="0 0 0" rpy="0 0 0" />
            <geometry>
                <box size="0.5 0.5 0.25"/>
            </geometry>
        </visual>
    </link>
</robot>
```

Figure 2.52 (a) URDF inertial element

```
<?xml version='1.0'?>
<robot name="trim_robot">
      <!-- Base Link -->
      <link name="base_link">
            <visual>
                  <origin xyz="0 0 0" rpy="0 0 0" />
                  <geometry>
                        <box size="0.5 0.5 0.25"/>
                  </geometry>
            </visual>
            <collision>
                  <origin xyz="0 0 0" rpy="0 0 0" />
                  <geometry>
                        <box size="0.5 0.5 0.25"/>
                  </geometry>
            </collision>
            <inertial>
                  <mass value="5"/>
                  <inertia ixx="0.13" ixy="0.0" ixz="0.0" iyy="0.21"
                        iyz="0.0" izz="0.13"/>
                  </inertial>
      </link>
</robot>
```

Figure 2.52 (b) URDF inertial element.

2.13.1.3 URDF - Joint

It describes the relationship between the links

Joint elements define whether the joint is flexible (movable) or inflexible (fixed)

- **URDF Types**

1) **Fixed:** The two links doesn't move compared to each other ex. Circuit boards and the robot body.

Figure 2.53 Fixed link on robot body

2) **Continuous:** One link moves in a continuous matter compared to the robot body ex. Wheels and the robot body – Car wheels.
3) **Revolute:** One link moves in an exact angle range compared to another link ex. Robotic arm joints.

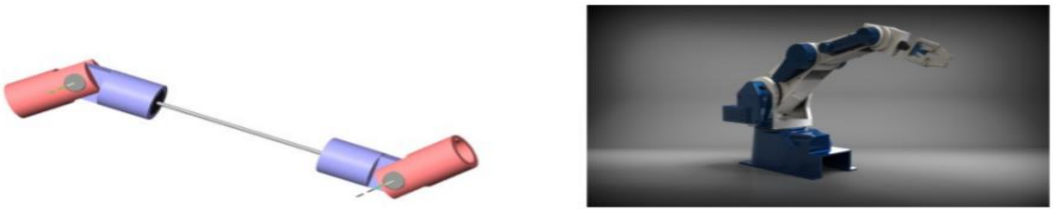


Figure 2.54 Robotic arm joints

4) **Prismatic:** One link moves in a linear range compared to another link ex. Robotic arm joints – 3D printer Axis.

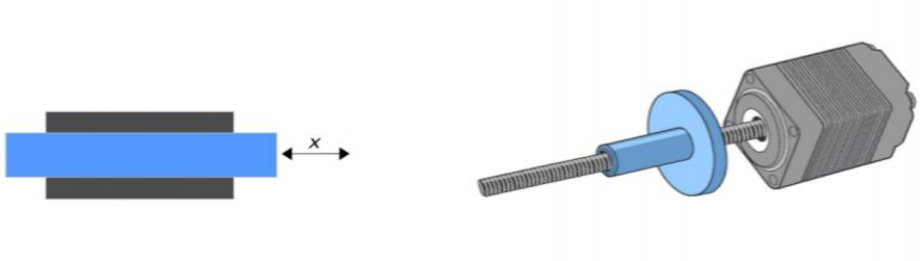


Figure 2.55 3D printer Axis

2.13.1.4 Motion type

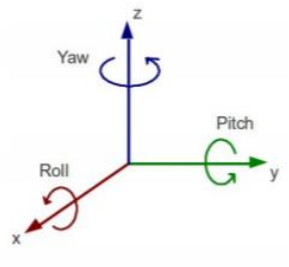
1) Parent and Child: The two links to be attached together.

2) Axis: The unit vector in which the motion will happen (here it's a rotation around the y-axis.

```
<joint name="joint_right_wheel" type="continuous">
  <parent link="base_link"/>
  <child link="right_wheel"/>
  <origin xyz="0 -0.30 0" rpy="0 0 0" />
  <axis xyz="0 1 0" />
</joint>
```

Figure 2.56 URDF's parents and child elements

3) Origin xyz: where the joint is located in space. rpy (roll, pitch, yaw): the orientation of the joint.




```
<joint name="joint_right_wheel" type="continuous">
   <parent link="base_link"/>
   <child link="right_wheel"/>
   <origin xyz="0 -0.30 0" rpy="0 0 0" />
   <axis xyz="0 1 0" />
 </joint>
```

Figure 2.57 URDF origin element

3.1 Introduction of swarm robotics

Sometimes it is impossible to complete a task by a single person or it becomes quite difficult to that person to complete the work. In such cases, there is need of a team or group of members that can collaboratively work and make the work of the person or the user very much easy.

Swarm intelligence (SI) is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems. The concept of **Swarm Robotics** is based on this basis of grouping of multiple robots or devices and perform the desired task. Swarm robotics is a new approach to the coordination of multi-robot systems which consist of large numbers of mostly simple physical robots. This approach emerged on the field of artificial swarm intelligence, as well as the biological studies of insects, ants and other fields in nature, where swarm behavior occurs.

3.2 Swarm Robotics Definition

It is interesting to notice that there seem to be no explicit definitions of swarms in the literature. Common contributions, for example, from the field of swarm intelligence dare to keep a gap in this concern. Instead one finds definitions of swarming behavior. In biology a common definition of swarming behavior is “aggregation often combined with collective motion” which mostly agrees with our above examples. The importance of swarms as a concept is also communicated in the language itself. For example, in English there are words for a number of swarm behaviors, such as flocking in the case of birds, shoaling or schooling in the case of fish, and herding in the case of quadrupeds. Consequently, we owe a definition of a swarm and conclude for now that a swarm is defined via its behavior.

3.3 Advantages of swarm robotics

The advantages and characteristics of the swarm robotics system are presented by comparing a single robot and other similar systems with multiple individuals. These characteristics are quite similar to that of nature swarm.

3.3.1 Comparing with a single robot

To complete a sophisticated task, a single robot must be designed with complicated structure and control modules resulting in high cost of design, construction and maintenance. Single robot is vulnerable especially when a small broken part of the robot may affect the whole system and it's difficult to predict what will happen. The swarm robotics can achieve the same ability through inter-group cooperation and takes the advantage of reusability of the simple agents and the low cost of construction and maintenance. The swarm robotics also takes the advantage of high parallelism and is especially suitable for large scale tasks.

A single robot is inspired from human behaviors by comparing the corresponding nature species of these researching areas, while the swarm robotics is inspired from the social animals. Due to the restriction of current technology, it's hard to simulate the human

interactions using machines or computers while the mechanisms in animal groups are easier to apply. This gives the swarm robotics a bright future in dealing with complex and large scale problems. The advantages of swarm robotics are described below.

3.3.1.1 Parallel

The population size of swarm robotics is usually quite large, and it can deal with multiple targets in one task. This indicates that the swarm can perform the tasks involving multiple targets distributed in a vast range in the environment, and the search of the swarm would save time significantly.

3.3.1.2 Scalable

The interaction in the swarm is local, allowing the individuals to join or quit the task at any time without interrupting the whole swarm. The swarm can adapt to the change in population through implicit task re-allocating schemes without the need of any external operation. This also indicates that the system is adaptable for different sizes of population without any modification of the software or hardware which is very useful for real-life application.

3.3.1.3 Stable

Similar to scalability, the swarm robotics systems are not affected greatly even when part of the swarm quits due to the majeure factors. The swarm can still work towards the objective of the task although their performances may degrade inevitably with fewer robots. This feature is especially useful for the tasks in a dangerous environment.

3.3.1.4 Economical

As mentioned above, the cost of swarm robotics is significantly low in designing, manufacturing and daily maintaining. The whole system is cheaper than a complex single robot even, if hundreds or thousands of robots exist in a swarm. Since the individuals in the swarm can be massively produced while a single robot requires precision machining.

3.3.1.5 Energy efficient

Since the individuals in the swarm are much smaller and simpler than a giant robot, the energy cost is far beyond the cost of a single robot compared with the battery size. This means that the life time of the swarm is enlarged. In an environment without fueling facilities or where wired electricity is forbidden, the swarm robotics can be much useful than traditional single robot.

3.3.2 Comparing to other multi-agent systems

There exist several research areas inspired from the nature swarm, which are often confused with swarm robotics, such as multi-agent system and sensor network. These research areas also utilize the cooperative behavior emerged from the multiple agents in the group for specialized tasks. However, there are several differences between these systems, which can distinguish these systems fundamentally, as shown in **Table 3.1**.

Table 3.1 Comparison of swarm robotics and other systems

	Swarm robotics	Multi-robot system	Sensor network	Multi-agent system
Population Size	Variation in great range	Small	Fixed	In a small range
Control	Decentralized and autonomous	Centralized or remote	Centralized or remote	Centralized or hierarchical or network
Homogeneity	Homogeneous	Usually heterogeneous	Homogeneous	Homogeneous or heterogeneous
Flexibility	High	Low	Low	Medium
Scalability	High	Low	Medium	Medium
Environment	Unknown	Known or unknown	Known	Known
Motion	Yes	Yes	No	Rare
Typical applications	Post-disaster relief Military application Dangerous application	Transportation Sensing Robot football	Surveillance Medical care Environmental protection	Net resources management Distributed control

From **Table 3.1**, it can be easily deduced that the main differences among swarm robotics and other systems are population, control, homogeneity and functional extension. Multi-agent and sensor network systems mainly focus on the behaviors of multiple static agents in the known environments while the robots in the multi-robot systems are quite small, usually heterogeneous and are externally controlled.

3.3.2.1 Autonomous

The individuals in swarm robotics systems must be autonomous, i.e. capable of interacting and motioning in the environment. With these key functions, the cooperative mechanisms inspired from the nature swarms can be introduced into the swarm robotics. Although the systems, like sensor networks, are far different from the swarm robotics from such point

of view, but the research on the area can indeed throw some lights on swarm robotics research.

3.3.2.2 Decentralization

With a good set of cooperative rules, the individuals can complete the task without centralized controls which promises the scalability and flexibility of the swarm. At the same time, the swarm can benefit more in the environments when communication is interrupted or lagged and improves the reaction speed and precision of the swarm.

3.3.2.3 Local sensing and communications

Due to the restriction of hardware and cost, the robots in the swarm usually have a limited range of sensing and communicating and thus the whole swarm is distributed in the environment. Actually, the use of global communications will lead to a significant decline in scalability and flexibility, as the communication cost is explode exponentially as the population grows. Nevertheless, certain controlling global communications are acceptable, for instance, updating the controlling strategies or sending the terminal signals, so long as it's not used in the interaction between individuals.

3.3.2.4 Homogenous

In a swarm robotics system, the robots should be divided into the roles as few as possible and the number of robots acting as each role should be as large as possible. The role here indicates the physical structure of the robot or other states that cannot be changed into one another dynamically during the task. A state in a finite state machine does not count in our definition. This definition indicates a swarm, no matter how large it is, is not considered as swarm robotics if the roles of robots are divided meticulously. For instance, the robots football usually is not considered as swarm robotics, since each individual in the team is assigned a special role during the game.

3.3.2.5 Flexibility

A swarm with high flexibility can deal with different tasks with the same hardware and minor changes in the software, as the nature swarms can finish various tasks in the same swarm. The individuals in the swarm show different abilities and cooperation strategy when they deal with different tasks. The swarm robotics should provide such flexibility, especially in similar tasks, such as foraging, flocking or searching. The swarm can switch to different strategies according to the environment. The robots can adapt to the environment through machine learning from the past moves and can change to a better strategy

3.4 Disadvantages of Swarm Systems

3.4.1 Non optimal

Because they are redundant and have no central control, swarm systems are inefficient. Resources are allotted higgledy-piggledy, and duplication of effort is always rampant. What a waste for a frog to lay so many thousands of eggs for just a couple of juvenile offspring! Emergent controls such as prices in free-market economy a swarm if there ever was one tend to dampen inefficiency, but never eliminate it as a linear system can.

3.4.2 Non controllable

There is no authority in charge. Guiding a swarm system can only be done as a shepherd would drive a herd: by applying force at crucial leverage points, and by subverting the natural tendencies of the system to new ends (use the sheep's fear of wolves to gather them with a dog that wants to chase sheep). An economy can't be controlled from the outside; it can only be slightly tweaked from within. A mind cannot be prevented from dreaming, it can only be plucked when it produces fruit. Wherever the word "emergent" appears, there disappears human control.

3.4.3 Non predictable

The complexity of a swarm system bends it in unforeseeable ways. "The history of biology is about the unexpected," says Chris Langton, a researcher now developing mathematical swarm models. The word emergent has its dark side. Emergent novelty in a video game is tremendous fun; emergent novelty in our airplane traffic -- control system would be a national emergency.

3.4.4 Non understandable

As far as we know, causality is like clockwork. Sequential clockwork systems we understand; nonlinear web systems are unadulterated mysteries. The latter drown in their self-made paradoxical logic. A causes B, B causes A. Swarm systems are oceans of intersecting logic: A indirectly causes everything else and everything else indirectly causes A. I call this lateral or horizontal causality. The credit for the true cause (or more precisely the true proportional mix of causes) will spread horizontally through the web until the trigger of a particular event is essentially unknowable. Stuff happens. We don't need to know exactly how a tomato cell works to be able to grow, eat, or even improve tomatoes. We don't need to know exactly how a massive computational collective system works to be able to build one, use it, and make it better. But whether we understand a system or not, we are responsible for it, so understanding would sure help.

3.5 Modeling methods for swarm robotics

Modeling is a method used in many research fields to better understand the internals of the system that is investigated. Modeling helps to the swarm robotics since a swarm robotic algorithm is supposed to be scalable to hundreds of thousands of robots in population. The time and money are limited for such scale of experiments, the experiments can be done in an easier way.

Considering the characteristics of swarm robotics, the modeling methods are divided into four types according to Ref.: sensor-based, microscopic, macroscopic and swarm intelligence-based. The four methods are described in detail in this section.

3.5.1 Sensor-based modeling

In the sensor-based modeling method, the sensors and actuators of the robots are modeled as the main components of the system along with the objects in the environment. Then the interactions of the robots are modeled as realistically and simply as possible. This modeling method is mostly used, and the oldest method is used for robotic experiment.

The earlier research using sensor-based modeling methods did not consider the real physical limitations, now the researchers introduce the physical principle into the model.

3.5.2 Microscopic modeling

In the microscopic modeling, the robots and interactions are modeled as a finite state machine. The behaviors of each robot are defined as several states, and the transfer conditions are based on the input from communication and sensing. Since the model is based on the behaviors of each robot, the simulation should be run for several times to obtain the averaged behaviors of the swarm.

In the most swarm robotics research, the probabilistic microscopic model is used, since noise can be modeled as probability in the model. In a probabilistic microscopic model, the probabilities are valued from the experiments of real robots, and the model is iterated with these probabilities for state transfer in the simulation to predict the behavior of the swarm.

3.5.3 Macroscopic modeling

Macroscopic modeling is a modeling method opposite to the microscopic modeling. In the macroscopic modeling, the system behavior is defined as difference equation, and a system state represents the average number of robots in this state at the time step.

The main difference between microscopic and macroscopic models is the granularity of the models. The microscopic model for the behavior at individual level is used to simulate the group behaviors while the macroscopic model simulates the behaviors at the swarm level. The microscopic model iterates the swarm behavior, and the macroscopic model can give out the final state of the swarm. In this way, the macroscopic model can have a global glance at the swarm while the microscopic model can show the details of the swarm behaviors.

Probabilistic macroscopic models are also widely used by the researchers. Martini et al. applied the macroscopic modeling to stick the pulling problem from a basic model which contains only two states up to the model with all states. They also compared the microscopic, macroscopic and sensor-based models and described the shortages of macroscopic model.

3.5.4 Modeling from swarm intelligence algorithms

Cooperative schemes from swarm intelligence algorithms have been introduced into the swarm robotics in many researches. Since the robots use the same or similar schemes with these algorithms, the models and other methods used to analyze these algorithms, which are quite mature than that in swarm robotics, can be used directly for robot research.

The most commonly used algorithm from swarm intelligence is the particle swarm optimization (PSO) [(14)] which mimics the flocking process of the birds. The particles fly in the field and search for the best. It can be found obviously that many commons remain between PSO and swarm robotics. A mapping between particle and robot can be presented easily.

Besides PSO, the researchers also introduce other swarm intelligence algorithms into swarm robotics. Many successful swarm models were inspired from the ant colonies. These inspired approaches provide an effective heuristics for searching in dynamic environment and routing.

However, there are still many problems when a cooperative scheme from swarm intelligence is introduced. The schemes in these algorithms consider the most global interactions and introduce large amount of random moves for high diversity. Some schemes also contain the operations to reset the positions of searching agents. However, these operations are unavailable for swarm robotics. How can the schemes in swarm robotics avoid such operations while taking full advantage of the scalability and flexibility is a future research direction.

3.6 Types of Swarm Robotics

- **Are classified to different types according to the following criteria:**

3.6.1 Swarm Size

1) Pair 2 Robots
2) Limited Group multiple Robots
3) Infinite Group n>>1 Robots

3.6.2 Communication Range

1) None-Communicative Robots
2) Near-Communicative Robots

3) Infinite communicative Robots

3.6.3 Communication Topology

1) **Top-Broad:** Every robot can communicate with all of the other robots, but it is not able to send a message to a particular robot.
2) **Top-Address:** Every robot can communicate with any arbitrary robot by name or address.
3) **Top-Tree:** Robots are linked in a tree & may only communicate through this hierarchy.
4) **Top-Graph:** Robots are linked in a general graph.

3.7 Applications of swarm robotics

Potential applications for swarm robotics are many. They include tasks that demand miniaturization (Nano robotics), like distributed sensing tasks in micro machinery or the human body. One of the most promising uses of swarm robotics is in disaster rescue missions. Swarms of robots of different sizes could be sent to places rescue workers can't reach safely, to detect the presence of life via infra-red sensors. On the other hand, swarm robotics can be suited to tasks that demand cheap designs, for instance mining or agricultural foraging tasks. Also some artists use swarm robotic techniques to realize new forms of interactive art.

More controversially, swarms of military robots can form an autonomous army. U.S. Naval forces have tested a swarm of autonomous boats that can steer and take offensive actions by themselves. The boats are unmanned and can be fitted with any kind of kit to deter and destroy enemy vessels.

Most efforts have focused on relatively small groups of machines. However, a swarm consisting of 1,024 individual robots was demonstrated by Harvard in 2014, the largest to date.

Another large set of applications may be solved using swarms of micro air vehicles, which are also broadly investigated nowadays. In comparison with the pioneering studies of swarms of flying robots using precise motion capture systems in laboratory conditions, current systems such as Shooting Star can control teams of hundreds of micro aerial vehicles in outdoor environment using GNSS [(15)] systems (such as GPS [(16)]) or even stabilize them using onboard localization systems where GPS is unavailable. Swarms of micro aerial vehicles have been already tested in tasks of autonomous surveillance, plume tracking, and reconnaissance in a compact phalanx. Numerous works on cooperative swarms of unmanned ground and aerial vehicles have been conducted with target applications of cooperative environment monitoring, convoy protection, and moving target localization and tracking

3.8 Introduction to IOT

Although the concepts we call on throughout this part are relatively straightforward, people have many different visions of what the phrase means, and many of the implications are hard to grasp. So we will take this question slowly in this chapter and look at it from a number of different angles.

