TEAM DCE

(team No 26)

presents

for **TechX Challenge**

Team Composition

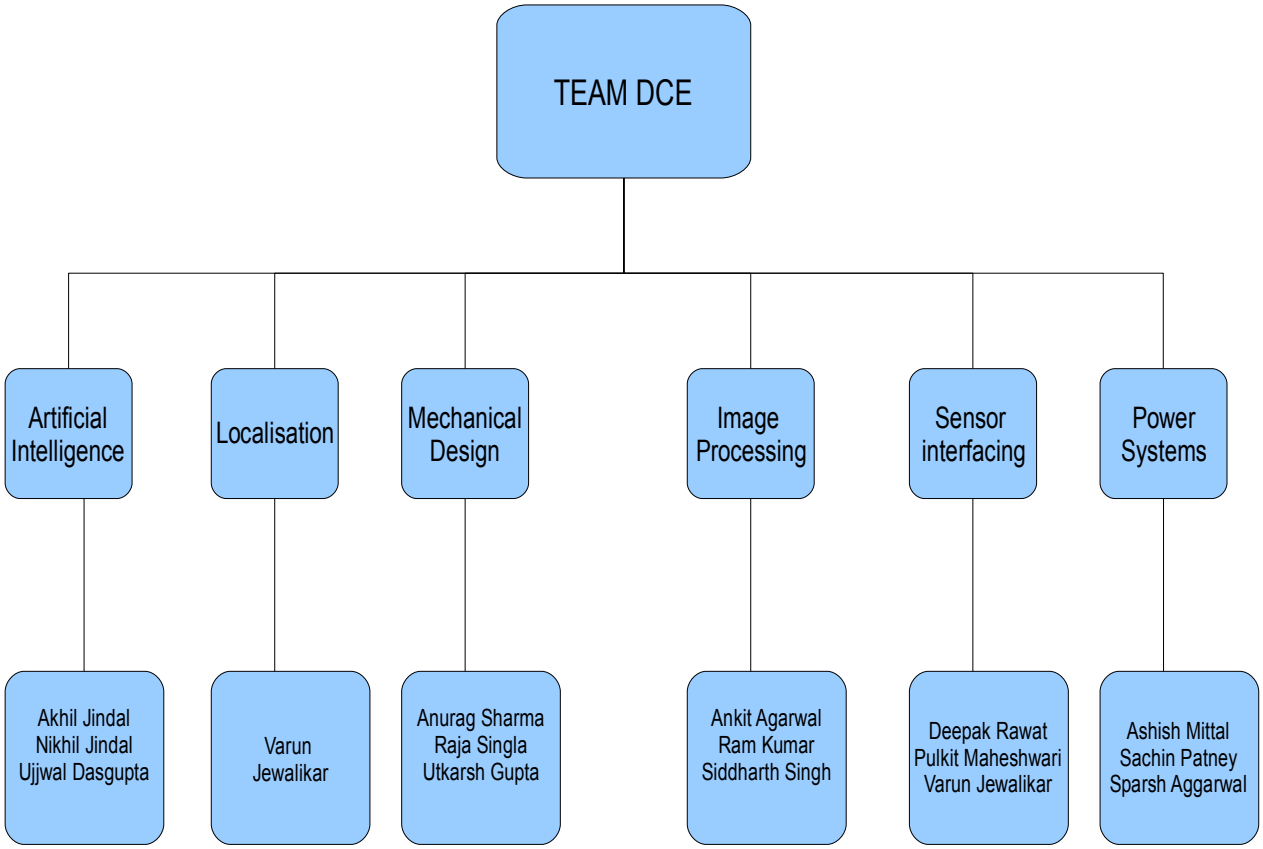
Faculty Advisors

1)Mrs. Rajni Jindal
Lecturer
Department of Computer Technology
Delhi College of Engineering

2)Mr.N.S Raghav
Lecturer
Department of Information Technology
Delhi College of Engineering

Team Manager

Sachin Patney
IInd year
Computer Engineering Department
Nanyang Technology of University


TEAM DCE
Artificial Intelligence
Localisation
Mechanical Design
Image Processing
Sensor interfacing
Power Systems
Akhil Jindal
Nikhil Jindal
Ujjwal Dasgupta
Varun Jewalikar
Anurag Sharma
Raja Singla
Utkarsh Gupta
Ankit Agarwal
Ram Kumar
Siddharth Singh
Deepak Rawat
Pulkit Maheshwari
Varun Jewalikar
Ashish Mittal
Sachin Patney
Sparsh Aggarwal

Index

1. Introduction:

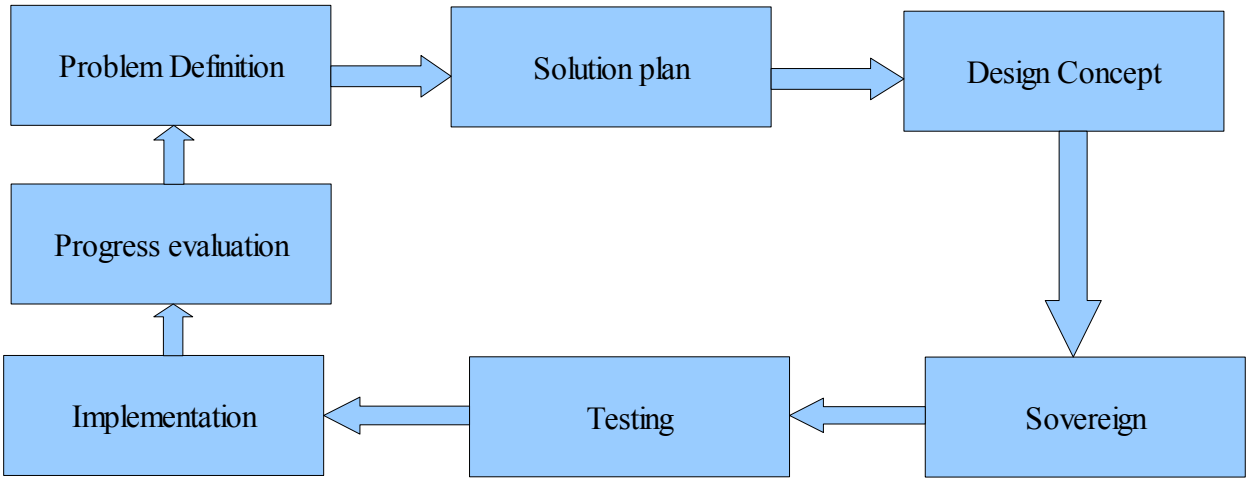
TEAM DCE from Delhi College of Engineering(DCE) has made a fully Autonomous Defence Robot(ADR) - SOVEREIGN. The robot has been specifically designed for the TechX Challenge organised by Defence Science And Technology Agency(DSTA), Singapore Government and many other potential applications in the offering.

Sovereign an intelligent autonomous robot is capable of autonomously navigating to a specified building using GPS way points, enter the building, avoid obstacles, operate elevator, climb stairs, engage and finish off targets and finally retracing its path back.

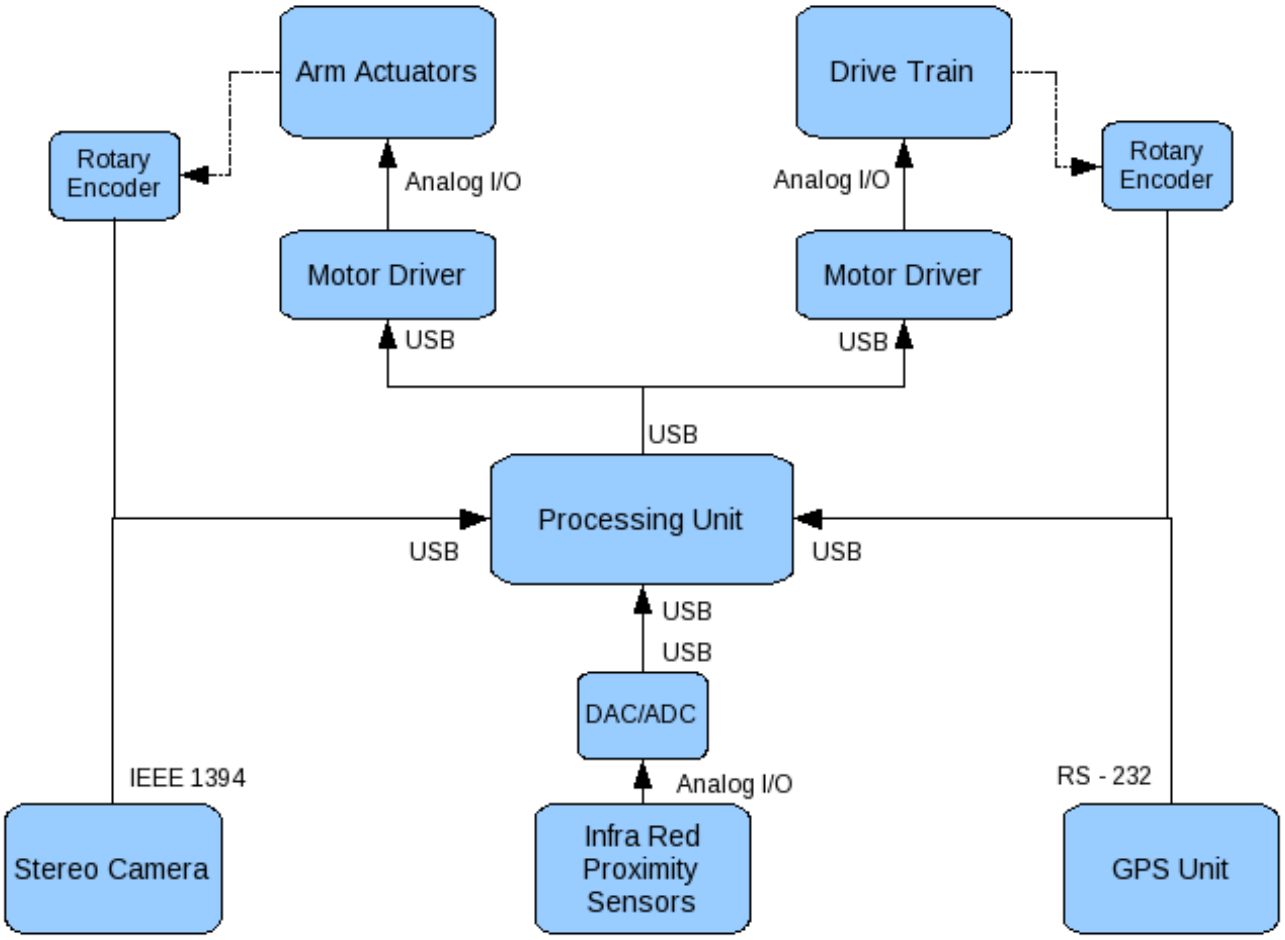
The team comprises of 17 members which have broadly been divided into sub-groups of Artificial Intelligence, Embedded systems, Image Processing, Power and Control System, and Mechanical Systems. Each member of the team is a specialist in his own field. The development of the robot, provided the team with a challenging task which tested their knowledge to the core.

