Vision Based Target Recognition Through Unmanned Ground Vehicle

By

CSUO Saad Bin Shafqat (Grp Ldr)

PC Abubakar Ali khan

NC Hannan Hassan

Supervisor:

Col Dr Faheem Arif

Co-Supervisor

Dr Imran Siddiqui

Submitted to the Faculty of Computer Science

National University of Sciences and Technology, Rawalpindi in partial fulfillment for the requirements

of a B.E Degree in Computer Software Engineering

July 2012

CERTIFICATE

Certified that the contents and form of project report entitled **“Vision Based Target Recognition Through Unmanned Ground Vehicle”** submitted by 1) Saad Bin Shafqat, 2) Abubakar Ali Khan, and 3) Hannan Hassan have been found satisfactory for the requirement of the degree.

Supervisor: **____________________**

Col. Dr. Fahim Arif

ABSTRACT

Human resources are a very important part of any field especially when it comes to Military. With the advancements in the world and the techniques and equipment utilized by the army of these modern times, every country needs to keep pace with the modern equipment.

One of the most developing technology and is used all around the world is drone technology. We are inspired by the drone technology which is to move an UGV (unmanned ground vehicle).

Image processing has been a very rapidly developing field of computer science. Its techniques and algorithms have been implemented into various practical systems such as gesture recognition, face recognition, and drone technology.

Target recognition is also an area where there has been a lot of work with many different products coming out. From drones to different ground vehicles to go from A point to B point and find and recognize a specific object. Automated video systems have been proven to perform effectively and with a high success rate in a controlled static environment. However the challenge is to deploy video surveillance cameras in the outdoors that maintain a high probability of target detection.

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose unflinching support and unstinting cooperation, a work of this magnitude would not have been possible

ACKNOWLEDEMENTS

We are eternally grateful to Almighty Allah for bestowing us with the strength and resolve to undertake and complete the project.

We gratefully recognize the continuous supervision and motivation provided to us by our Project Supervisors, Col Dr Fahim Arif from MCS and Dr Imran Siddiqui from Bahria University. We are highly gratified to our committee members Asst Prof. Dr Awais Majeed, Dr Hammad Afzal and Lec Bilal Rauf for their continuous and valuable suggestions, guidance, and commitment. We are highly thankful to all of our professors whom had been guiding and supporting us throughout our course and research work. Their knowledge, guidance and training enabled us to carry out this research work.

We would like to offer our admiration to all our classmates, and our seniors who had been supporting, helping and encouraging us throughout our thesis project. We are also indebted to the MCS system administration for their help and support.

We deeply treasure the unparallel support and tolerance that we received from our friends for their useful suggestions that helped us in completion of this project.

In the end we are also deeply obliged to our families for their never ending patience and support for our mental peace and to our parents for the strength that they gave us through their prayers.

INDEX

LIST OF FIGURES

Figure No **Page No.**

Chapter 1: Introduction

1.1. INTRODUCTION

Today, the medical industry, astronomy, physics, chemistry, forensics, remote sensing, manufacturing, and defense are just some of the many fields that rely upon images to store, display, and provide information about the world around us. The challenge to engineers is to quickly extract valuable information from raw image data. This is the primary purpose of image processing – converting images to information.

A digital image is composed of a grid of pixels and stored as an array. A single pixel represents a value of either light intensity or color. Images are processed to obtain information beyond what is apparent given the image's initial pixel values.

Image processing has been a very rapidly developing field of computer science. Its techniques and algorithms have been implemented into various practical systems such as gesture recognition, face recognition.

Target Recognition generally refers to the use of computer processing to detect and recognize target signatures in a given set of data. In terms of image processing this data in is the form of image or as frames that are extracted from a video.

Target recognition is also an area where there has been a lot of work with many different products coming out. From drones to different ground vehicles to go from A point to B point and find and recognize a specific object. Automated video systems have been proven to perform effectively and with a high success rate in a controlled static environment. However the challenge is to deploy video surveillance cameras in the outdoors that maintain a high probability of target detection.

Human resources are a very important part of any field especially when it comes to Military. With the advancements in the world and the techniques and equipment utilized by the army of these modern times, every country needs to keep pace with the modern equipment.

1.2. PROBLEM DOMAIN

Vision based target recognition systems are very complicated and costly. Furthermore, the requirements of processing power for image processing are very high, operating system overloads and other system activities on most operating systems means that the possible utilizable memory for image processing reduces.

1.3. MOTIVATION

The requirement for developing a highly accurate, robust and cost effective target recognition system is a requirement for most organizations and even countries. Incorporating a low cost UGV model that is assembled using cheap and easily available parts further adds to the cost effectiveness and completeness of the target recognition system.

1.4. GOALS AND OBJECTIVES

1.4.1. Goal

The overall goal of the project is to assemble an unmanned ground (UGV) vehicle that will follow a specified path. A camera mounted on the UGV will provide an input video feed to an onboard laptop. The onboard laptop will have the target recognition software (TRS) running on it. The desired target's image will be provided to the TRS system. Frames will be extracted from the input video feed and image processing algorithm (i.e. SIFT) will process those frames to match with the targets given image. Once a target is recognized a notification alert will be generated.

1.4.2. Objectives

- Developing a target recognition software with a very high level of accuracy.
- Assembling a cost effective Unmanned Ground Vehicle model
- Implementing the SIFT algorithm for target recognition.
- Developing a program on a platform where computing power can be used more for image processing.
- Reducing need for technical savvy and experienced staff to operate remotely controlled security equipment.

1.5. DELIVERABLES

- Path following Unmanned Ground Vehicle
- Software for Target recognition
- Documentation

1.6. SYSTEM OVERVIEW

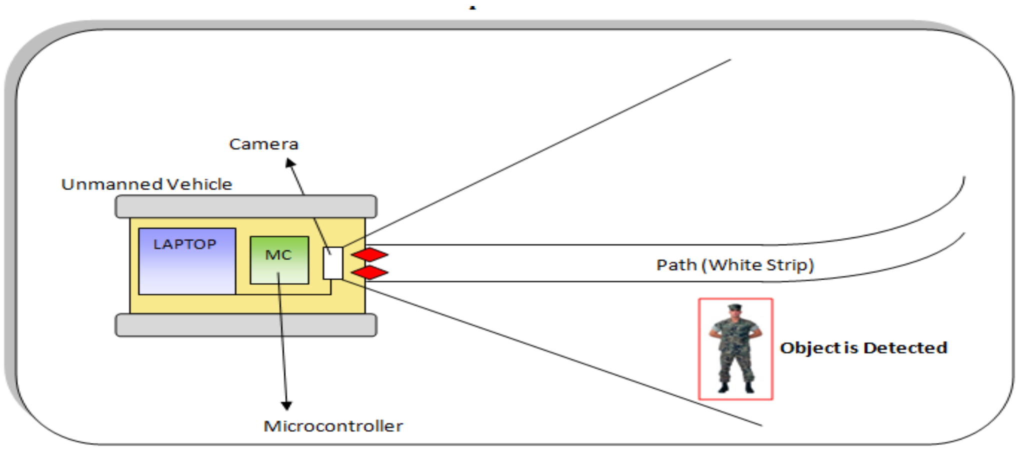


Fig 1.1. Top View

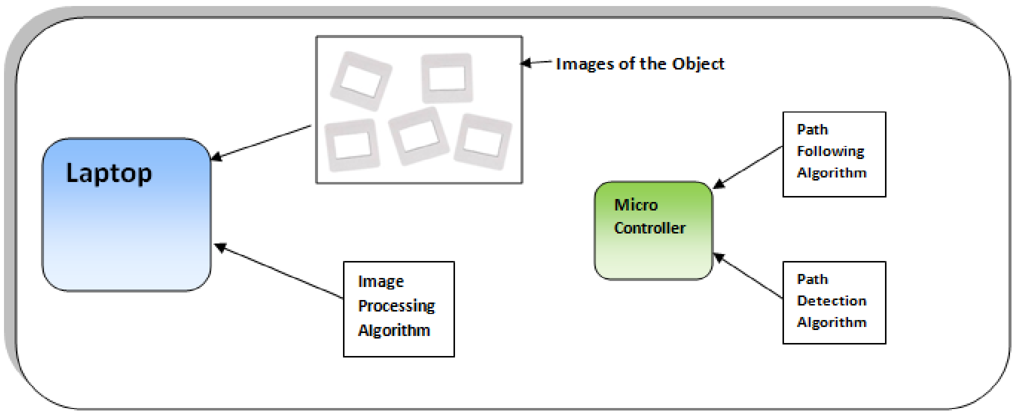


Fig 1.2. Detailed View

Chapter 2: Literature Review

2.1. INTRODUCTION

A lot of research has been done for target detection through vision. A number of different algorithms have been provided each with its own pros and cons. For this project we intend to try to use the better part of these different algorithms and try to avoid or counter the issues arising in them.

Automatic Target Recognition by Matching Oriented Edge PixelsClark F. Olson and Daniel P. Huttenlocher

IEEE Transactions on Image Processing, Vol. 6, no. 1, January 1997

Describes the technique of representing images by their edge maps and generating a 3-D model of the image by the given 2-D standard images.

One of the algorithms used for image recognition is to represent the target objects and their images by their edge maps, with a local orientation associated with each edge pixel. A three-dimensional object will be modeled of the basic of a set on standard 2-D images of the object. For a full approximation of the overall 3-D view created, concepts of linear algebra like rotation, translation and scaling will be used.

Development of a vision-based ground target detection and tracking system for a small unmanned helicopter LIN Feng, LUM Kai-Yew, CHEN Ben M.t&LEE Tong H. 2009 Science is China Press

Describes the use of a CMOS camera and a laser point for real time target recognition using pattern recognition.

Another school of thought considers pattern recognition to be the most appropriate approach for target recognition. Target recognition is done on the basis of clustering and matching them to a database of target and non target images or feature vectors. Pattern recognition depends on a lot of statistics.

Vision-Based Target Recognition And Autonomous Flightsthrough Obstacle Arches With A Small Uav

Florian-M. Adolf, Franz Andert, Lukas Goormann, andJorg S. Dittrich_German Aerospace Center (DLR), Institute of Flight Systems D-38108 Braunschweig, Germany 2006

Provides and algorithm for aerial vehicles to pass through a gate or a small corridor autonomously by using a GPS self localization and alignment according to the sensors on the plan.

A number of implementations of target recognition have been developed for both land and air. Cameras and a multitude of sensors are assembled onto a vehicle or a small helicopter or drone and after that using pattern recognition by identifying the color, shape and appearance signatures and comparing them to database consisting of a set of images of the target item.

Distinctive Image Features from Scale-Invariant Key points by David Lowe, Accepted January 22, 2004, International Journal of Computer Vision 60(2), 91-110,2004

Related to Scale Invariant Feature Transform, its general method and basic operations.

Scale Invariant Feature Transform is method to extract features of objects that can be stored in form of a database. These features can be later used to identify those objects in a frame or image containing a large number of objects. The benefit of using SIFT is that it is invariant to distortion (to an extent), rotation, noise and illumination. Another advantage is that the features are distinctive i.e. it is very less probable that features of one object will match another object in the database. SIFT matches the features using the following technique:

1. Matching individual features to database of known features using nearest neighbor algorithm.
2. Hough transform to identify clusters belonging to the same object
3. Performing verification for consistent pose perimeters through least square solution.

2.2. ISSUES

The problem is that this technique is a little slow i.e. it takes about 1 second to search through an image for an object. To improve its efficiency, we will first segment the image. Any segment that is not likely going to be the object will be discarded so from an image, the background (which will be most of the part of image) will be discarded. This will reduce the processing time needed for each image and hence the object will be searched in more images in less time.

Achieving maximum efficiency by reducing the amount of data given to the system is the main subject of this project.

Chapter 3: Project Plan

3.1. PROJECT OVERVIEW

3.1.1. Project Organization

- **Project Manager**

Role	**Organization: Name**
Project Leader	CSUO Saad Bin Shafqat
Project Manager	NC Hannan Hassan
Technical Project Mgr.	PC Abubakar Ali Khan

- **Project-internal Functions**

Function	**Organization: Name**	**Comment**
Quality Assurance	Hannan Hassan	
System Test Lead	Saad Bin Shafqat	
Validation Lead	Abu Bakar Ali	
Configuration Mgmt	Abu Bakar Ali	
Change Mgmt	Hannan Hassan	
Software Development Mgmt	Saad Bin Shafqat	
Hardware Development Mgmt	Hannan Hassan	
Interfacing Mgmt	Hannan Hassan	

- **Project Team**

Organization: Name	**Availability**	**Comment**
CSUO Saad Bin Shafqat	Full-Time	
PC Abu Bakar Ali	Full-Time	

Organization: Name	**Availability**	**Comment**
NC Hannan Hassan	Full-Time	

- **Steering Committee**

The Steering Committee (SteCo) of the project is responsible for managing the overall project and ensuring the project is completed in the proper time frame. Apart from it the SteCo committee is responsible for providing references and other help if the Project Team get stuck somewhere

The SteCo consists of the following members:

Organization	**Name**	**Comment**
CS Department	Col Dr. Faheem Arif	
Bahria University	Dr. Imran Siddiqui	Subject to Availability

3.1.2 Project Budget

- **Schedule and Milestones**

Milestones	Description	Milestone Criteria	Planned Date
M 0	Start Project	Proposal Accepted	<2011-08-05>
	Project goals and scope defined Basic Requirements defined	SRS reviewed Stakeholders identified	<2011-11-1>
M 1	Start Planning		<2011-11-02>
	Basic Design, Architecture Dividing into Modules, Responsibility Division	Detailed Design Completed, Tasks to group members assigned.	<2011-11-10>
M 2	Start Execution		<2012-01-04>
	Detailed Design defined, Development Model chosen	Basic Hardware and Software developed. Training completed for set objects.	<2012-02-20>
M 3	Interfacing and Confirm Execution		<2012-02-22>
	Working Separate Software and Hardware	Architecture reviewed and stable. Both Hardware and Software working together	<2012-03-10>
M 4	System Testing and Documentation		<2012-03-11>

Milestones	Description	Milestone Criteria	Planned Date
	Integration testing and performance testing	Improving product efficiency, Coding of new functionality finished(if time available), Draft documentation	<2012-03-12>
M 5	Release Product		<2012-04-10>
		Product system tested, documentation reviewed	
M 6	Close Project		<2012-05-01>

- **Budget**

Category	**Budget for Period in Rs.**					
	M0-M1	**M1-M2**	**M2-M3**	**M3-M4**	**M4-M5**	**M5-M6**
Human Resources (internal)	-	-	-	-	-	-
Human Resources (external)	-		1200	1800	-	-
Purchases (COTS)	-	-	10000	10000	-	5000
Equipment	-	-	-	-	-	-
Tools	-	-	3000	-	-	-
Travel costs	1000	1000	1000	1000	1000	1000
Training						
Review activities	-	-	2500	1500		1500

3.2. WORK BREAKDOWN STRUCTURE

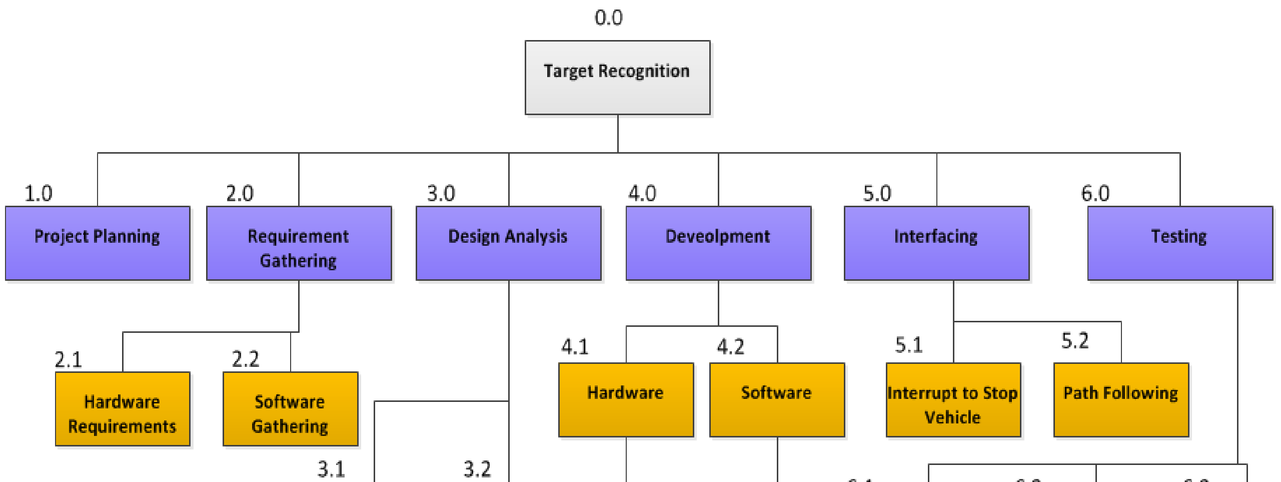


Fig 3.1. Work Breakdown Structure

Chapter 4: Software Requirement Specification Document

4.1. PURPOSE

The project under consideration is about target recognition through an unmanned ground vehicle. The main theme of the project is that an unmanned vehicle will be provided with a set of images of a particular stationary object which it would search along the path it is trained to work on, after recognizing the object it will send back an alert message.

Surveillance activities require a large number of human resources .There is also a high element of risk involved if surveillance is for military purpose. This project aims at making an unmanned ground vehicle that can be trained on a particular path or terrain where it can search for, detect and recognize a stationary object whose image had been provided to it through a remote server.

The main objective of the project is to make an effective and efficient algorithm for image recognition that can be used in different varied areas. The project aims at using the principles of image processing and implementing them at real world scenarios such as mine detection, target detection and locking, surveillance etc.

The basic purpose of this project is to remove the excess use of human resources used for surveillance, and other military purposes and lower the danger to the human lives.

The SRS covers the complete system including both software and hardware requirements as well as the basic features of the system. This document describes the overall system functionality describing the basic characteristics of the system, the scope of the project, the main features of the system, an explanation as to what are the objectives of the system. Along with this, this document also describes certain assumptions and limitations regarding the implementation of the product

This is the first release of such a system.