“Internet of Things “Almost all, every area, device, sensor and software are connected to each other. The ability to access these devices through a smartphone or through a computer is called IOT [(13)]. These devices are accessed remotely. IoT is basically a platform where we connect everyday things embedded with electronics, software, and sensors to the internet enabling them to collect and exchange data. In this way, each of your devices will learning from the experiences of other devices, just as humans do. With IoT, the interdependence in the human will expand- i.e. to interact, collaborate and contribute to things. The ‘Thing’ in IoT can refer to any device that might comprise any kind of built-in-sensors with the ability to collect and transfer data over a network or internet without manual intervention.

Shortly, IoT is a network of smart devices, sensors, and actuators that can interconnect with each other.

3.9 Internet of Things Ecosystem

All the components that enable businesses, governments, and consumers to connect to their IoT devices, including remotes, dashboards, networks, gateways, analytics, data storage, and security is part of the Internet of Things ecosystem.

IoT ecosystem is like a community that consists of data and monetary flows that helps in connecting enterprises and vendors together. This new chain of development is the best way to connect companies together. Even in future, the companies will be offering the IoT ecosystem similar to risk management and cyber security.

IoT is not just transforming connectivity among devices and objects but it is also allowing the people to get remote access easily. With so many advantages of IoT, it is interesting to see the main ecosystem components that IoT works on. Here are the main components on which IoT works on.

3.9.1 Gateway

Gateway enables easy management of data traffic flowing between protocols and networks. On the other hand, it also translates the network protocols and makes sure that the devices and sensors are connected properly.

It can also work to preprocess the data from sensors and send them off to next level if it is configured accordingly. It is essential to configure it as the presence of TCP/IP [17] protocol allows easy flow.

Not only this, it gives proper encryption with the network flow and data transmission. The data flowed through it is in the higher order that is protected by using latest encryption techniques. You can assume it like an extra layer between the cloud and devices that filter away the attack and illegal network access.

3.9.2 Analytics

The analog data of devices and sensors are converted into a format that is easy to read and analyze. This is all possible due to the IOT ecosystem that manages and helps in improving the system. The main factor that is influenced is security.

The most important function of IOT technology is that it supports real-time analysis that easily observes the irregularities and prevents any loss or scam. Preventing the malicious things to attack the smart devices will not only give you a sense of security but also it will save all your private data from being used for illegal purposes.

The big companies collect the data in bulk and analyze it to see the future opportunity so that they can easily develop more business advancement and gain something out of it. This analysis easily helps in setting future trends that have a capability to rule the market. From this analysis, they can be one step ahead of the time and easily achieve success. Data may be a small word but it holds the power to make or break the business if used correctly.

3.9.3 Connectivity of Devices

The main components that complete connectivity layer are sensors and devices. Sensors collect the information and send it off to the next layer where it is being processed. With the advancement of technology, semiconductor technology is used that allows the production of micro smart sensors that can be used for several applications.

- **The main components are :**

1) Proximity detection
2) Humidity or Moisture Level
3) Temperature sensors and thermostats
4) Pressure sensors
5) RFID [18] tags

The modern smart sensors and devices use various ways to be connected. The wireless networks like Wi-Fi [19] and Bluetooth makes it easy for them to stay connected. They have their own advantages and drawbacks that are classified in various forms like efficiency rate, data transfer, and power.

3.9.4 Cloud

Cloud computing is a major pattern for large data storage and analytics, with the help of internet of things ecosystem, companies are able to collect bulk data from the devices and applications. There are various tools that are used for the purpose of data collection that can collect, process, handle and store the data efficiently in real time. It is also responsible for making a tough decision that can easily break the deal. This all is done by one system that is IoT Cloud.

It is an intimidating high-performance network that connects servers together to optimize the performance of data process that is being processed by many devices at once. It also helps in controlling traffic and delivering accurate data analytics results.

One of the most important components of IoT cloud is database management that is distributed in nature. The cloud basically combines many devices, gateways, protocols, devices and a data store that can be analyzed efficiently. These systems are used by many companies in order to have improved and efficient data analysis that can help in the development of the services and products. In addition to this, it also helps in forming an accurate strategy that can help in building an ideal business model.

3.9.5 User Interface

This is another factor on which IoT ecosystem depends immensely. It provides a visible and physical part that can be easily accessed by the user. It is important for the developer to create a user-friendly interface that could be accessed without putting any extra efforts in it and that can help in easy interaction.

With the help of advancement, there are various interactive designs that could be used easily and that can easily solve any complex query. For examples, at home people have started to use the colorful touch panels instead of the hard controls that were used earlier. It is increasing day by day as now the touchpads are also launched that can switch on the air conditioners from a distance.

This has set out a trend for the digital generations and have managed to hype up today's competitive market. The user interface is the first thing that user pay attention to before buying a device. Even customers are oriented to buy the devices that are user-friendly and less complex that could be used with wireless connectivity.

3.9.6 Standards and Protocols

The webpages are now using the HTML format with the cascading style sheet. This has made the internet more stable and reliable service to use. They are the most used standard protocols making it no just friendly but easily acceptable. However, IoT doesn't have that standard.

It is important to choose a platform the IoT that can help in determining the way your platform will interact with the system. Thus, you will be able to have an interaction with devices and networks with the same standard as yours. It is important for having the same protocol to have a successful interaction.

3.9.7 Database

Internet of things are increasing dynamically and is all dependent on data that are used immensely in the data centers. It is essential to have a proper database system that can store and manage the data that is being gathered from various device and end-users. There are also various management tools that offer many automated features that help in easy accumulation of data stored and managed in bulk at the same place.

3.9.8 Automation

As mentioned above, the database system is using the automotive features that help in managing data and accumulating it. However, the data management is the only limited thing that is used by the internet of things. It is now used for a much more advanced version that allows the automatic adjustment of the wireless things. For example, you can easily control light with a click of the remote. The air conditioner is now connected to your smartphone and you can switch them on and off whenever you want. Even it is possible to play with the temperature.

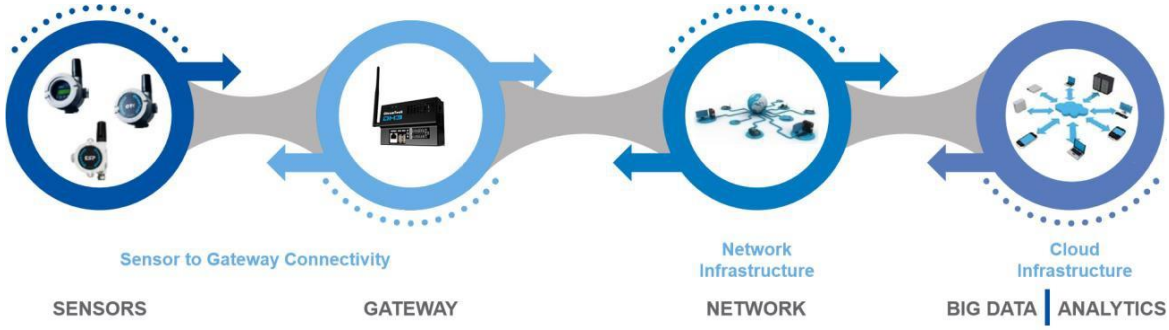


Figure 3.1 IOT Ecosystem

3.10 Working of IOT

The entire process starts with the devices themselves, such as smartphones, digital watches, electronic appliances which securely communicate with an internet of things platform. IoT platform collects and combines data from multiple devices and platforms and applies analytics to share the most valuable data with applications to address industry-specific needs. Just like Internet has changed the way we work & communicate with each other, by connecting us through the World Wide Web (internet), IoT also aims to take this connectivity to another level by connecting multiple devices at a time to the internet thereby facilitating man to machine and machine to machine interactions.

People who came up with this idea, have also realized that this IOT ecosystem is not limited to a particular field but has business applications in areas of home automation, vehicle automation, factory line automation, medical, retail, healthcare and more.

Let's start with a simple real-life example- a car after taking a long road trip, the driver notices that his check engine light and finds that it needs to look by a mechanic, but doesn't know if there is something minor or serious. As it finds out, the sensor that triggered the check engine light monitors the pressure in the inner break line. This sensor is one of the many sensors present in the car which are constantly communicating with each other.

A component called the diagnostic bus gathers the data from all these sensors and then passes it to the gateway in the car. The gateway collects and sorts the data from different sensors. This way only the most relevant diagnostic information will transmit to the manufactures platform.

Before this connection to happen, the car's gateway and platform must register with each other and confirm a secure communication. The platform keeps on constantly gathering and storing information from hundreds of cars worldwide, building a record in a database. The manufacturer has added rules and logic to the platform. The platform triggers an alert in his car, after sensing the brake fluid has dropped below the recommended level. The manufacturer then sends him an appointment for servicing of his car, and the car's problem is rectified.

- **Here, 4 fundamental components of IOT system, which tells us how IOT works:**

3.10.1 Sensors/Devices

First, sensors or devices help in collecting very minute data from the surrounding environment. All of this collected data can have various degrees of complexities ranging from a simple temperature monitoring sensor or a complex full video feed.

A device can have multiple sensors that can bundle together to do more than just sense things. For example, our phone is a device that has multiple sensors such as GPS, accelerometer, camera but our phone does not simply sense things.

The most rudimentary step will always remain to pick and collect data from the surrounding environment be it a standalone sensor or multiple devices.

3.10.2 Connectivity

Next, that collected data is sent to a cloud infrastructure but it needs a medium for transport.

The sensors can be connected to the cloud through various mediums of communication and transports such as cellular networks, satellite networks, Wi-Fi, Bluetooth, wide-area networks (WAN [20]), low power wide area network and many more.

Every option we choose has some specifications and trade-offs between power consumption, range, and bandwidth. So, choosing the best connectivity option in the IOT system is important.

3.10.3 Data Processing

Once the data is collected and it gets to the cloud, the software performs processing on the acquired data.

This can range from something very simple, such as checking that the temperature reading on devices such as AC [21] or heaters is within an acceptable range. It can sometimes also be very complex, such as identifying objects (such as intruders in your house) using computer vision on video. But there might be a situation when a user interaction is required, example- what if when the temperature is too high or if there *is* an intruder in your house? That's where the user comes into the picture.

3.10.4 User Interface

Next, the information made available to the end-user in some way. This can achieve by triggering alarms on their phones or notifying through texts or emails.

Also, a user sometimes might also have an interface through which they can actively check in on their IOT system. For example, a user has a camera installed in his house, he might want to check the video recordings and all the feeds through a web server.

However, it's not always this easy and a one-way street. Depending on the IoT application and complexity of the system, the user may also be able to perform an action that may backfire and affect the system. For example, if a user detects some changes in the refrigerator, the user can remotely adjust the temperature via their phone.

There are also cases where some actions perform automatically. By establishing and implementing some predefined rules, the entire IOT system can adjust the settings automatically and no human has to be physically present. Also in case if any intruders are sensed, the system can generate an alert not only to the owner of the house but to the concerned authorities.

3.11 IOT Communication Protocols

Several Communication Protocols and Technology used in the IOT. Some of the major IoT technology and protocol (IOT Communication Protocols) are Bluetooth, Wi-Fi, Radio Protocols, LTE-A [22], and Wi-Fi-Direct. These IoT communication protocols cater to and meet the specific functional requirement of an IoT system.

3.11.1 Bluetooth

An important short-range IoT communications Protocols / Technology. Bluetooth, which has become very important in computing and many consumer product markets. It is expected to be key for wearable products in particular, again connecting to the IoT albeit probably via a smartphone in many cases. The new Bluetooth Low-Energy (BLE [23]) – or Bluetooth Smart, as it is now branded – is a significant protocol for IoT applications. Importantly, while it offers a similar range to Bluetooth it has been designed to offer significantly reduced power consumption.

3.11.2 Wi-Fi

Wi-Fi connectivity is one of the most popular IoT communication protocol, often an obvious choice for many developers, especially given the availability of Wi-Fi within the home environment within LAN [24]s. There is a wide existing infrastructure as well as offering fast data transfer and the ability to handle high quantities of data. Currently, the most common Wi-Fi standard used in homes and many businesses is 802.11n, which offers range of hundreds of megabit per second, which is fine for file transfers but may be too power-consuming for many IoT applications.

3.11.3 NFC

NFC [25] (Near Field Communication) is an IoT technology. It enables simple and safe communications between electronic devices, and specifically for smartphones, allowing consumers to perform transactions in which one does not have to be physically present. It helps the user to access digital content and connect electronic devices. Essentially it extends the capability of contactless card technology and enables devices to share information at a distance that is less than 4cm.

3.11.4 LoRaWAN

LoRaWAN [26] is one of popular IoT Technology, targets wide-area network (WAN) applications. The LoRaWAN design to provide low-power WANs with features specifically needed to support low-cost mobile secure communication in IoT, smart city, and industrial applications. Specifically meets requirements for low-power consumption and supports large networks with millions and millions of devices, data rates range from 0.3 kbps to 50 kbps.

3.11.5 MQTT (Message Queue Telemetry Transport)

MQTT [27] is a simple messaging protocol, designed for constrained devices with low-bandwidth. So, it's the perfect solution for Internet of Things applications. MQTT allows you to send commands to control outputs, read and publish data from sensor nodes and much more.

Therefore, it makes it really easy to establish a communication between multiple devices.

➢ MQTT Basic Concepts

In MQTT there are a few basic concepts that you need to understand:

- Publish/Subscribe
- Messages
- Topics
- Broker

1) MQTT – Publish/Subscribe

The first concept is the publish and subscribe system. In a publish and subscribe system, a device can publish a message on a topic, or it can be subscribed to a particular topic to receive messages.

Figure 3.2 Communication between devices

2) MQTT – Messages

Messages are the information that you want to exchange between your devices. Whether it's a command or data.

3) MQTT – Topics

Topics are the way we register interest for incoming messages or how we specify where we want to publish the message.

4) MQTT – Broker

At last, you also need to be aware of the term *broker*.

The broker is primarily responsible for receiving all messages, filtering the messages, decide who is interested in them and then publishing the message to all subscribed clients.

3.11.5.1 MQTT Features

1) A publish/subscribe messaging model that facilitates one-to-many distribution. The sending applications or devices do not need to know anything about the receiver, not even its address.

2) Ideal for constrained networks (low bandwidth, high latency, data limits, fragile connections). MQTT message headers are kept as small as possible; the fixed header is just 2 bytes. Its on demand, push-style message distribution keeps network utilization low.

3) Multiple service levels allows flexibility in handling different types of messages. Developers can designate that messages will be delivered “at most once”, “at least once”, or “exactly once”.

4) Designed specifically for remote devices with little memory or processing power. Minimal headers, a small client footprint and limited reliance on libraries make MQTT ideal for constrained devices.

5) Easy to use and implement with a simple set of command messages. Many applications of MQTT can be accomplished using just CONNECT, PUBLISH, SUBSCRIBE and DISCONNECT.

6) Built-in support for loss of contact between client and server. The server is informed when a client connection breaks abnormally, allowing the message to be re-sent or preserved for later delivery.

3.11.5.2 Disadvantages

Security: MQTT is unencrypted by default. This makes it natively unsecured and requires you to take additional steps and absorb some overhead to make sure TLS/SSL [28] is implemented. If not, any communication over MQTT, including username and password, is open to hackers.

3.12 IOT Hardware

IoT Hardware includes a wide range of devices such as devices for routing, bridges, sensors etc. These IoT devices manage key tasks and functions such as system activation, security, action specifications, communication, and detection of support-specific goals and IoT Hardware components like the Arduino Uno, Raspberry Pi and Beaglebone.

3.12.1 Node MCU

NodeMCU [29] is an open source IOT platform, the NodeMCU development board is a powerful solution to program microcontrollers and be part of the Internet of Things (IoT). The NodeMCU development board, based on ESP8266EX, is a cute module with a microcontroller, integrated Wi-Fi receiver, and transmitter. NodeMCU supports several programing languages; hence, it is very easy to upload programs from any computer over a micro-USB port.

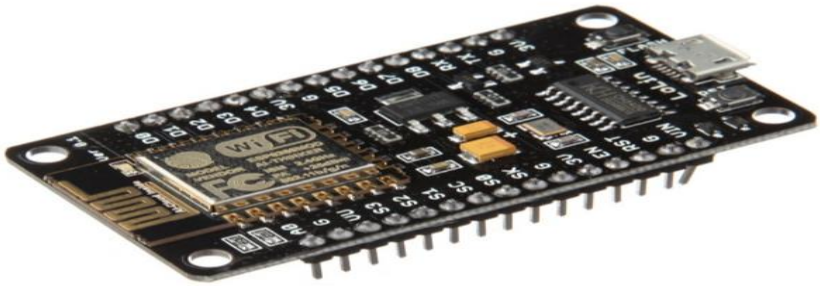


Figure 3.3 NodeMCU

3.12.2 ZigBee

ZigBee is an open global standard for wireless technology designed to use low-power digital radio signals for personal area networks. ZigBee operates on the IEEE 802.15.4 specification and is used to create networks that require a low data transfer rate, energy efficiency and secure networking. It is employed in a number of applications such as building automation systems, heating and cooling control and in medical devices.

ZigBee is designed to be simpler and less expensive than other personal are network technologies such as Bluetooth.

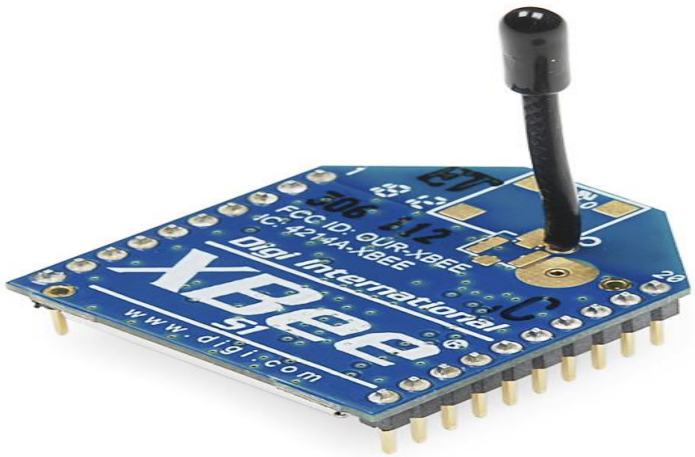


Figure 3.4 ZigBee

3.13 IOT Software

The software and the programming languages on which IoT works uses very common programming languages that programmers use and already know.

IoT software encompasses a wide range of software and programming languages from general-purpose languages like C++ and Java to embedded-specific choices like Google's Go language or Parasail.

3.13.1 C and C++

The C programming language has its roots in embedded systems—it even got its start for programming telephone switches. It's pretty ubiquitous, that is, it can be used almost everywhere and many programmers already know it. C++ is the object-oriented version of C, which is a language popular for both the Linux OS and Arduino embedded IoT software systems. These languages were basically written for the hardware systems which makes them so easy to use.