The purpose of this report is to describe the conceptual design of the robot and its components and highlight the unique innovative aspects of the design and design process.

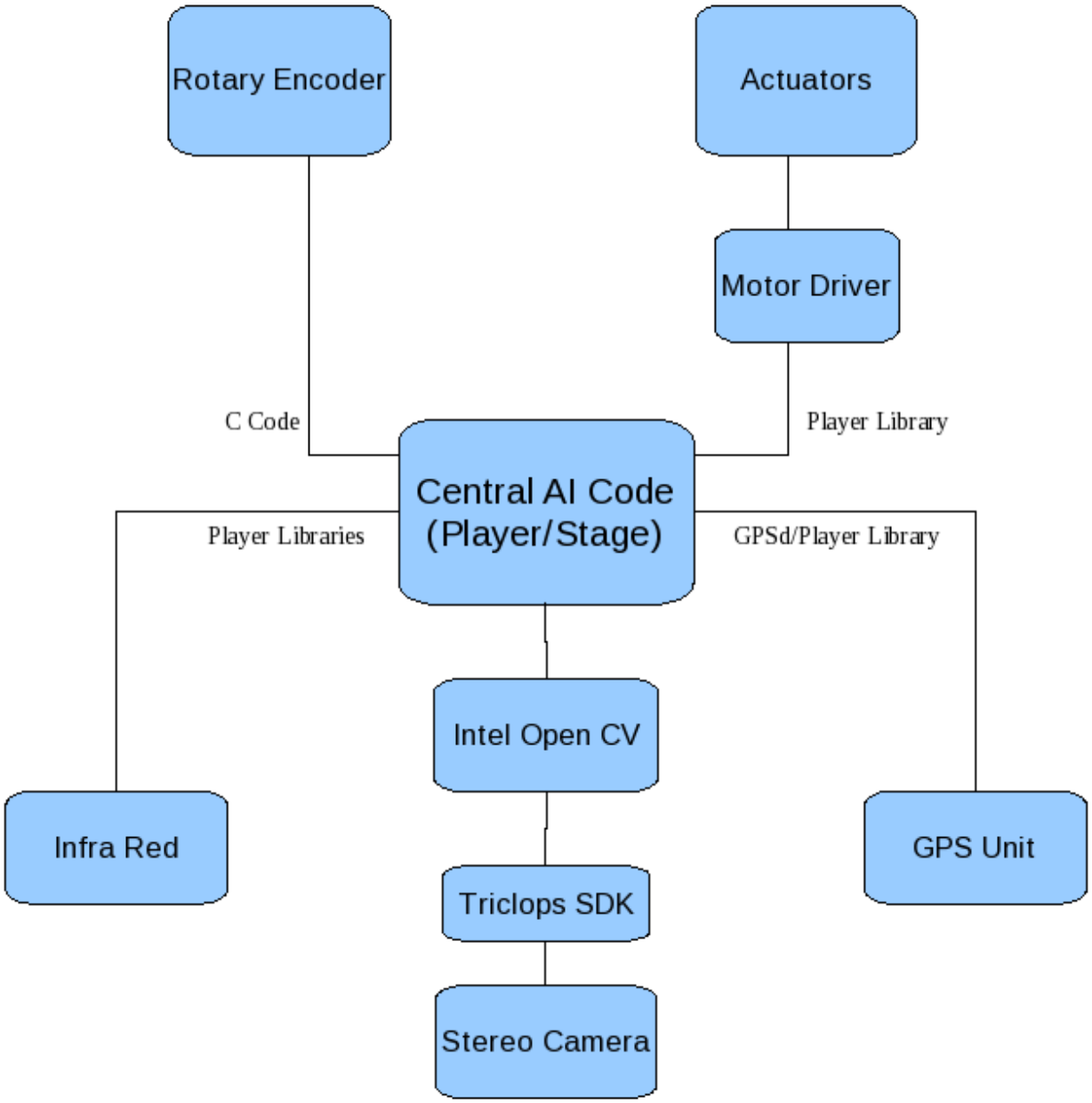
By adhering to the design process, a reliable, robust and efficient robot has been built. The problem statement was divided into many sub-problems and accordingly different modules were prepared which were integrated to form the complete robot.



2.System Configuration



3.Software Configuration



Player --- robot device interface

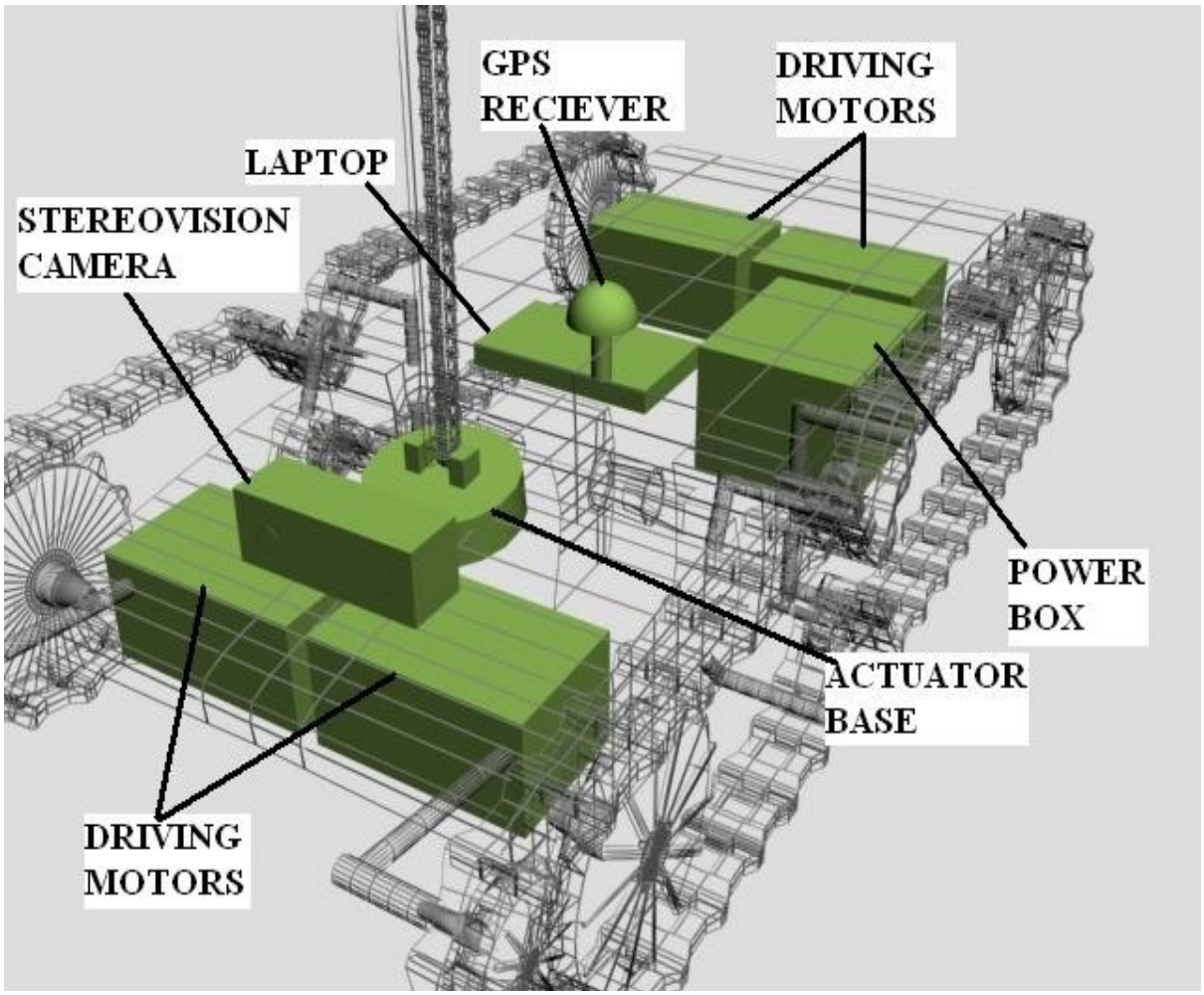
Player is a network server for robot control. Running on our robot, Player provides a clean and simple interface to the robot's sensors and actuators over the IP network. Our client program talks to Player over a TCP socket, reading data from sensors, writing commands to actuators, and configuring devices on the fly. Player is language and platform independent.
Player allows multiple devices to present the same interface. On-the-fly device requests allow our client to gain access to different sensors and actuators as needed for the task at hand. The behavior of the server itself can also be configured on the fly.