4.2. DOCUMENT CONVENTIONS

The Conventions used are as follows:

- ‘Times New Roman’ font style has been used with font size ‘12’
- Main headings are bold and capitalized while sub-headings are numbered in hierarchy in bold.
- Indentations have been used to show clearly which part of the document belongs to which heading.
- Clear numbering has been used to identify the topics and subtopics.

4.3. INTENDED AUDIENCE AND READING SUGGESTIONS

The document is intended for the faculty of the CSE Department, for the CI, for managers and owners of Surveillance Company’s and Military personnel. This document is also intended for the developer, project manager, potential users, testers, and documentation writers.

The suggested sequence for the readers is the same as per the order in which the document has been written. As in:

i. Introduction

ii. Overall description

iii. System Features

iv. External Interface Requirements

v. Other Non Functional Requirements

4.4. PROJECT SCOPE

Development of a working model of a vehicle for object detection through image processing that will search for and then identify the given object without any manual intervention on a given path via a webcam attached on the vehicle.

The project will reduce the amount of human resources spent on surveillance as well reduce the risk element involved for humans.

Business value for this can be termed as:

- Making the surveillance job for effective
- Ideally providing same results with using less human resource
- Cost of human resource is reduced significantly

The scope of the project includes making an efficient and effective image processing algorithm that can recognized the desire object while the camera is on a moving vehicle. The scope of the project also includes making a simple ground vehicle with a camera mounted on it and also designing its basic structural model and also the interfacing between the hardware and software.

The major goals of this project are to acquaint ourselves to implement the basic concepts of Image Processing that we have learned during this course, to develop a detailed understanding of Image Processing, to apply the SDLC in the right way and to practice with that the major techniques of project management and software engineering. Another important goal of this project is to learn the basic programming at microcontroller level and to learn how software systems are embedded into a desired hardware.

4.5. REFERENCES

- Literature Review
- Project Plan

4.6. OVERALL DESCRIPTION

4.6.1. Product Perspective

Surveillance and monitoring are very cumbersome tasks requiring a lot of man power. Certain military tasks involving surveillance and are very risky such as mine sweeping. The risk factor of human life can be reduced significantly by using unmanned ground vehicles to recognize the desired object. Hence, the human resource freed through this can be better utilized elsewhere.

This is the first version of this system. The SRS describes the specifications of the entire project rather than a particular component.

4.6.2. Product Features

The main features of the project are object recognition and path following. Object recognition has certain sub features such as frame extraction and video feed. Each feature is described in details in the next chapter of the SRS.

These features describes as to how the system will work. The order of importance of the features is given as

- Object Recognition
- Video feed and Frame extraction
- Training for target objects
- Region of Interest
- Feature Matching and Object Recognition
- Path Following

Details of the features are described in the next chapter of the SRS.

4.6.3. User Classes and Characteristics

- Military
- Surveillance Companies

4.6.4. Operating Environment

The system is being designed to be operated both indoors and outdoors mostly on plain terrain. The operating system on which the system will work on will be Linux/Windows. The interaction between the hardware and the software system will be through ARP's or 8051 Processor. The webcam input will be directed to a computer placed within the vehicle and the image processing algorithm will work entirely independent of the other operations such as motion or path following.

4.6.5. Design and Implementation Constraints

- Constraints of processing power
- Limitations of the features and resolution of the camera being used
- Amount of time required to train for a particular environment
- Lack of knowledge on microcontrollers.
- Hardware reliability

4.6.6. User Documentation

The product will with it contain a number of documents which will include

- Project Background
- SRS
- Project Plan
- Design Document
- Integration Test Plan
- Source Code
- User Manual

4.6.7. Assumptions and Dependencies

The following assumptions are being made related to this project

- Path should be flat
- Object to be recognized should be stationary
- If only a single camera is used, the object would be on one side where the object is supposed to be (for demonstration only)
- The path will be with contrast to the ground for easier path recognition

4.7. SYSTEM FEATURES

4.7.1. Training for Target Objects

- Description and Priority

Being a target recognition system, our system requires multiple target images from different angles for system training. This feature will allow the users to add images of additional objects. Upon adding the images, their features will be calculated through SIFT algorithm and stored under one label for one object.

- Stimulus/Response Sequences
  - ➢ User opens the ‘New Target Training’ feature of the system.
  - ➢ System requests for images of the target
  - ➢ User enters ‘n’ images of the target. Value of ‘n’ depends on the complexity of the object.
  - ➢ System uses these images to extract features of the new objects for recognition and then stores them under a label along with features calculated from other images under the same label.
- Functional Requirements

- ➢ Add Image

This allows the users to add multiple new images under the same label for the new target. If the image is not present at the given path, an error message will be given and the system prompts the user to give the correct path.

- ➢ Extract Features

The features of the object from the image will be extracted using this function. In case the image received is not clear or there are multiple objects in the image, the system will give an error and prompt the user to enter another image for extraction of features.

- ➢ Store Features

The features extracted will be stored under a label. For an object, there will be multiple images and their features will be stored under on label for future reference when finding and recognizing targets.

4.7.2. Video Feed and Frame Extraction

- Description and Priority

The vehicle will have a camera through which continuous video feed will be given to the system. This feature will receive the video feed and extract frames from it for the system to find the Region Of Interest (ROI) and then match the features of the objects in frame with the features already stored. It is of high priority.

- Stimulus/response sequence

  - ➢ Camera connected to the system inside the vehicle sends a video stream of the outside scenario to the system.
  - ➢ Individual frames will be extracted at a very low rate such as 1 fps and will be sent to the system.

  - If in case there is too much noise in the frame or it is blurred, the system will discard this frame and extract a new frame from the continuous video stream it is receiving.

- Functional Requirements

  - Video Capture

  This function allows the system to receive a continuous stream from the camera connected to it. If in case the camera is not sending any stream or there is some error in receiving the video, user will be prompted about the error and the possible solution.

  - Frame Extraction

  Individual frame will be extracted from the video stream using this function. It will extract 3-4 fps and this frame will be used by the system as an image to attain the Region Of Interest and then feature matching will be done on it. If a frame is faulty i.e. blurred or too much noisy, this frame will be discarded and new frame will be taken from the video stream.

4.7.3. Region of Interest

- Description and Priority

A frame might contain back ground in large proportion or objects that are entirely different from our target. Thus from every frame, we will use only the regions which have the possibility of matching our target and then feature matching will be used on those regions. If this feature is not used, a lot of time will be wasted in checking possible match of feature against the entire image.

- Stimulus/response sequence

  - A frame will be given to this module

  - This module will use segmentation technique to find the Region Of Interest.

- These Regions of Interest will then be given to the feature matching and image recognition module for matching features and recognizing targets.

- Functional Requirements

  - Receiving frame

  This function will receive the frame from the frame extraction module and check whether the image is suitable for performing any operation i.e. there is not too much noise in it. In case the image is not suitable, it will request for a new image.

  - Finding Region of Interest

  This function will be used to segment the image and find the regions in the image that interest us i.e. background and other features that are irrelevant will be discarded and only the suitable regions will be kept for passing on to the feature matching and image recognition module.

4.7.4. Feature Matching and Object Recognition

- Description and Priority

This is the core module of the system. It will use Scale Inline Feature Transform (SIFT) feature matching techniques for the system to match the features of targets already stored and the regions of interest received from the above module.

- Stimulus/Response Sequence

  - An ROI will be given to this module.
  - It will extract the features of this ROI using the SIFT feature detection technique
  - This module will then match the features of this ROI with those of the target objects that are already stored
  - It will then check whether the features of the objects match or not

- If the features of both objects match, an interrupt will be generated and the vehicle will be stopped.
- If the features do not match, the next ROI will be used and the same steps will be applied on it until a possible match is found.

- Functional Requirements

  - Receiving the Region Of Interest

  The Region of Interest will be given to this function. This function will then use other functions to extract features. In case the region of interest is not received, it will not stop rather request again for a region of interest.

  - Extracting features of ROI

  Features of the ROI will be extracted using SIFT feature detection technique. These features will then be used to cross match against those of the target already stored.

  - Matching Features

  This function will perform the feature matching between the target features already stored and the ROI features extracted in the previous function. There can be following cases:

    - The object and the ROI are different and their features do not match. In this case, a new ROI will be taken and same functions will be applied on it.
    - The Object and ROI are same and their features match. In this case, an interrupt will be generated that the object has been found and the vehicle will stop and wait for the next instruction.
    - The object and ROI are same and their features do not match. In this case, there is no need to panic because the object might not be clearly visible or there might be some other problem. The object will appear in the next few

frames also until the vehicle crosses it and the highest probability is that it will be detected in the next frame.

- Checking for Match

If a match is detected, this function will be called and it will generate an interrupt for the vehicle to stop and take the next action. In case a match is not there, next ROI will be taken and the above steps will be performed on them.

4.7.5. Path Following

- Description and Priority

This module is responsible for the movement of the vehicle. The vehicle will move on the path which is a (White/Black) Strip. On the basis of values gathered from the sensors, this module will make the path plan which is used to move the vehicle on the path. It is of High priority. The rating for the benefit is “7”.

- Stimulus/Response Sequence

  - Sensors data are received in this module.
  - Sensor returns a value of 1 or 0 to microcontroller.
  - In response to this value (sensor) command will be given to motors. The vehicle will start moving.
  - If the path is making a turn the sensor will give the values and appropriate maneuver will be planned.
  - Again the commands are given to the both motors whether to slow down or stop.
  - This module will keep on generating values and following the path until the object is detected.
  - As soon as the object is detected an interrupt is generated and the vehicle will stop.

- Functional Requirements

  - ➢ Receive Sensor Value

  Sensor will return high when high contrast is detected. The sensor will return low if low contrast is detected. There will few sensors mounted on the vehicle which will send values simultaneously to the microcontroller.

  - ➢ Microcontroller Action (Maneuvering of Vehicle)

  The values acquired from the sensors are used by microcontroller. A program is burned into the microcontroller which will compute the commands that are to be given to the motors.

  - ➢ Receive Interrupt

  The Interrupt will occur when the object is detected. This Interrupt is issued by the Feature Matching and Object Recognition module when the object is recognized. The interrupt will tell the microcontroller to stop and wait for further instruction. The microcontroller will give stop command to the motors.

4.8. EXTERNAL INTERFACE ENVIRONMENT

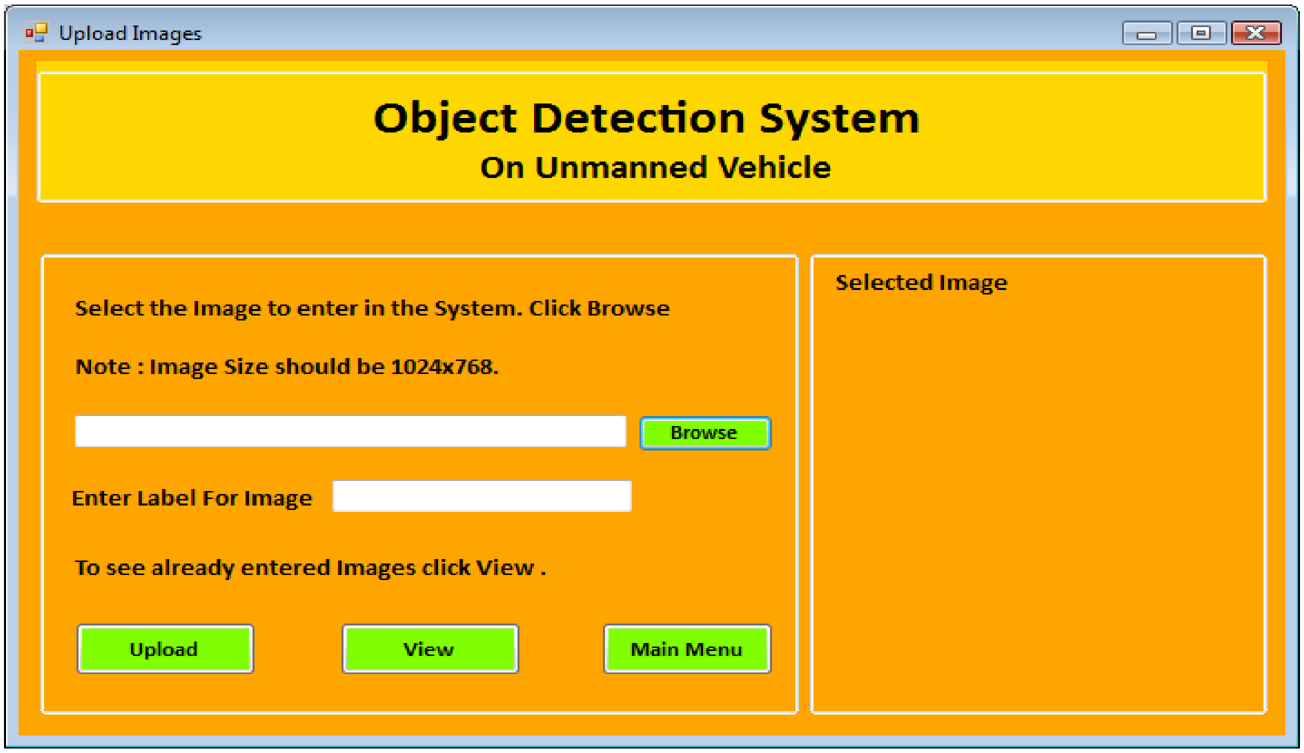
4.8.1. User Interface (Optional)

The interface will based on forms with menu, text buttons and radio buttons etc. The user interfaces will be designed and judged on the basis of the usability attributes such as”

- Learnability
- Speed of operation
- Robustness
- Recoverability
- Adaptability

The user screens will be required by the user to upload the images of the target object, to start the search operations and also if possibly the user screen will provide a live stream video of the vehicles camera (subject to time).

Sample of screen as to how the interface might look to the user are shown on next page.


Upload Images

Object Detection System
On Unmanned Vehicle

Select the Image to enter in the System. Click Browse

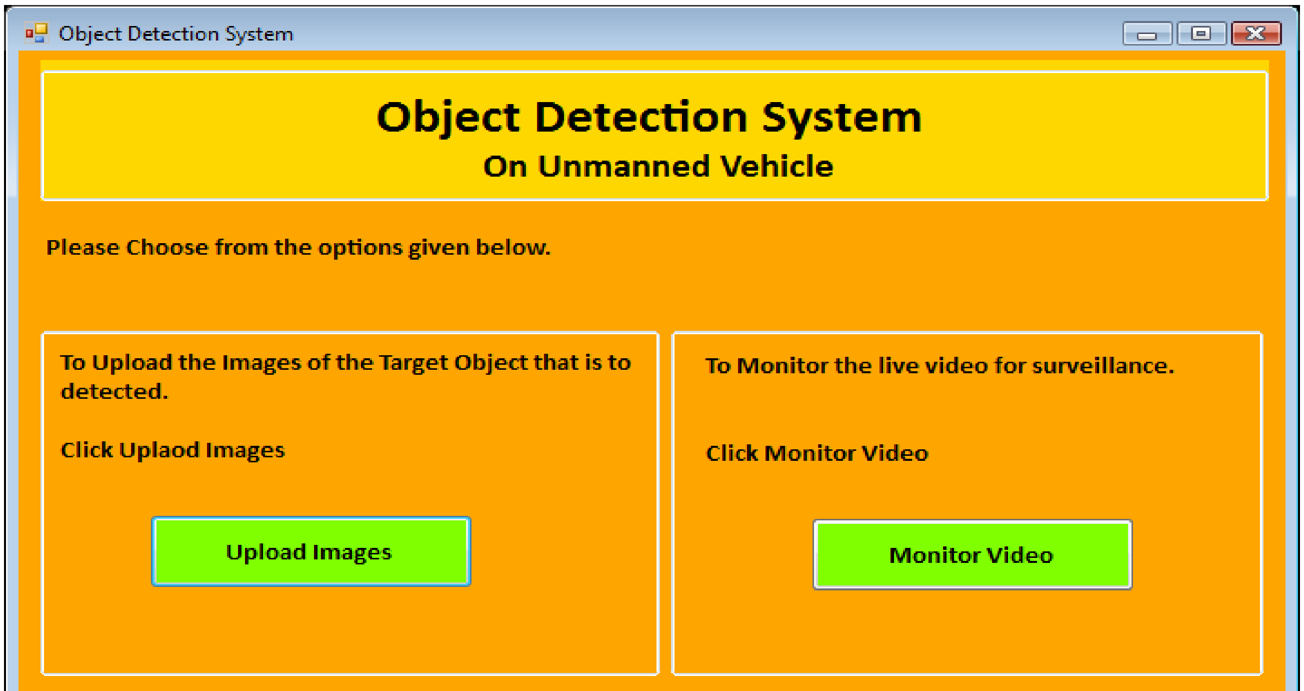
Note : Image Size should be 1024x768.

Enter Label For Image

To see already entered Images click View .

Selected Image

Fig 4.1. User Interface: Target Selection screen


Object Detection System

Object Detection System
On Unmanned Vehicle

Please Choose from the options given below.

To Upload the Images of the Target Object that is to detected.

Click Uplaod Images

To Monitor the live video for surveillance.

Click Monitor Video

Fig 4.2. User Interface: Main Screen

4.8.2. Hardware Interfaces

The motion of the tank will be controlled through a micro-controller. The interfacing among the software system and the microcontroller will be done via a processor (an ARP or an 8051 or any processor).

The motion will be simply based on the values of zero and one. For path following certain calculations will need

4.8.3. Software Interfaces

Name	Version
Linux	Ubuntu
Open CV	2.2 or later
Socket Programming (Optional)	

- **Data Items/messages**
  - Image-data item
  - Recognition interrupt
  - Turning interrupt
  - Video streaming
- **Services**
  - Net beans IDE
  - Open CV libraries
  - VMLab
  - Proteus

4.8.4. Communication Interfaces (Optional)

FTP will be used for exchanging messages with the onboard computer and the command centre, messages such as notifying about the detection of the desired object.