3.13.2 Python

There has been a recent surge in the number of python users and has now become one of the "go-to" languages in Web development. Its use is slowly spreading to the embedded control and IoT world—specially the Raspberry Pi processor. Python is an interpreted language, which is, easy to read, quick to learn and quick to write. Also, it's a powerhouse for serving data-heavy applications.

3.14 Applications of IOT

The ability to network embedded devices with limited CPU, memory and power resources means that Internet of Things (IoT) finds applications in nearly every field. Few examples,

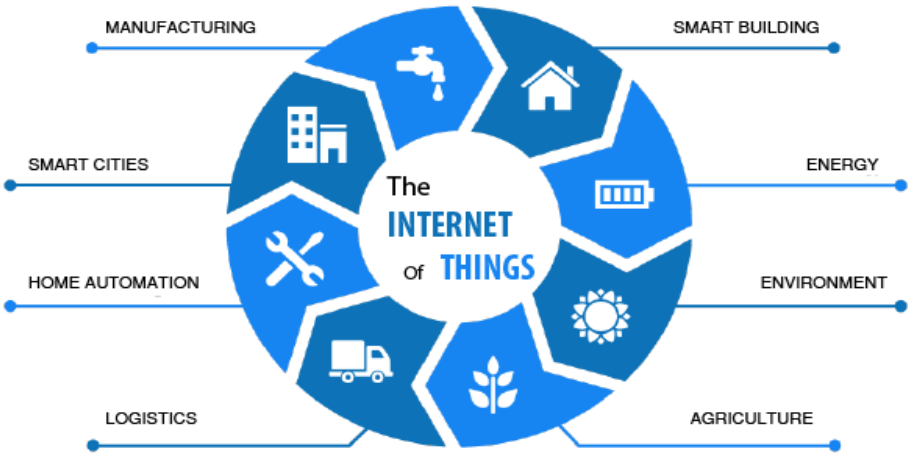


Figure 3.5 Applications of IOT

Home Automation (also known as smart home devices) such as the control and automation of lighting, ventilation, air conditioning (HVAC [30]), robotic vacuums, air purifiers, ovens or refrigerators that use Wi-Fi for remote monitoring.

1) **Wearable technology** which includes smart watches, fitness trackers, VR [31] headsets and more…
2) **Environment:** IoT application technologies that have sensors can be used to monitor air and water quality, soil or atmospheric conditions, and even the movements of wildlife. The IoT devices can also be used in applications such as tsunami early-warning systems to enable authorities to offer more effective responses and aid.
3) **Utilities** doing real time grid assessment from devices sitting on the grid
4) **Logistics** firms designing real-time visibility into location and condition of assets
5) **Infrastructure:** The IoT can also be applied to monitor and control operations of urban and rural infrastructure such as bridges and railway tracks. IoT can help to schedule repair and maintenance activities in a well-organized manner.
6) **Manufacturers** predicting when equipment will need maintenance
7) **Insurers** increasing revenue through asset monitoring
8) **Medical and Healthcare:** IoT devices can facilitate remote health monitoring and emergency notification systems such as blood pressure and heart rate monitors.
9) **Retail:** Retailers creating more personalized in-store shopping experiences
10) **Banks** providing better offers and more engagement via tellers and ATMs.

3.15 Advantages and Disadvantages of IOT

3.15.1 Advantages

1) **Data:** The more the information, the easier it is to make the right decision. Knowing what to get from the grocery while you are out, without having to check on your own, not only saves time but is convenient as well.
2) **Tracking:** The computers keep a track both on the quality and the viability of things at home. Knowing the expiration date of products before one consumes them improves safety and quality of life. Also, you will never run out of anything when you need it at the last moment.
3) **Time:** The amount of time saved in monitoring and the number of trips done otherwise would be tremendous.
4) **Money:** The financial aspect is the best advantage. This technology could replace humans who are in charge of monitoring and maintaining supplies.

3.15.2 Disadvantages

1) **Compatibility:** As of now, there is no standard for tagging and monitoring with sensors. A uniform concept like the USB or Bluetooth is required which should not be that difficult to do.

2) **Complexity:** There are several opportunities for failure with complex systems. For example, both you and your spouse may receive messages that the milk is over and both of you may end up buying the same. That leaves you with double the quantity required. Or there is a software bug causing the printer to order ink multiple times when it requires a single cartridge.
3) **Privacy/Security:** Privacy is a big issue with IoT. All the data must be encrypted so that data about your financial status or how much milk you consume isn't common knowledge at the work place or with your friends.
4) **Safety:** There is a chance that the software can be hacked and your personal information misused. The possibilities are endless. Your prescription being changed or your account details being hacked could put you at risk. Hence, all the safety risks become the consumer's responsibility.

3.16 The Future of IOT

The future of IoT presents an opportunity to connect all sorts of different devices, collect many different types of data, and learn from it without having to sort it all out first. In the future of IoT, we may be able to learn from things like wind turbines, scissor lifts or blood analyzers before we even know what we're looking for or trying to accomplish.

4. Introduction

You can enable artificial swarms of robots, new approaches for search and rescue missions. Our project is a new and innovative idea to help people know where mine without prejudice, and useful hard to reach environments. And implemented by Internet objects, industrial automation, artificial intelligence, and embedded systems.

Can be developed according to the needs of the market and the environment. For example, if it is possible to change the wheels with stronger wheels to withstand desert heights. The main robot can replace drones that scan Earth site, locate mines and send data to the slave. This includes higher resolution on mine sites and speed of implementation. The sensor can be placed inside the robot based on the function used.

This application deals with our project we'll use "Particle Swarm" swarm inspired by this approach from other insects swarm consists of one robot (master) and two slaves in detecting mines in an unfamiliar environment, on the surface or underground using metal detector.

4.1 Metal Detector

Metal detectors are portable electronic devices used for detecting the presence of any metal within close range. These instruments operate by sensing changes in magnetic waves caused by being in close proximity to metal.

This project focuses on the adaptation, simulation and construction of a commonly available schematic for a Pulse Induction (PI [32]) metal detector.

The background information of the history and uses of metal detectors is presented as well as the design criteria for out particular project.

The theory behind how a basic PI metal detector works is examined, along with the basic details of a readily available design for a detector. A detailed examination of the chosen schematics and the function of each component is examined and explained, as well as explanations for certain choices of component values. The results of a computer simulation using Pspice are shown, and then the results of the actual construction of a breadboard prototype, along with the problems encountered are examined.

The aim of our project was to create a circuit that was capable of detecting metal. It had to be battery powered and use a commonly available and understood design.

In addition to this, the circuit design had to be relatively simple and compact so as to fit on a size limited Printed Circuit Board (PCB [33]) board (due to the use of the EAGLE PCB layout tool).

4.1.1 History of the Metal Detector

Metal detectors apparently date back to the shooting of US President James A. Garfield in July 1881. One of the bullets aimed at the President lodged inside his body and couldn't be found. Telephone pioneer Alexander Graham Bell promptly cobbled together an electromagnetic metal-locating device called an induction balance, based on an earlier invention by German physicist Heinrich Wilhelm Dove. Although the bullet wasn't found and the President later died, Bell's device did work correctly, and many people credit it as the very first electromagnetic metal locator. However, it was Gerhard Fischer's portable metal detector that became widely used after 1925.

4.1.2 Existing Mine Clearing Methods

Detecting the precise location of each individual mine is the first task in clearing the landmines. Demining is a process involving surveys, mapping and minefield marking and its actual removal from the ground. Mines clearness divided into (Military, Humanitarian)

The military clearance process just clears their strategic pathways for the soldier's movements at war which is also known as breaching.

Humanitarian clearance process is a social cause to restore peace and security at a community level. It is a challenging task to clear out the land from landmines and unexploded ordnance which.

4.1.3 Mine Detection and Sensing Technologies

Demining steps cannot be carried out if location of mines are unknown. Thus, mine detection, and in general, mine sensing technologies are a major step and currently a slow process in demining. The target of technology progress is to increase the probability of the detection with low probability of false alarms. Improvements are targeted on sensor fusion and careful study of the limitations of the tools from each.

Location, environment and soil composition. The outcome of developing effective detection is to speed up the detection rate, reliability, accuracy and efficient distinguishing between explosives and other buried debris ensuring more safety of the de-miners and the environment. Along with the development, testing is also important to simulate the minefield conditions for real scenarios to validate the latest technology for its reliability and

limitations. The procedures of actual operations in the mine field region help benchmarking the performance of the equipment, systems and the algorithms leading to enhancements.

The sensing and detection of landmines from this challenge would enable the use of autonomous robots instead of manual clearance. This project became the main motivation for working on the strategy towards the optimization, neural network classification and autonomous navigation.

4.1.4 Remote Sensing Technology

The most demanding and challenging problems is reliable detection of the mines in all kind of terrain. Some of the sensors used for detection are ground penetrating radar, laser Doppler vibrometer, Polari-metric Infrared (IR [34]) and hyper spectral methods, electromagnetic induction sensor - metal detector, explosive vapor detection, X-ray backscatter, and Acoustic/seismic systems uses sound or seismic waves for identifying mines.

4.1.4.1 Ground Penetrating Radar (GPR)

With a manufacturing of plastic based mines to make the mine detection difficult, there came a need to use GPR [35] s for detecting small amounts of metal being used in the evolved AP [36] mines. The advantage is that they detect both metallic and non-metallic objects buried in the ground with an additional factor of recognizing its shape. GPRs were initially applied to study subsurface structures or buried objects in civil engineering, geology, archeology and soil and environment science. Many anti-personnel mines requires the use of large frequency signal with GPR systems. GPR helps in providing a 3D depth image of the ground. However there are limitations. For detecting small objects, high frequency signals are required for better resolution which reduces soil penetration and increases image clutter.

GPR is one of the best technology for soil research. However, mine detection becomes complicated due to various reasons. The results are affected by the soil surface irregularities and the soil conditions. GPR in fusion with metal detector have been very fruitful in identification of mines. Also signal processing techniques using GPR which filters out the signal for mine by nullifying the soil conditions (soil type, moisture content, and weather conditions) reduces false alarms. Few of the techniques developed are Automatic Target Recognition, methods based on 2D and 3D texture analysis, background removal, and many other statistical approach.

4.1.4.2 Infrared (IR) and hyper spectral methods

Mines retain or release heat at a different rate from their surroundings. Infrared methods can detect this electromagnetic radiation over the mine surface or the soil immediately over the mine. Thermal sensors are of use for identification between soil and mines based on temperature variation. This method is limited to homogeneous soil. Light is the medium of data collection which makes image acquisition faster. Anthropogenic materials with the

smooth surface feature can preserve polarization; allowing mine identification. Also the light scattering may be affected due to the mutation of the soil with mines present.

Thermal detection methods have also been exploited due to variations in temperature near mines compared to its surrounding; considering the fact that the mine. Would be hotter during the day were as would cool off faster in the night. The discrepancy in the temperature can be removed by inducing laser illumination or microwave radiation

4.1.4.3 X-Ray Backscatter

This technique can create a high quality image by passing photons through mines due to its short wavelength. Although X-ray imaging on subsurface is impossible but the backscatter of X-rays can showcase buried and irradiated objects. It relies on the fact of mines and soil having the different mass densities and atomic numbers.

4.1.5 Uses of Metal Detector

1) Don't use metal detectors just to find coins on the beach.
2) You can see them walking through scanners at airports.
3) Scientific research. Archaeologists often frown on untrained using metal detectors for breach of important artifacts, but used correctly, with respect, and metal detectors can be valuable tools in historical research.
4) Used in government agencies and office buildings and schools.
5) Use the most important is the detection of surface and underground mines, and differentiate between mines and metals.

4.1.6 Types of Metal Detectors

Metal detectors were mainly designed and developed for detecting explosives, Refinement of Alexander Bells original invention have led to different types, but they all still work on the same concept of electricity and magnetism.

4.1.6.1 Best Frequency Oscillation

The oscillation detector is very easy to understand, and perhaps the most basic. Its depth of reading is around 2 Feet. All metal detectors have a ring of coil copper rolled around iron or steel. There is one large ring at the base of the detector and one smaller one a little higher up. The coils are both connected to oscillators that generate a frequency.

4.1.6.2 Very Low Frequency Detectors

This detector is most favored by the treasure hunters out there. It is an accurate sensitive device due to its coil, which has a pair of coils. The first coil is the transmitter with an intense electrical current. The second coil acts as the receiver collecting the signals and amplifying them. The transmitter coil works by producing a magnetic field above and

below the ground. Again as the best frequency oscillation this can reach about 2 feet under the ground.

The magic on this detector is that it picks up a different frequency depending on the source.

4.1.6.3 Pulse Indicator

This is one of the latest innovations of metal detecting. Usually used by door staff or guards to detect concealed weapons. It is not that efficient in detecting types of metals like the others. The portable device relies on echo or echolocation only. For example if you were to make a sound in an open room such as a hall the sound would repeat. If you were to do the same in a padded room the sound would be lost. Other detectors will use two coils, the portable detector only uses one or several that work as one.

An electric pulse is sent through the coils where it disappears and the magnetic field is reversed. This generates a short electric current and then disappears. It is the pulsing motion that detects any metal objects although it cannot determine what this maybe. For a portable detector it is one of the most expensive. These are also called pinpoint detectors.

Industrial Metal Detectors – Better known as door-frame detectors. They are often found at airports or secure buildings such as a courthouse. It works solely on the principles of electromagnetism. Not one used for treasure hunting due to its size of course, but it is a great deterrent for criminals trying to enter the building with weapons.

4.1.7 Depth of Metal Detector

- There's no exact answer to that question, unfortunately, because it depends on all kinds of factors, including:

1) The size, shape, and type of the buried metal object: bigger things are easier to locate at depth than small ones.
2) The orientation of the object: objects buried flat are generally easier to find than ones buried with their ends facing downward, partly because that creates a bigger target area but also because it makes the buried object more effective at sending its signal back to the detector.
3) The age of the object: things that have been buried a long time are more likely to have oxidized or corroded, making them harder to find.
4) The nature of the surrounding soil or sand you're searching.
5) Generally speaking, metal detectors work at a maximum depth of about 20–50cm (8–20in).

4.1.8 Types of Metal Detecting Coils

4.1.8.1 Concentric Coils

The transmit coil is on the outside and the receiver coil is on the inside. These benefits the user as it gives the advantage of the coils being wound as large as possible in search diameter and depth detection.

They also provide the most symmetrical field; this makes it easier to pinpoint objects. It's one of most common and versatile search coils because it gives consistency in target identification. The design itself is not so flawless and is more susceptible to interference from ground minerals.

4.1.8.2 Imaging Coils

This is an advanced option of the concentric search coils. It is almost the same accept it has a second receive coil. This enables additional target information for true target sizing as well as target depths capabilities. All the above allow the investor to differentiate trash from treasure of the same conductivity. This is only limited to one type of detector provide by one company.

4.1.8.3 Mono Coils

These coils are only available for Pule Induction detectors. The difference between this coil and the above coils is that the transmitter and receiver coils are located together. Or one coil which acts as both the transmitter and receiver.
The mono coil provides maximum sensitivity but the performance in the mineralized ground.

4.1.8.4 2-Box Coils

In this design the receiver and transmit are separated by a quite a distance of feet. This search coil has a 3 to 4 foot diameter. The 2 box search coils are often used for deeply buried treasure such as relics. It is great for detecting larger objects but may find it difficult detecting smaller objects.

4.1.8.5 Double D Coils

This design was built to reduce any ground interference, recovering performance which may be lost by the concentric coil in mineralized soil. The reason they are called Double D coils is because they are shaped like the letter D. With that set up the coils generate a cancelling effect of ground signals.

The positive detection field runs front to back beneath the center of the design, the other part of the coil which that is negative, cancels the effect of the ground signals. This is what helps maintain the performance over the mineralized ground.

Due to the positive field being small the Double D design is less sensitive than the concentric design of the same size. However over mineralized ground the Double D is much better. This design would be used for such things as relic hunting.

4.1.9 Theory

PI systems may use a single coil as both transmitter and receiver, or they may have two or even three coils working together. This type of metal detector sends powerful, short bursts (pulses) of current through a coil of wire. Each pulse generates a brief magnetic field. When the pulse ends, the magnetic field reverses polarity and collapses very suddenly, resulting in a sharp electrical spike.

This spike lasts a few microseconds (millionths of a second) and causes another current to run through the coil. This current is called the reflected pulse and is extremely short, lasting only about 30 microseconds. Another pulse is then sent and the process repeats. A typical PI-based metal detector sends about 100 pulses per second, but the number can vary greatly based on the manufacturer and model, ranging from a couple of dozen pulses per second to over a thousand.

If the metal detector is over a metal object, the pulse creates an opposite magnetic field in the object. When the pulse's magnetic field collapses, causing the reflected pulse, the magnetic field of the object makes it take longer for the reflected pulse to completely disappear. This process works something like echoes: If you yell in a room with only a few hard surfaces, you probably hear only a very brief echo, or you may not hear one at all; but if you yell in a room with a lot of hard surfaces, the echo lasts longer.

In a PI metal detector, the magnetic fields from target objects add their "echo" to the reflected pulse, making it last a fraction longer than it would without them. A sampling circuit in the metal detector is set to monitor the length of the reflected pulse.

By comparing it to the expected length, the circuit can determine if another magnetic field has caused the reflected pulse to take longer to decay. If the decay of the reflected pulse takes more than a few microseconds longer than normal, there is probably a metal object interfering with it.

The sampling circuit sends the tiny, weak signals that it monitors to a device call an integrator. The integrator reads the signals from the sampling circuit, amplifying and converting them to direct current (DC).

The direct current's voltage is connected to an audio circuit, where it is changed into a tone that the metal detector uses to indicate that a target object has been found. PI-based detectors

are not very good at discrimination because the reflected pulse length of various metals is not easily separated. However, they are useful in many situations in which other non PI based metal detectors would have difficulty, such as in areas that have highly conductive material in the soil or general environment. Also, PI-based systems can often detect metal much deeper in the ground than other systems.

4.1.10 Balance Concepts

Taken from one of the original Tesoro patents, this represents the phase related target response of some common metals, showing two components – "R" and "X", or conductive and magnetic Metallic items and the ground can have characteristics of both. Ideal Ferrite (purely magnetic) would be our 0° reference point.

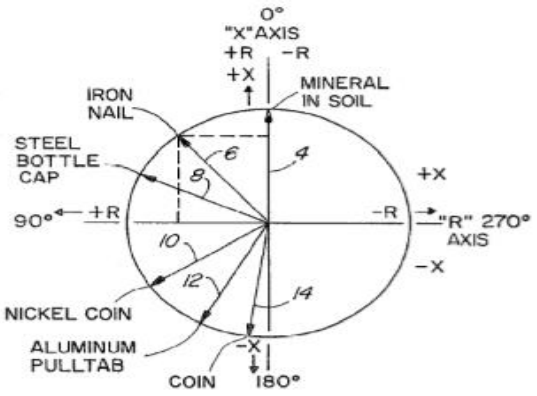


Figure 4.1 Conductivity Scale

This diagram is being used to illustrate a point. In order for a single channel Induction Balance metal detector to reject ground mineralization, it must accept anything that is "more positive" on the phase scale. Unfortunately, when a single channel IB metal detector is adjusted to tune out ground mineralization, iron will be accepted as well.