Player supports a wide variety of components/devices whose drivers are written in Player/stage

a)GPS
b)Proximity sensors
c)Firewire Camera
d)Motor Drivers

Stage --- simulation interface
Simulates a population of mobile robots, sensors and objects in a two-dimensional bit-mapped environment. It is used a a Player plug-in module, providing populations of virtual devices for Player.
It has been used extensively for in system simulation of various types of code.

4.Hardware Configuration Layout



5.Communication and Computing Resources

a)Network Architecture

The control program which will be running on the onboard laptop will be written using the Player/Stage library. This is an opensource framework which allows an abstraction of each sensor/actuator used in the form of an interface or a driver. The AI code which communicates with the hardware need not know the specifications of each hardware, but can communicate using standard protocol, which is interpreted by the driver. Each sensor module communicates independently with the central control program, which uses the data collected from each to build a representation of the world in its memory.

b)Computers

The processing unit for Sovereign shall consist of a single laptop computer (Asus EEE PC 4G, x86 platform). The choice was made because of its small size and weight, and because it's high battery life prevents addition power supply considerations. Detailed system specifications are provided in the next section.

The operating system shall be linux based, and as no graphical interface or additional services shall be running, considerable memory savings are made.

The software will be coded in C, using additional libraries (playerstage, opencv) for different tasks. This will ensure efficiency of the code. The resource-intensive image processing procedures shall be activated only when required.

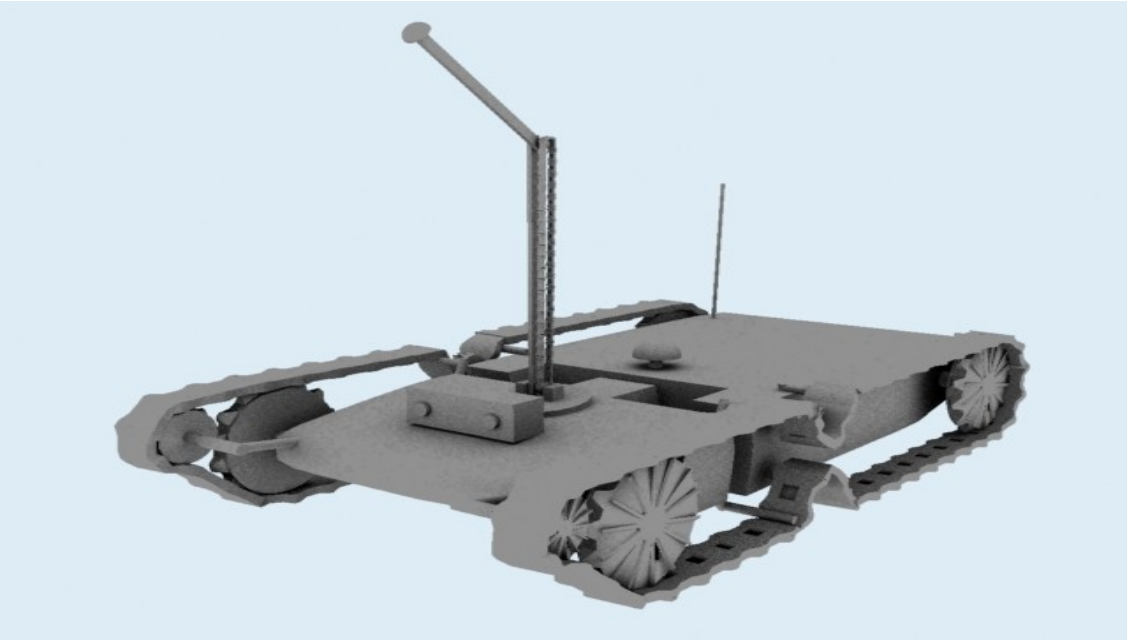
c) Detailed Specifications

Specifications of the ASUS EEE PC

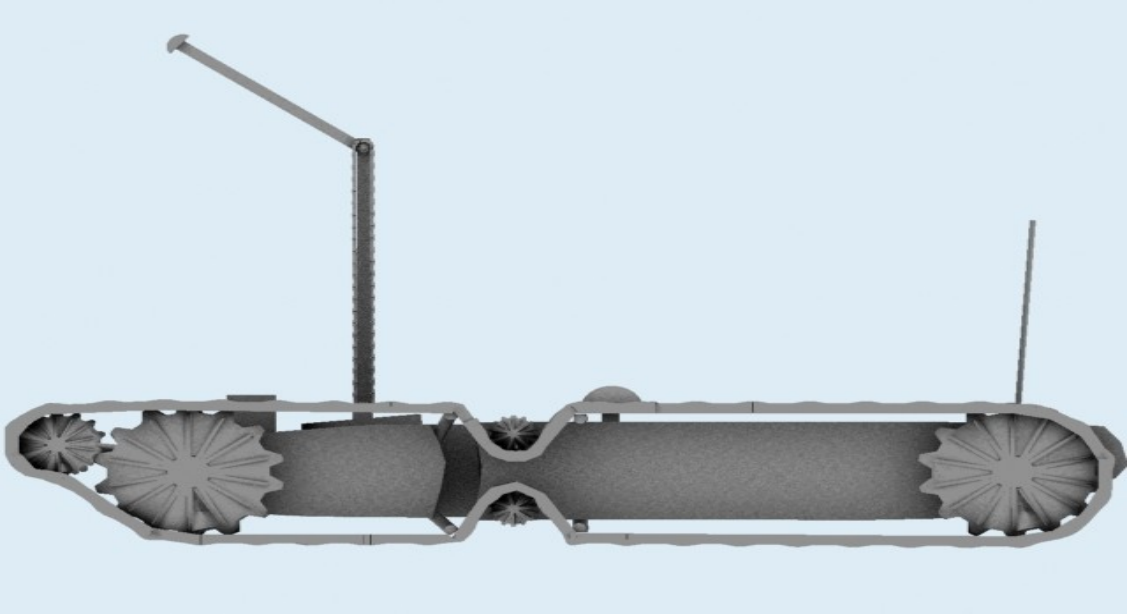
Processor : Intel Celeron-M ULV 353, 900 MHz power and 512 kb L2 Cache
Chipset : Intel 910GML series
RAM : 512 MB DDR2 RAM
Media : 4GB Solid State Disk (SSD)
External Storage Options : MMC/SD card reader

Graphics : Integrated Intel GMA 900 graphics processor
Display : 7 inch, 800 X 480 TFT LCD
Power : 4 cell, Lithium Ion, 5200 mAh battery
Dimensions : 225 X 165 X 21~35 mm
Weight : 920 grams
Connectivity : 10/100 Ethernet ad Wi-Fi 802.11b/g support
I/O Ports : 3 USB 2.0, VGA, RJ45, 0.3 megapixel video camera

6. Mechanical Design



CAD model



Side view

a)Key Locomotion Features

i)Design Approach

As the robot had to overcome various hurdles like staircase, curbs and slopes, our team spent some hard time thinking about an efficient and reliable design.
After performing a series of tests on various small modular designs, we decided to design the robot in two parts- the front part or the child part and the rear part or the parent part. The two part were hinged together, allowing relative movement between the two in the vertical plane. This particular design was found to be very successful while climbing the stairs. As the front part approached the stairs and got lifted off the ground, the rear part provided it the necessary force as it had better traction with the ground. Once the robot is completely on the stairs, it is supported by the tread belts that carry it further up.

ii)Chassis and Frame

The chassis for Sovereign is designed to have a lower center of gravity and a strong frame. Size, position and weight of all components were taken into consideration and were designed with this in mind. The frame is constructed of square Aluminium tubing that has been welded together and polished. The body has been constructed using aluminium sheets bolted to the frame. Aluminium was chosen as it is lightweight and a thick sheet provides appreciable strength, stiffness and rigidity. Most of the parts are detachable and the inner components can be easily accessed.

b)Drive

The drive train consists of two custom fabricated tread belts that power the Sovereign over any terrain, slope, staircase and curbs. The belt is supported over two pulleys - one each in the front and rear part. Modified belt tensioners have been employed to meet our requirement of relative movement between the two parts.
Sovereign employs Differential steering mechanism, as it is the simplest and the best steering method for pinpoint turning.
Four DPM h500 stepper motors were selected to power these tread belts, two motors

driving each belt. These motors have a stall torque of 9 Nm and rotate at 1000 RPM. They are connected to gearboxes having a gear ratio of approximately 10:1 so as to bring the maximum speed of Sovereign down to 1.31 m/s i.e. within the speed limit of 2m/s.
Timing belts are employed to transfer the power from the motor to the pulley axle. The use of timing belts brings shock resistance to the body.

c)Physical Characteristics

The load of Sovereign has been divided into the front and the rear part. The front part holds two driving motors, the elevator activating mechanism and the camera. The rear part holds the laptop, the GPS system, the power module and two driving motors.
The dimensions of the components are as follows:

Component	Weight (kg.)	Dimensions (mm)
camera	0.342	157x36x47.4
GPS	0.200	
Laptop	0.945	225x165x35
Elevator operating mechanism	2.000	1650x160x160
Battery	2.200	
Driving Motors	15.200	86mm sq. frame;250 mm long