4.9. NON FUNCTIONAL REQUIREMENTS

4.9.1. Performance Requirements

Performance requirements can be categorized as

- Response Time

  For any image processing system response time is a very important. Especially in a target recognition system, it would not be considered successful if it takes a considerable amount of time. Setting a particular time target is hard at this initial stage but the target would be to make the response time at about half a second.
- Throughput

  Throughput is defined as the rate at which incoming requests are completed. For this particular project the throughput will vary because if a few factors such as:
  - Total area in which the find an object
  - Type of terrain
  - The speed with which the vehicle moves
- Concurrency

  Because of the orientation and aims of the project only perform one task at a time that will be given to it
- For Batch jobs

  How often the vehicle will be asked to perform a job depends upon the battery; how long it last.

4.9.2. Safety Requirements

Possible loss, damage or harm by use of this product can be:

- Battery may falter because of excessive use.
- Relays or wiring may burn or short circuit

4.9.3. Security Requirements

- User Authentication for use of vehicle
- Encoded images sent(Optional)

4.9.4. Software Quality Attributes

Additional quality characteristics for the product that will be important to either the customers or the developers to be considered are:

- Adaptability

  The system should be designed in such a way that in can adapt to various different conditions such as weather and location etc. This at this moment will not be possible therefore we have made some assumptions regarding it which are described in chapter 2.

- Correctness

  The system should have a high success rate of detecting the required targets. the operation time can be sacrificed somewhat for higher recognition rate of the project.

- Maintainability

  The system should be easier to maintain. The design of the hardware should be such that fault recognition should be easy. The code of the system should be rhetoric enough so that the tester or a third person can understand and identify any faults if they are present.

- Reliability

  The system should be reliable i.e. it should have a considerable period of time between maintenance and service, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.

Chapter 5: Architecture and Design

5.1. PURPOSE

A solid foundation is very important for building any complex software system. Considering key scenarios, designing for common problems, or appreciating the long term consequences of key decisions need to be done in order to avoid risks. The risks exposed by poor architecture include software that is unstable, is unable to support existing or future business requirements, or is difficult to deploy or manage in a production environment.

Like any other field architecture for vision based projects it also very important. Most vision based projects work in a very systematic and coherent manner with the different modules having specific pre defined routines. A particular order it very crucial in such vision based systems and this defines the result of the system. Therefore, having a solid architecture is very crucial system and it would provide a basic blue print for the design and development of the project and act as the main foundation of the project. Furthermore it will act as a ground zero in case of any major risks or errors come up later during design or implementation.

5.2 SCOPE

The document describes the basic architecture that will act as the foundation on which the system will be developed. It defines the major processing blocks or modules of the system as well as describes the flow of data (frame data) along the whole algorithm. This document addresses the architecture of the image processing algorithm that we will be implementing. The hardware specifications of the document have been specified in another document.

5.3 REFERENCES

Hardware Specification Document

Software Requirement Specification Document

5.4. ARHCITECTURAL REPRESENTATION

The architectural pattern selected for this project is Pipes and Filters. In a pipe line each component has a particular input and an output. A component reads data on it input and processes its particular function or method and provides the resultant data on its output. These local transformations are called as “filters” and the connectors that move data between filters are known as “pipes”

For the vision based system using the Pipes and Filters patterns suits well as we will need to perform operations in a sequential manner with the order of the operations very important. The data will be in the form of the extracted frames from the video feed and the processing filters will be in the form of image processing filters.

5.4.1. Reasons

- Pipes and Filters allow designer to understand the overall input/output of a system as a simple composition of the behavior of individual filters.
- They support reuse, a filter is implementing a particular method or a sub-functionality of the method, hence it can be replicated and place in any other similar system with minimal efforts.
- Enhancements and modification are easier as new filters can be added to existing systems.
- The implementation details of each filter are irrelevant to the high level understanding of the overall system, as long as a logical definition of their behavior is specified (input, transformation and output).
- Implementation and testing can become easier. As the complex system is divided into small less complex filters.

- It also supports concurrent execution. Each filter can be implemented with a separate task and can be executed in parallel with the other filters.

5.4.2. Basic Architecture

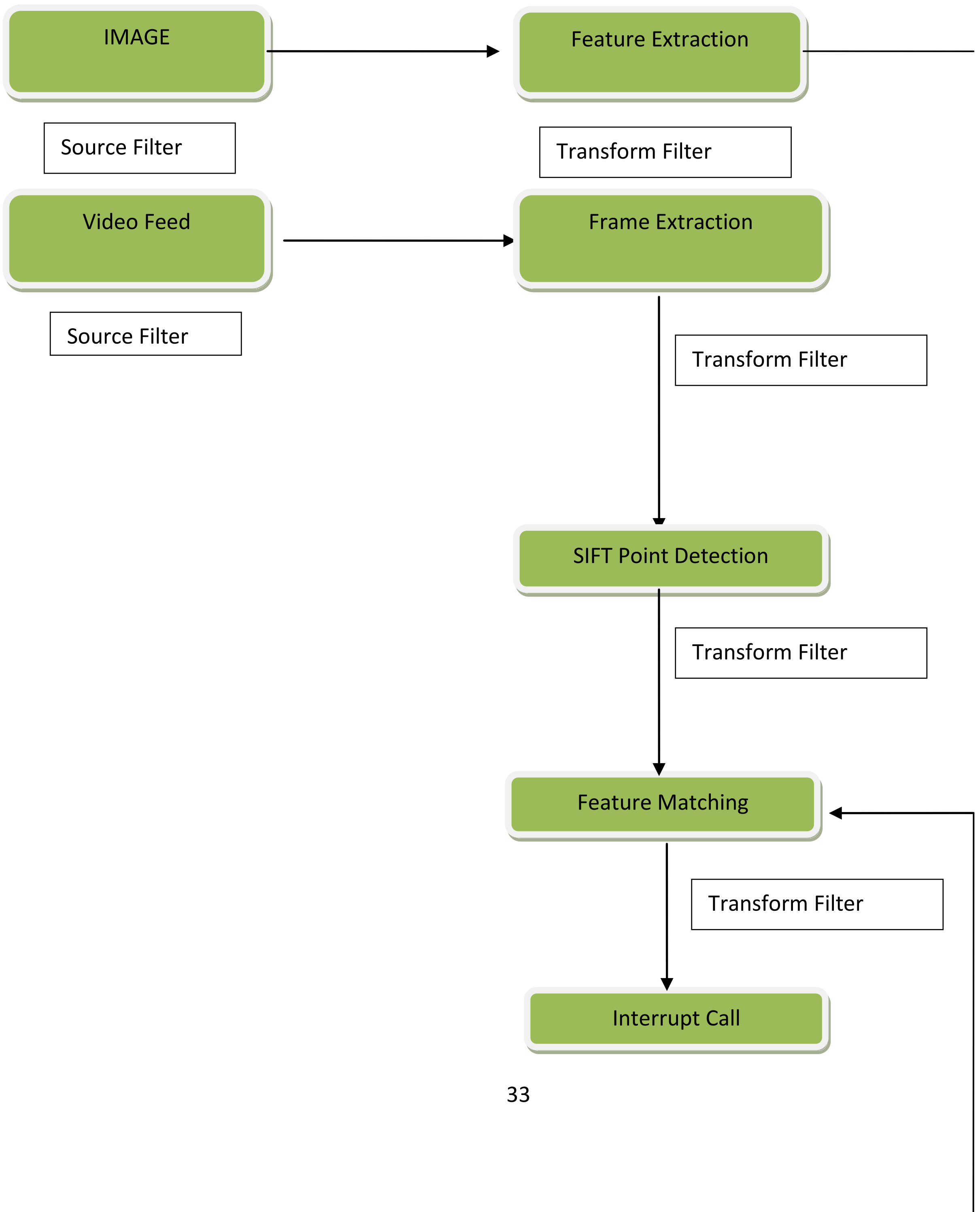


Fig 5.1. Basic Architecture

5.4.3. Description

- The overall system will be basically based on functional programming, data moving through the pipes will be in the form of videos frames, and each filter has a specific task that needs to be performed in the described ordered. Filters can be characterized by their input/output behavior: source filters produce a stream without any input; transform filters consume an input stream and produce an output stream.
- The system being implemented is on the basis of functional programming therefore pipes and filters architecture fits quite well to it.
- This architecture provides a sequential baseline on which to design and implement the algorithm. Each filter in the architecture contains a particular step of the image processing algorithm which may need to be further subdivided.
- An image will be provided to the system. Key points will be extracted from the given feature. As soon as the system in asked to search for the object whose image is provided, the UGV will start moving. A video feed will come through the web cam attached to the UGV. Frames would be extracted from the feed where frames one at a time will be moved to the algorithm for their key point detection. Both the pipes containing data as set of key points pass the data into the matching filter which compares the key points and seeing if they match.
- What this architecture does is that it divides a large processing task into a sequence of smaller independent processing steps. The image processing algorithm that will be implemented would be divided into small group of functions that are affected only by the set inputs rather than method invocations that will make the processing simpler and faster.

5.5. ARCHITECTURAL GOALS AND CONSTRAINTS

The major requirements with impact on the system's architecture are enlisted below.

5.5.1. Response time

For any image processing system response time is a very important. Especially in a target recognition system, it would not be considered successful if it takes a considerable amount of time. Setting a particular time target is hard at this initial stage but the target would be to make the response time at about half a second.

5.5.2. Throughput

Throughput is defined as the rate at which incoming requests are completed. For this particular project the throughput will vary because if a few factors such as:

- Total area in which the find an object
- Type of terrain

5.5.3. Hardware

The hardware dynamics such as its speed, stability etc will be needed to cater for in the architecture. **The hardware design and specification are described in a separated document.**

5.5.4. Interfacing Issues

Interfacing among hardware and software is difficult to describe in an abstract form in this document is very difficult.

5.6. USE CASE VIEW

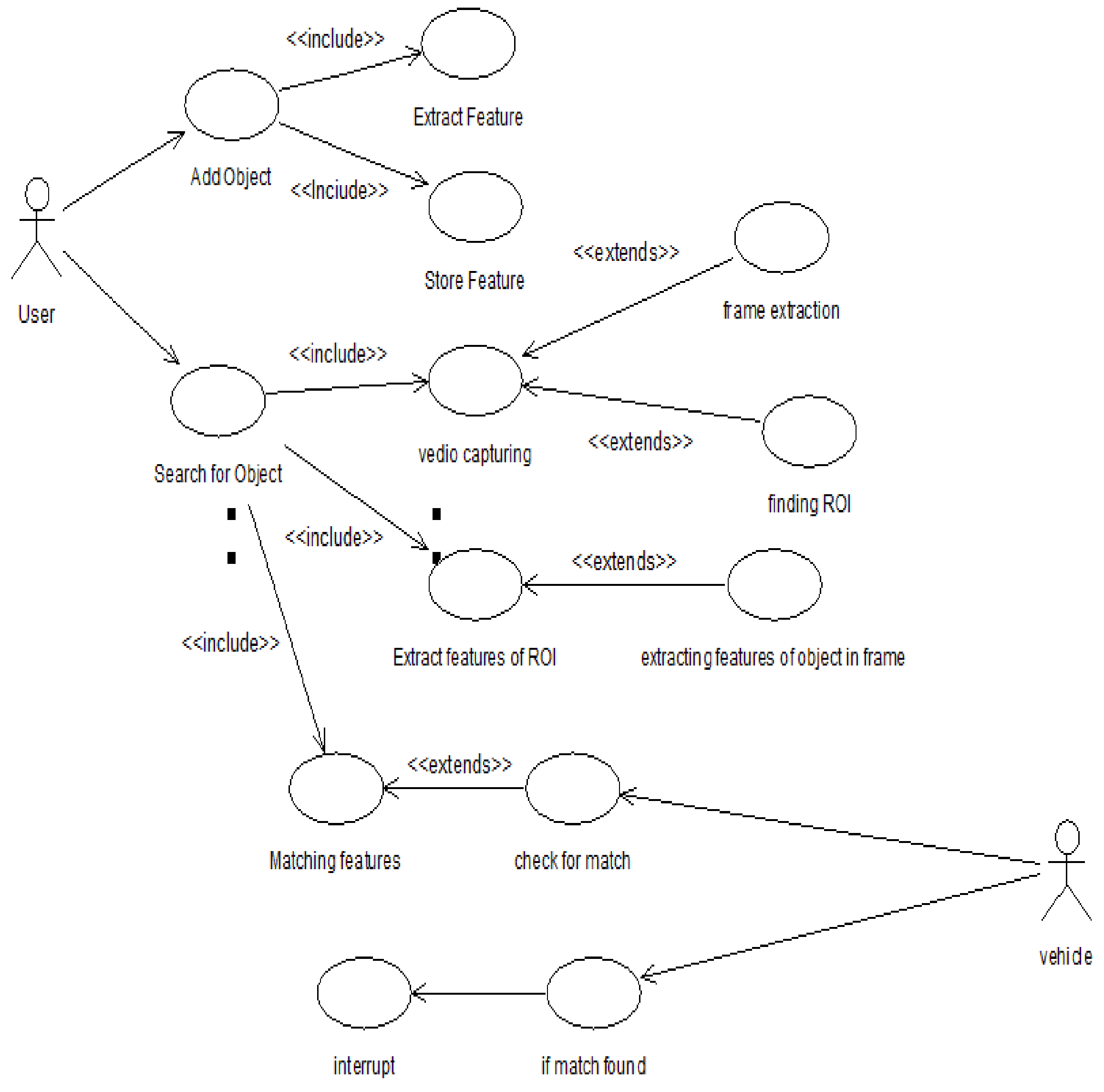


Fig 5.2. Use Case Diagram

5.7. USE CASE SPECIFICATIONS

5.7.1. Use Case: 1- Add Object

- Brief Description. This use case describes how a user will add object image into the system having meaningful label.
- Actor. User.
- Precondition. The user should have clear images of the object which is to be identified. And the image should be of specific size.
- Basic Flow of Events
  - The Main Menu prompts the user to make a choice for the tasks to be performed. User chooses Add Object.
  - The user will browse the location of the image and click upload.
  - After successful upload the system will extract features from the images that are uploaded. (Include)
  - Next the system will store features under a unique label for that object for further use in the processing. (Include)
  - The system will prompt a message that the features are extracted and stored.
  - The use case will end successfully.
- Alternative Flows
  - The image is not clear. In step (3) of the basic flow, if the user uploaded an unclear image the system will prompt the user that the particular image is not clear for features extraction please enter another image. The user will upload another image or simply cancel the operation and the particular image is discarded from the system. The system will resume from step (4).
  - The format is not supported. In step (1) of the Basic flow, if the user mistakenly selects an unsupported file or image the system will prompt the user select correct format image. After that the system will resume from step 2.
  - The image size is too large. If, in step (1), the user selects the image that has very large resolution the system will prompt the user to select the image

having specified resolution or lower. After correct selection the system will resume form step (2) of the basic flow.

- Duplicate Image. If the new image matches to an already existing image, an error message asks the user to either quit or select different image.
- Key Scenarios. NA
- Post-conditions
  - Successful Completion. The system will show the massage that the features are extracted and stored properly.
- Special Requirements.NA

5.7.2. Use Case: 2- Search for Object

- Brief Description. This use case describes how a user can monitor the video and the object detection. More important is how the system detect the object the flow of the system is described in this use case.
- Actor. User.
- Precondition. There should be some images of the object and the features of that object.
- Basic Flow of Events
  - The Main Menu prompts the user to make a choice for the tasks to be performed. User chooses Monitor video.
  - Camera is connected to the system, when user choose monitor video option the system start capturing video feed form the camera.(Include)
  - After that the system extract frames from the video stream (the rate of the video frame is 4fps).
  - The next step is the system will segment the frame and extract region of interest ROI from every frame that is received.
  - Next the system will extract the features of the ROI (SIFT feature detection technique) which is previously extracted from frames. In this step the system will extract feature of the object in the ROI.(Include)
  - The system will now match the features of the ROI with the features of the target already saved by the user.

        - The object and the ROI are different and their features do not match. In this case, a new ROI will be taken and same functions will be applied on it.
        - The Object and ROI are same and their features match. In this case, an interrupt will be generated that the object has been found and the vehicle will stop and wait for the next instruction.
        - The object and ROI are same and their features do not match. In this case, there is no need to panic because the object might not be clearly visible or there might be some other problem. The object will appear in the next few frames also until the vehicle crosses it and the highest probability is that it will be detected in the next frame.
    - After the feature matching the object is identified in the video and the user can see it on the screen and a sound notification will be generated.
    - The use case end successfully here.
- Alternative Flows
    - Problem in video feed.   In step (2) of the basic flow, if there is problem in the video i.e. the camera is not sending the video the user will be prompted to check the cable of the camera. After the user find the problem then the system will resume from step (3).
    - Faulty Frame.   In step (3) of the basic flow if the frame which is extracted is noisy or blurred the system will discard this frame and will extract the next suitable frame this step will repeat itself until a proper frame is extracted.
    - Features do not match.   In step (7) if the features of the frame and the features of the object already stored do not match then the system will go to step (2) and again receive the video feed from the camera and will resume from there. Basically system will be in loop until the features match correctly.
- Key Scenarios.   NA
- Post-conditions
    - Successful Completion.   Features match correctly.
    - Failure Condition. The features do not match.
- Special Requirements.NA

5.7.3. Use Case: 3- if match found

- Brief Description. This use case describes that if the features matches then what will vehicle do.
- Actor. Vehicle.
- Precondition. The vehicle should be in running mode and following the path.
- Basic Flow of Events
  - ➢ The system will match the features of the object already stored and the features of which are extracted from the frame.
  - ➢ If the features are matched the vehicle will get an Interrupt from the system.(Include)
  - ➢ The microcontroller which is controlling the movement of the vehicle will immediately issue the command or instruction to the motors to stop.
  - ➢ The use case will terminate successfully.
- Alternative Flows
  - ➢ If feature do not match. In step (1) of the basic flow, if the features do not match the system will not give the interrupt. The vehicle will continue following the path until the features match with any frame of the video feed. The use case will terminate here.
- Key Scenarios. NA
- Post-conditions
  - ➢ Successful Completion. The vehicle will stop after the recognition of the object.
- Special Requirements.NA

5.8. LOGICAL VIEW

This section presents an overall structure and logical view of the project. It describes overall structure of the model, decomposition into further subsystems as well as their responsibilities.