The way around this problem is to use 2 Channels Consider the block diagram below. In this configuration, the phasing of ØB can be adjusted to match the induced phase shift of the receive signal caused by ground mineralization. This channel will then respond to all metals but not to the ground.

Likewise, the phasing of ØA can be adjusted to ignore unwanted metals, lower on the phase scale but this channel will still see a phase shift from the ground as well when the search coil is moved over the ground. Now only go positive at the same instant will there be detection. There are also filters employed (not depicted) that makes this a "motion mode" only detector. This is the basic scheme used by the TGS [37].

4.1.11 Explanation Schematic

- The schematic can be broken down into about 12 functional blocks:

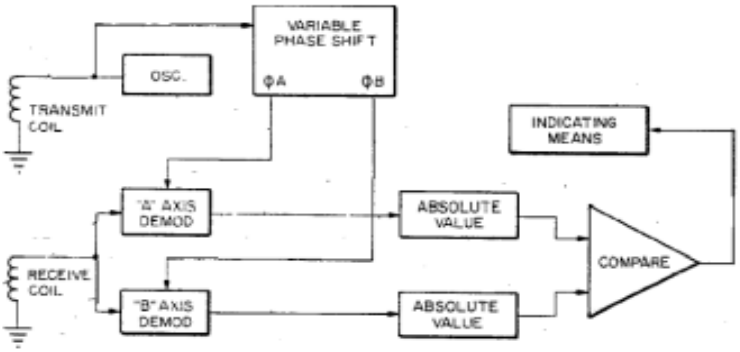


Figure 4.2 Block Diagram of the TGSL

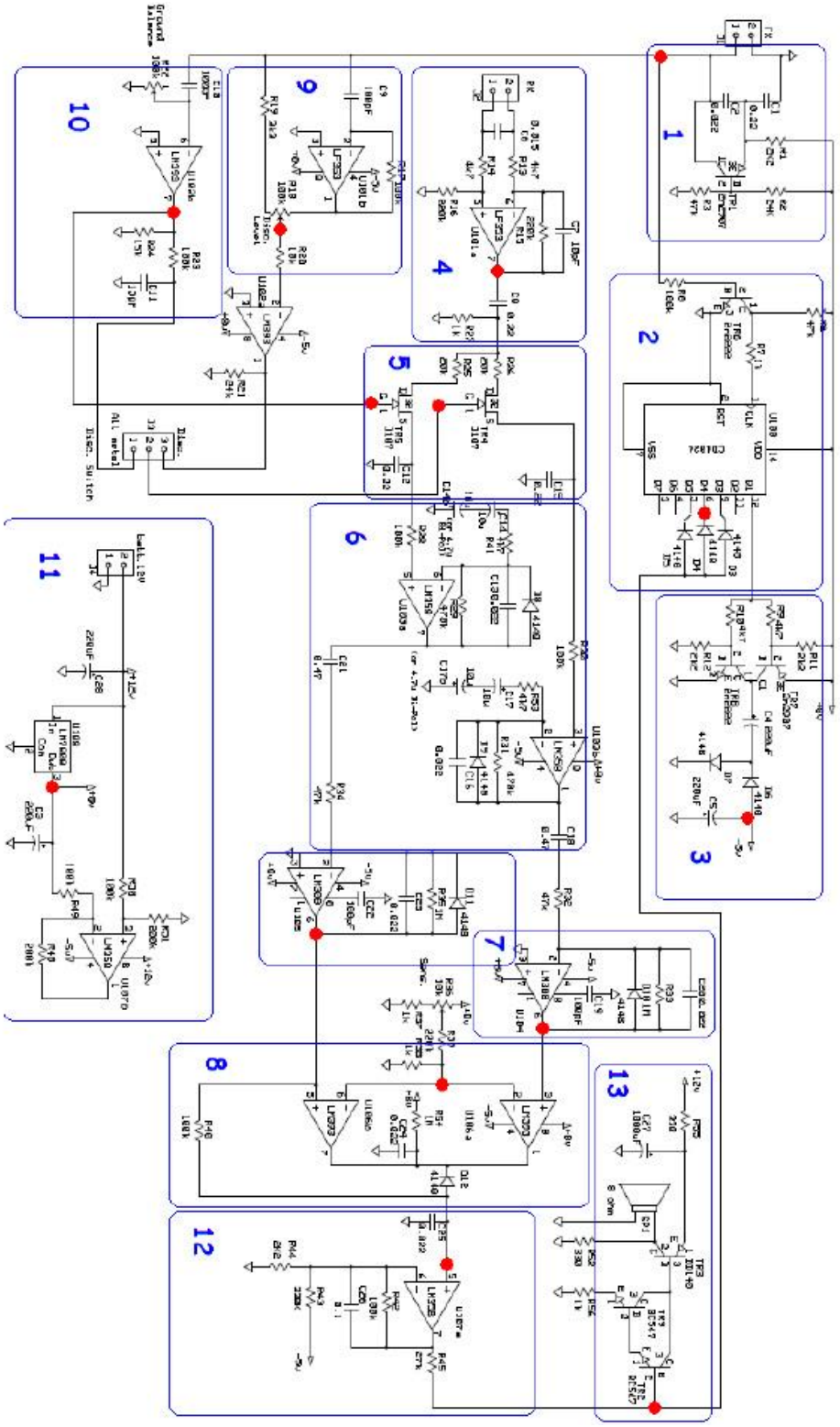


Figure 4.3 Schematic of the TGSL

1) The basic colpits oscillator. Together with the transmit coil attached to J1, it provides a 14.5 kHz low power T x signal to the coil, audio circuit, -5v converter and a signal to use as a phase comparison for the receive circuit. T x amplitude should be in the 16V range.

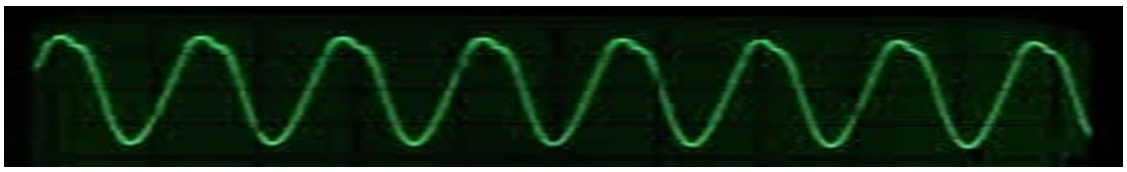


Figure 4.4 J1-1 5V and 20uS/Div.

2) The CD4024 is a binary counter that provides a clocking signal to the -5V converter, block 3. It also provides a synchronous audio signal to the audio driver, circuit block 13. The 14.5 kHz signal from the oscillator is divided by 32 down to about 453Hz by the CD4024. Furthermore, the 3 diodes connected to pins 5, 6 and 9 work together create a very short duty cycle to save on audio power.

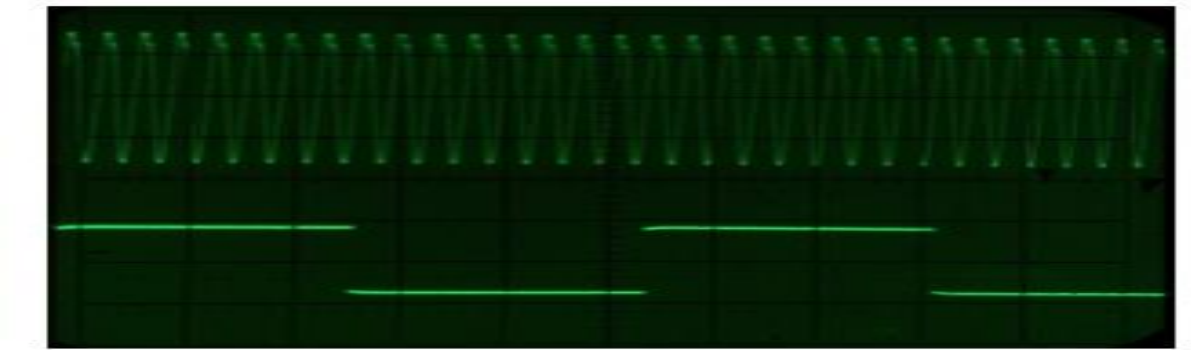


Figure 4.5 J1-1 (top) & U100 Pin 6 (bottom) 5V & .2mS/DIV

3) Converts +8V to -5V to be used as a negative voltage source. This test point may not be exactly -5V but should be relatively close.

4) This is the Rx preamp, U101a. Keep in mind the output, pin 7 as we will see this one again! Also, C6 is very important as this tunes our Rx coil. C6 in parallel with the Rx coil tunes our Rx circuit to about 16.12 kHz (off resonance). This is required to get the proper phase relationships for disc and ground balance to work properly.

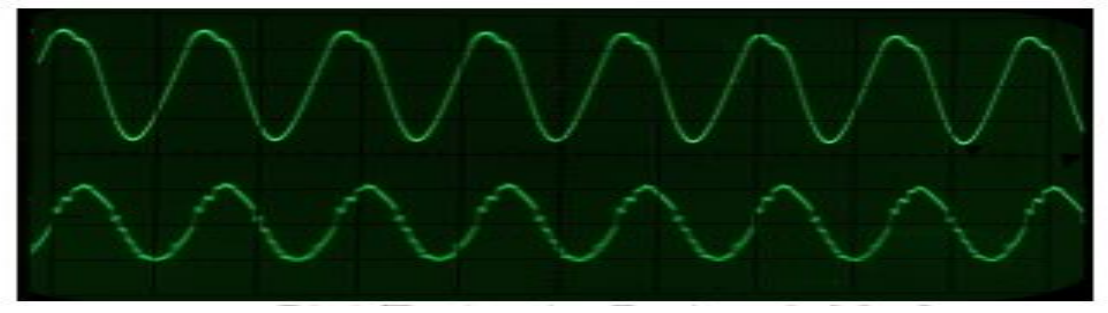


Figure 4.6 J1-1 TX (top) 5v/Div. & 20Us & U101 Pin 7 Rx (bottom)

5) TR4 and TR5 along with C12 and C1 make up our synchronous demodulators. They basically "gate" a sample of the Rx signal by a phase related timing signal from block 9 (disc channel) and block 10 (ground balance channel). Furthermore, C12 and C15 turn our sampled signal into a small D.C value that varies with any phase shift from the Rx signal from block 4.

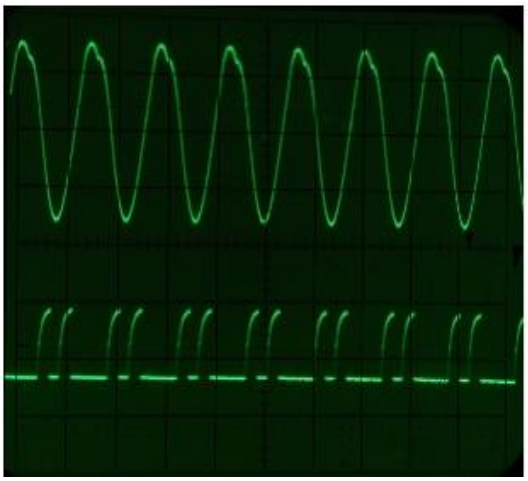


Figure 4.7 (a) Gate – TR4 Disc set at Min

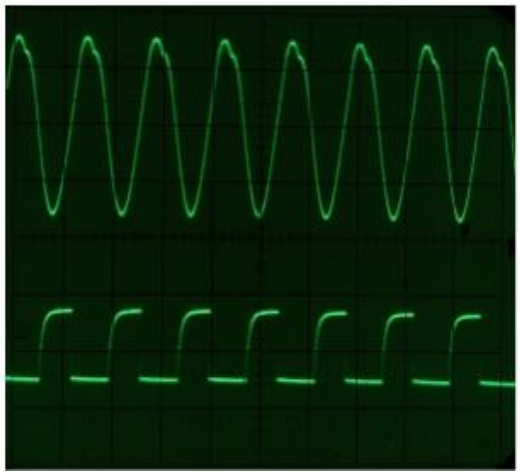


Figure 4.7 (b) Gate – TR4 Disc set at Midrange

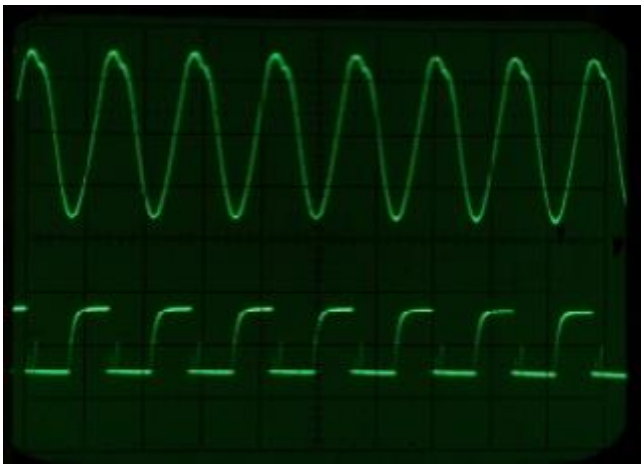


Figure 4.8 Gate – TR4 Disc set at Max

6) U103a and U103b further amplify and filter our D.C. phase related signals this functions as a band pass filter with a frequency centered around 11Hz. This allows a low frequency pulse to pass to the next stage that should be in tune with the approximate sweep speed of the coil passing over a target, this is why proper sweep speed of the coil is important.

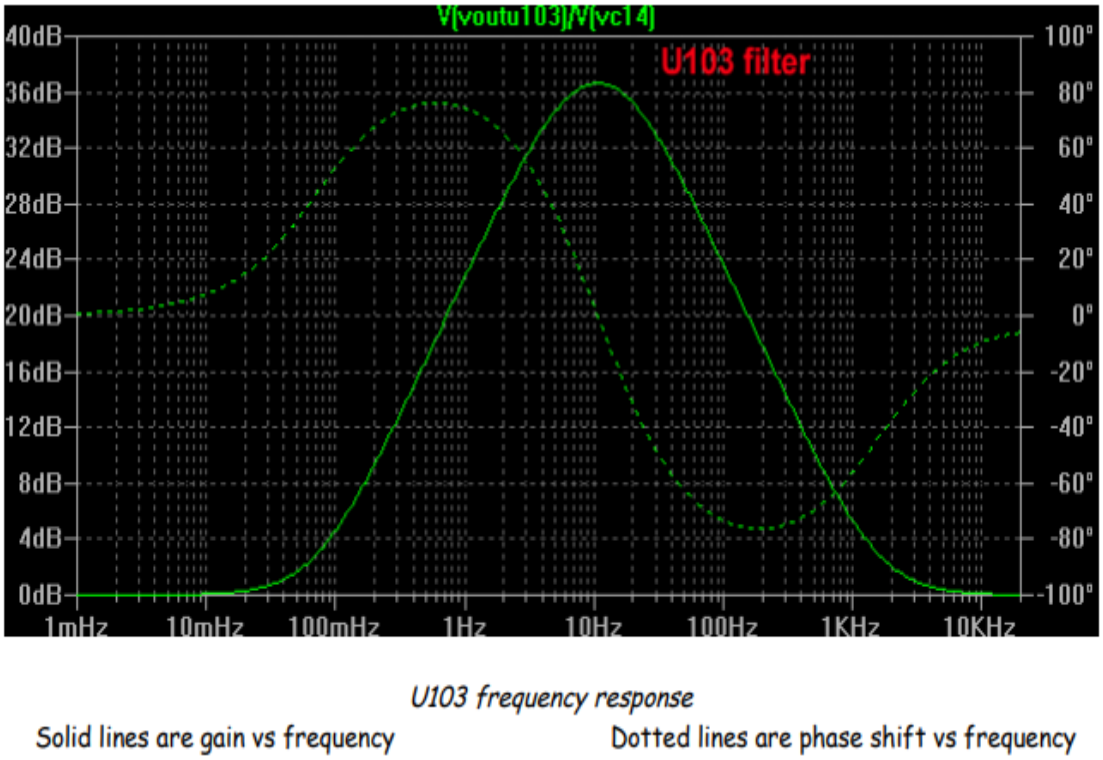


Figure 4.9 Complements of Simon Baker

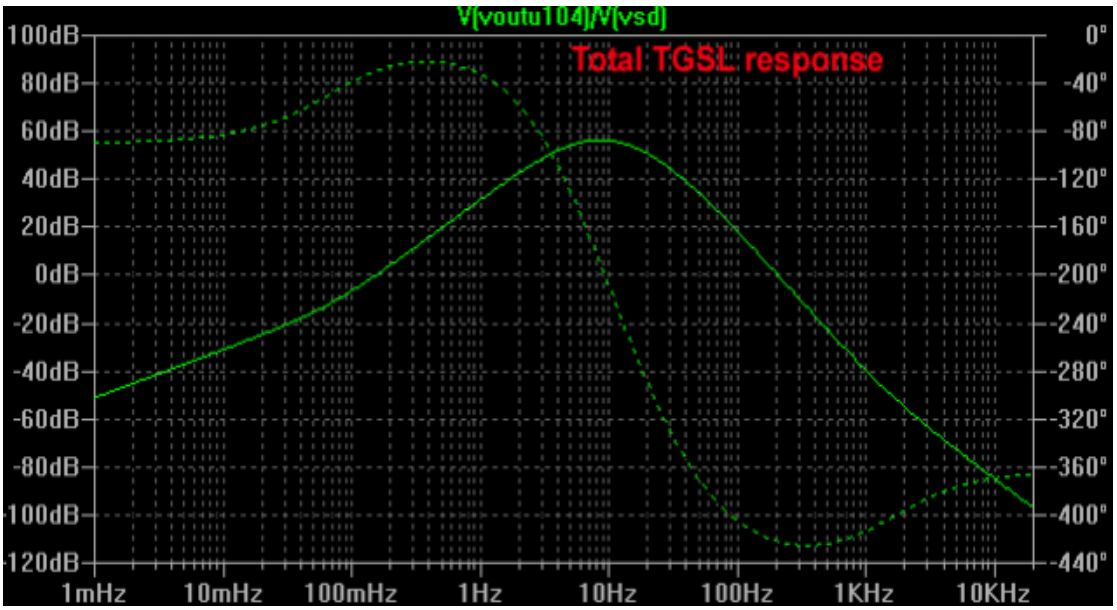


Figure 4.10 Total TGSL response of all stages

7) Provides more D.C. amplification, filtering and limiting. Signals can be measured here.

8) As a D.C. Voltage that will swing + or – 1V or so as targets pass the coil. Keep in mind that U104 Pin 6 is the disc channel and U105 Pin 6 is the GB channel. Also, these test points can give a general indication of electrical "noise" in your environment. High noise levels can make your TGS perform poorly! In general, a measurement 5mV p-p of noise can be considered very well.

9) Here is where some interesting things happen that are not always obvious to the casual observer. Pins 2 and 6 are tied to a reference voltage and the LM393 compares the inputs on pins 3 and 5. The LM393 has the characteristic of being able to "sink" current from the opposite LM393 when the outputs are tied together. This type of IC is commonly referred to as an open collector design. So in order for the diode D12 to stop conducting (and ultimately provide forward bias for TR2 in block 13), BOTH U106 Pin1 and 7 must go high at the same time.