The body designed using aluminium tubing and sheets, weighs 3.6 kg. The total weight of Sovereign is 25.23 kg. It is 800mm long, 500mm wide and 300mm high.
This structure has been designed to support and evenly distribute a load of up to 25 kgs.
The center of gravity has been kept quite low so that it does not topple while climbing stairs and overcoming other obstacles.

d) Specifications

1) Driving motor

86SH118-6004A

2) Driving motor driver

Specifications

	MODEL		86SH118-6004A			
1	Rated Voltage	V	2,7	-	-	-
2	Current/Phase	A	6	-	-	-
3	Resistance/Phase	Ω	0,45	-	-	-
4	Inductance/Phase	mH	5,1	-	-	-
5	Holding Torque	Nm	8,7	-	-	-
6	Rotor Inertia	g-cm²	2700	-	-	-
7	Weight	Kg	3,8	-	-	-
8	Number of leads	n°	4	-	-	-

AMS 4850

Maximum step frequency	200KHz on Microstepping
Logic level -0-	<2V
Logic level -1-	>3.5V
LED Outputs	Green - Power Red - Pulse
Current setting	5.0A rms, Customization possible
Supply Voltage	18-48Vdc

3) Actuator Driving Motor

1	Step angle		1,8°
2	Step angle accuracy (full step, no load)	%	±5%
3	Rated Voltage	V	2,3
4	Current/Phase	A	2,8
5	Resistance/Phase	Ω	0,83
6	Inductance/Phase	mH	2,2
7	Detent Torque	mNm	35
8	Holding Torque	Ncm	101
9	Rotor Inertia	g-cm²	275
10	Weight	Kg	0,65
11	Number of leads	n°.	4

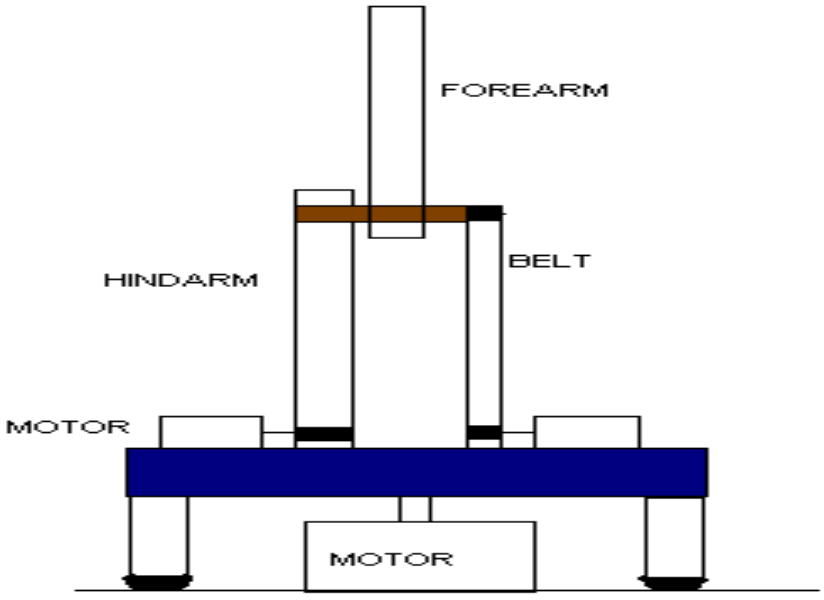
4) Actuator motor driver

AMS-243

Specific technical data

Power supply voltage	12V to 30Vdc, Nominal 24 Vdc
Phase current (peak value)	0.7A to 3A
Optocoupler inputs:	Clock. Direction. Enable.
Logic input voltage	5V to 12Vdc
Chopping frequency	18 kHz
Max. clock frequency	20 kHz in Half step mode with a minimum pulse of 10µs.
LED indicator	Pulse (green), Fault (red)
Protection	Short-circuit, between phases Thermal
Operating temperature	0 to 60°C
Termination	L type screw terminals in case of chassy mounting.
Mode selection	Auto and Manual selection through on board DIP switch

d)Elevator Activating Mechanism



I) Mechanism Design

We are employing a robotic arm to operate the elevator. The robotic arm consists of two parts- forearm and hindarm. Both these parts are placed on a platform which is kept on the robot's base. The platform is capable of rotating through full 360^0, making the robotic arm capable of moving in all directions. The hindarm is placed on the platform and move through 180^0. The forearm is attached to the shoulder having free rotation between the two and hence the arm can move through 360^0.

For the movement of the parts three DPM 57SH51-4A stepper motors are used. One motor will be moving the platform, one will be employed for hindarm and the third one for the forearm.

Motor for driving the platform is placed below it on the robot base. The base platform is supported by two Castor wheels. The motor for driving the hind arm pulley is kept on the base platform. The motor for driving the forearm pulley is also kept on the platform and connected to the rear end of forearm pulleys by means of a driving chain. This reduces the load on the parts of the robotic arm and all the load falls on the platform. This design enables us to use light weight material for the fabrication of robotic arm.

ii) Degrees Of Freedom

The degrees of freedom, or DOF, is a very important term to understand. Each degree of freedom is a joint on the arm, a place where it can bend or rotate or translate. The number of degrees of freedom can be identified by the number of actuators on the robot arm.
There are only two motions a joint could make: translate and rotate. There are only three axes this could happen on: x, y, and z (out of plane).
As all the motions in our robotic arm are rotational, we can say that our robotic arm has three degrees of freedom.
But as the driving motion of the robot itself can increase the space coverage of the arm, its degrees of freedom will also be included in the robotic arm's degrees of freedom.
The DOF of vehicle moving in a plane is two, hence, the net degrees of freedom comes out to be five.

This design has been chosen because it provides us with maximum space coverage along with least amount of actuators
As a result the control system of the robotic arm becomes relatively simpler.

7.Sensor Interfacing

a. GPS

The GPS unit being used is the Garmin GPS 17HVS. A vital element for the navigation , the Garmin GPS 17 provides the autonomous vehicle a trusted and reliable input of positioning data. It can also serve as a tool to adjust the positioning data during the robot's run. It can also serve as our feedback system.

SPECIFICATIONS

GPS Performance

Receiver: WAAS-enabled, 12 parallel channel GPS receiver continuously tracks and uses up to 12 satellites to compute and update your position

Update Rate: 1 to 900 seconds between updates; programmable in 1-second increments

GPS accuracy:

Position: < 15 meters

DGPS (USCG) accuracy:

Position: 3-5 meters

DGPS (WAAS) accuracy:
Position: < 3 meters

This GPS unit features a 12-channel receiver that reports its precise position through the continuous tracking of satellites. It is capable of receiving WAAS/DGPS differential corrections, improving its horizontal precision to less than 3 meters.

The GPS module is implemented using the open-source software GPSd . Using this approach, we are able to use any NMEA compatible GPS unit on the robot. This is the pseudo-standard Linux interface for GPS. The GPS interfaces directly to the computer via an RS232 serial connection using a standard NMEA-0183 protocol.

b.BumbleBee2 Stereo Camera

Computer Vision system
Camera: Point Grey - BumbleBee2 Stereo Camera (BB2-03S2C-XX)

Technical Specifications:

Sony 1/3" Color CCD
Resolution- 640X480
FPS- 48
Focal Length- 3.8mm with 66° HFOV
Interface- 6-pin IEEE-1394a for camera control and video data transmission. 2 x 9-pin IEEE-1394b for camera control and video data transmit.
Voltage Requirements- 8-32V via IEEE-1394 interface or GPIO connector
Power Consumption- 2.5W at12V
Dimensions- 157 x 36 x 47.4mm
Mass- 342 grams

PointGrey's BumbleBee2 stereo vision camera is a widely used hardware for depth mapping and image processing.
The cameras ablity to instantaneously generate depth maps is highly beneficial for the robot.
Low power consumption, light weight and compactness make this camera the best option.

The camera is positioned at the front part of the bot. The Field of View is around 50°.

c.Proximity Sensors

Sharp GP2D12 Infrared Sensors

The GP2D12 infrared sensors takes a continuous distance reading and reports the distance as an analog
voltage with a distance range of 10cm to 80cm.

The Phidget 8/8/8 is used for interfacing IR sensors. The 8/8/8 is easily controlled by a computer through a USB computer port which provides

8 Analog input, 8 Digital input and 8 Digital output

8.Localization Module

Localization module of an automated vehicle helps answer the very simple yet elusive question “Where am I?” from the vehicle's perspective. This question becomes even more difficult to answer when sensors and odometry being utilized is moderately accurate as the noise inherited from the inaccurate data provided by these devices makes the picture more hazy.

The fact that during the return retracing of the path traveled has to be carried out makes mapping of the initial course traversed an absolute necessity.

As is suggested by the breakthrough research of Cheesman, Chatila, and Crowley which proved the
existence of a correlation between feature location errors (a.k.a mapping) due to errors in motion, i.e. localization, which in turn affect all feature locations. This suggests an inseparable link between the two important processes of Localization and Mapping. More specifically one process gives an output which is the input for the other process and therefore justice can be done to both the processes if they are carried out simultaneously.