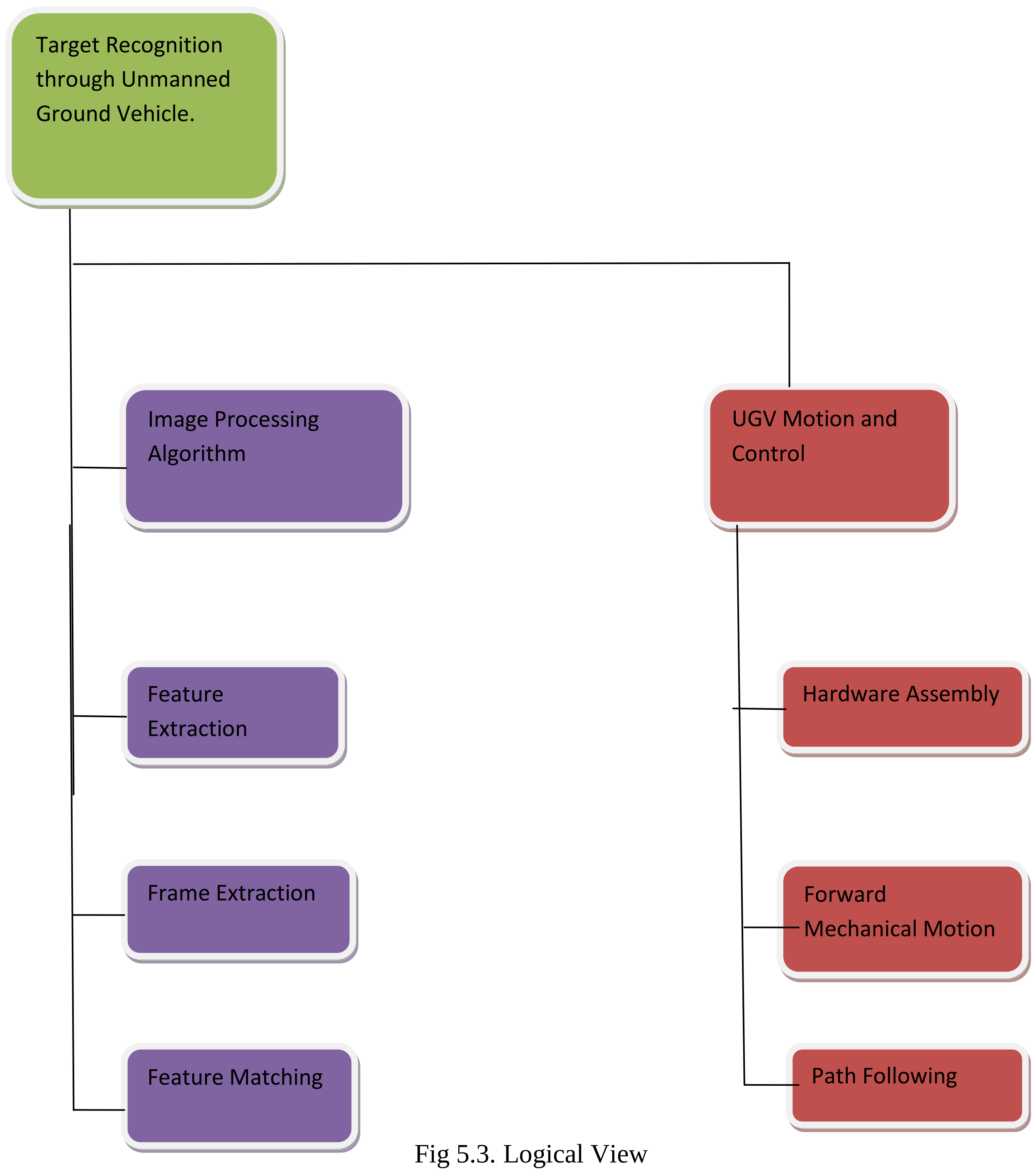


Fig 5.3. Logical View

5.9. RESPONSIBILITIES

5.9.1. Image Processing Algorithm

Responsible for the overall target recognition software system. Consists of the basic image processing techniques that need to be applied to the image of the object given and to the video frames.

- Feature Extraction

  Take out key point/features out of the given image and the video frames to create a set of key points to match the given image with the video feed frames

- Frame Extraction

  Extract frames at a set rate from the live video feed from the web cam, frames are required by the feature extraction system to identify keep points in that very frame.

- Feature Matching

  This is where the main result of the system is generated, the results of the feature extraction of the given image will be matched with each frame, as certain minimum number of key points will be required consider to consider the frame as the desired object.

5.9.2. UGV Motion and Control

Responsible for the main hardware development, its operation and functioning. The control of speed and motion of the vehicle and path following.

- Hardware Assembly

  Developing the basic hardware of the UGV. Assembling the main frame of the UGV its overall body. Mounting motors and other circuit operations that need to be developed and attached to the UGV in order for it to work properly.

- Forward Mechanical Motion

  This part will deal with making the UGV work. Developing the basic processor circuit. Creating and implementing the code for the motion of the vehicle. The main task of this part will be to burn the motion code on the microcontroller and also have a port on the processor board for communication with the laptop which is required in order for the vehicle to be stopped once target has been recognized.

5.10. SOFTWARE DESIGN

5.10.1. Introduction

- Purpose

The purpose of this document is to provide a design for the implementation of Vision Based target Detection via unmanned ground vehicle. The system is designed to search for (on a given path) and detect any object whose images have been provided to the system. It is a vision based software system that is being developed over a physical hardware, creating a single main commanding system for both the hardware and software hence making them work coherently.

- Scope

The scope of the document contains an overview of the major modules; it describes the main data flow as well as the overall states the system is in during operation. This document is only related to the software design and its interaction with hardware. The hardware design is provided in a separate document.

- Goals and Requirements

The model presented in this document is intended to address the following goals and requirements:

- Simple module division to simplify operations
- Reduced development costs through reuse and separation of functional and technical architectures
- Simplification of instrument setup/take-down to reduce the chance for errors and increase observing efficiency.
- Improve efficiency of the image processing algorithm.
- Maintain a high accuracy rate

5.10.2. Design Overview

5.10.2.1. System Overview

The overall system is a vision based system along with a central nucleus to control the vehicle's motion; (when to start it and when to stop it). The overall overview of the system is shown below.

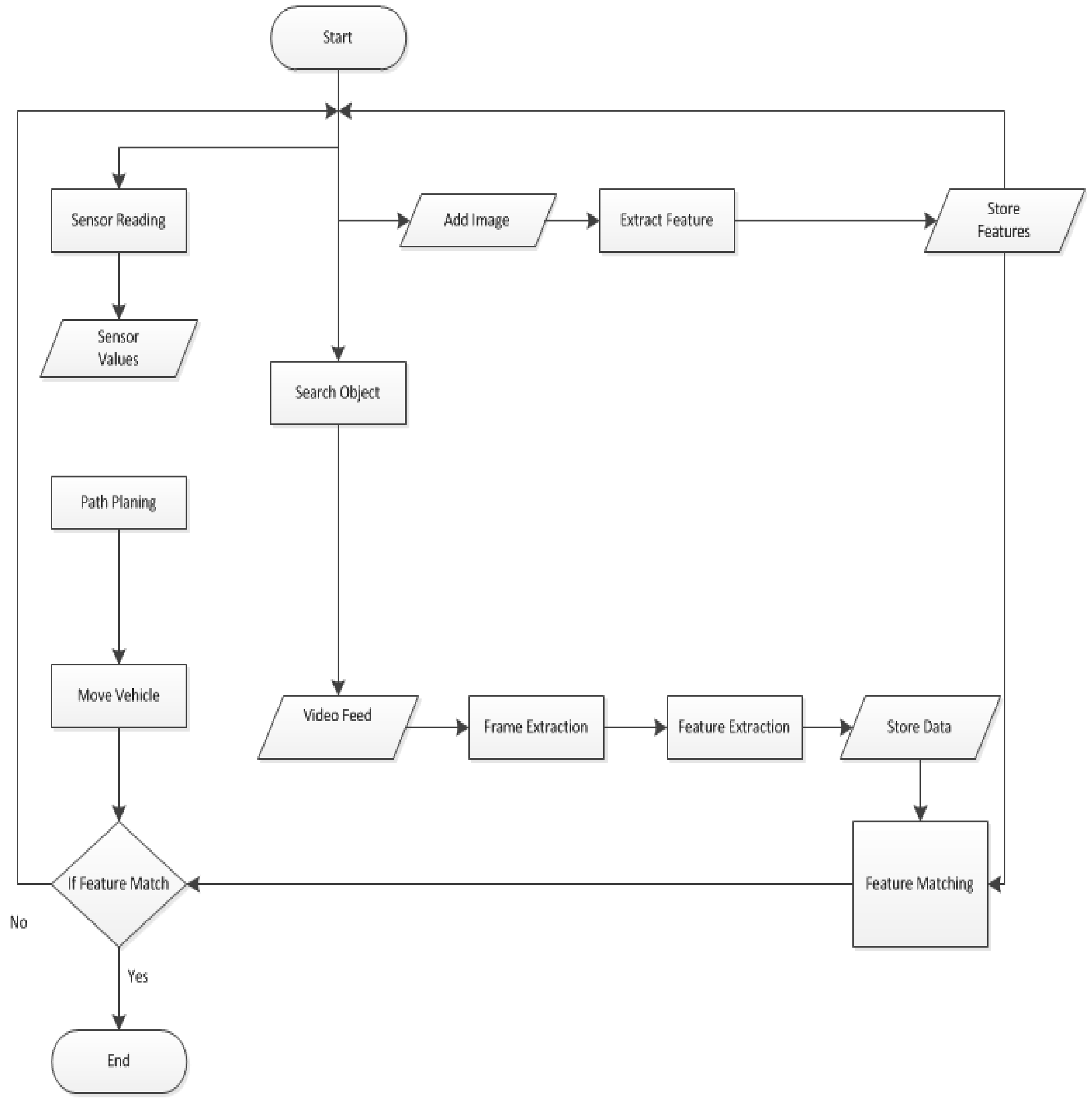


Fig 5.4. System Overview

5.10.2.2. Data Flow

Data in this project is in the form of video frames, different filters will be applied on each frame at different steps of the implementation.

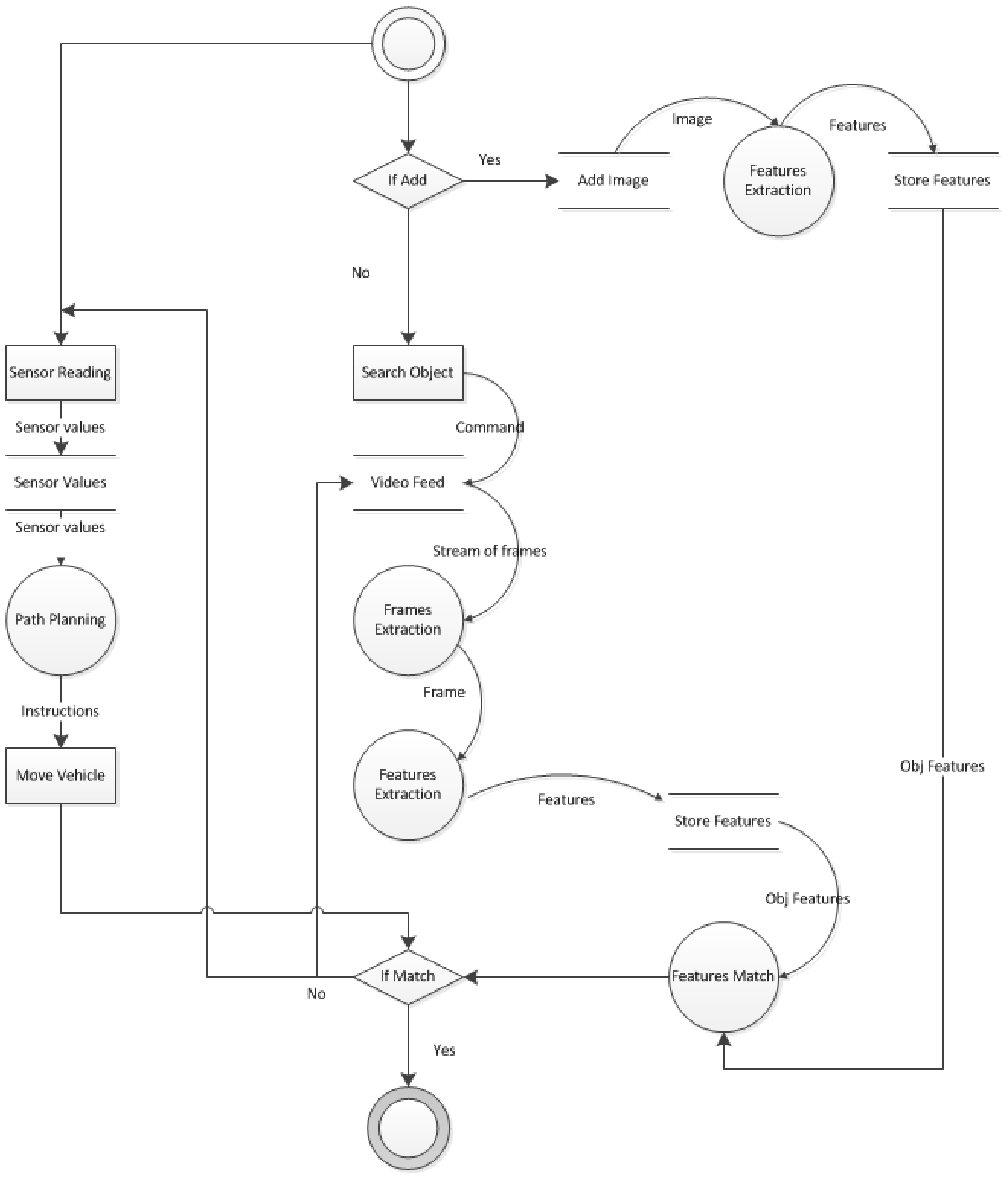


Fig 5.5. Data Flow Diagram

5.10.2.3. System Sequence

The operational sequence of the system is described below:

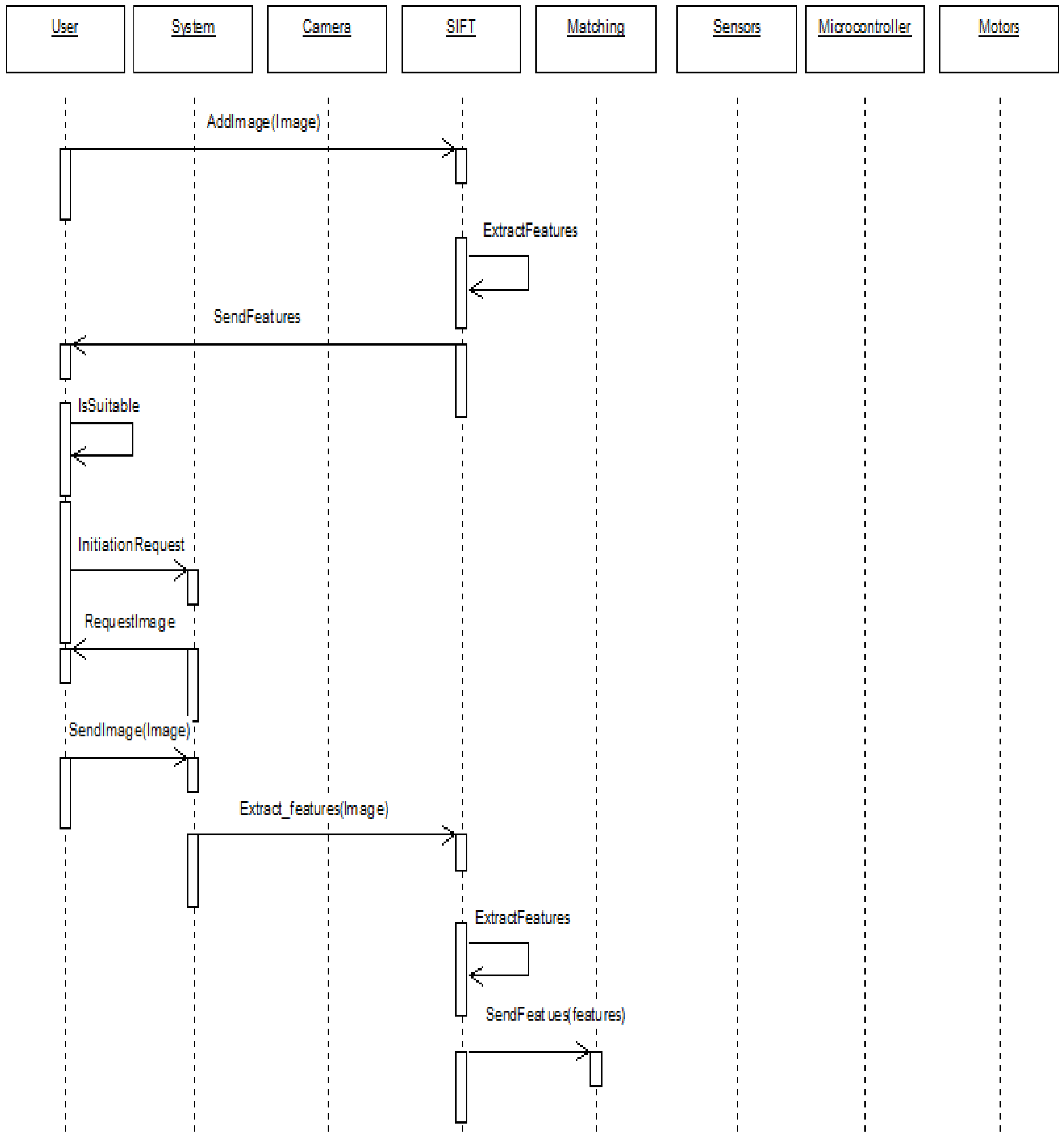


Fig 5.6 (a). Sequence Diagram

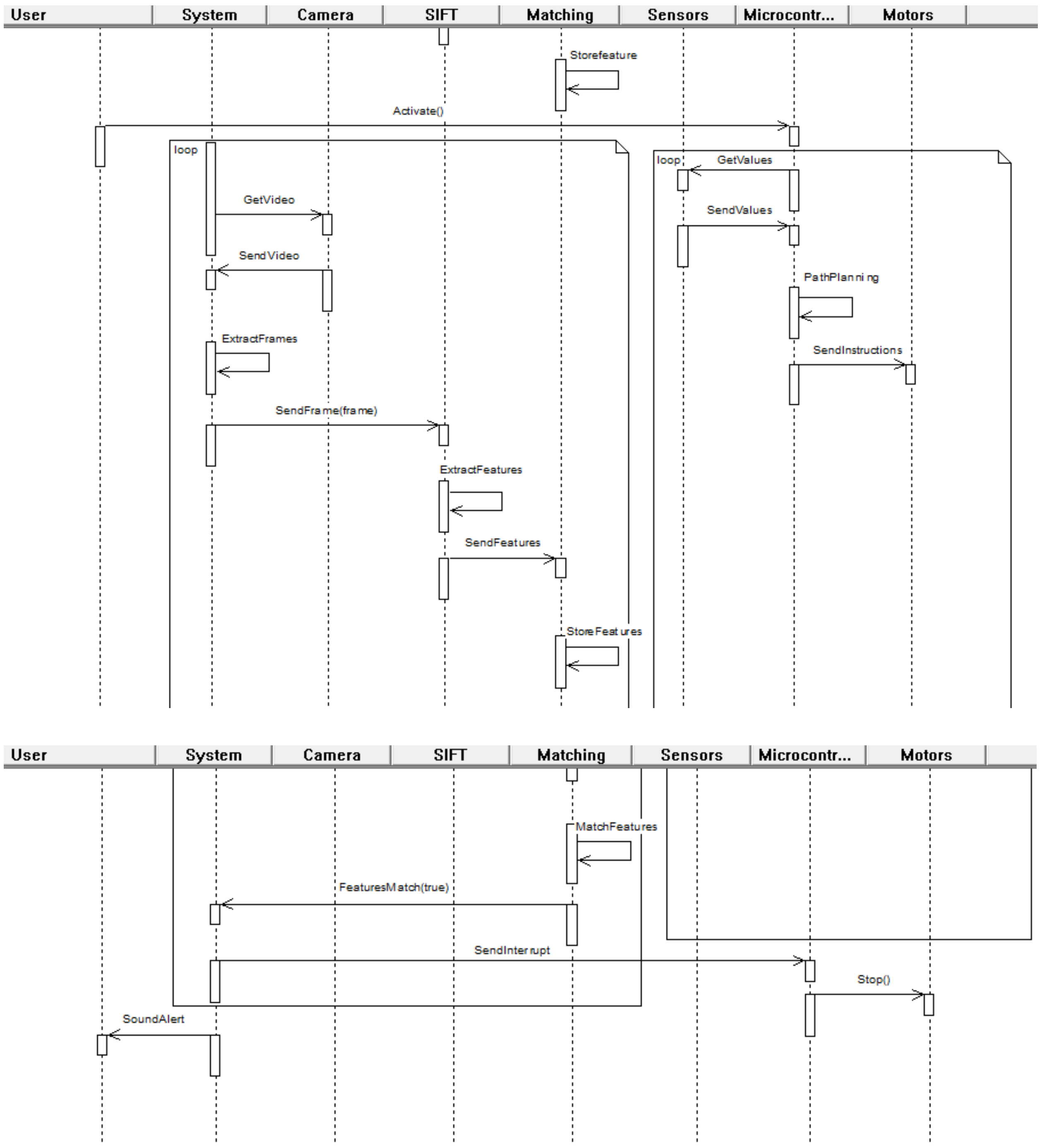


Fig 5.6 (b). Sequence Diagram (continued)

5.10.2.4. State transition Diagram

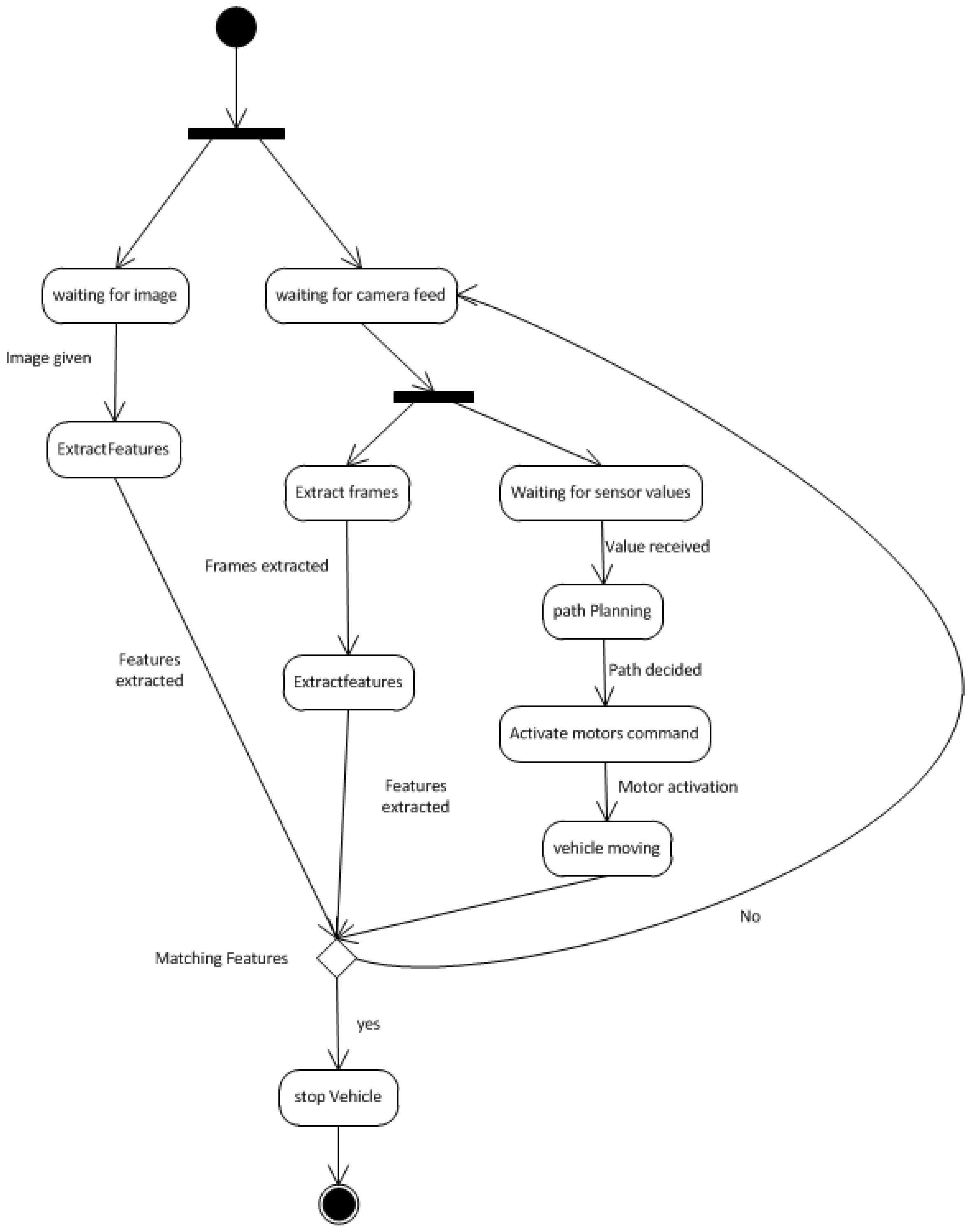


Fig 5.7. State Transition Diagram

5.10.3. Developmental Method

Functional programming is the developmental methodology that will be used in this system. The main components of the software are functions, not objects. The complex system is divided into smaller easier modules which can be implemented in functional programming paradigm in a given language(C language in this case).

The project basically is a set of sequential operations that need to be performed on one image at a time, therefore there is no need to introduce complexities by adding object oriented paradigms and design patterns.

C is the language used for image processing in our project. The three main reasons are

- The most widely used Image Processing library is OpenCV. It is written in C language. Our implementation will use these OpenCV and SIFT libraries and therefore, programming in C will make the task simpler and faster to compile with the pre written libraries.
- As our project is based on Linux OS, the built in compiler for C (Gcc) in Linux makes it easier to compile the code using all the libraries. For other languages, Compilers would have to be installed.
- Our project also deals with hardware (microcontroller for the vehicle) and all this hardware is programmed in C and Assembly language. It will make interoperability easier and simpler for us if we us C for Image Processing.

Libraries used:

- OpenCV libraries by Intel and Willow Garage
- SIFT libraries by Robb Hess

5.11. HARDWARE SPECIFICATION AND DESIGN

5.11.1. Introduction

- Purpose

This document describes the model of the vehicle being developed for the Vision Based Target recognition Project. This document highlights the basic components of the vehicle, its structure and working. This document also describes the interaction between the software and the vehicle itself.

- Scope

This document describes the model of the vehicle being developed for the Vision Based Target recognition Project. This document highlights the basic components of the vehicle, its structure and working. This document also describes the interaction between the software and the vehicle itself.

- Objectives

The document provides a complete design of the hardware components to be fabricated. It also provides the overview on the operations of the vehicle; its motion and the procedure of path following

- Benefits

The benefits of the hardware design document are to:

  - ➢ Enable project team to fabricate the hardware components.
  - ➢ Give an overview of the hardware to the stakeholders
  - ➢ Form the basis for hardware unit testing

5.11.2. Overall Architecture

The UGV is build on a wooden frame that acts as the main chassis of the vehicle with a ratio of 2:1(length: width) .Axel bolts consisting of nut bolts and washers are the responsible for the movement of the vehicle. The vehicle is being designed in a shape quite similar to a tank

5.11.2.1. Basic Structure

The UGV is being designed following the basic structure of a tank. The reason for such a design is that it enables the vehicle to be used on different terrains and can still be kept stable on uneven surfaces. Two independent motors are bolted one on each side for the motion of the UGV. The motors are discussed in detail later. They motors are operated through a motor control relay board that will be responsible for the motion of the motors. A battery will be connected to power the motors and the rest of the circuitry. A microcontroller based circuit will be responsible for sending commands to start and stop the vehicle. This microcontroller will be coded with the basic operation of forward motion. A laptop mount and a camera mount will be present on the upper side of the tank to accommodate the laptop and the camera.

The details of the elements being used in the UGV are given below:

- Frame

  The frame of the UGV is made out of wood for simplicity. The axle bolts are the most important part of the frame and they keep the entire UGV model together. They consist of Nuts and washers that fasten the main axle bolts.

- Motors

  In order to move the vehicle motors are required, the motors being used for this UGV are DC Gear motors. Gear motors are complete motive force systems consisting of an electric motor and a reduction gear train integrated into one easy-to-mount and -configure package. The reason to use these motors is because of

  - Their simple working and easy to configure.
  - The range of their RPM is quite wide and can go from very high to very low RPM hence improving the control over vehicle speed.

- ➢ Lower purchase cost, as similar motors are also used in the power windows etc of automobiles they are cheaper and easily available new and used.
- ➢ They are reliable and do not breakdown or stop working regularly.

Two independent motors are connected one on each side. The reason to make them independent is to make the UGV turn easier as slowing down one motor respect to the other motor will make the UGV turn towards the lower speed motor. Calibration will need to be done in order to calculate the gradual amount of speed reduced as stopping one motor might result in turning the vehicle over.

- Power Source

A battery would be connected to provide the power for the motors to work. The capacity of the battery is will be decided upon implementation time keeping in view the time it will need to power the motors for the operation.

- Microcontroller Circuit

A microcontroller would be responsible for the controlling the movement of the UGV at a particular speed. The processor will have a port for the connection with the laptop that will trigger the interrupt to stop the UGV once the target image has been recognized. The processor to be used will be either Intel 8051 or AVR.

- Motor Controller Relay Board

The relay board consists of a set of 8 relays, 4 for each motor that are responsible for triggering the motion of the UGV. Since the microcontroller cannot by itself produce enough power to make the motor move relays are used. In order to move the UGV the microcontroller output will be passed on to the relay at the positive end. This will trigger the connection of the battery to the relay hence passing the required power to the motor

- Camera

A high resolution webcam will be mounted any side of the UGV, the video feed will be then passed to the onboard laptop which would perform the image processing algorithm.

5.11.3. Dimensions

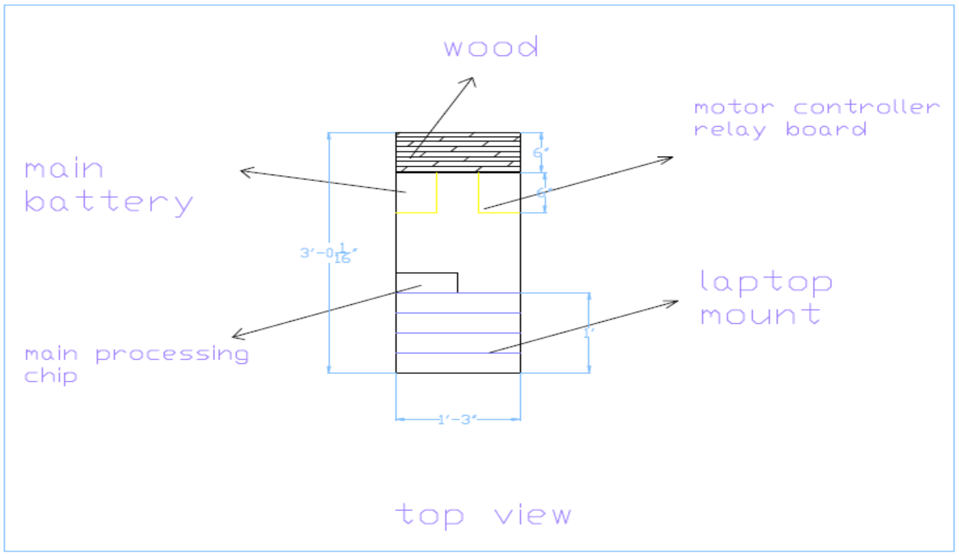


Fig 5.8 (a). Dimensions (Top View)

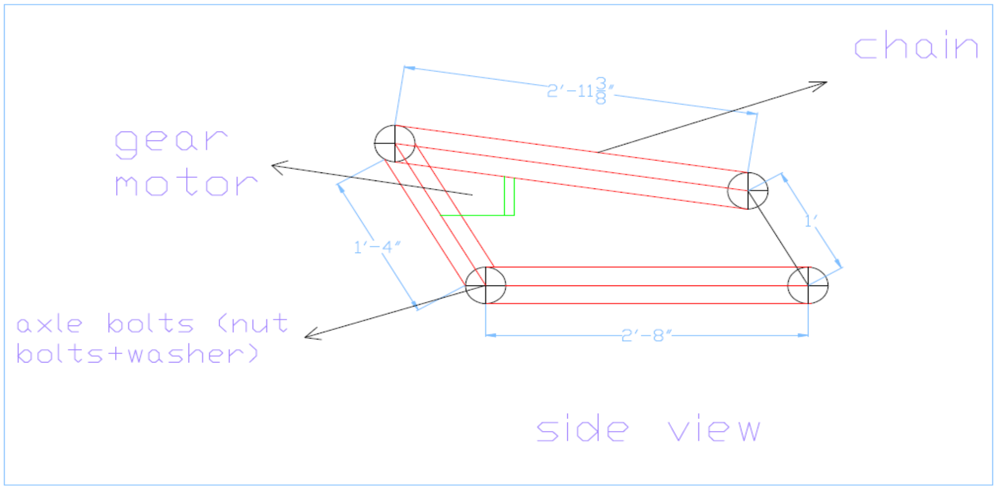


Fig 5.8 (b). Dimensions (Side View)

5.11.4. Processor

For the UGV AVR ATmega16 will be used. The high-performance, low-power Atmel 8-bit AVR RISC-based microcontroller combines 16KB of programmable flash memory, 1KB SRAM, 512B EEPROM, an 8-channel 10-bit A/D converter, and a JTAG interface for on-chip debugging. The device supports throughput of 16 MIPS at 16 MHz and operates between 4.5-5.5 volts.

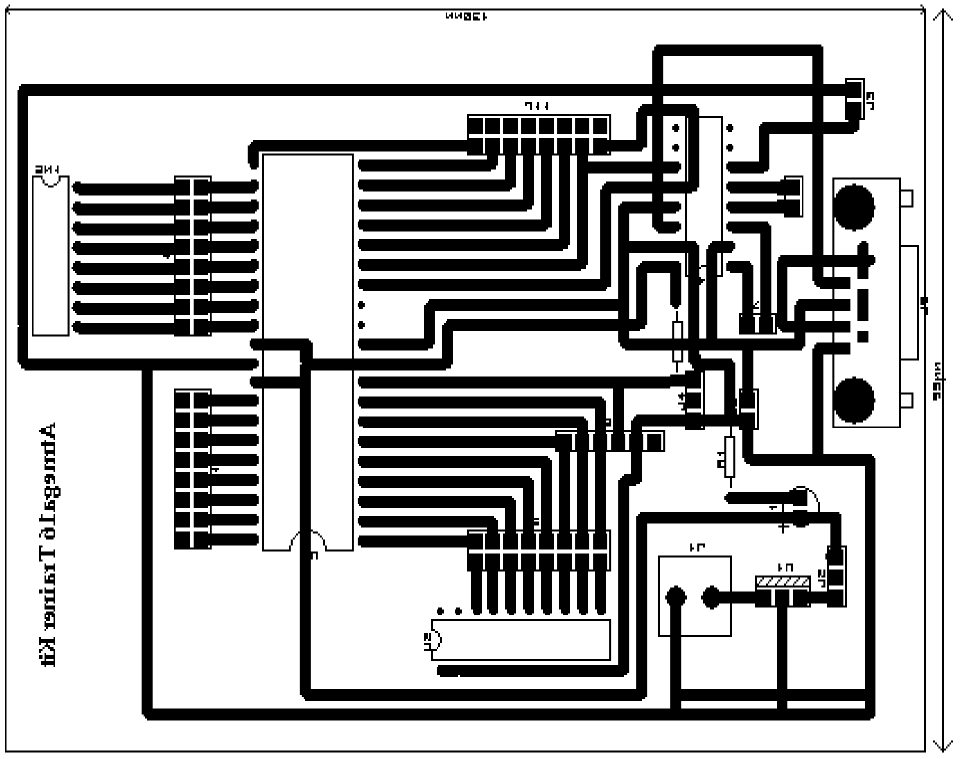


Fig 5.9. Processor Diagram

5.11.5. Design Methodology

5.11.5.1. Step 1

The first step in the making a working hardware model is to first ensure the motion of the UGV before we add the constraint of path following. This approach will make error identification and analysis easier. As soon as the main code and mechanical part of the UGV provide the required result i.e. motion at a particular speed, the second step i.e. path following will be implemented.