   Now in practice, the disc channel swings wildly as the loop is passed over the mineralized ground while the ground balanced channel only goes positive when the loop passes any type of metal. If the disc channel agrees with the ground balance channel at the same instant, we have detection. The voltage at the test point here should change from about .036V to close to 0v as the sensitivity pot is varied.

10) Provides for positive phase adjustment to the reference signal for the disc channel.

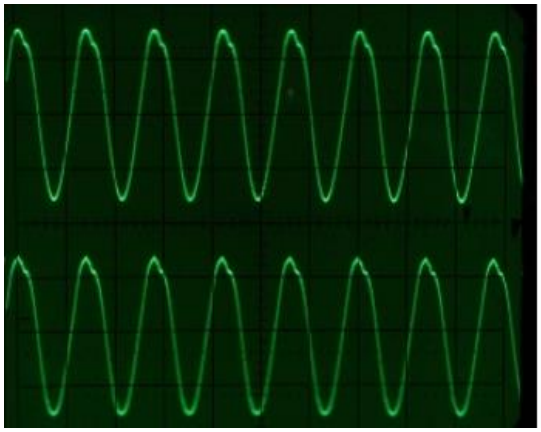


Figure 4.11(a) Test point 9 DISC set at Minimum

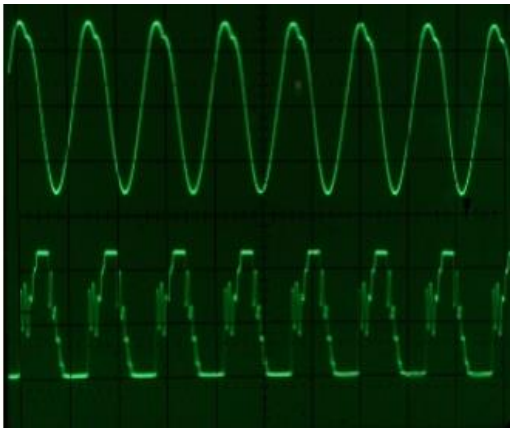


Figure 4.11(b) Test point 9 DISC set at Maximum

11) Provides for negative phase adjustment to the reference signal for the ground balance channel.

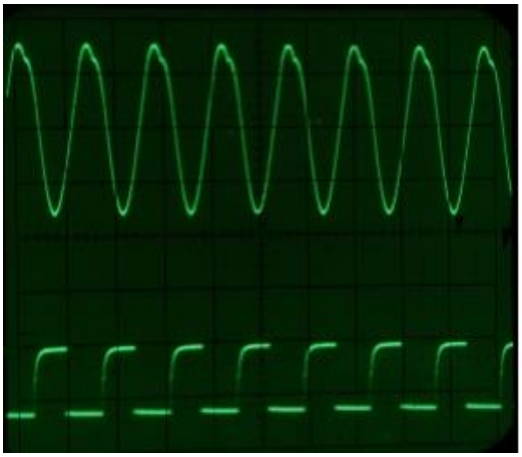


Figure 4.12 (a) GB Potentiometer Min

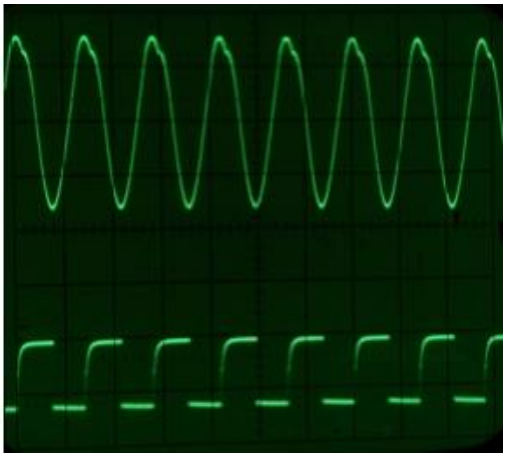


Figure 4.12 (b) GB Potentiometer Max

12) +8V Voltage regulator circuit. TP 11 should test at 8V. Provides a D.C. bias signal to TR2 in block 13 upon target detection. Normally will be at about 6V and will drop to 0V as targets pass the coil.

13) Audio driver circuit. This test point varies from -5V to + Battery voltage as targets pass the coil.

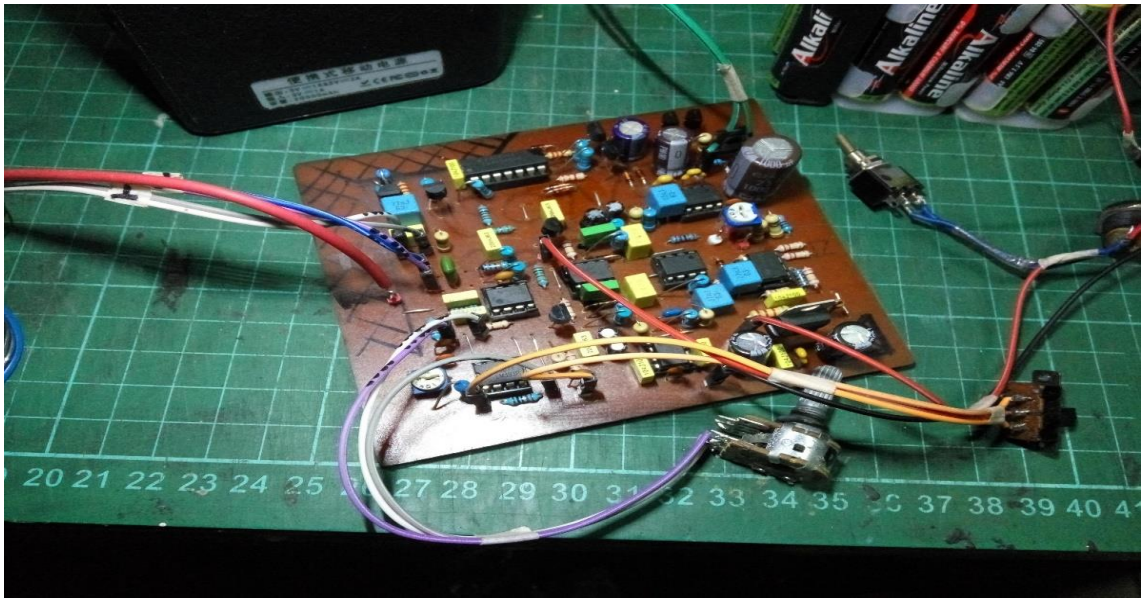


Figure 4.13 Circuit of TGSL

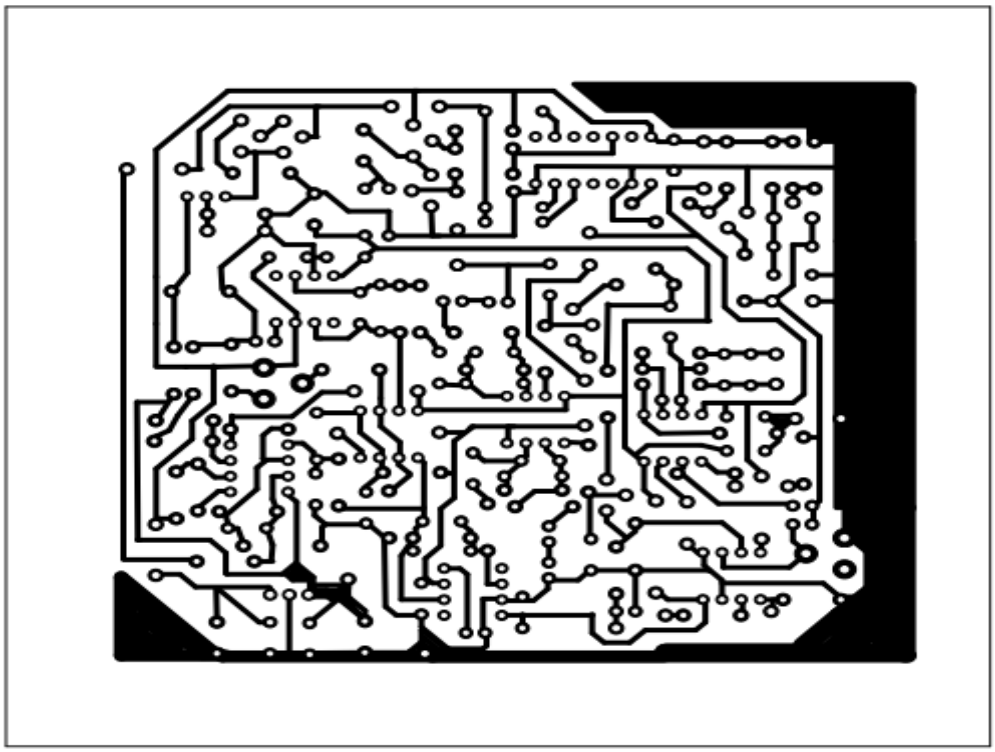


Figure 4.14 PCB of TGSL

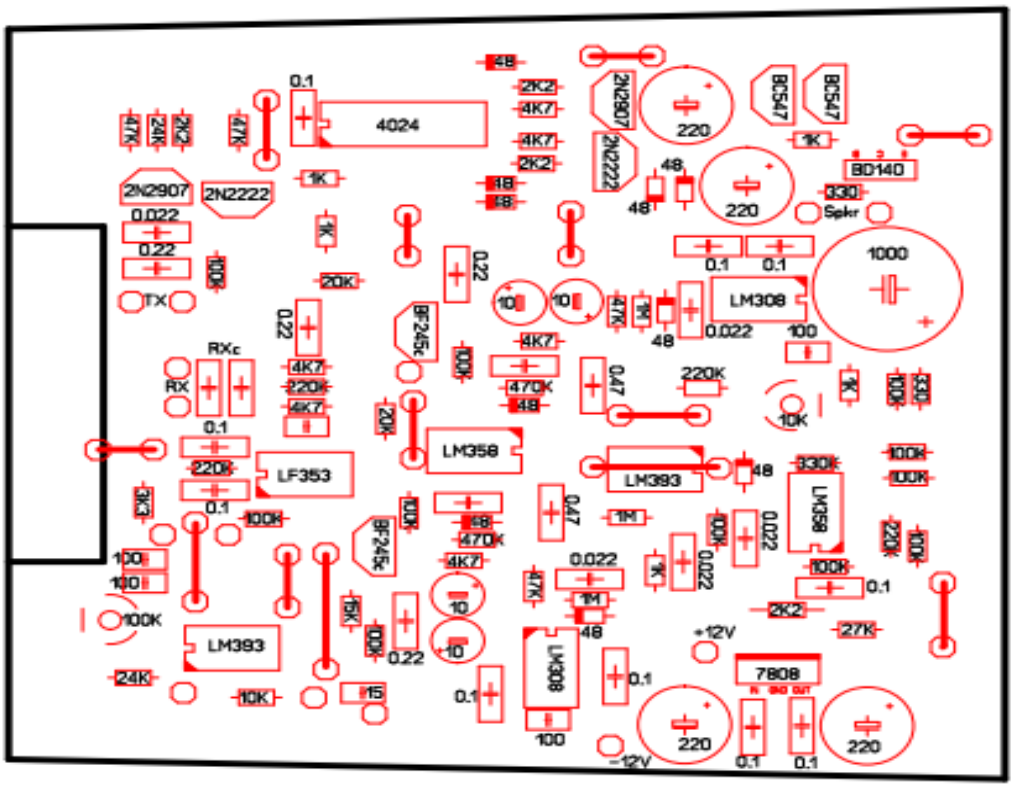


Figure 4.15 Placement of TGSL

Table 4.1 Values TGSL

1	c1	0.22
2	c2	0.022
3	c3	220uf
4	c4	220uf
5	c5	220uf
6	c6	0.015
7	c7	10pf
8	c8	0.22
9	c9	100pf
10	c10	100pf
11	c11	15pf
12	c12	0.22
13	c13	0.022
14	c14	4.7uf
15	c15	0.22
16	c16	0.022
17	c17	4.7uf
18	c18	0.47
19	c19	100pf
20	c20	0.022
21	c21	0.47
22	c22	100pf
23	c23	0.022

24	c24	0.022
25	c25	0.022
26	c26	0.1
27	c27	1000uf
28	c28	220uf
29	D3	4148
30	D4	4148
31	D5	4148
32	D6	4148
33	D7	4148
34	D8	4148
35	D9	4148
36	D10	4148
37	D11	4148
38	D12	4148
39	J1	TX
40	J2	RX
41	J3	Disc Switch
42	J4	batt.12V
43	R3	47K
44	R4	1K
45	R5	330
46	R6	47K

47	R6	22K
48	R6	2K2
49	R7	1K
50	R8	100K
51	R9	4K7
52	R10	4K7
53	R11	2K2
54	R12	2K2
55	R13	5K1
56	R14	5K1
57	R15	220K
58	R16	220K
59	R17	100K
60	R18	100K
61	R19	3K3
62	R20	10K
63	R21	24K
64	R22	100K
65	R23	100K
66	R24	15K
67	R25	20K
68	R26	20K
69	R27	1K

70	R28	100K
71	R29	470K
72	R30	100K
73	R31	470K
74	R32	47K
75	R33	1M
76	R34	47K
77	R35	1M
78	R36	10K
79	R37	1K
80	R38	1K
81	R39	220K
82	R40	100K
83	R41	4K7
84	R42	100K
85	R43	330K
86	R44	4K2
87	R45	27K
88	R48	200K
89	R49	100K
90	R50	100K
91	R51	200K

92	R53	4K7
93	R54	1M
94	SP1	8 ohm
95	TR1	2n2907
96	TR2	BC547
97	TR3	BD140
98	TR4	J107
99	TR5	J107
100	TR6	2n2222
101	TR7	2n2907
102	TR8	2n2222
103	TR9	BC547
104	U100	CD4024
105	U104	LM308
106	U105	LM308
107	U108	LM7808
108	U101a	LF353
109	U101b	LF353
110	U102a	LM393
111	U102b	LM393
112	U103a	LM358
113	U103b	LM358
114	U106a	LM393
115	U106b	LM393
116	U107a	LM358
117	U107b	LM358

4.1.12 Coil Construction

By far, coil making is the most challenging and will most likely determine the overall performance of the completed TGSL [38]. Equally important is the final mechanical construction as the success of the project depends on it. Don't get discouraged though. With a little effort and attention to detail, it can be done! Of course an original coil for the Tesoro Golden Sabre can be purchased, but that would defeat our purpose. Besides, we can build one that outperforms the original.

Diameter: 255mm x 137 mm (one coil)
Wire size: .25mm (30 AWG) enameled

Tx inductance: 6.0mH
Tx resistance: 18 – 25 Ohms
Tx resonance - About 14.5 kHz

Rx Inductance: 6.5mH
Rx resistance 18 – 25 Ohms
Rx resonance with 15nF – About 16.1 kHz

Figure 4.16 the specifications for the most popular coil on the Geotech forum

Regardless of the exact dimensions or number of turns, most importantly we must have the proper inductances and nulling or phasing will be off and proper ground balance setting and/or disc setting may not be possible. Before attempting to wind coils, we must have a way to measure or calculate inductance, operating frequency or both. Most commonly used are inductance meters, oscilloscopes or frequency counters. An alternate method will also be shown.

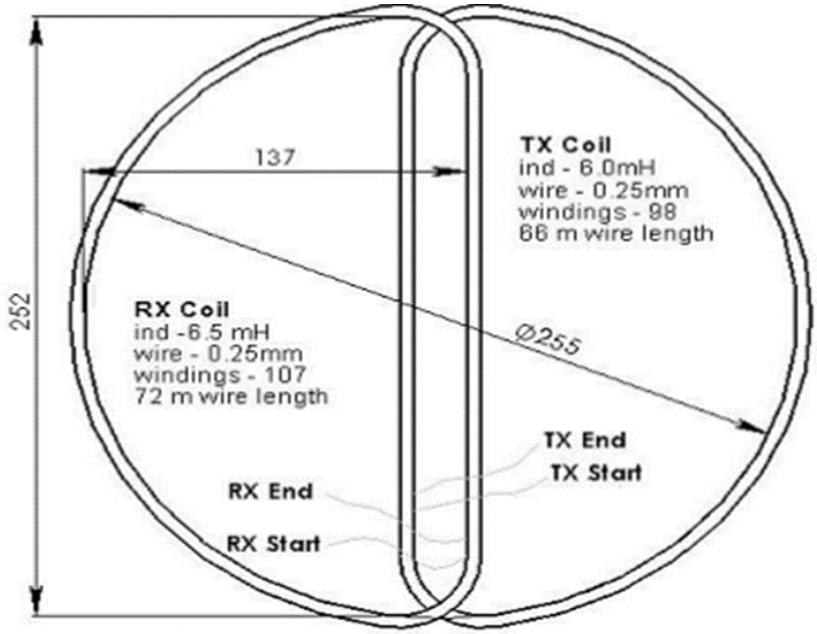


Figure 4.17 Measure inductance

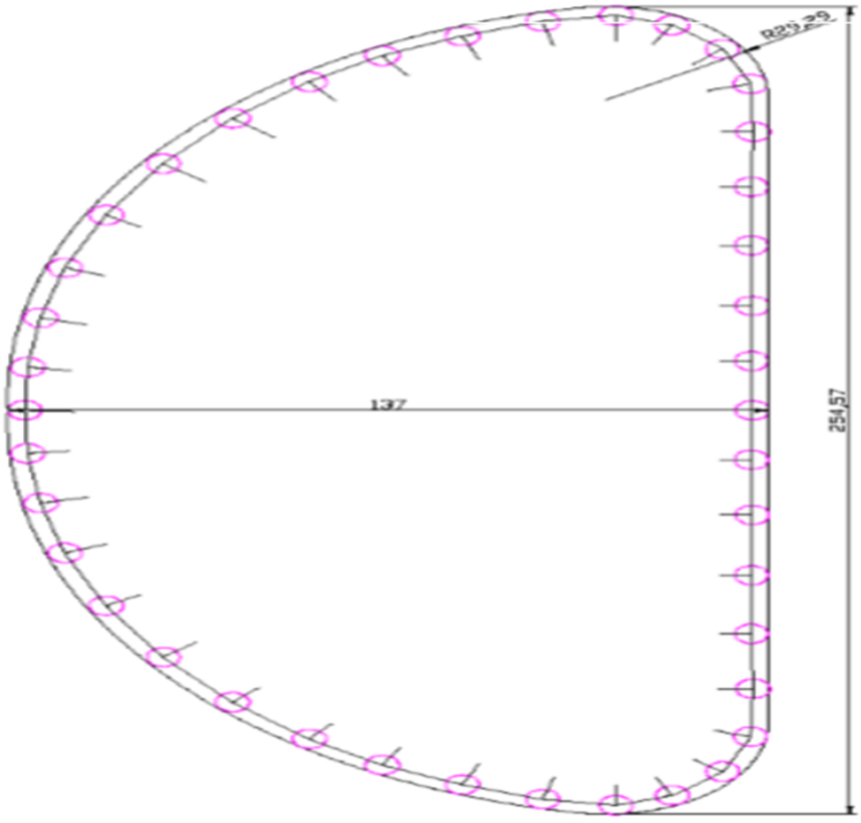


Figure 4.18 Basic Coil pattern below drawn by “Pip3c"

So, how many turns of wire? For this pattern, I am going to suggest 99 turns for TX and 104 turns for Rx (30 AWG enameled wire). In running the calculations for coil inductances a difference of just 5 turns is usually the answer for difference of .5mH between the two coils. I have been able to consistently build "good" coils this way. Due to variations in wire and pattern building, your number of turns needed to achieve proper inductances may vary. This is just a suggested starting point. One method of winding on a form is depicted below.