Using this line of thought a group of methodologies termed as Simultaneous Localization And Mapping (SLAM) have been analyzed and tested out by our team. All of these make use of recursive Bayesian filters in one way or the other. There are basically two types of approaches that have been developed and used by research groups all over the world. One of them makes use of a Kalman Filter in the form of an Extended Kalman Filter (due to the non linearity of the processes at hand) and other one uses particle filters which again make use of Bayesian techniques.
After rigorous simulation and consideration our team has arrived at the following breakdown to the mapping and localization problem:

- Phase A-- Initial outdoor traversal from the starting point until the entrance of the building:
  The fact that a reactive VFH based algorithm is being used for the obstacle avoidance during this phase of the journey and the fact that the number of features (landmarks) available outdoors is very erratic (either too many or too less) a rather simplified Extended Kalman Filter will be used to statistically optimize the robot pose information i.e. the most accurate estimate of the robots x ,y and theta coordinate that can be extracted from the on board sensors will be arrived at . The measurement model of this filter would be filtered from the data obtained from

the on board stereo camera and the GPS sensor. The prediction model or the system model would be using feedback data being obtained from the encoders on the four stepper motors propelling the robot. This data would also be used to generate and update a map of the environment which has been explored. This map would be in the form of an occupancy grid which would make the task of retracing it much easier. A Fast SLAM or other vision based SLAM techniques have been kept out of consideration owing to the fact that all processing has to done onboard and such techniques would be stretching the onboard computers too far given the other algorithms already running on it.

- Phase B-- Exploration of the building:
  Since the availability of features indoor is much less of a concern indoors thus a real time implementation of Fast SLAM would be carried out during this phase of the journey. This technique has been chosen because of the fact that this type of a SLAM technique makes the most judicious use of processing power as compared to other alternatives i.e. DP-SLAM or other vision based SLAM techniques. Once again the measurement model of this filter would be filtered from the data obtained from the on board stereo camera and the GPS sensor and the prediction model or the system model would be using feedback data being obtained from the encoders on the four stepper motors propelling the robot.

- Phase C-- Retracing of the path back to the starting point:
  The map generated during Phase B would be appended to the map obtained during Phase A to generate a complete representation of the environment that has been explored so far. This map would be treated as the boundary for the robot during this phase as further exploration of the environment would only be a waste of time and power. The map would be used as an input for an Adaptive Monte Carlo Localization driver which is natively supported in the Player/Stage client server architecture being utilized. Apart from the fact that it is natively supported the other major advantage is the fact that this technique has shown encouraging results during simulation compared to other map based localization techniques.

Further mathematical details of the above module are stated in Annexure A.

9. Path Planning and Navigation Module

a)Navigation architecture:

The robot has to avoid the obstacles and reach the destination in the minimum possible time. For this we have divided this module into 2 sub-categories. A ‘global path planner’ which would try to take the robot to destination as quickly as possible. While a ‘local planner’ which would try to avoid obstacles. The ‘global path planner’ will provide the shortest possible route to the ‘local planner’ which would try to execute that task while avoiding obstacles. For ‘local planner’ we have used Virtual Force Fields (VFF) algorithm and for ‘global path planner’ we have used A* search algorithm.

Initially, we are provided with the start and goal coordinates. Here we will be using Virtual Force Field (VFF) for obstacle avoidance. The goal coordinates will provide the desired heading throughout. During initial outdoor navigation this is going to be our main algorithm. This is based on purely reactive architecture. During outdoor navigation, a series of latitude/longitude way points shall be provided along which the robot needs to move. The current subgoal shall be the next uncleared way point of the robot. The relative heading and distance of the subgoal can be calculated by subtracting the subgoal coordinates from the robot's current coordinates.

While navigation indoors, GPS coordinates are unreliable and shall not be used. As there is no fixed indoor goal, the robot shall enter its “wander” procedure. Here, a random subgoal will be calculated periodically. This shall continue until the required target (lift, button or other targets) have been localized using image processing. Once this has been done, the relative location of the current target shall become the next subgoal.

When it reaches outside again, we have the whole map this time. We use A* algorithm as the global path planner which provides way points to the VFF which acts as the local path planner. Here ours is an Hybrid architecture.

We have represented our world map in the form of square grids which is stored in the form of a 2D array. The array has a value 0 in case of free space and 1 otherwise.

b) Navigation Performance

Our hybrid reactive/delibrative algorithm is efficient and can execute without powerful hardware, as a reactive approach has been used for most of the motion planning. Some potential problems may arrive due to local minima problems, that is, the robot may get stuck in an area where the repulsive and attractive fields cancel out, and the potential field has a local minima at that point. Such problems will be tackled by temporarily assigning a random subgoal to the robot if its position does not change over a long time.Also A* suffers from the problem that the optimal path computed by it consists of sharp turns which would not be the actual path taken by the robot. This problem is overcome by allowing the local path planner to take care of navigation between successive waypoints. The advantage of using A* is that by using an intelligent Heuristic function it restricts the no of vertices's being explored in the graph, and thus proves to be a fast and efficient algorithm for optimum path planning.

c.)**Navigation Capabilities**

i)**Local motion and obstacle avoidance**

Once the current subgoal has been established, the robot needs to move towards it. To achieve this, an algorithm called Virtual Force Fields has been used. A polar map containing the location of all obstacles and the goal relative to the robot shall be constructed by interpreting the data from the sensors. The goal shall be considered a source of an attractive virtual force, which is directed towards the goal and constant in magnitude (regardless of distance from goal). Each obstacle detected on the map shall be a source of a repulsive virtual force, which is directed away from the obstacle, and linearly decreases with the distance from the obstacle. The superposition of each virtual force vector shall result in a net vector, which is the direction in which the robot must move. The turn-rate of the robot shall be proportional to the difference between this force vector and the current heading of the robot. The net speed of the robot shall be in the forward direction if the angle between the heading and the resultant is acute, and in the reverse direction otherwise.

ii)**Path Planning and Retro traverse**

We have used A* algorithm as our global path planner. Given the complete map of the world, it provides the shortest possible path. This will be used to calculate a local

subgoal when the map of the world is known.

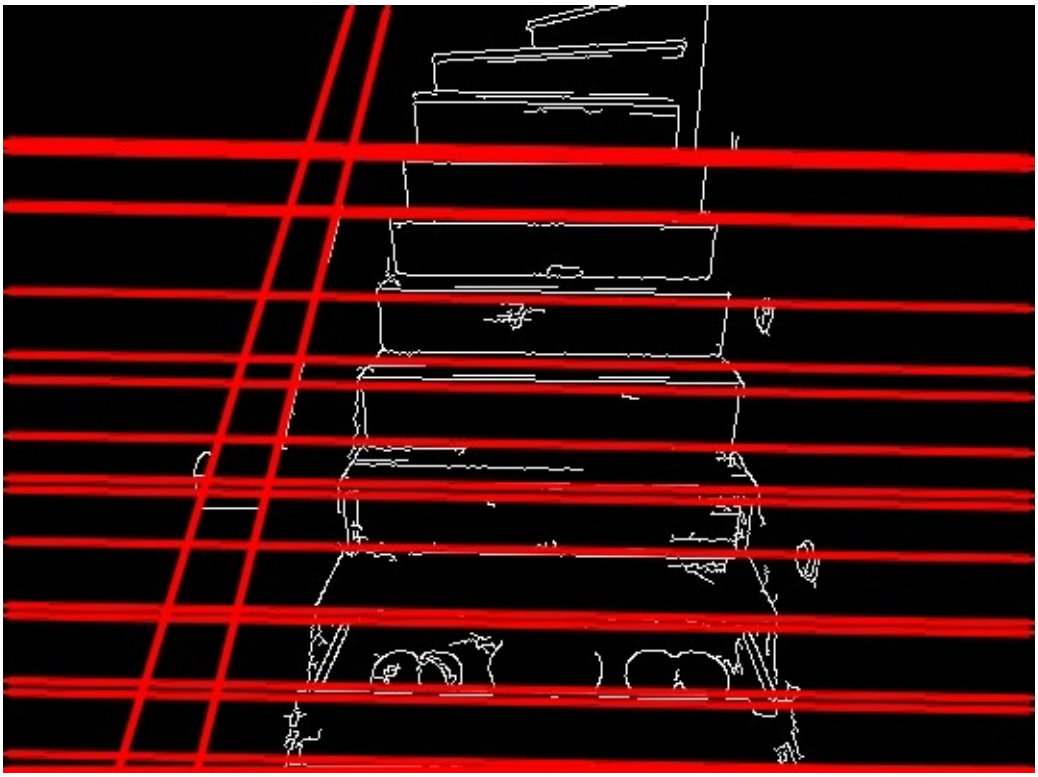
It starts from the source as the current node, gives it some cost and iterates through all its neighbours. A node is considered as neighbour if it can be reached from the current node directly. It gives appropriate costs and a corresponding number to these neighbouring nodes and adds them to its list of nodes, if it is not already there. This list of nodes is initially empty. Now it searches for the vertex with the minimum cost in its list of nodes and sets the current node as its parent. Then it sets this as the current node and repeat all the above mentioned steps until the goal has been set as its current node.
For fast and efficient implementation of the algorithm, the open list has been maintained in the form of a binary heap in which the node with the minimum cost is always at the top. Also, the heap is such that the cost of parent node is less than the costs of both the child nodes. Among the child nodes cost of left child is the least. This helps in easy and fast accessing of required nodes.

10. Image Processing module

a. Staircase Detection

The image captured by the camera will be transformed by the Canny edge detection algorithm, and then lines present shall be extracted using the Hough transform. A stack will be used to store each line, ordered by the angle with respect to the horizontal. A staircase shall be present when there is a large density of lines present with an angle which is close to zero. Such an algorithm shall also be applied to the depth map corresponding to the particular image.

Once the staircase has been found, the distance to it shall be determined from the depth map. The coordinates of the staircase shall be fed to the path planner as the new goal. Once the robot is adjacent to the staircase, it will turn in place until it is aligned parallel to the staircase, and the motors shall be driven forward.
An example of a staircase image scanned by our algorithm:



b.Elevator Button Detection

After the robot is aligned in parallel to the elevator door plane the button identification algorithm will start working.