We designed the circuit in following sections.

- At first we designed the serial communication circuit block(sending of data to microcontroller serially)
- Invert logic and H Bridge relay circuit was designed at second level.
- Then interfacing the Relay circuit with tank motors.

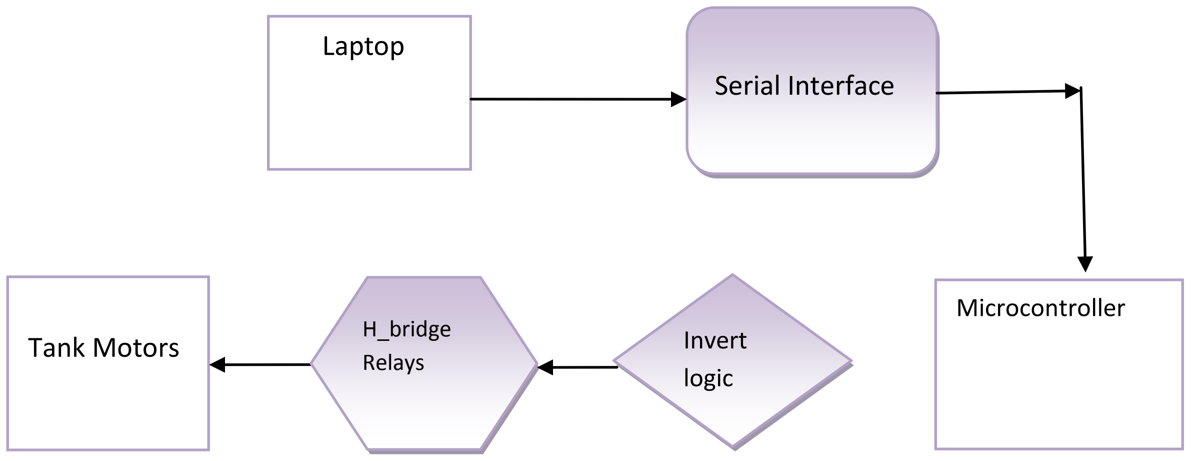


Fig 5.10. Block Diagram

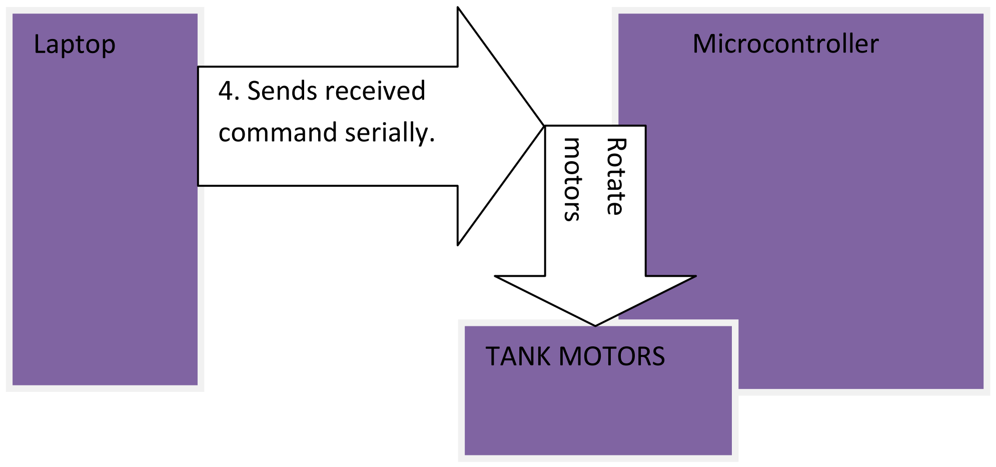


Fig 5.11. Flow Chart

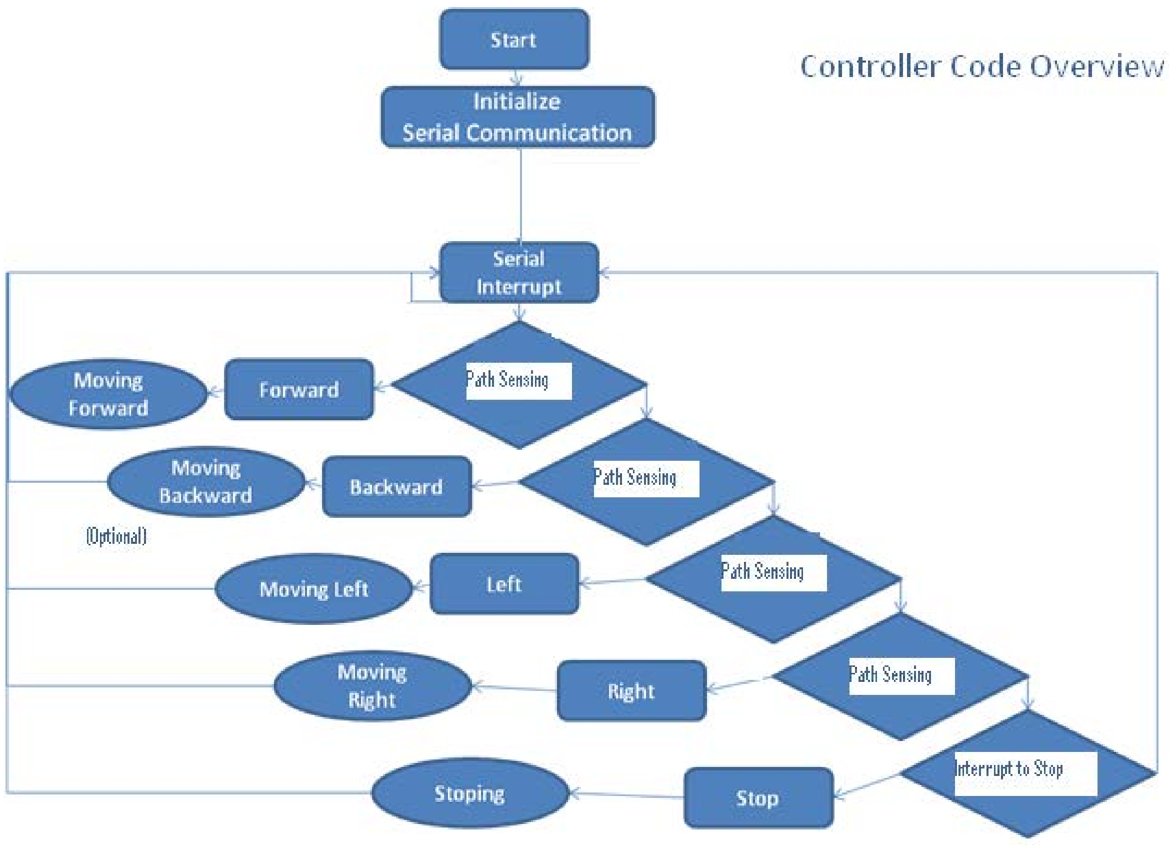


Fig 5.12. Controller Code Overview

5.11.5.2. Serial Communication

The UGV will get commands (data) from laptop to microcontroller which will give command to motors to run in forward direction. For this communication we will use full duplex serial port of microcontroller. The laptop sends data to comport which is attached serially to microcontroller.

5.11.5.3. Step 2

After the assembly and basic functioning check of the UGV and its interfacing with the laptop. The UGV hardware and code both will be modified for path following. The basic design of the path following system is described below

- Line Tracking

Phototransistors will be mounted at very narrow angle to get more precise data for decision making of the UGV.

These sensors sense the light in there external environment. They are made of photo diodes and photo transistors. Light dependent resistor can also use for light sensing. Their resistance decreases as light falls on it.

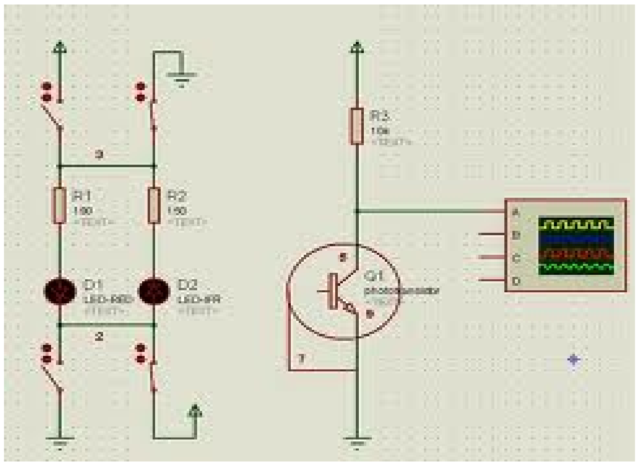


Fig 5.13. Photo Transistor Proteus Diagram

5.11.5.4. Comparators

Operational amplifiers can be used as comparators. They can compare two voltages or current and will be used for comparing the output of phototransistors with a high reference voltage Vref for line tracking.

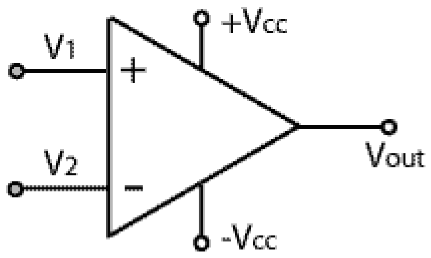


Fig 5.14. Schematic of Comparator

From figure, if V1 is greater than V2 then the output will be +VCC and If the V1 is less than V2 then output will be –VCC. Suppose if phototransistors output is V1 is high then the output of OP-amp will be high.

This will provide the basic distinction on the contrasting path.

5.11.5.5 Comparator Board

Comparator board consists of LM324 IC. LM324 takes input from the emitter of Phototransistor, compares it with reference voltage and pass on the resulting output voltage to microcontrollers input pin. Reference voltage in the comparator circuit board is adjusted by variable resistance.

5.11.5.6. Line Tracking Circuits

To complete the task, our robot should follow the white line. To achieve this goal we have designed our line tracking circuits by using used silicon phototransistor L14G3 and power LED's.L14G3 is basically a silicon phototransistor, whose collector leg is connected to applied voltage, base is grounded and Emitter leg is connected to LM324. We designed one circuit for forward line tracking and two circuits for junction line tracking.

5.11.5.7. Basic Flow

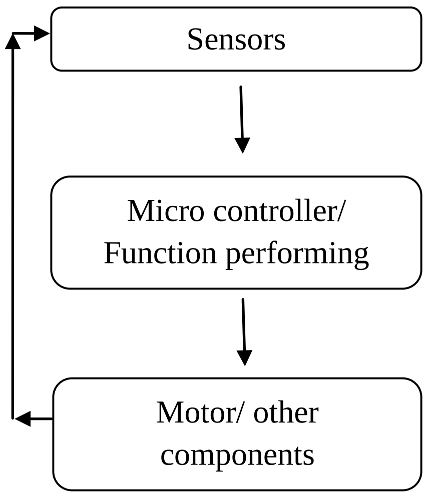


Fig 5.15. Basic Flow

Chapter 6: Implementation

6.1. SOFTWARE IMPLEMENTATION

6.1.1. Image Processing Algorithm (SIFT)

The main benefits of SIFT are that the features are invariant to image scaling and rotation, and partially invariant to change in illumination and 3D camera viewpoint. They are well localized in both the spatial and frequency domains, reducing the probability of disruption by occlusion, clutter, or noise. Large numbers of features can be extracted from typical images with efficient algorithms. In addition, the features are highly distinctive, which allows a single feature to be correctly matched with high probability against a large database of features, providing a basis for object and scene recognition. The cost of extracting these features is minimized by taking a cascade filtering approach, in which the more expensive operations are applied only at locations that pass an initial test.

Following are the major stages of computation used to generate the set of image features:

- Scale-space extrema detection

  The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.

- Keypoint localization

  At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.

- Orientation assignment

  One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

- Keypoint descriptor

  The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination. This approach has been named the Scale Invariant Feature Transform (SIFT), as it transforms image data into scale-invariant coordinates relative to local features.

For image matching and recognition, SIFT features are first extracted from a set of reference images and stored in a database. A new image is matched by individually comparing each feature from the new image to this previous database and finding candidate matching features based on Euclidean distance of their feature vectors. This paper will discuss fast nearest-neighbor algorithms that can perform this computation rapidly against large databases.

The keypoint descriptors are highly distinctive, which allows a single feature to find its correct match with good probability in a large database of features. However, in a cluttered image, many features from the background will not have any correct match in the database, giving rise to many false matches in addition to the correct ones. The correct matches can be filtered from the full set of matches by identifying subsets of keypoints that agree on the object and its location, scale, and orientation in the new image. The probability that several features will agree on these parameters by chance is much lower than the probability that any individual feature match will be in error. The determination of these consistent clusters can be performed rapidly by using an efficient hash table implementation of the generalized Hough transform.

Each cluster of 3 or more features that agree on an object and its pose is then subject to further detailed verification. First, a least-squared estimate is made for an affine approximation to the object pose. Any other image features consistent with this pose are identified, and outliers are discarded. Finally, a detailed computation is made of the probability that a particular set of features indicates the presence of an object, given the accuracy of fit and number of probable false matches. Object matches that pass all these tests can be identified as correct with high confidence.

6.1.2. Libraries Used

- OpenCV libraries
  - Cv.h
  - Highgui.h
  - Cxcore.h
- Open source Sift libraries
  - Imgfeatures.h
  - Kdtree.h
  - Sift.h

6.1.3. Function Documentation

- import_features()
  - Reads image features from file. The file should be formatted either as from the code provided by the Visual Geometry Group at Oxford or from the code provided by David Lowe.
  - Parameters:
    - filename location of a file containing image features
    - type determines how features are input. If type is FEATURE_OXFD, the input file is treated as if it is from the code provided by the VGG at Oxford:http://www.robots.ox.ac.uk:5000/~vgg/research/affine/index.html, If type is FEATURE_LOWE, the input file is treated as if it is from David Lowe's SIFT code: http://www.cs.ubc.ca/~lowe/keypoints

        - feat pointer to an array in which to store imported features; memory for this array is allocated by this function and must be released by the caller using free(*feat)
    - Returns:
        - Returns the number of features imported from filename or -1 on error

- export_features()
    - Exports a feature set to a file formatted depending on the type of features, as specified in the feature struct's type field.
    - Parameters:
        - filename name of file to which to export features
        - feat feature array
        - n number of features
    - Returns:
        - Returns 0 on success or 1 on error
- draw_features()
    - Displays a set of features on an image.
    - Parameters:
        - img image on which to display features
        - feat array of Oxford-type features
        - n number of features

- kdtree_build()
  - A function to build a k-d tree database from keypoints in an array.
  - Parameters:
    - features an array of features; this function rearranges the order of the features in this array, so you should take appropriate measures if you are relying on the order of the features (e.g. call this function before order is important)
    - n the number of features in features
  - Returns:
    - Returns the root of a kd tree built from features.
- kdtree_bbf_knn()
  - Finds an image feature's approximate k nearest neighbors in a kd tree using Best Bin First search.
  - Parameters:
    - kd_root root of an image feature kd tree
    - feat image feature for whose neighbors to search
    - k number of neighbors to find
    - nbrs pointer to an array in which to store pointers to neighbors in order of increasing descriptor distance; memory for this array is allocated by this function and must be freed by the caller using free(*nbrs)
    - max_nn_chks search is cut off after examining this many tree entries
  - Returns:

        - Returns the number of neighbors found and stored in nbrs, or -1 on error.

- kdtree_release()
    - De-allocates memory held by a kd tree.
    - Parameters:
        - kd_root pointer to the root of a kd tree
- sift_features()
    - Finds SIFT features in an image using default parameter values. All detected features are stored in the array pointed to by feat.
    - Parameters:
        - img the image in which to detect features
        - feat a pointer to an array in which to store detected features; memory for this array is allocated by this function and must be freed by the caller using free(*feat)
    - Returns:
        - Returns the number of features stored in feat or -1 on failure
- _sift_features()
    - Find SIFT features in an image using user-specified parameter values. All detected features are stored in the array pointed to by feat.
    - Parameters:
        - img the image in which to detect features

- feat a pointer to an array in which to store detected features; memory for this array is allocated by this function and must be freed by the caller using free(*feat)
- intvls the number of intervals sampled per octave of scale space
- sigma the amount of Gaussian smoothing applied to each image level before building the scale space representation for an octave
- contr_thr a threshold on the value of the scale space function , where is a vector specifying feature location and scale, used to reject unstable features; assumes pixel values in the range [0, 1]
- curv_thr threshold on a feature's ratio of principle curvatures used to reject features that are too edge-like
- img_dbl should be 1 if image doubling prior to scale space construction is desired or 0 if not
- descr_width the width, , of the array of orientation histograms used to compute a feature's descriptor
- descr_hist_bins the number of orientations in each of the histograms in the array used to compute a feature's descriptor

➢ Returns:

- Returns the number of keypoints stored in feat or -1 on failure

6.2. HARDWARE IMPLEMENTATION

6.2.1. Basic Model

The basic model of the UGV has been assembled using cut out wood, with both side panels attached to the main lower base. The side panels are reinforced through upper wooden beams. The UGV has been manufactured with an inclined shape. DC gear motors have been mounted on the front facing side of each side panel. 4 bolts have been bolted on side panel on which the chain will be wrapped.