It's basically two slabs of wood clamped together and the holes for the pattern are drilled. Then wooden dowel pins are driven into the holes. Between the two halves, spacers are used to allow a uniform thickness (4mm) – about the thickness of 2 U.S. Nickels. You can use just one slab of wood and insert dowel pins but it is advantageous to use two sides for a number of reasons.

1) When winding a coil on a single sided form, it is more difficult to wind coils consistently and once constrained, inductances are far more likely change.
2) When winding a coil on a double sided form, turns can be easily added or removed to achieve the proper inductance before constraining.

4.1.13 Double sided form

Once wire is put on a form, it's a good time to verify inductance. This can be done with an inductance meter. Or... my favorite method is just attach the coil to your completed circuit board (J1) and measure the running frequency. The Tx should run at 14.5 kHz.

Next, half of the form must be removed and the wire must be constrained. I use simple dental floss. Note: If wire is wound on a two sided form tightly, very little change in inductance will occur once constrained. At most, it should increase by only 1 mH.

Build your Rx coil in the same manner, except for the number of turns. Also, to check your Rx coil, you can attach it to your completed circuit as well, but attach your Rx coil to J1 (the Tx circuit). Since this is 6.5mH, the oscillator should now run at 13.95 kHz.

Figure 4.19 Double sided form

Again, this is just a quick check and will then be removed! Following constraining the wires, they must be infused with cyan glue, so the coils become very stiff. This is very important as IB detectors are very unforgiving if not mechanically rigid! Once dry, we must add an insulating layer of vinyl electrical tape. This is important as it helps prevent shorts to the shields that will be applied next.

4.1.14 Coil Shielding

Metal detectors are portable electronic devices used for detecting the presence of any metal within close range. These instruments operate by sensing changes in magnetic waves caused.

Coil shielding is not optional. It is necessary to eliminate the effects of ground capacitance, static charges and to help eliminate noise from external electrical sources. The idea is for the coils to "see" the capacitance of the shields, which will be at the same potential as the circuit ground, and not earth ground, rocks or vegetation.

Choices for shielding material can be aluminum foil, aluminum, copper, or lead tape, "Scotch 24" shielding tape or metalized Mylar (polyester film), just to name a few. Plain aluminum kitchen foil is usually the easiest to obtain and works just fine. I prefer metalized Mylar as final sensitivity of the coil is noticeably better with this material. Sources of conductive Mylar include Mylar balloons, ribbons or bows from a party store, or reflective.

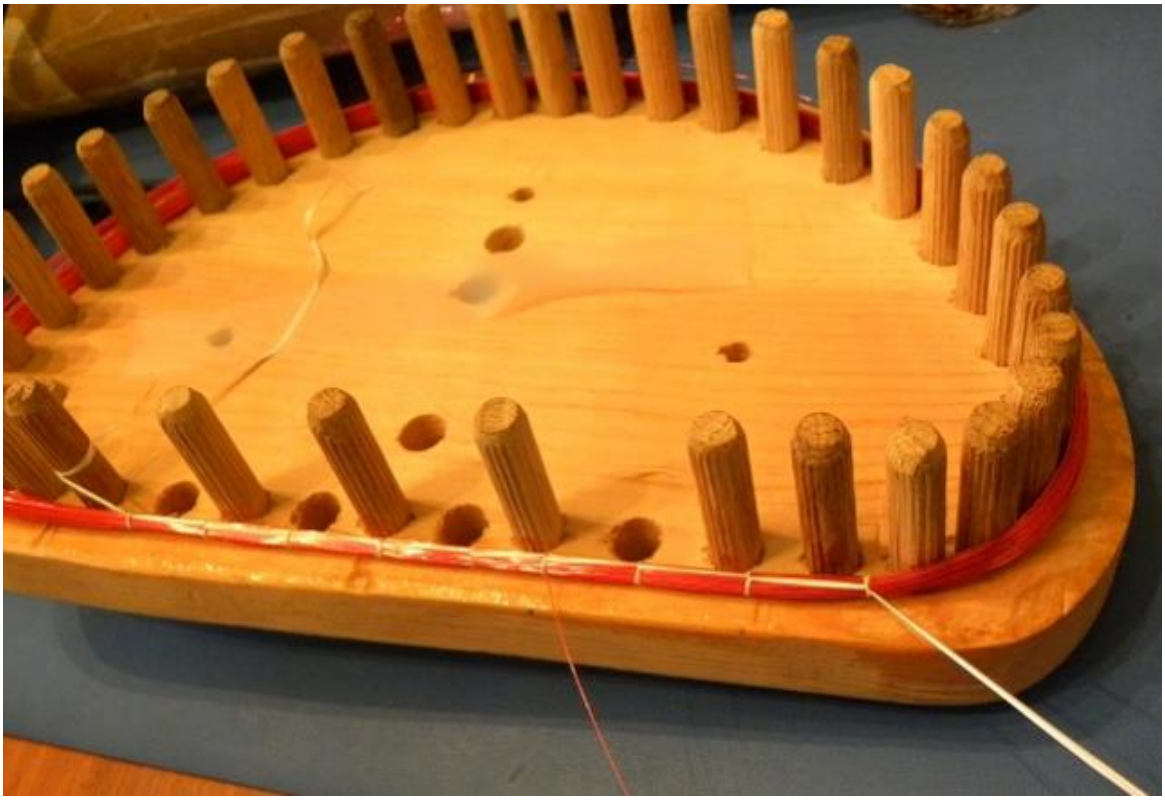


Figure 4.20 Form with half removed – Binding with string

Mylar survival blankets from a camping or outdoors shop. Keep in mind that in general, only ONE side of the Mylar is conductive and must be in contact with any wires needed to attach to the cable ground!

- **Once insulating is finished, you can apply shielding in the following manner:**

1) Wrap a thin, bare wire on top of the electrical tape but stop about 1cm short of completing a full circle. Leave enough of a "pig tail" to solder to the cable ground.
2) Wrap one spiraling layer of aluminum or Mylar around the coil, making sure that the conductive side (if using Mylar) makes contact with the bare wire. Again, make sure to stop short of completing a full circle.

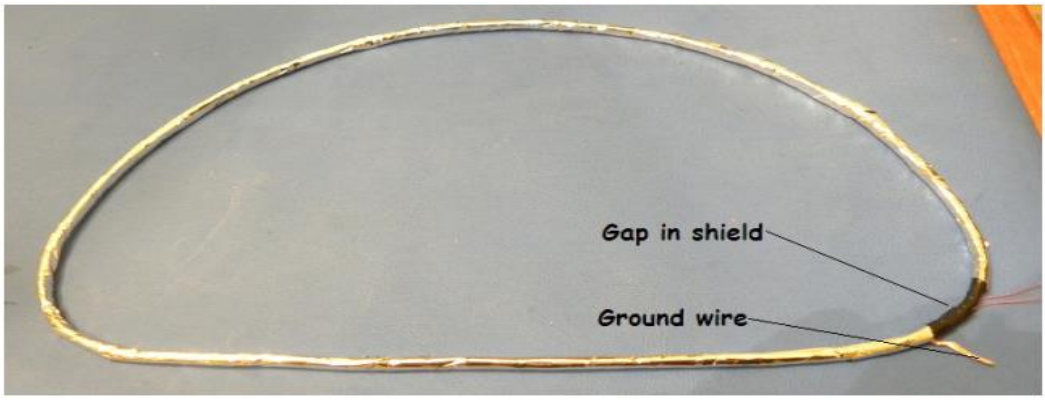


Figure 4.21 Shield installed

3) Apply a final layer of vinyl electrical tape.

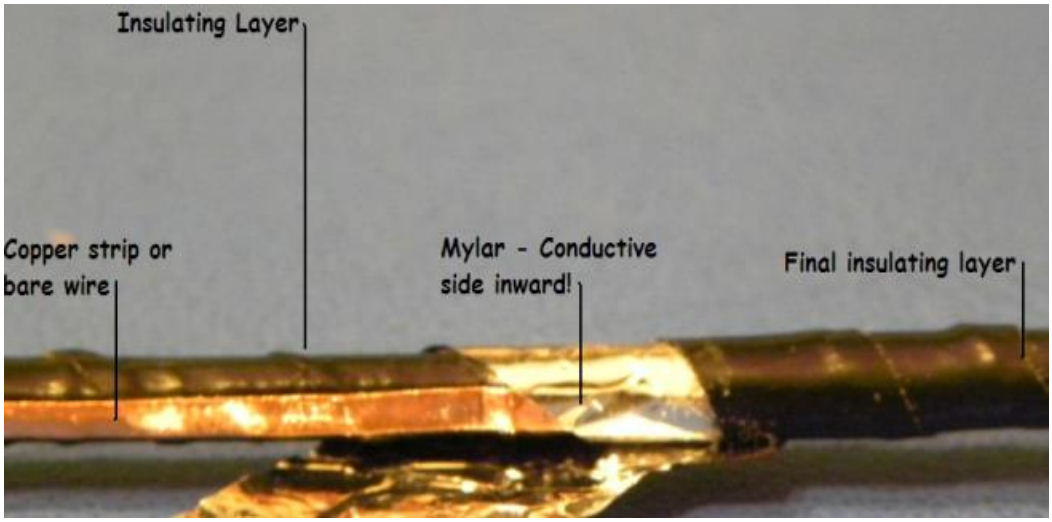


Figure 4.22 final layer of vinyl electrical tape.

4.1.15 Cabling and Coil Grounding

There are several options when it comes to coil cabling. Below are just two methods that work well. While testing my first couple of coils, I was disappointed with the performance

of the TGSL when the ground was wet. The bottom image is my preferred method of coil ground and offers improved performance in wet grass. Optionally, Use Belden-M (8723) cable.

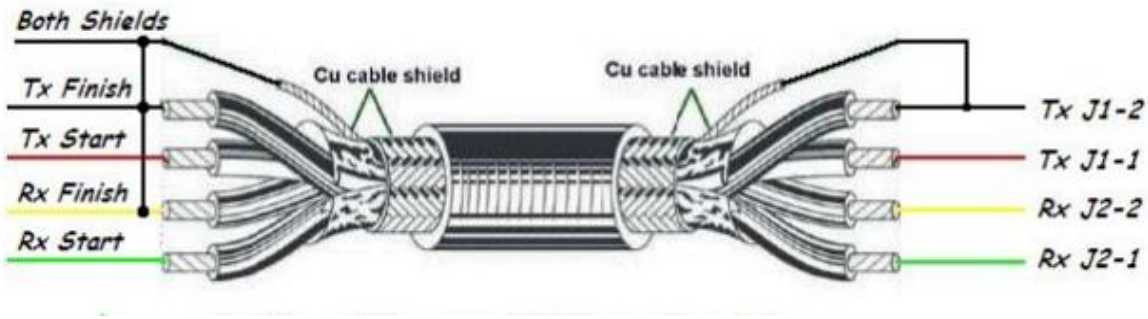


Figure 4.23 Coil wiring per TGSL coil making

4.1.16 Coil Nulling

More controversy seems exists on coil nulling than any other aspect of this project. Basically, this is the process of overlapping the two coils so that they are in an "Induction Balance" state and the Rx coil will be under minimum influence of the alternating magnetic field of the TX coil.

This is because part of the Rx coil will be "inside" the Tx coil and part will be "outside" or part will see a positive magnetic flux and part will see negative flux at any instant in time. This may seem trivial, but I have seen so many new builders completely miss this concept.

4.1.16.1 Basic Nulling

To null coils, lay them out on a workbench free from the influence of any metal objects.

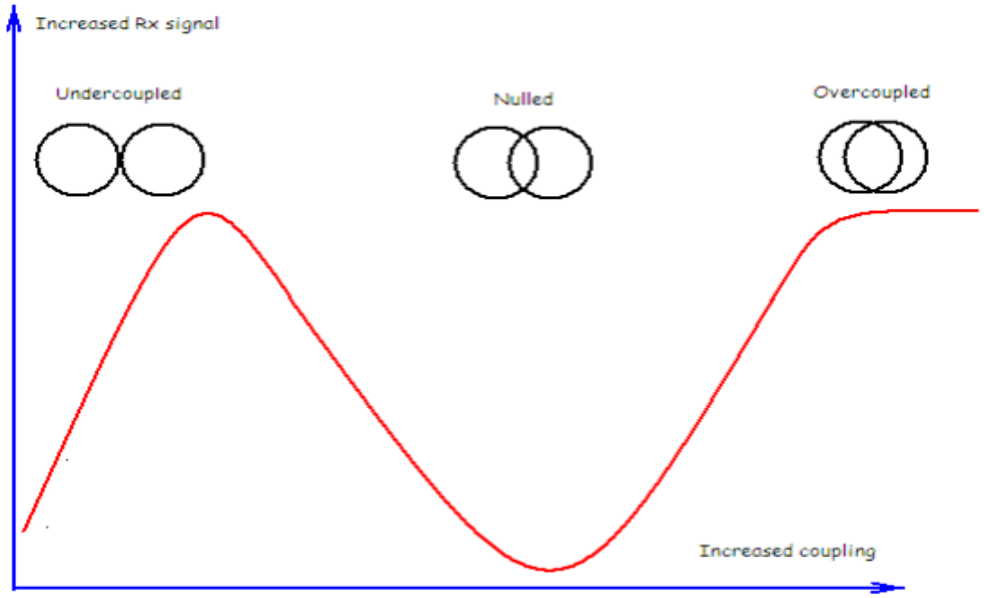


Figure 4.24 Relationship between RX single and Coupling

The final USB-2 cable to the circuit should be used and wires should be kept short as possible. Ideally, they should be placed inside the coil shells and the bottom coil should be secured. Power on the TGSL and adjust coil overlap while monitoring U101a, pin 7 with a DVM [40] in the A.C. setting. The goal is to find a sharp null in the voltage between U101a pin 7 and the PCB ground. You should be able to get the signal down to about 4mV but the actual reading may be subject to the frequency response of the DVM used.

When you have found the deepest null while positioning the overlap of the coils, it is time to secure the coils mechanically and check the ground balance setting. For this, you will need a small ferrite slug scavenged from an old radio or possibly a toroid. Verifying correct circuit operation would very difficult without a ferrite slug! Start with the TGSL switched to the all metal mode and set the GB [39] pot fully counter clockwise. Start waving the ferrite slug over the coil and advance that GB pot slowly clockwise until the ferrite slug is rejected or the sound starts to break up. Advancing too much further and silver may be rejected in the DISC mode! This will be a good place to start for rejecting the return signal associated with ground.

Possible problems: If ferrite cannot be rejected or accepted (in the all metal mode) at around the mid-range setting of the GB pot (or maybe just a little Clock wise of the mid-range setting) then possibly either the coils are not nulled or the difference in resonant frequency between Tx and Rx circuits is not correct (phasing is off).

4.2 Idea of project

In our project we have used many techniques and combined to get unique project so we used the idea of swarm robots this concepts has not been used in project of detection for mines before and we developed the project with **AI** (Artificial Intelligence) to give the project advantages to accomplish tasks faster and more accurately so we will discuss how we used swarm and AI in our project..

We used swarm robots so we can dispose many bombs in the same time and we can also secure a specific area that we doubt if this a bomb or just a normal metal. At the end, we've been in this research through how we could create swarm robots with Artificial Intelligence that we could use them in our project much better than single robot without AI.

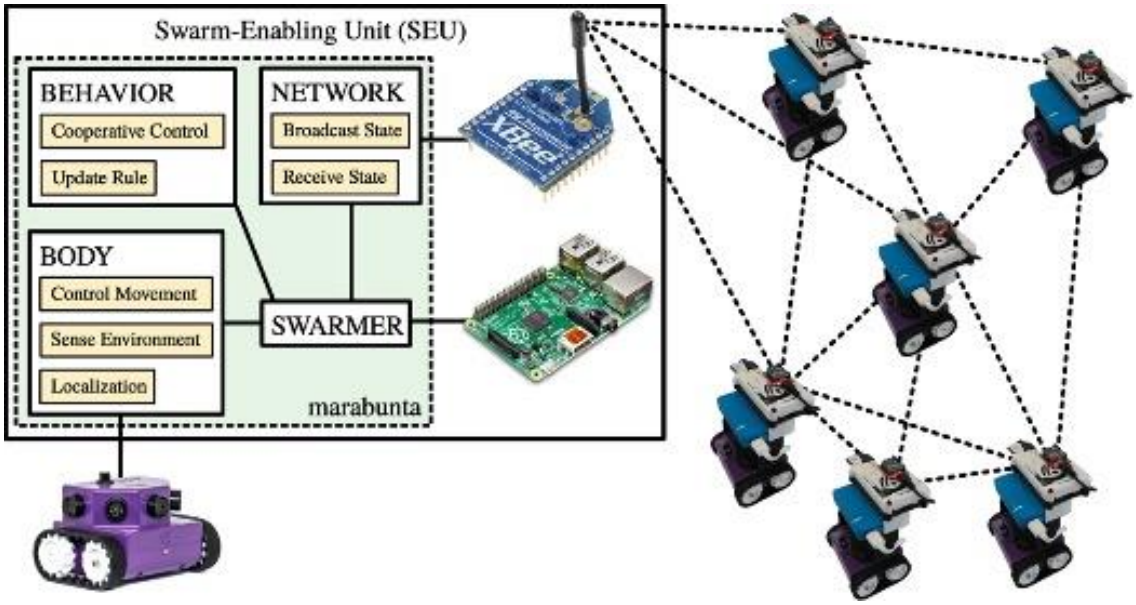


Figure 4.25 the system of swarm

4.2.1 Techniques usage in project

Techniques are used to get powerful project that have never been exist and save many resources and life's and also do the task with many opportunity to accomplish it with higher accuracy and time will be gained a lot.

4.2.1.1 Swarm robots

We used swarm as 3 robots will be connected together and each one will have specific task to do and the others will know what the others achieved. (More details in ch.3).

4.2.1.2 AI (Artificial Intelligence)

We used AI as improvement for project so we use AI as SLAM and connection between robots and automation for the slaves without interference and updating positions and Achieved tasks for all robots. (More details in ch.2).

4.2.1.3 ROS (Robotic Operation System)

We used this technique so we have ability that we apply the same package that we Used for one robot with other and modify what we need for new robots and they all Available packages. (More details in ch.2).

4.2.1.4 SLAM (simultaneous localization and mapping)

One technique or application of ROS that we used to get and build map for Unknown environment. (More details in ch.2).

4.2.1.5 IOT (Internet of Things)

We used this technique to communicate between slave's robot and master one so they know what every one of them have accomplished. (More details in ch.3).

4.2.1.6 Image Processing

We used this technique to detect the mines using camera (For more details in ch.2).