For this we are using Intel's OpenCV libraries along with the PointGrey BumbleBee2 stereo camera.
The button detection algorithm will work as follows:

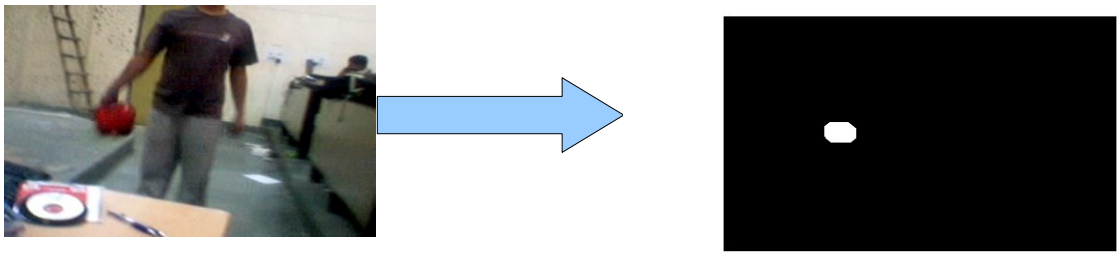
-Feature extraction (basic geometrical shapes) from the given image using canny edge filter and hough transform.
-Image scan after alignment.
-Feature matching in the scanned image.
-In case of affirmitive match, the depth at the point will be fed to the main code along with the horizontal and vertical coordinartes with respect to the floor and the robot.
-The mechanical arm will now approach the button coordinates.

c.Target Detection Methodology

BumbleBee 2 camera is used for target identification in the indoor environment.
Intel's OpenCV library are used for the image processing functions.
The target identification algorithm starts scanning the environment as soon as the robot enters the room.
The approach being used is as follows:

-Each image is scanned (in RGB colorspace).
-The designated targets are identified as colored blobs in the scanned image.
-A basic blob detection algorithm then searches simultaneously for the specified color blobs in each scanned image.
-The blob detector has a threshold value of color ratios for every color.(like for red R/G and R/B values are calculated)
-The image is then converted into a greyscale image with different values for each color blob.
-The centre of the greyscale blobs are then looked for depth values in the simulataneoulsy generated depth map.

-The robot then approaches the blob centre until a depth threshold value is reached. (threshold is the mechanical arm reach).An example of the Target Detection is given below:



Original Image showing Red target in hand

Target Detected

11. Power Management Module

Type of battery	Amp Hours	Durations (Hours)	Weight	Device Controlled
25.9V Li-PO battery	12 AH	1.5	1.8 kg	Motors
11.1 V Li-PO battery	3.8 AH	2	.3 kg	Sensors, Camera, GPS, electronics

The schematic diagram shown represents the electrical power system of the proposed robot. One,25.9 volt 12 Amp hours and one 11.1 V Li-Po rechargeable battery power the motors and electronics indirectly through a custom electrical box . The team has made an electrical box. It holds all the electrical circuits at one place.All wires coming out of the box have been colour coded for easier identification .It has been kept at a place to experience least vibration, shocks, and noises. It facilitates easy serviceability. It identifies problems easily.We are using 25.9 volt battery for giving power to motors and one battery of 11.1 volt for giving power to other electronic circuits. Two LED`s on the front of the electronical box provide instant verification of the electrical systems proper functioning. The size of main power supply board is around 30*25 cm. Predicted time required before a recharge is calculated to be 1-2 hours at normal speed for a fully charge battery. This prediction is verified by experimental tests. The time varied is based on the tasks the vehicle is supposed to perform. A backup battery is always charged and ready to replace a weak battery. We can have chosen any other battery like lead acid, NIMH, NICD etc. But they don't fulfill our requirement .Because we want, most important, a light battery which can give high power so we chose Li-Po battery. For regulation we use dc-dc converters instead of regulators or resistances.

a.**Wireless E-Stop**

When creating any autonomous vehicle it is very important to take safety into account. A careful safety analysis is essential to ensure that no one will be harmed while operating, observing, or working on the vehicle .

The robot includes two forms of emergency stopping, wireless and manual. For the wireless E-stop, we use RF modules. Whenever the "OFF" button is pressed on the remote control ,power to the relay is interrupted, which in turn open the relays internal switch ,thus cutting the DC power to the motors. It can operate up to a distance of 100m. The total weight of wireless stop is .2Kg of dimension 15X10cm.The location of antenna is in the rear of the robot so that the signals are not interrupted by GPS.

b.Manual Stop

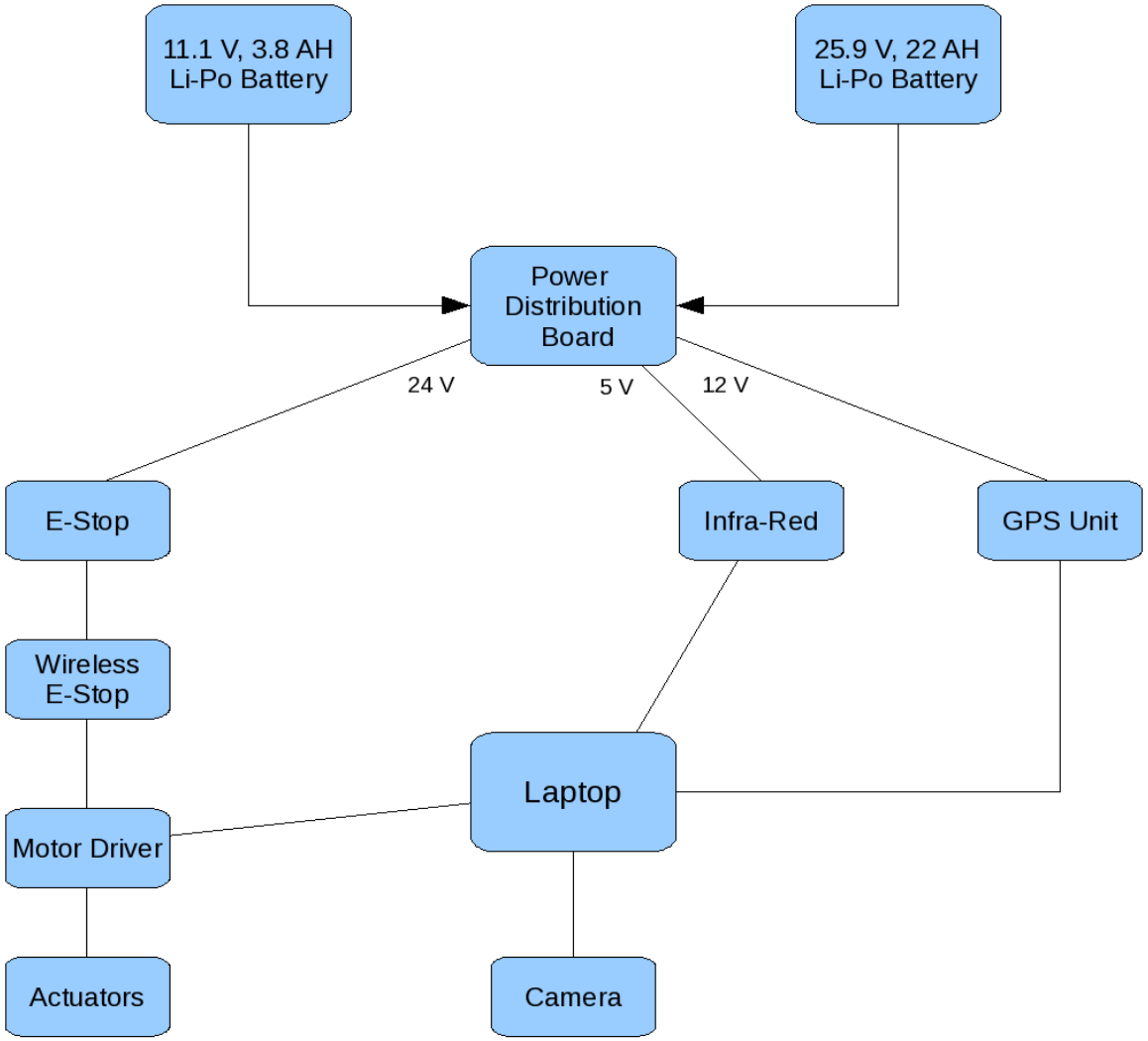
The mechanical switch is connected to the main line which provides power to the relays; hence, when the mechanical switch is pressed, power is cut to the relays, thus, opening the internal switches and cutting DC power to the motors.

c.**E-STOP Communication**

The frequency of RF modules which we have used for E-STOP communication is 315MHz and range of the module is at most 100m.

d.**Safety Consideration**

Another consideration of safety for the robot is that we are using PCM (protection circuit monitoring) for battery so that a maximum current, can flow through the circuit. If current flowing is above the maximum limit the circuit breaks, which prevents our equipments and batteries. Fuses, circuit breakers and disconnect switches protect all of the robot's components from overload, noise spikes, and short circuits. Our overall system of interrelating sensors provides us the opportunity for safe navigation because of the inherent redundancy applied through software.