6.2.2. Motors

2 DC gear motors have been mounted on the vehicle, one on each side. The two motors are independent of each other. The motors are responsible for the motion UGV. The motors operate through pulses or continuous input provided to it via a microcontroller.

6.2.3. Microcontroller

ATMEGA-16 40 pin package is used as main controlling brain of UGV. Selection of this specific micro controller is made with a lot of study and practical experience. Aqua needs total of 14 I/O pins for controlling of its components. The operating frequency chosen of ATMEGA 16 is 8-Mhrz, and ATMEGA 16 resides on generic board inside UGV.

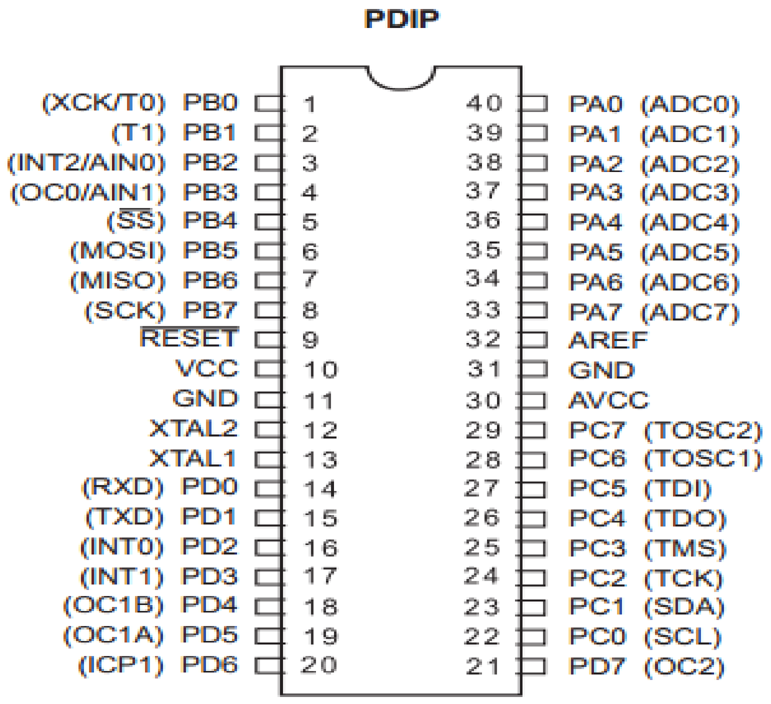


Fig 6.1. ATmega16 Pin Configuration

- Port B

Port B is used for debugging to display the LED’s on the MCB. The LED shows which phototransistor is on white or on blue. This helps us debugging the code.

- Port A

Port A is used to read the values of the phototransistors pa0-pa3 is used buy 4 sensors

- Port D:

Port D is used to generate the signal for motors this signal is amplified by the amplifier circuit which is discussed later.

6.2.4. Sensors

4 Phototransistors are used in UGV to sense the path under. This phototransistor works when LED’s light is reflected from the path under UGV. The voltage it returns is used by the microcontroller (ATmega16) to plan the movement of the UGV. L14G3 are used for line tracking. These phototransistors are mounted at very narrow angle to get more precise data for decision making of the robot.

These sensors sense the light in there external environment. They are made of photo diodes and photo transistors. Light dependent resistor can also use for light sensing. Their resistance decreases as light falls on it.

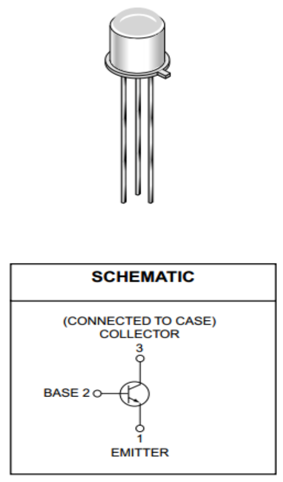


Fig 6.2. Phototransistor

6.2.5. Amplifier Circuit

The amplifier circuit is used in UGV so that the signal generated by microcontroller is of 5v and the H-bridge (motor controller) is activated (switched) at 12 volts. So to convert from 5v to 12v we used amplifier circuit.

6.2.6. Code for the Microcontroller

- Header Files:

```
#include <avr\io.h> // Most basic include files

#include <avr\interrupt.h> // Add the necessary ones

#include <util\delay.h> // here
```

- Threshold Values:

5v = 1024

So we scale the voltage return by the sensor for processing

1v = 204.5

4.86v = 993.87 round off 994

```
/***********speed of motor variables****************/

#define thre_r1 994 // threshold value right 1 4.86v

#define thre_r2 949 // threshold value right 2 4.50v

#define thre_l2 960 // threshold value left 2 4.56v

#define thre_l1 477 // threshold value left 1 2.07v

//********************Global variables***************

unsigned int right,left,right_mid,left_mid,speed_right,speed_left;
```

```
//**********************Functions*******************

void ini_adc (void)

{

ADMUX= (1<<REFS0);

// enable ADC | prescalar 2

ADCSRA= (1<<ADEN)| (1<<ADPS0);

}


unsigned int read_adc (unsigned char adc_input)

{

ADMUX=adc_input | (1<<REFS0);

// Delay for stabilizing the v

_delay_us(10);

// Start the AD conversion

ADCSRA|= (1<<REFS0);

// conversion to complete

while ((ADCSRA & 0x10) ==0);

//Disabling flag for new conversion

ADCSRA|= (1<<ADIF);

//retrunnng result
```

```
return ADCW;

}


int main (void)

{

ini_adc (); // initializing adc

DDRA=0x00; // initializing port A for input in ADC mode

DDRD=0xff; //Initializing data directional register for PortD as out put

PORTD=0xff; // initializing portD for output

DDRB=0xff; //Initializing data directional register for PortB as out put
```

- Read the sensor values:

```
while (1) // forever loop

{

    right= read_adc (0); //adc pa0 reading first sensor right 1

    right_mid= read_adc (1); //adc pa1 reading first sensor right 2

    left_mid= read_adc (2); //adc pa2 reading first sensor left 2

    left= read_adc (3); //adc pa3 reading first sensor left 1
```

- Five Conditions:
    - ➢ Ideal condition (Blue White White Blue)

```
if ((right>thre_r1) && (right_mid<thre_r2) && (left_mid<thre_l2) &&
(left>thre_l1))

{

PORTB=0b00001001; // debugging port

PORTD=0b11000000; // both will go in straight direction

}
```

- (White White Blue Blue) UGV moving Right

```
else if (((right>thre_r1) && (right_mid >t hre_r2) && (left_mid < thre_l2)
&& (left<thre_l1)) || ((right >t hre_r1) && (right_mid > thre_r2) &&
(left_mid < thre_l2) && (left>thre_l1)))

{

PORTB=0b00000011;

PORTD=0b10000000; // left motor will hold down and right motor will be
running

}
```

- (White Blue Blue Blue) UGV moving Right

```
else if ((right t >thre_r1) && (right_mid > thre_r2) && (left_mid > thre_l2)
&& (left<thre_l1))

{

PORTD=0b10000000; // left motor will hold down and right motor will be
running

PORTB=0b00000111;

}
```

- (Blue Blue White White) UGV moving Left

```
else if(( (right t< thre_r1) && (right_mid < thre_r2 )&& (left_mid> thre_l2)&&(left >thre_l1))|| ((right>thre_r1) &&(right_mid<thre_r2) &&(left_mid> thre_l2)&& (left>thre_l1)))

{

PORTD=0b01000000; // right motor will hold down and left motor will be running

PORTB=0b00001100;

}
```

- (White Blue Blue Blue) UGV moving Left

```
else if ((right<thre_r1) && (right_mid > thre_r2) && (left_mid > thre_l2) && (left> thre_l1))

{

PORTD=0b01000000; // right motor will hold down and left motor will be running

PORTB=0b00001110;

}

}

return 0;

}
```

Chapter 7: Testing and Result Analysis

7.1. UNIT TESTING

For unit testing the main functions of the source code were given a particular input and then outputs were matched against the expected outputs.

The system has following Unit Test Cases:

7.1.1. TestCase1

- _sift_features()
  - Purpose: Find SIFT features in an image using user-specified parameter values.
  - Pre-Requisite: The system is running.
  - Test Data: target image, image features, sift intervals, control threshold, descriptor width, descriptor histogram
  - Steps:
    - Run the program
    - Load the image
    - Run _sift_features() to calculate features
  - Expected output: no. of features (integer value)
  - Status: Successful.

7.1.2. TestCase 2:

- kdtree_build( )
  - Purpose: A function to build a k-d tree database from keypoints in an array.

- Pre-Requisite: The system is running and the features have been extracted.
- Test Data: object features, no of features.
- Steps:
    - Run the program
    - Load the image
    - Run _sift_features() to calculate features
    - Run kdtree_build() to make feature tree
- Expected output: feature tree
- Status: Successful.

7.1.3. TestCase 3

- find_matches( )
    - Purpose: to find matches between object features and frame features
    - Pre-Requisite: The system is running and object and frame features have been calculated.
    - Test Data: frame features, no. of frame features, object feature tree, frame, object
    - Steps:
        - Run the program
        - Load the image
        - Run _sift_features() to calculate features
        - Run kdtree_build() to make feature tree
        - Extract frames

        - Extract frame features
        - Build frame features tree
        - Expected Output: image pointer containing matches
    - Status: Successful.

7.1.4. TestCase 4

- ransac_xform( );
    - Purpose: Calculates a best-fit image transform from image feature correspondences using RANSAC
    - Pre-Requisite: The system is running, features have been extracted and matches have been found.
    - Test Data object features, matches
    - Steps:
        - Run the program
        - Load the image
        - Run _sift_features() to calculate features
        - Run kdtree_build() to make feature tree
        - Extract frames
        - Extract frame features
        - Build frame features tree
        - Find matches between two using this function.
        - Compare object features and matches matrix to check where the object is placed in the image and draw a boundary around it.

- ➢ Expected Output: matrix containing points enclosing the object
- ➢ Status: Successful.

7.1.5. TestCase 5

- draw_xform( )
    - ➢ Purpose: draw a boundary at the points in the xform matrix to surround the object and create a boundary around them.
    - ➢ Pre-Requisite: The system is running.
    - ➢ Test Data: frame, target object, matrix of matching points
    - ➢ Steps:
        - Run the program
        - Load the image
        - Run _sift_features() to calculate features
        - Run kdtree_build() to make feature tree
        - Extract frames
        - Extract frame features
        - Build frame features tree
        - Find matches between two using find_matches() function
        - Find the xform matrix by ransac_xform() function.
    - ➢ Expected Output: a box enclosing the object
    - ➢ Status: Successful.

7.1.6. TestCase 6

- kdtree_bbf_knn()
    - Purpose: To Find an image feature's approximate k nearest neighbors in a kd tree using Best Bin First search.
    - Pre-Requisite: system is running and object and frame features have been calculated
    - Test Data: root node of feature tree, features, tree and feature maximum matches
    - Steps:
        - Run the program
        - Load the image
        - Run _sift_features() to calculate features
        - Run kdtree_build() to make feature tree
        - Extract frames
        - Extract frame features
        - Build frame features tree
    - Expected Output: no. of matches (integer)
    - Status: Successful.

7.1.7. TestCase 7

- UGV run test
    - Purpose: To check whether the UGV moves in a straight line

- Pre-Requisite: UGV assembled, battery charged,
- Test Data: UGV
- Steps:
    - Connect wires of motor to Microcontroller
    - Connect battery wires to microcontroller
    - Trigger switch on MCB
- Expected Output: Vehicle moves in a straight line
- Status: Successful.

7.1.8. TestCase 8

- UGV run test
    - Purpose: To check whether the UGV follows path
    - Pre-Requisite: UGV assembled, battery charged ,defined path
    - Test Data: UGV, defined path
    - Steps:
        - Connect wires of motor to Microcontroller
        - Connect battery wires to microcontroller
        - Connect sensor wires to the Microcontroller
        - Set up path using Duck tap
        - Trigger button on MCB
    - Expected Output: Vehicle moves in a given path
    - Status: Successful.

7.2. SCENARIO BASED TESTING

Object recognition and feature matching is affected by a number of factors. Most important of them are angle of the object and lighting and the number of extracted features of the object. SIFT algorithm is used because it recognizes the objects by matching features even if the angle is changed or part of the object is visible. It matches the features even if the angle of the object is changed or inclination is different.

However, the issue of lighting is one which cannot be resolved. If there is too much light then the camera will not pick up the actual features of the object. Only a reflection of the light source will come in the video stream and the no details of the object will appear. This scenario can only be resolved if the system works in an environment where there is no excess of light.

Given on next pages are some of the scenarios and their outputs:

Case 1: When the object appears in front of camera and the light source is also normal.

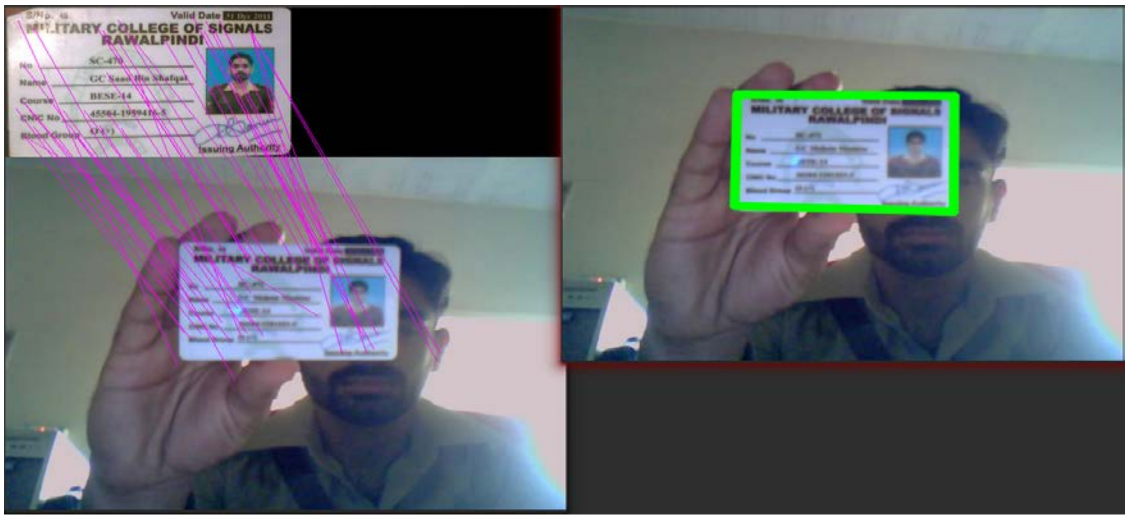


Fig 7.1. Case 1 (Best Case Scenrio)

The lines show the features being matched between the frame and the object image. The window on the right shows the object has been detected perfectly and a boundary has been drawn around it.

Case 2: When the object appears in front of the camera in an angle. Light conditions are normal.

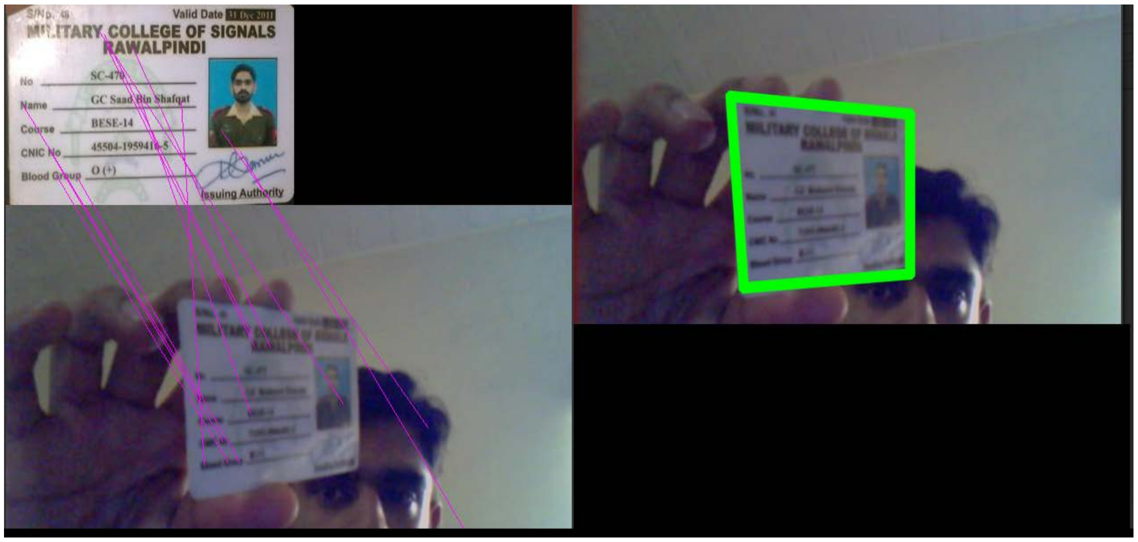


Fig 7.2. Case 2 (object with an Angle)

The image shows the object in front of the camera in an angle as compared to the image of the target object. Due to the efficiency of sift, the features were still compared and matched. Enough matches were found when the objects were tested with angles up to 60 degrees as compared to the image and the objects were matched

Case 3: Lighting conditions are unsuitable.