4.2.2 Project Description

As we said we have 3 robots as swarm idea one robot is the master and two will be slaves and here the concept of project will be that main robot (master) will be controlled by Joystick (controller) with many channels to control it with every channel control specific part in robot for example channel for control the arm and channel for motion of the body and so on ,so the motion of robot usually be by Embedded but we use ROS (Robotic Operation System) which more easily and the best advantage for it that the same library that we used for one robot.

we could use the same for the others with simple modification, so we move the robot (master) to unknown environment with no map and start to scan the area of the map block by block to secure some locations on the map so the robot can move and continue scanning for the full map and every location will detect for metal will be marked as point so the main robot will move away or around this point so he can keep scanning and securing the area.

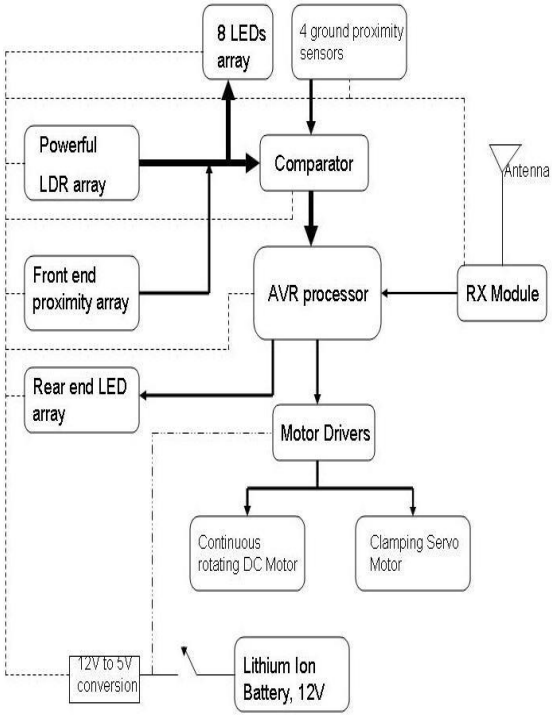


Figure 4.26 Control System at the Master

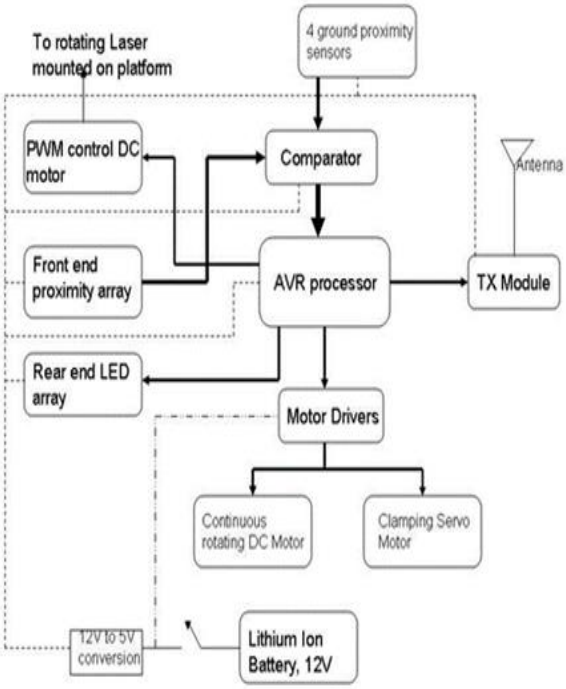


Figure 4.27 Control System at the Slave

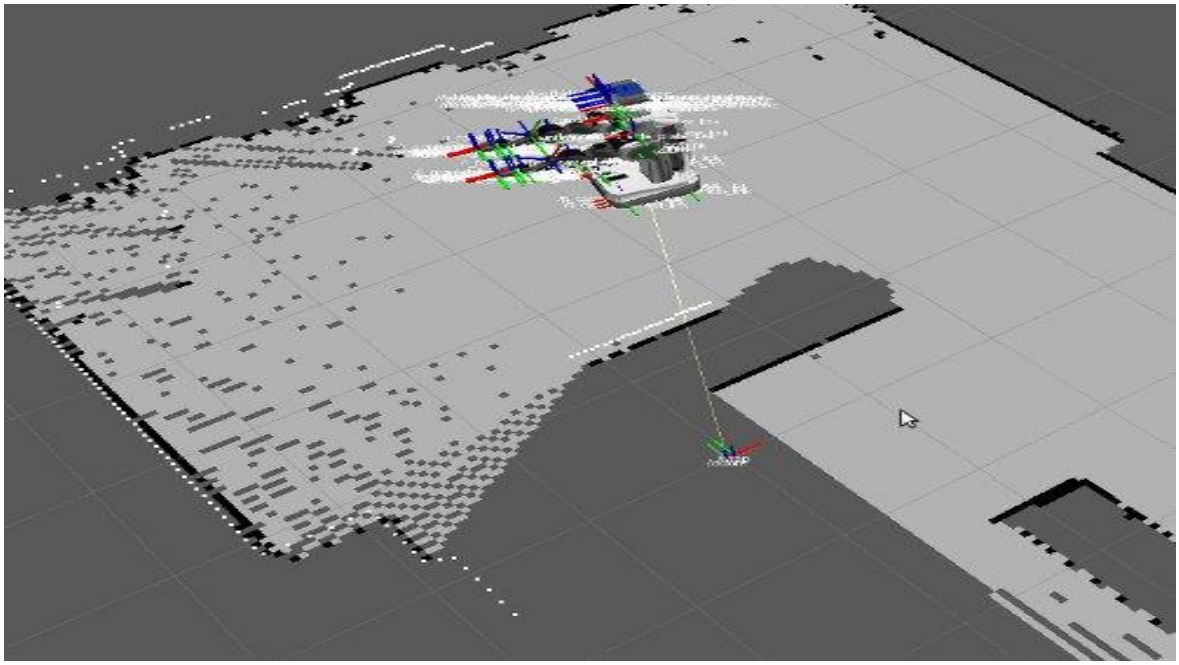


Figure 4.28 the robot make mapping using ROS

and the process of scanning and searching for metals will be by metal detector that type is PI (Pulse Induction) and here the coil will sense the electromagnetic field of the metal that will pass by and give indication that we receive and the metal detector will be in the arm of the robot which will have the ability to detect mines for range like circle because this arm with servo motors will be in specific angels that will be allowed to move in and in the same time of motion for the master robot it will save the map so it will not be unknown for the salves robots after it finishes the scan of the environment and save the map of it here we used one technique of ROS which is SLAM (simultaneous localization and mapping) that we used to scan and detect the map for the robot and in it there is many other technique.

That we used so we could apply SLAM and the one is used here is the project is GMapping [41] which is outdoor technique, we have to set many parameters in ROS so we can get the map and detected the possible points that could have mines and for the connection between robots we used IOT (Internet Of Things) so that robots will be known what others have accomplished and then slaves robots will enter the map according to what the master have built for them and they got it by the connection.

that we set and then they will scan the metal as if it just metal or mines and detect by Image processing when the slave robot determine if it is mine or not the others robots must be known for what this robot get far so the connection between all robots is very important.

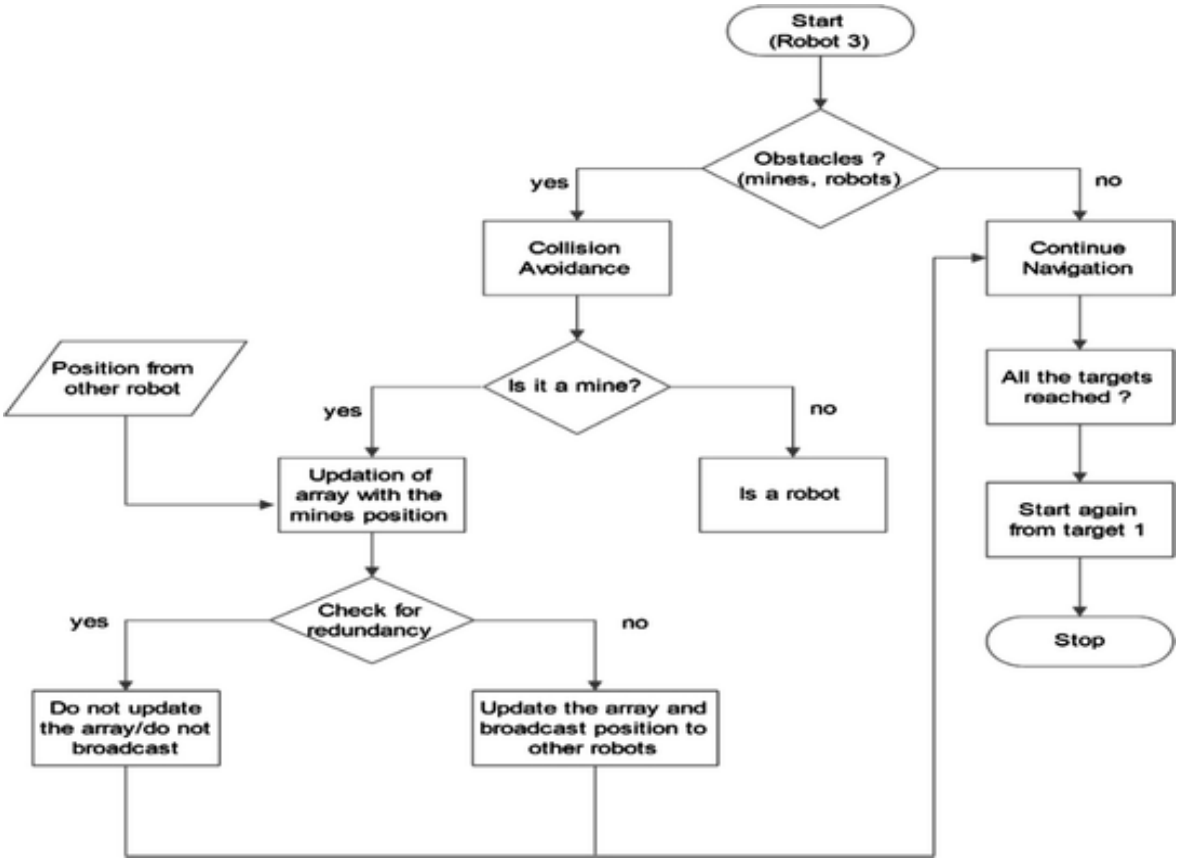


Figure 4.29 Flow diagram depicting the working of application

also supplied the power supply of circuit of the robot which is batteries for the motors of motion and motor of arm and metal detector all will be connected to solar cell to recharge it with no need to replace them or any problem happen due to battery have no power to continue the mission so recharging them is best solution to make the circuits stable and not have any problem of power supplies and it have many calculation and equation to know what you need of solar cell to be enough for the hole robot.

4.3 Hardware

The robot used for the challenge is a **Husky robot** from Clear path Robotics.

The main purpose of the robot is to detect mines. Hence a Vallon VMP3 pulse inductor metal detector is installed on the Husky.

It is mounted on a 2DOF robotic arm that sweeps the detector in the front.

The robot runs with a robot operating system ROS framework on an Ubuntu based machine. ROS provides libraries and tools to create robot applications helping out with device driver communication, visualizations, package management, message passing, etc.

Figure 4.30 Husky Robot

4.3.1 Components

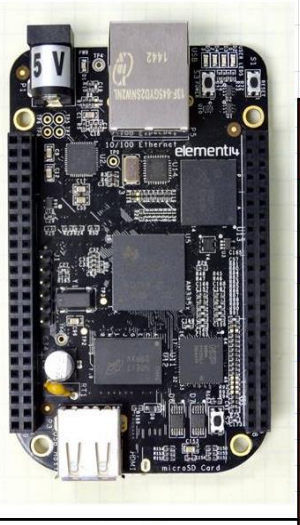
4.3.1.1 Beagle bone Black

The Beagles are tiny open-hardware (you could make one yourself), open-software computers that plug into whatever you have around the house.

Beagles mean big functionality in small packages because these little PCs can be used for all kinds of applications you've been tinkering with... and can handle many of the same tasks as your desktop PC [42].

P9

Function	Physical Pins		Function
DGND	1	2	DGND
VDD 3.3 V	3	4	VDD 3.3 V
VDD 5V	5	6	VDD 5V
SYS 5V	7	8	SYS 5V
PWR_BUT	9	10	SYS_RESET
UART4_RXD	11	12	GPIO_60
UART4_TXD	13	14	EHRPWM1A
GPIO_48	15	16	EHRPWM1B
SPIO_CSO	17	18	SPIO_D1
I2C2_SCL	19	20	I2C_SDA
SPIO_DO	21	22	SPIO_SLCK
GPIO_49	23	24	UART1_TXD
GPIO_117	25	26	UART1_RXD
GPIO_115	27	28	SP11_CSO
SP11_DO	29	30	GPIO_112
SP11_SCLK	31	32	VDD_ADC
AIN4	33	34	GND_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	ECAPWMO
DGND	43	44	DGND
DGND	45	46	DGND



LEGEND
Power, Ground, Reset
Digital Pins
PWM Output
1.8 Volt Analog Inputs
Shared I2C Bus
Reconfigurable Digital

P8

Function	Physical Pins		Function
DGND	1	2	DGND
MMC1_DAT6	3	4	MMC1_DAT7
MMC1_DAT2	5	6	MMC1_DAT3
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
EHRPWM2B	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
EHRPWM2A	19	20	MMC1_CMD
MMC1_CLK	21	22	MMC1_DAT5
MMC1_DAT4	23	24	MMC1_DAT1
MMC1_DATO	25	26	GPIO_61
LCD_VSYNC	27	28	LCD_PCLK
LCD_HSYNC	29	30	LCD_AC_BIAS
LCD_DATA14	31	32	LCD_DATA15
LCD_DATA13	33	34	LCD_DATA11
LCD_DATA12	35	36	LCD_DATA10
LCD_DATA8	37	38	LCD_DATA9
LCD_DATA6	39	40	LCD_DATA7
LCD_DATA4	41	42	LCD_DATA5
LCD_DATA2	43	44	LCD_DATA3
LCD_DATA0	45	46	LCD_DATA1

Figure 4.31 Beaglebone Pinout Diagram

4.3.1.2 GigE Vision Cameras

GigE is the most common camera interface for industrial image processing. For high-performance industrial cameras. It provides a framework for transmitting high-speed video and related control data over Ethernet networks, it is well equipped with sensors including laser range finder and a stereo pair of GigE cameras.

Figure 4.32 GigE Vision Cameras

4.3.1.3 Atmega3

ATmega32 is an 8-bit high performance microcontroller of Atmel's Mega AVR family. ATmega32 has 32 KB programmable flash memory, static RAM of 2 KB and EEPROM of 1 KB. The endurance cycle of flash memory and EEPROM is 10,000 and 100,000, respectively.

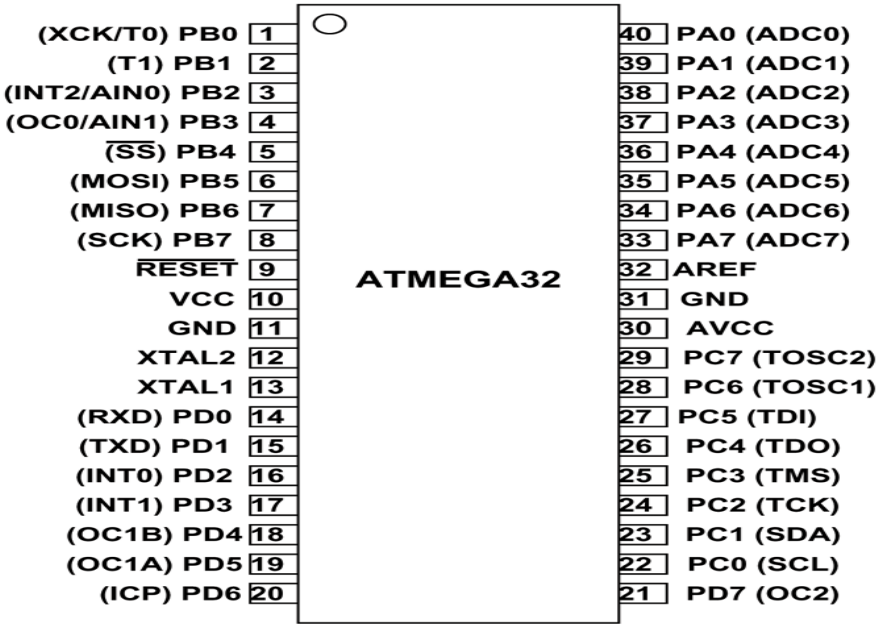


Figure 4.33 ATmega32 pin configuration

4.3.1.4 GPS module

It is a device that is capable of receiving information from GPS satellites and then to calculate the device's geographical position, GPS pinpoints its location in latitude and longitude which is converted to the minefield frame by the given UTM (Universal Transverse Mercator) Coordinate desired.

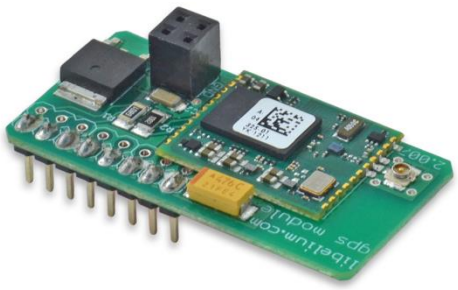


Figure 4.34 GPS module

4.3.1.5 Ultrasonic sensors

Ultrasonic sensors are a type of acoustic sensor divided into three broad categories: transmitters, receivers and transceivers.

Figure 4.35 Ultrasonic sensor

4.3.1.6 Lidar sensor

Lidar (also called LIDAR, LiDAR, and LADAR) is a surveying method that measures distance to a target by illuminating the target with pulsed laser light and measuring the reflected pulses with a sensor. Differences in laser return times and wavelengths can then be used to make digital 3-D representations of the target.

Figure 4.36 Lidar sensor

4.3.1.7 IMU sensor

Inertial measurement unit IMU [43] is an electronic device it's function to estimate the position and orientation of the robot using the velocity and angle rate information from the accelerometer and gyro sensor.

Figure 4.37 IMU sensor

4.3.1.8 Thermal sensor

A device that detects temperature. Thermal sensors are found in many laptops and desktop PCs in order to sound an alarm when a certain temperature has been exceeded.

Figure 4.38 Thermal sensor

4.3.1.9 Metal detector

A metal detector is an electronic instrument which detects the presence of metal nearby. Metal detectors are useful for finding metal inclusions hidden within objects, or metal objects buried underground.

Figure 4.39 Metal Detector

4.3.1.10 Accelerometer module

MEMS[44] (Micro-electromechanical systems) stands for Micro-Electro-Mechanical Sensors that can measure acceleration. In truth, the ADXL335 accelerometer module is a 3 axis accelerometer that has 6 pins, three pins for 3-axis sensing, one for Vcc, one for ground, and one for self-test.

Figure 4.40 Accelerometer Module

4.3.1.11 Wireless access point

In computer networking, a wireless access point, or more generally just access point, is a networking hardware device that allows other Wi-Fi devices to connect to a wired network. The AP usually connects to a router as a standalone device, but it can also be an integral component of the router itself.