11.1 V, 3.8 AH
Li-Po Battery
25.9 V, 22 AH
Li-Po Battery
Power
Distribution
Board
24 V
5 V
12 V
E-Stop
Infra-Red
GPS Unit
Wireless
E-Stop
Laptop
Motor Driver
Actuators
Camera

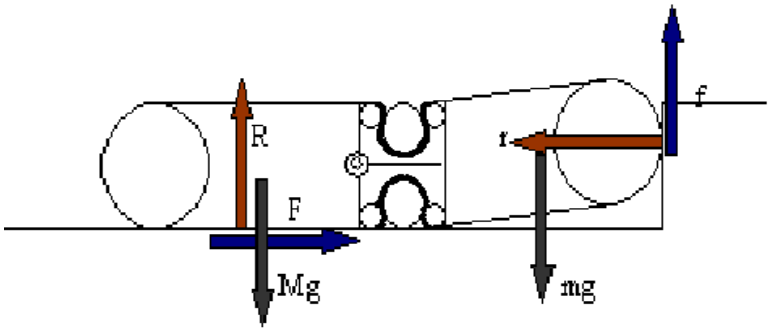
ANNEXURE A

CALCULATIONS

1.**Mechanical Design:**

Specifications And Calculations

Designing the robot in two parts helped in easy ascending of the stairs as is proved by the following force diagrams:



Let r be the reaction force exerted by the stair on the tread of the robot, R be the reaction force exerted by the ground on the tread, m be the mass of the front part, M be mass of rear part, f be the force exerted due to friction by the stairs and F be force of friction due to ground. ‘g’ is acceleration due to gravity (9.81 m/s2). Consider stairs and ground to be of the same material. Let u to be coefficient of friction between the stairs and the treads. If we consider ground to be of concrete and treads of rubber, the value of u is found to be between 0.6 - 0.85.

Balancing the forces in the vertical plane

$$2f + 2R = mg + Mg$$

Or $2\mu r + 2R = (m + M)g$ ……1

Balancing the forces in the horizontal plane

$$F = r$$

Or $\mu R = r$ ……..2

Solving equation 1 and 2, we get

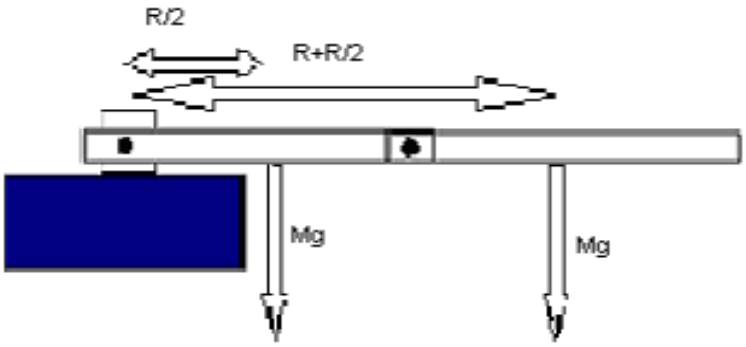
$R=(m+M)g / 2(\mu^2+1)$
$r= \mu (m+M)g / 2(\mu^2+1)$
$F= \mu (m+M)g / 2(\mu^2+1)$
$f= \mu^2(m+M)g / 2(\mu^2+1)$

So, the required torque = (F+f)*Pulley radius

As two motors are driving a single belt, therefore the torque required per motor will be reduced to half.

Net motor torque = (F+f)*Pulley radius*1/2

2.<u>ACTUATOR TORQUE CALCULATIONS:</u>



Let M be the weight of one arm of the robotic arm. As both arms have same length and made up of same material, both of them have same mass. Let R be the length of each arm. Therefore, the torque at the point of intersection of the two arms is:

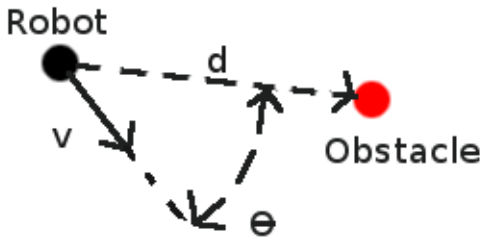
T' = Mg*(R/2) .....(1)

The torque required by the motor driving the hind arm pulley will be:

T" = Mg*(R/2) + Mg*(R+R/2) .....(2)

The maximum torque required by a motor to drive this robotic arm will be at the point when the arm is in horizontal position. Hence, the torque required by the motor will be T" as its the maximum torque required by a motor.

2.Virtual Force Fields



We can assume the robot to be a point object. Let its velocity vector be V and distance from obstacle d. The angle between these vectors is Θ. The velocity component of V along d is Vcos(Θ). The time taken to avoid collision with the obstacle is 2d/Vcos(Θ). The strength of the force field vector corresponding to a particular obstacle is inversely proportional to the time taken to avoid collision, and hence our virtual force field for an obstacle with coordinates (d,Θ) is Vcos(Θ)/2d, in the direction opposite to d. We sum up the virtual field due to each obstacle to get the net repulsive vector.

3.A* algorithm

The cost it assigns to each node is calculated as below:
The cost assigning process takes into account the present cost of reaching the current node from the source and even estimates the future cost it will take to reach the destination. The future cost is at present calculated through the Manhattan method which calculates the absolute difference in the coordinates of the 2 nodes.

Hence during an iteration when it reaches a new neighbour some of the variables may look like this:

```
parent[neighbour] = current;
cost_to_destination[neighbour] = absolute(x_coordinate[neighbour]-destination_x) +
absolute(y_coordinate[neighbour]-destination_y) ;
cost_from_source[neighbour] = cost_from_source[current] + 1;
Total_cost[neighbour] = cost_from_source[neighbour] +
cost_to_destination[neighbour];
```

The correctness of A* star depends a lot on the heuristic estimated to reach the destination from the current node. Hence, a special emphasis was given to it during coding. After some extensive research and continuous testing Manhattan method was found to be the best and hence was incorporated in the code.

This algorithm has been fully tested during In-system simulations and has passed all those tests by providing correct results in real-time.

4.<u>Localization Module:</u>

For the Kalman filter described above the equations for the state update or the prediction model would be given by (all symbols have their usual meanings):

a) $x_i = x_{i-1} + G_i * u_i$

Here u_i represents the velocity input during the time interval i-1 to i but it can be taken as any other actuator input (acceleration,etc) but G_i which basically transforms the actuator control input to the form of x_i will take a different form.

b) $P_i = P_{i-1} + Q_i$

Here P_i keeps track of the uncertainty of the members of x_i i.e. it keeps track of all the binary relations of the members of the matrix x while Q_i represents the covariance of the process noise. This equation seems pretty intuitive, it just adds the process noise to the uncertainty of the world state at each iteration of the filter.

The update equations for the measurement model would be given by:

c) $x_i = x_{i-1} + K_i * v_s$

Here K_i represents the Kalman weight or the Kalman gain and v_s is the innovation or the adjustment vector which is suggested by the new sensor measurements and the

Kalman gain represents our confidence in these new measurements i.e. it basically models how accurate our sensors are.

d) $P_i = P_{i-1} - K_i * S_i * (K_i)^T$

This equation basically represents the fact that with more and more iterations of the filter, the uncertainty in the world state (P_i) , keeps reducing i.e. our representaion of the world state converges to perfection but due to equation (2) the uncertainty keeps increasing but if

$$Q_i =< K_i * S_i * (K_i)_T$$

then the filter tends towards a statistically optimal estimate of the world state. This condition has to be ensured while constructing the prediction and the measurement models.

Here the prediction model is linear but this is never a possibility in a real world scenario which forces us to use an Extended Kalman Filter which is a simple extension of the above filter.

Although the above explanation is an oversimplification of the actual implementation but then a rigorous discussion of these equations is well beyond the scope of this document.

5.<u>Power Module Calculations:</u>

We used four 86SH118 motors and three DPM 57SH51-4A having current rating 6A and 4A respectively. And we used 2 Li-Po batteries of 3.8Ah and 22Ah.For driving motors, we use 22Ah battery.

Let us use 6A battery for 55 minutes then it uses total current = 22A …… 1

If us use 4A battery for 5 minutes then it uses total current = 1A …… 2

Let max .current dissipated in 1 hour = 2A …… 3

Total current consumed in circuit = 25A …… 4

Time = 22*60/25 =52.8 minutes

So 52.8 minutes is theoretical time, but actually practical time is more than 1 hour because these calculations are on the basis of continuous current taken by the motors but

actually current taken by motors are discontinuous.

For electronic circuits, we used 3.8 Ah battery.