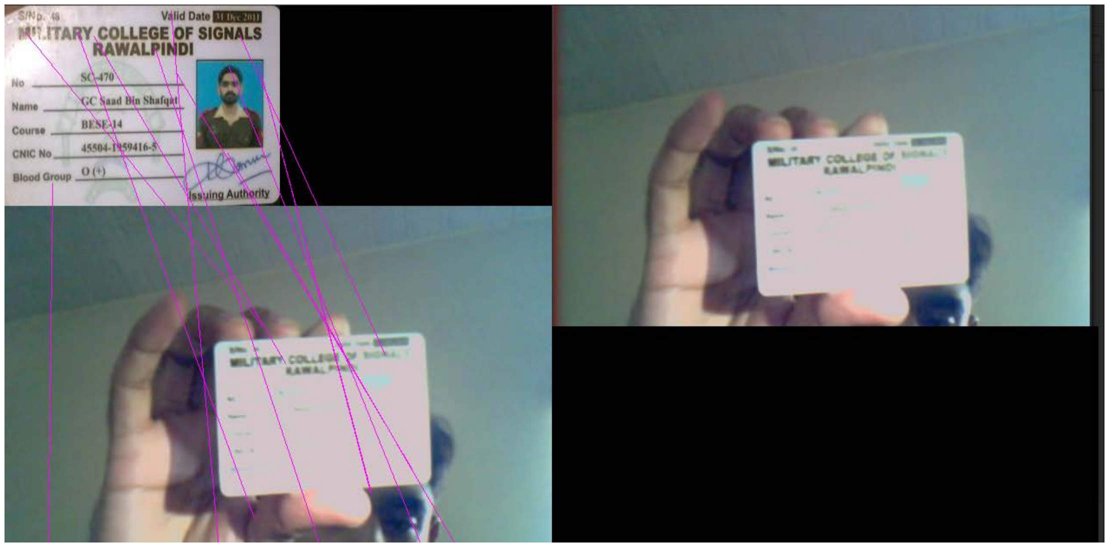
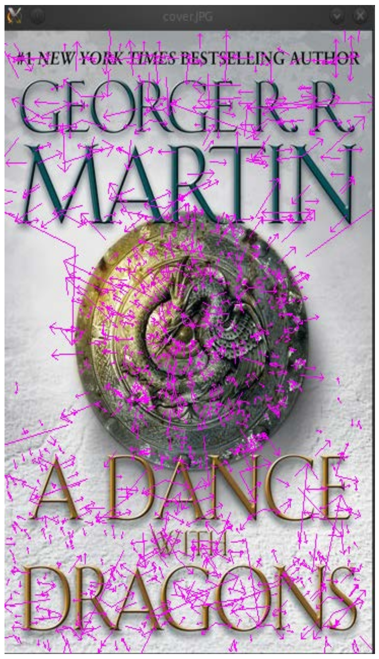
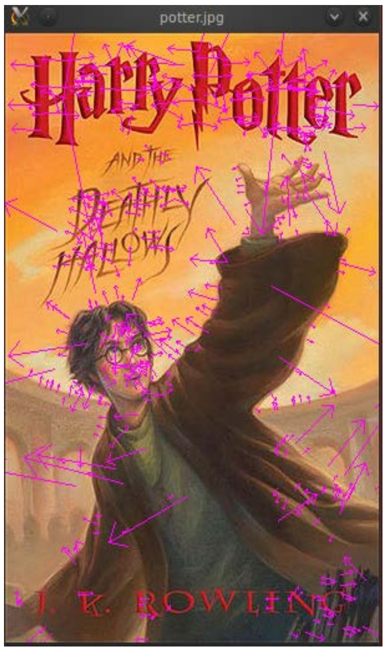


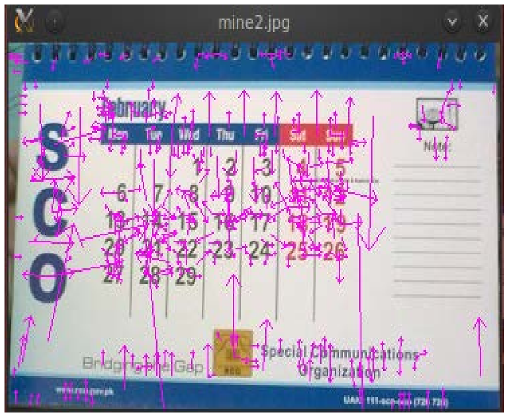
Fig 7.3. Case 3 (Extra Light)

As shown in the above scenario, there was excessive light and the features of the object were not clearly visible to the camera, therefore, matching could not be done and the object was not recognized.

Another important factor was the number of features extracted by sift for matching. It was observed that if the features extracted were less than 150, there was difficulty while matching them with the object in the video. Therefore, a pre requisite of this system is that the image of the object must contain more than 200 distinct features so that they can be easily matched. If the features extracted are any less, the results will not be correct.







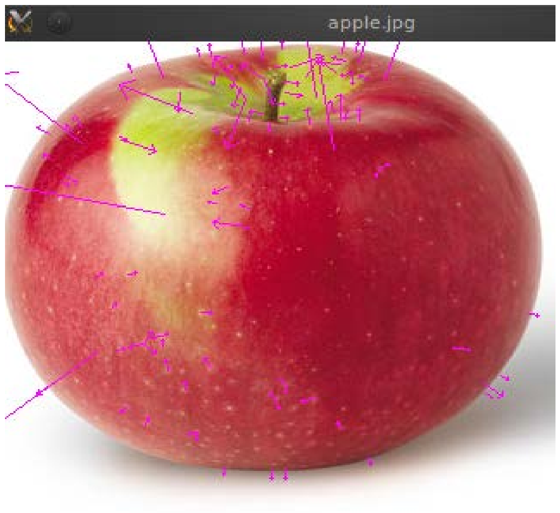
(a) Dance of Dragons book (1095 features)

(b) Harry Potter book (479 features)

(c) A Table Calendar (495 features)

Fig 7.4. Features Extraction of Objects with Enough Features

Features of these objects are shown with pointers. These features were later used in matching and the objects were matched successfully.



Below is an image of an apple

Fig 7.5. Features Extraction of Apple

In case of this apple, the features extracted were only 95 and when an apple was placed in front of the camera, it was not successfully detected.

Although the jitter and vibrations due to the movement of vehicle distorted the frames, however, the success rate was more than 90 % i.e. the correct objects were recognized successfully in 83 attempts out of 90.

A few readings and results are given in Table.

- card is the id card used in recognition
- book1 is dance of dragons book
- calendar is the table calendar
- book2 is the harry potter book

These experiments were done using these 4 objects in different scenarios but the light was considered suitable. Below are the results collected when the objects were recognized from moving vehicles.

Sample no	Object	Angle of object (In degrees approx)	Classified as	Binary Result	% matched
1	card	0	card	true	83
2	card	15 (horizontal)	card	true	75
3	card	30 (horizontal)	card	true	71
4	card	60 (horizontal)	card	true	65
5	card	-15 (horizontal)	card	true	74
6	card	-30 (horizontal)	card	true	73
7	card	-60 (horizontal)	card	true	62
8	card	15 (vertical)	card	true	73
9	card	30 (vertical)	card	true	75
10	card	60 (vertical)	card	true	62
11	card	-15 (vertical)	card	true	76
12	card	-30 (vertical)	card	true	77
13	card	-60 (vertical)	card	true	61
14	book1	0	book1	true	85
15	book1	15 (horizontal)	book1	true	75

16	book1	30 (horizontal)	book1	true	71
17	book1	60 (horizontal)	none	false	22
18	book1	-15 (horizontal)	book1	true	78
19	book1	-30 (horizontal)	book1	true	73
20	book1	-60 (horizontal)	book1	true	62
21	book1	15 (vertical)	book1	true	82
22	book1	30 (vertical)	book1	true	75
23	book1	60 (vertical)	book1	true	62
24	book1	-15 (vertical)	book1	true	76
25	book1	-30 (vertical)	book1	true	77
26	book1	-60 (vertical)	book1	true	64
27	calendar	0	calendar	true	89
28	calendar	15 (horizontal)	calendar	true	72
29	calendar	30 (horizontal)	calendar	true	76
30	calendar	60 (horizontal)	calendar	true	67
31	calendar	-15 (horizontal)	calendar	true	73
32	calendar	-30 (horizontal)	calendar	true	74
33	calendar	-60 (horizontal)	none	false	12
34	calendar	15 (vertical)	calendar	true	77

35	calendar	30 (vertical)	calendar	true	72
36	calendar	60 (vertical)	calendar	true	62
37	calendar	-15 (vertical)	calendar	true	73
38	calendar	-30 (vertical)	calendar	true	78
39	calendar	-60 (vertical)	calendar	true	62
40	book2	0	book2	true	89
41	book2	15 (horizontal)	book2	true	71
42	book2	30 (horizontal)	book2	true	72
43	book2	60 (horizontal)	book2	true	62
44	book2	-15 (horizontal)	book2	true	78
45	book2	-30 (horizontal)	book2	true	73
46	book2	-60 (horizontal)	book2	true	62
47	book2	15 (vertical)	book2	true	84
48	book2	30 (vertical)	book2	true	72
49	book2	60 (vertical)	book2	true	67
50	book1	-15 (vertical)	none	false	30
51	book1	-30 (vertical)	book1	true	75
52	book1	-60 (vertical)	book1	true	64

Chapter 8: Conclusion and Future Work

8.1. CONCLUSION

Target recognition is a very important part of software engineering. The applications of target recognition are numerous. It is a generalized field that can be implemented for face recognition, target marking and many other various applications.

Security and surveillance are a very important functionality of such systems. Along with the software, the assembled hardware also presents the concept of mobility to image processing. Stationary surveillance though important is not always the solution for all security issues. Hence having a moving vehicle is very important.

Because of the amount of data required to develop and test target recognition systems for robustness, and the cost of mounting realistic data collection efforts, the development of accurate synthetic data-generation tools is also being addressed.

8.2. FUTURE WORK

The future work in this project can be done in the following ways:

8.2.1. Using Embedded Systems

As for now a laptop is being placed on the UGV for image processing. A further enhancement in this project can be implementing the image processing algorithm on a dedicated processor that is embedded on the UGV. In this way the dependency of placing a laptop will be reduced as placing a laptop in external environment is not feasible. Moreover a dedicated embedded processor would also increase the efficiency of the system.

8.2.2. Artificial Intelligence for Path Search

As for now the UGV is working on a particular path that for now has been defined as a contrast of blue and white. The future work in it can be using Artificial Intelligence search algorithm such as A* or Best First Search. In such a way the vehicle will become entirely autonomous in finding its own path. And hence can be used in entirely unknown environments.

8.2.3. Using GPU

Utilizing faster systems with GPU’s would make the algorithm more power and efficient, running multiple instances would be more plausible.

Chapter 9: User Guide

9.1. UGV

To operate the UGV properly the following steps should be fulfilled. Otherwise the UGV will not work correctly or it will damage its electrical parts.

9.1.1. Microcontroller Connections

On Microcontroller board there are pins of the microcontroller which are used for the input and outputs. We used the Port D for motor signal and Port B is for reading the sensor values. The wires which are to connect power amp (for converting 5v to 12v for switching the rely board of the motor driver) and microcontroller is connected at port D PD05 and PD06.

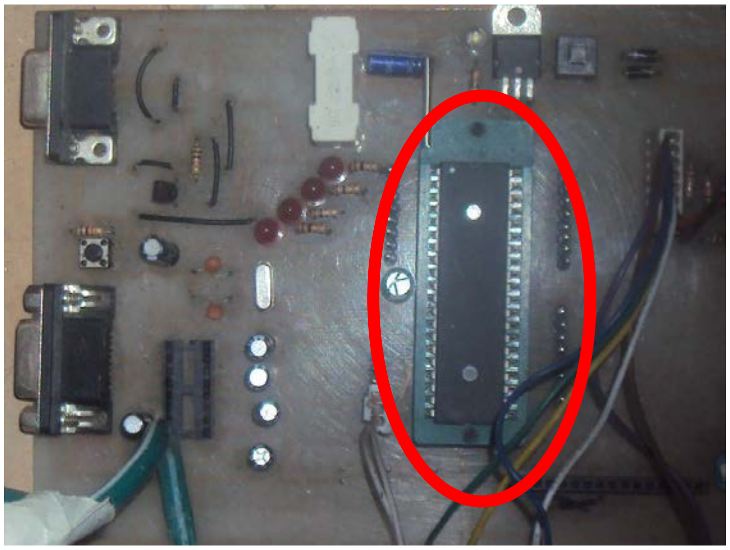
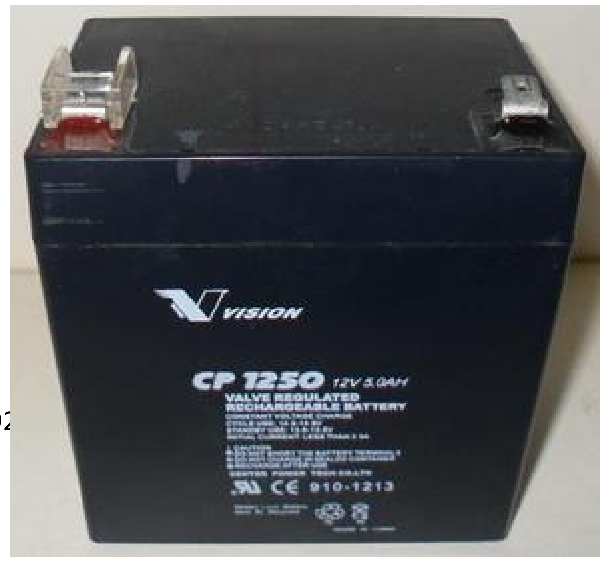


Fig 9.1. Microcontroller on Circuit Board

The sensors are connected at Port A PA01, PA02, PA03, PA04 because there are four sensors.

9.1.2. Battery connections

The battery used in UGV is a 12volts (5mA) which is enough to give power



to two DC motors and sensors (phototransistors) and the MCB (Microcontroller Board). The battery should be attached properly so that the UGV will work correctly.

Fig 9.2. Battery

The Green Wire from rely board (motor driver) is connected to positive terminal and the Blue wire is connected with the negative terminal.

Two wires from the microcontroller are connected with the battery, red wire is connected to positive and black wire is connected to negative terminal.

9.1.3. Press the ON button on the MCB

When all the above steps are done now the UGV is ready to move. There is a button on the microcontroller board with will pressed so that the UGV is activated and it starts moving.

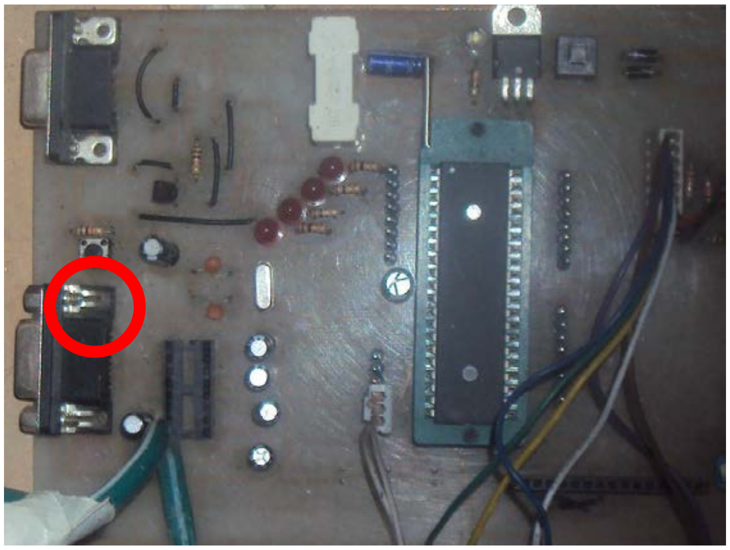


Fig 9.3. Start Switch

9.2. IMAGE RECOGNITION

- Install OpenCV 2.2 or later on your linux system. Instructions can be found at http://opencv.willowgarage.com/wiki/InstallGuide_Linux.
- Compile the header and library files given in the CD in linux (preferably Ubuntu).

- After the files have been compiled, there will be a library file in lib folder and header files in include folder
- For ease, do the following steps
  - Copy all the header files in include folder to /usr/local/include/opencv
  - Copy the library files in lib folder to /usr/local/lib
- After doing the above steps, come to the folder where you main source code is placed
- Edit the main source code (as given in the CD) and enter the name and path of your target object in 'target' field.
- Compile the main program (source code given in CD) using the following command **gcc -I/usr/local/include/opencv -L/usr/local/lib -o sift -lcv -lcxcore -lhighgui –lfeat**
  - -I gives the path of the folder from where the header files will be included
  - -L gives path of the folder from where library files will be included
- Run the resulting by typing **'./sift'**.
- Two separate windows will open in front of you on the screen.
  - One will show the target object and feature matches
  - Other will show if the object has been recognized or not
- To start target recognition, press 'space bar'.
- The second window will mark the object if it will match with the target image.

REFERENCES

- Distinctive Image Features from Scale-Invariant Key points by David Lowe, Accepted January 22, 2004, International Journal of Computer Vision 60(2), 91-110,2004
- Automatic Target Recognition by Matching Oriented Edge PixelsClark F. Olson and Daniel P. Huttenlocher

  IEEE Transactions on Image Processing, Vol. 6, no. 1, January 1997
- Development of a vision-based ground target detection and tracking system for a small unmanned helicopter LIN Feng, LUM Kai-Yew, CHEN Ben M.t&LEE Tong H. 2009 Science is China Press
- Vision-Based Target Recognition And Autonomous Flightsthrough Obstacle Arches With A Small Uav

  Florian-M. Adolf, Franz Andert, Lukas Goormann, andJorg S. Dittrich_German Aerospace Center (DLR), Institute of Flight Systems D-38108 Braunschweig, Germany 2006
- A Simple Local Path Planning Algorithm for Autonomous Mobile Robots

  Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohamad Z. Volume 5, 2011
- Software Architecture for Computer Vision:

  Beyond Pipes and Filters Alexandre R.J. Francois Institute for Robotics and Intelligent Systems University of Southern California afrancoi@usc.edu July 2003
- An Introduction to Software Architecture

  David Garlan and Mary Shaw

  January 1994 School of Computer Science, Carnegie Mellon University

- http://www.codingthearchitecture.com/2008/03/18/software_architecture_document_guidelines.html
- https://www.i2b2.org/software/projects/datarepo/CRC_Architecture_10.pdf
- http://www.cmcrossroads.com/bradapp/docs/sdd.html
- http://www.haskell.org/haskellwiki/Functional_programming
- http://opencv.willowgarage.com/documentation/cpp/index.html
- http://blogs.oregonstate.edu/hess/code/sift/
- http://www.atmel.com/devices/ATMEGA16.aspx
- http://www.atmel.com/dyn/products/product_card.asp?PN=ATmega16
- www.pxisa.org/Spec/PXIHWSPEC21.pdf
- http://products.ihs.com/cis/Doc.aspx?AuthCode=&DocNum=267772
- http://www.cse.iitk.ac.in/users/vision/dipakmj/code/cslam/documentation/html/class_features.html