Figure 4.41 Wireless Access Points

4.3.1.12 Buck converter

A buck converter is a DC-to-DC power converter which steps down voltage from its input to its output. It is a class of switched-mode power supply typically containing at least two semiconductors and at least one energy storage element, a capacitor, inductor, or the two in combination.

Figure 4.42 Buck converter

4.3.1.13 LIPO battery

A lithium polymer battery, or more correctly lithium-ion polymer battery (abbreviated as LiPo, LIP [45], Li-poly, lithium-poly and others), is a rechargeable battery of lithium-ion technology using a polymer electrolyte instead of a liquid electrolyte.

Figure 4.43 LIPO battery

4.3.1.14 Solar Cell

Many solar cells make a solar panel. A solar cell is the basic electricity generating unit of the solar photovoltaic system, Solar cells form the basic electricity generating unit of the solar photovoltaic system.

Figure 4.44 Solar cell unit

4.3.1.15 solar cell phone charger

Solar cell phone chargers use solar panels to charge cell phone batteries. They are an alternative to conventional electrical cell phone chargers and in some cases can be plugged into an electrical outlet.

Figure 4.45 Solar cell phone charger

4.3.1.16 Pushbutton

A push-button or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed.

Figure 4.46 Push-Button Switch

4.3.1.17 H Bridge

An H bridge is an electronic circuit that switches the polarity of a voltage applied to a load. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backwards.

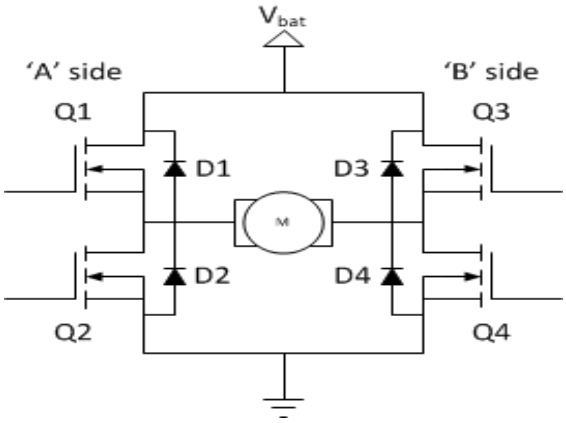


Figure 4.47 H-Bridge

4.3.1.18 DC motor

A DC motor is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields.

Figure 4.48 DC Motor

4.3.1.19 Node MCU

It's open source platform part of the Internet of Things (IOT). The NodeMCU development board, based on ESP8266EX, it integrated Wi-Fi receiver, and transmitter, Also supports several programing languages, hence, it is very easy to upload programs from any computer over a micro-USB port.

Figure 4.49 Accelerometer Module

5. Features and Future

5.1 Features of Project

In our Swarm Robotics system, we have three Robots: one is master, and two are slaves.

- **Our system work practically in field as follow:**

Firstly the master enter the desired land, The main robot draws a map of places where mines and metals and kept score for separating places where mines and the minerals, and also paints a safe route for slaves, so they pass through without collision or explosion, and store all the information about the land and Moving to the cloud.

Function of the cloud is the process of connecting the main robot slaves, and any other information that is identifiable from the main Android working at automating the transfer of information to the cloud, these cloud analyses and process data then displayed it on the GUI [46].

Slaves begin to store all information of the main robot and dealt with at the beginning of entering the land so they can deal with the mine or metal whatever, knowing that the slaves they draw another map of certainty to map drawn by the main robot only to confirm information.

Knowing that the slaves are connected together, and store its information and information the rest of the slaves, and this so as not to deal with the place that was spotted again, and if there is any malfunction in one of the slaves is another slave function.

Clarification of group robots it's getting easier, faster and access to places with high accuracy and security.

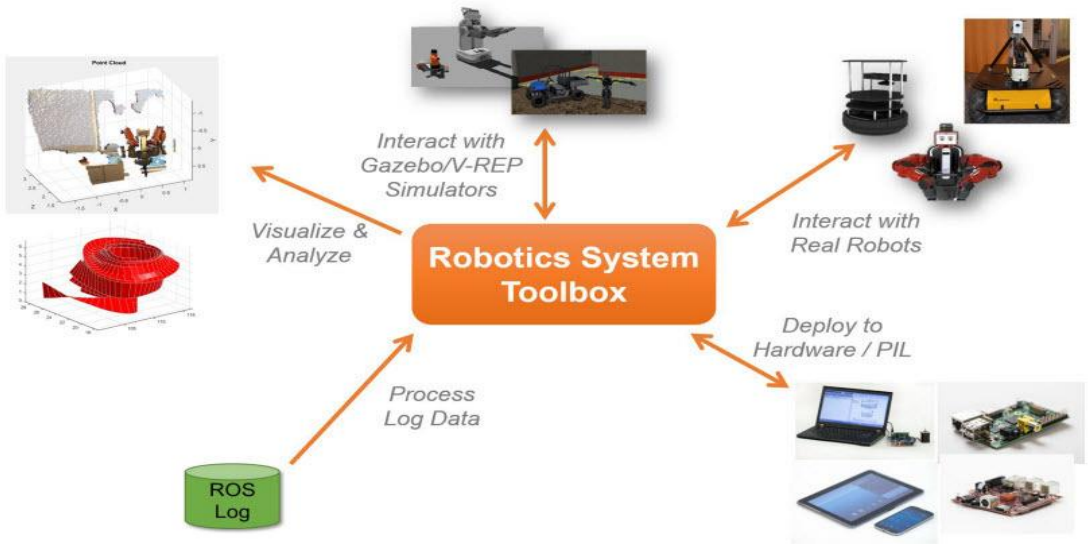


Figure 5.1 Robotics System Toolbox

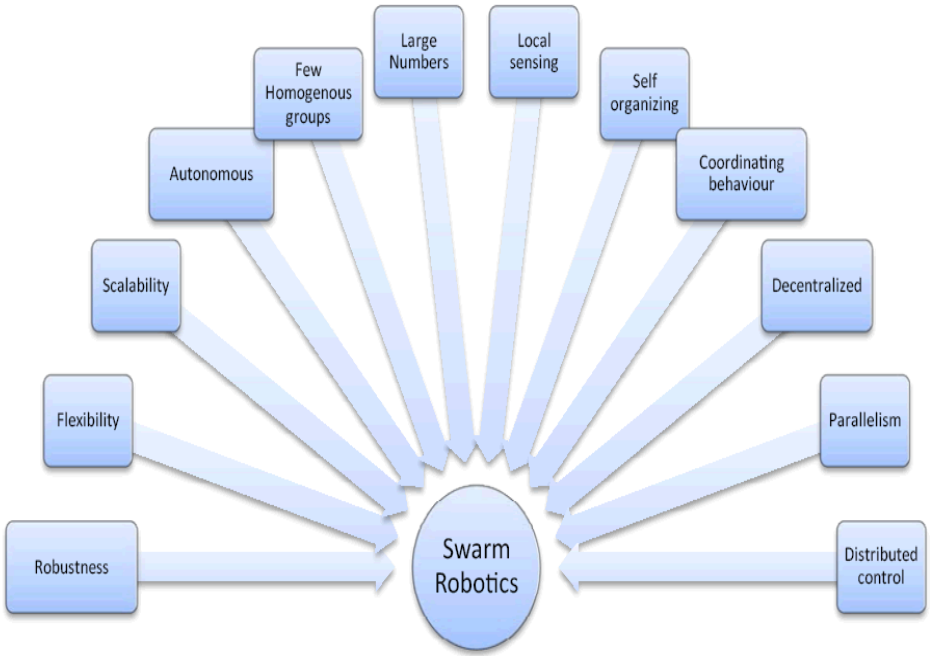


Figure 5.2 Characteristics of swarm robotics

5.2 Future Works

➢ Our project can be used in many another fields and applications **as follow**:

- Disaster rescue missions is one of the most important applications of swarm's robots. Swarms of robots of could be sent to places rescue workers can't reach this would save lives.

- Swarm robots have many applications in mining tasks Swarms robots can be used in military to form an autonomous army.

- Swarm robots are also useful for autonomous surveillance and environment monitoring to investigate environmental parameters, search for survivors, and locate sources of hazards such as chemical or gas spills, toxic pollution, pipe leaks, and radioactivity.

- A robotic spacecraft designed to make scientific research measurements is often called a space probe. Many space missions are more suited to tele robotic rather than manned operation, due to lower cost and lower risk factors, so tele robotic probes are the only way to explore them.

- Swarm robots can perform tasks in which the main goal is to cover a wide region. The robots can disperse and perform monitoring tasks, for example, in forests.

- In Military applications, it is very much useful in detecting bomb which is most dangerous task for the humans.

- Robotics is expected to play a major role in the agricultural/farming domain. Swarm robotics, in particular, is considered extremely relevant for precision farming and large-scale agricultural applications. Specifically, a decentralized monitoring/mapping scenario, and implement a use case for the detection and mapping of weeds in a field by a group of small unmanned aerial vehicles (UAVs [47]).

- Toxic waste clean-up, search and rescue (SAR [48]) and collection of terrain samples are some of the important applications of Swarms robots.

- Swarms robots can be for Exploration and mapping it would save money and time. Presently, most of the efforts are being made in surveillance, reconnaissance and hazard detection.

- In medical fields, a use of Nano-robots moving through human veins and arteries (e.g. to fight certain types of cancer).

- There are many industries which use dangerous things like burning furnace, chemicals, making nuclear weapons etc. Here usage of swarms can reduce danger in such types of industries.

- The possible real applications of swarm robotics will take special importance when robots get to be mass produced and the costs of building swarms of robots decrease.

- The development of technologies such as MEMS (Micro-Electro Mechanical Systems) will allow to create small and cheap robots. In this way swarms of robots can be really useful for dangerous tasks. For example, for mining detection and cleaning. The number of possible applications is really promising, but still the technology must firstly be developed both in the algorithmic and modelling part, and also in the miniaturization technology.

Conclusions

Swarm robotics they have many applications, bomb disposal is one of the most dangerous application where the risk of death lurks with every move. We have many bomb disposal robots but what make our project more intelligent and creative we added two main ideas that made the project more powerful than ever.

In our project we used swarm robots so we can dispose many bombs in the same time and we can also secure a specific area that we doubt if this a bomb or just a normal metal. At the end, we've been in this research through how we could create swarm robots with Artificial Intelligence that we could use them in our project much better than single robot without AI.

The swarm robotic can achieve the same capacity through collaboration among groups, taking advantage of the low cost of construction and maintenance.

This suggests that the swarm can perform tasks involving multiple targets is distributed in a vast range of environment, interacting in a local swarm and allowing individuals to join or terminate the task at any time without interrupting the whole swarm.

The result of the project is to facilitate the detection of metals and mines in a faster, safer, scalable and less error.

References

1. KavitaRitika Saroha and Rajani Bala, 2013, Image Processing and Image Segmentation.Availabe at: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwis5aucrt7fAhUJ26QKHeobD0cQFjAAegQIChAC&url=http%3A%2F%2Fijrcar.com%2FVolume_1_Issue_7%2Fv1i703.pdf&usg=AOvVaw2j11GM-l7uXp8o1cSlAwM5 (Accessed at: 25/8/2018) https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwis5aucrt7fAhUJ26QKHeobD0cQFjAAegQIChAC&url=http%3A%2F%2Fijrcar.com%2FVolume_1_Issue_7%2Fv1i703.pdf&usg=AOvVaw2j11GM-l7uXp8o1cSlAwM5 (Accessed at: 25/8/2018)

2. IBM Corporation,2008, Parallel&serial communication.Available at: https://en.wikipedia.org/wiki/Parallel_communication (Accessed at: 25/8/2018)

3. Circuit Basics,2016, BASICS OF UART COMMUNICATION.Available at: http://www.circuitbasics.com/basics-uart-communication/ (Accessed at: 30/8/2018)

4. Alexander Mordvintsev & Abid K,2017, OpenCV-Python Tutorials Documentation.Available at: file:///E:/FINAL%20PROJECT/image%20processing/opencv-python-tutroals.pdf (Accessed at:1/9/2018)

5. John Sturtz,2018, Introduction to Python.Available at: https://realpython.com/python-introduction/(Accessed at : 1/9/2018)

6. Zhong Y, Zhang L, Huang B, Li P ,2006,an unsupervised artificial immune classifier for multi/hyper spectral remote sensing imagery. IEEE Trans. Geosci. Remote Sens (Accessed at: 30/8/2018)

7. Robot Operating System (ROS),2016, Available at: http://www.ros.org/wiki/. (Accessed at 15/9/2018)

8. Natick. (2015) Math Works Introduces Robotics System Toolbox for Complete Integration with Robot Operating System (ROS) Available at: https://www.mathworks.com/company/newsroom/mathworks-introduces-robotics-system-toolbox-for-complete-integration-with-robot-operating-system-ros.html (Accessed at 15/9/2018)

9. IBM Corporation,2008, Parallel&serial communication.Available at: https://en.wikipedia.org/wiki/Parallel_communication (Accessed at: 25/9/2018)

10. Circuit Basics,2016, BASICS OF UART COMMUNICATION.Available at: http://www.circuitbasics.com/basics-uart-communication/(Accessed at: 25/9/2018)

11. **Alexander Mordvintsev & Abid K,2017, OpenCV-Python Tutorials Documentation.Available at: file:///E:/FINAL%20PROJECT/image%20processing/opencv-python-tutroals.pdf (Accessed at:1/10/2018)**

12. **John Sturtz,2018, Introduction to Python.Available at: https://realpython.com/python-introduction/(Accessed at:1/10/2018)**

13. **Staff Writer,2018, Top 5 Uses of Python.Available at: https://pctechmag.com/2018/05/top-5-uses-of-python/ (Accessed at:20/10/2018)**

14. **ROS.org,2018, ROS Tutorials.Available at : http://wiki.ros.org/ROS/Tutorials (Accessed at: 25/10/2018)**

15. **Dr. Alaa Khamis, http://alaakhamis.org/teaching/SPC418/schedule.html.Available at: http://alaakhamis.org/teaching/SPC418/schedule.html (Accessed at: 30/10/2018)**

16. **Kevin M. Lynch, Randy Freeman, Matthew Elwin, Samhitha Poonacha, Andrew Kessler. 2013. Swarm Robotics. Neuroscience and Robotics Laboratory (NxR) Science blog.** https://nxr.northwestern.edu/research/swarm-robotics**(Accessed at: 13/11/2018)**

17. Kevin Ashton ,2008, Getting Started with IoT , **Available at:**
Andrew Minteer, July 2017, Analytics for the Internet of Things (IOT) Available at: **https://learning.oreilly.com/l(Accessed at: 13/11/2018)**

18. **Don Bowers. (2012) an introduction to the Tesoro Golden Sabre Light E-book library [online]. Available at: https://www.hobbielektronika.hu/forum/getfile.php?id=224173(Accessed at:20/11/2018)**

19. **Don Bowers. (2012) Making coil for TGSL E-book library [online]. Available at** http://www.abcelectronique.com/forum/attachment.php?attachmentid=28668**(Accessed at: 25/11/2018)**

20. **Natick. (2015) Math Works Introduces Robotics System Toolbox for Complete Integration with Robot Operating System (ROS) Available at: https://www.mathworks.com/company/newsroom/mathworks-introduces-robotics-system-toolbox-for-complete-integration-with-robot-operating-system-ros.html Accessed at:20/11/2018)**

21. **Yazdani Hassan. (2017) Swarms Robots and their applications Available at: http://www.iosrjournals.org/iosr-jce/papers/Vol19-issue1/Version-2/G1901024647.pdf (Accessed at:25/11/2018)**

22. **Zhong Y, Zhang L, Huang B, Li P (2006) an unsupervised artificial immune classifier for multi/hyper spectral remote sensing imagery** https://realpython.com/python-introduction **(Accessed at:25/11/2018)**

23. **Staff Writer,2018, Top 5 Uses of Python.Available at: https://pctechmag.com/2018/05/top-5-uses-of-python/ (Accessed at:30/11/2018)**

24. **Yazdani Hassan. (2017) Swarms Robots and their applications**
Available at: https://en.wikipedia.org/wiki/Artificial_intelligence/**(Accessed at:2/12/2018)**

25. **Staff Writer,2018, Top 5 Uses of Python.Available at:** https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligent_systems.htm**(Accessed at:2/12/2018)**

26. **John Sturtz,2018, Introduction to Python.Available at:**http://www.abcelectronique.com/forum/attachment.php?attachmentid=28668**(Accessed at:2/12/2018)**

27. **Yazdani Hassan. (2017) Swarms Robots and their applications**
Available at:http://www.numpy.org/#numpy **(Accessed at:15/12/2018)**

28. "MECHANICAL DEMINING,(2012) **Available at** https://www.freelancinggig.com/blog/2017/08/19/best-programming-languages-face-recognition/ **(Accessed at:15/12/2018)**

29. HRATC2014, (2015), **Available at** https://wiki.dd-wrt.com/wiki/index.php/Wireless_Bridge **(Accessed at:20/12/2018)**

30. **Zhong Y, Zhang L, Huang B, Li P (2006)**https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx_ug01.html **(Accessed at:25/12/2018)**

31. COSS Solutions,2004,Application IOT **Available at:** https://www.freelancinggig.com/blog/2017/08/19/best-programming-languages-face-recognition/ **(Accessed at: 27/12/2018)**

32. Tutorials Point,2016, Artificial Intelligence - Intelligent Systems **Available at:** https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligent_systems.htm **(Accessed at: 27/12/2018)**

33. HOKUYO Photo Sensor URG-04LX-UG01,2012, Available at : https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx_ug01.html **(Accessed at: 30/12/2018)**

34. Tutorials Point,2016, Artificial Intelligence - Intelligent Systems **Available at:** https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligent_systems.htm **(Accessed at: 30/12/2018)**

35. **ROS.org,2018, ROS Tutorials.Available at : http://wiki.ros.org/ROS/Tutorials (Accessed at:30/12/2018)**

36. **Scott Helmer and David Lowe,2018, Using Stereo for Object Reco/gnition.Available at: /https://www.cs.ubc.ca/~lowe/papers/10helmer.pdf (Accessed at :10/11/2018)**

37. AlexJinlei,2018, Stereo Vision Camera.Available at: https://github.com/AlexJinlei/Stereo_Vision_Camera (Accessed at : 6/1/2018)

38. AlexJinlei,2018, Stereo_Vision_Camera.Available at: https://github.com/AlexJinlei/Stereo_Vision_Camera/blob/master/stereo_cam_opencv/stereo_cam_disparity_map.py (Accessed at: 6/1/2018)

39. P. F. Felzenszwalb and D. A. M. andDeva Ramanan, "A discriminatively trained, multiscale, deformable part model," in CVPR, 2008. P. F. Felzenszwalb and D. A. M. andDeva Ramanan, "A discriminatively trained, multiscale, deformable part model," in CVPR, 2008.

40. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results,"

41. AlexJinlei,2018, Stereo_Vision_Camera.Available at: https://github.com/AlexJinlei/Stereo_Vision_Camera/blob/master/stereo_cam_opencv/stereo_cam_disparity_map.py (Accessed at: 6/1/2018)