I.R. sensors takes max current =.2 A ……. 5

GPS takes max current =.065 A ……. 6

Let current dissipated = 1A ……. 7

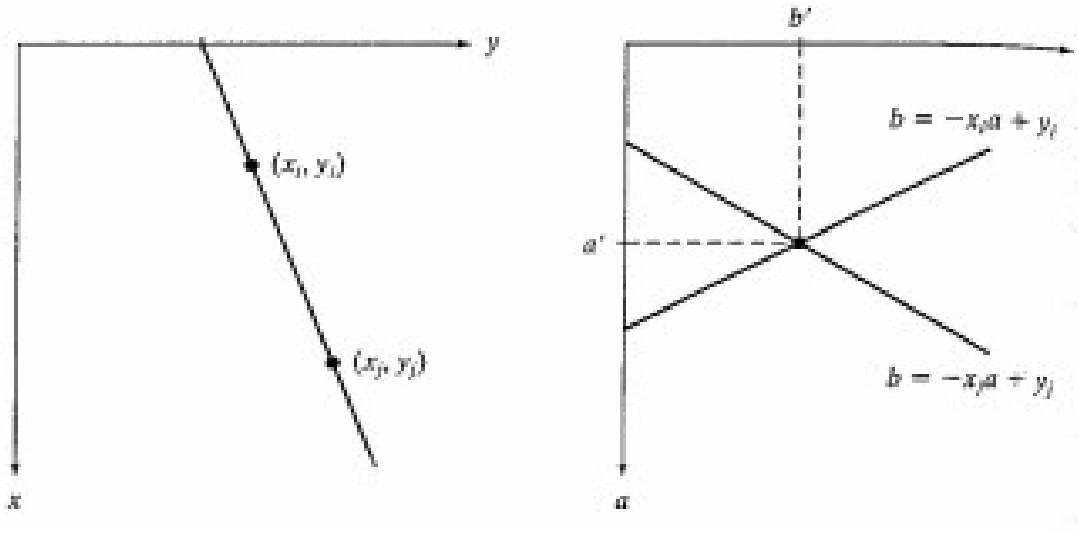
Total current consume = 1.265A ……. 8

Time = 3.8/1.265= 3 hours

6.Image processing calculations

a.Hough Transform

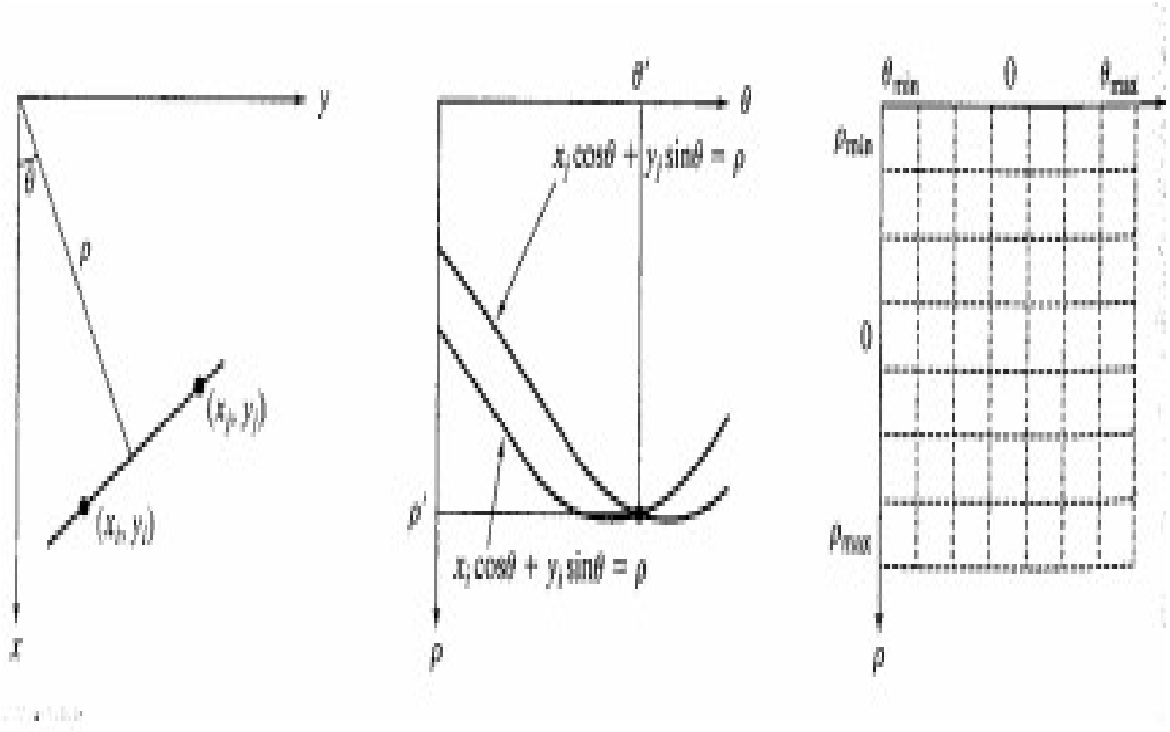
The following explains it's functioning:
Given a set of points in a image , suppose that we want to find the subsets of these points that lie on a straight line.
we consider a point (xi,yi) and all the lines that pass through it. Infinetely many lines pass through (xi,yi) , all of which satisfy the slope intercept equation yi=a*xi +b for some values of a and b. Writing this equation as b=-xi*a +yi and considering the ab plane yields the equation of single line for a fixed pair (xi,yi).Furthermore , a second point (xj,yj) also has a line in parameter space associated with it, and this line intersects



the line associated with (xi,yi) at (a',b'), where a' is the slope and b' the intercept of the line containing both (xi,yi) and (xj,yj) in the xy plane.

The normal form of the line is:
xcos(theta)+ysin(theta) =p

The figure illustrates the geometrical interpretation of the parameters p and theta . A horizontal line has theta =0 with p =+ve x intercept.similarly a vertical line has theta =90 with p being equal to the +ve y intercept , or theta =-90 with p being equal to -ve y intercept. Each sinusoidal curve in the figure represents the family of lines that pass through a particular points (xi,yi). The intersection point(p',theta') corresponds to the line that passes through both (xi,yi) and (xj,yj).

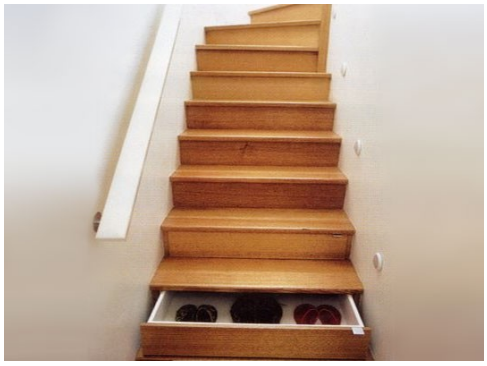


The computational attractiveness of the hough transform arises from the subdividing ptheta parameter space into so called accumulator cells , as shown in the figure. where (p_min,p_max) and (theta_min.,theta_max.) are the expected range of parameter values . Usually the(i,j),with accumulator value A(i,j), corresponds to the square associated to the parameter space coordinates (p_i,theta_j). Initially these cells are set to 0.the for non background point (x_k,y_k) in the image plane , we let theta equal each of the allowed y_k subdivision values on the theta axis and solve for the corresponding p using the equation p=x_k*cos(theta)+ y_k*sin(theta).the resulting pvalues are then rounded off to the nearest allowed cell value along the row axis. the corresponding accumulator cell is then incremented at the end of this procedure,a value of Q in A(i,j) means that the q point in the xy plane lie on the line x cos(theta_j)+ y sin(theta_j)= p_i.The no. of subdivisions in the p theta plane determines the accuracy of the co linearity of these points.

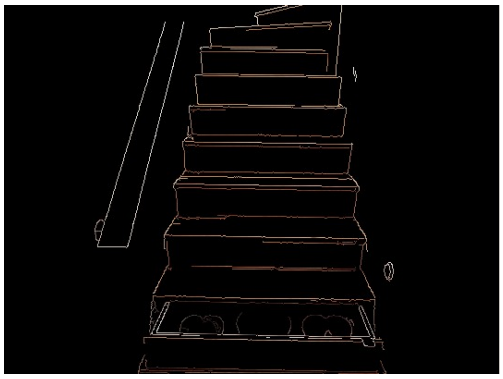
b.Canny Edge

The canny edge detection algorithm is used for the abstracting a given image into its edges, or regions of sharp change, which are of interest to us in feature extraction.

First, the image is blurred using the gaussian filter, after which, the image is differentiated along the x and y directions, to produce two differential vectors d1 and d2. If d1^2 + d2^2 is above a given threshold value, that pixel is assumed to be a part of an edge, and is marked white. Otherwise, it is marked black.

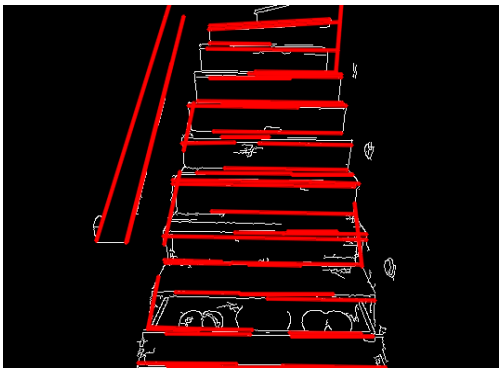


Original Image



After Canny Edge Operation

Grayscale of original image



After Hough Transformation

ANNEXURE B

References and Links :

1.Path planning and navigation module:

a) VFF and VFH--Fast Obstacle Avoidance for Mobile Robots
(www-personal.umich.edu/~johannb/vff&vfh.htm)

b) The vector field histogram-fast obstacle avoidance for mobile robots
(www.cs.uml.edu/~holly/91.549/Fall2004/borenstein91vector.pdf)

c) A* Pathfinding for Beginners by Patrick Lester
(www.policyalmanac.org/games/aStar Tutorial.htm)
(www.gamedev.net)

d) A-star Algorithm (www-cs-students.stanford.edu/~amitp/Articles/AStar2.html)

e) Tony Stenz's Papers on D* (Dynamic A*) Path finding (www.frc.ri.cmu.edu/~axs/)

f) Using Binary Heaps in A* Pathfinding by Patrick Lester
(www.policyalmanac.org/games/binaryHeaps.htm)

g) A Guide to Heuristic-based Path Planning by A. Stenz
(www.ri.cmu.edu/pub_files/pub4/ferguson_david_2005_2/ferguson_david_2005_2.pdf)

2.Localisation

a) Durrant-Whyte, Hugh "Localization, Mapping, and the Simultaneous Localization and Mapping Problem." Australian Center for Field Robotics. Sydney. 2002.

b) Heibert-Treuer, Bradley "An Introduction to Robot SLAM (Simultaneous Localization And Mapping" (http://dspace.nitle.org/handle/10090/782)

c) Riisgaard, Søren and Rufus Blas, Morten " SLAM for Dummies A Tutorial Approach to Simultaneous Localization and Mapping "

d) Maybeck, Peter S. Stochastic, Models, Estimation, and Control. New York: Academic
Press,1979

3.Player/stage

(http://www.playerstage.sourceforge.net)

4.Image Processing

Digital image processing using MATLAB by Gonzalez and Woods.