Robot Navigation

Bagadi Tarun Kumar
1501013

Electronics and Communication Engineering Department

Indian Institute of Information Technology Guwahati

A Report submitted in partial fulfillment of
Bachelor of Technology

30th April, 2019

Certificate of Approval

This is to certify that the 8th Sem B.Tech. project report entitled **"Robot Navigation"** is a record of bonafide work carried out by **"Mr.Bagadi Tarun Kumar"** under my supervision and guidance.

The report has partially fulfilled the requirements towards the degree of Bachelor of Technology in Electronics and Communication Engineering at Indian Institute of Information Technology, Guwahati.

Signed:

Dr. SURAJIT PANJA
Assistant Professor
Department of Electronics and Communication
Indian Institute of Information Technology, Guwahati
Guwahati 781001, Assam, India.

Contents

Chapter 1

Introduction

Navigation in general usage is know as the process or activity of accurately ascertaining one's position and planning and following a route. Navigation using directions to known sources has been in use for centuries. The observation of the sun and constellations of stars to infer position and heading are among the most ancient techniques for localization and navigation. For any mobile device, the ability to navigate in its environment is important. Avoiding dangerous situations such as collisions and unsafe conditions ,the robot has a purpose that relates to specific places in the robot environment it must find those places,and navigate autonomously.Navigation comprises everything a robot needs to get from point A to point B as efficient as possible.

Navigation is a special task for mobile robots, after all, isn't getting somewhere; And getting somewhere in a complex environment means navigation. You may not have given much thought to how you manage to walk across a path, but a robot to do the same thing is challenging. Coordinating sensor and motor skills to perform a specific task. Where am I? Where am I going? Where are you? What is the best way to get there? When will I get there? technology can answer all these questions. The technology employed in mobile robot navigation is rapidly developing. Here are two relatively modern systems satellite based Global Positioning Systems(GPS) and image-based Light Detection and Ranging (LIDAR)or cameras.

This report gives a detailed attempt to develop an autonomous robot navigation using GPS along with velocity consensus and successful understanding of slam and implementation of robust HECTOR -SLAM in real time on the FIREBIRD VI mobile robot using a RPLIDAR sensor, with the help of open source software hector-slam modules which is used for the development of such complex capabilities. This main part focused on the implementation of an environment modelling technique

1 GPS

Global Positioning System (GPS) is a space-based navigation system developed and operated by the United States government which is used to establish a position at any point on the globe. The GPS is also known as a satellite-based navigation system made up of a network of 24 satellites placed into orbit by the U.S. Department of Defense. GPS was originally intended for military applications, but in the 1980s, the government made the system available for civilian use. GPS can show you the exact position on the earth at any time, in any weather, no matter where you are! The GPS technology has made an impact on navigation and positioning needs with the use of satellites and ground stations the ability to track aircraft, cars, cell phones, boats and even individuals has become a reality. A system of satellites, computers, and receivers that is able to determine the latitude and longitude of a receiver on Earth by calculating the time difference of the signals. GPS uses these "Man-made stars" as reference points to calculate positions accurate to a matter of meters. In fact, with advanced forms of GPS, you can make measurements to better than a centimetre! In a sense, it's like giving every square meter on the planet a unique address.

Any position on the earth can be described by its latitude and longitude which are the lines of an imaginary grid laid on the earth's surface. Grid lines running east to west and parallel to the equator are known as parallels of latitude with the equator itself being 0 latitude. Lines that run north to south, between the North and South Poles, are known as meridians of longitude. The prime meridian or 0 longitude, runs through the Greenwich Royal Observatory in the United Kingdom from which it takes its name. Measured by the angle that they form at the centre of the earth, lines of latitude and longitude are known by degrees (), minutes ('), and tenths of a minute there being 60' in 1, and 360 in a circle. The angle of latitude is measured from the centre of the earth along the prime meridian from the equator (0), ranging from 0 to 90 north and to the south. The angle of longitude is measured at the centre of the earth along the equator from the prime or Greenwich meridian (0'), ranging from 0 to 180 east and to the west. The main advantage of using GPS for localization is that the data gathered does not depend on previous readings and therefore errors in localization do not grow over the course of time as they would with localization methods such as dead reckoning with wheel encoding

The GPS is a high performance of MTK positioning engine with direct USB interface. GPS receiver gives output in standard NMEA format with an update rate of 1 second at 9600 bps. This GPS receiver has an onboard battery for memory backup for quicker acquisition of GPS satellites. The standard output of GPS receivers uses NMEA strings. There are many NMEA messages that can be published. The one that is the most interesting to us is typically the GPGGA (sometimes just called GGA) that contains the position and time.

2 SLAM

The ability to learn a model of the environment and to localize itself is one of the most important abilities of truly autonomous robots able to operate within real-world environments. Localization and mapping are fundamental requirements to enable mobile robots to operate robustly in their environments. Mapping is basically determining the location of objects in the environment and localization is establishing the robot's position with respect to these objects is. The SLAM problem can be posted in the metaphorical chicken-and-egg sense; for a robot to know its current location it needs an accurate map, An unbiased map is needed for localization while an accurate pose estimate is needed to build that map.SLAM is a technique used by robots and autonomous vehicles to build a map within an unknown environment or to update a map within a known environment while keeping track of their current location. A robot has to address both the mapping and the localization problems in order to autonomously solve the map learning problem. Due to the coupling between these two tasks, autonomous map learning has been a challenging problem. The mapping and localization problem, called SLAM. It solves the problem of navigating a vehicle in an unknown environment, by building a map of the area and using this map to deduce its location, without the need for a priori knowledge of the location. The solution to this problem is of great importance to the field of autonomous robots operating in GPS-denied environments, Maps could be made in areas which are too dangerous or inaccessible to humans, like deep-sea environments or unstable structures. It would make robot navigation possible in places like space stations and other planets. The solution of these two problems will allow robots to assist humans in important tasks such as autonomous driving or hazardous search-and-rescue missions.

SLAM is similar to a person trying to find his or her way around an unknown place. First, the person looks around to find familiar markers or signs. Once the person recognizes a familiar landmark, he or she can figure out where they are in relation to it. If the person does not recognize landmarks, he or she will be labelled as lost. However, the more that person observes the environment, the more landmarks the person will recognize and begin to build a mental image, or map, of that place. The person may have to navigate this certain environment several times before becoming familiar with a previously unknown place. In a related way, a SLAM robot tries to map an unknown environment while figuring out where it is at. The complexity comes from doing both these things at once. The robot needs to know its position before answering the question of what the environment looks like. The robot also has to figure out where it is at without the benefit of already having a map

SLAM is not a particular algorithm or piece of software, but rather it refers to the problem of trying to simultaneously localise (i.e. find the position/orientation of) some sensor with respect to its surroundings, while at the same time mapping the structure of that environment. This can be done in a number of different ways, depending on the situation; so when referring to a 'SLAM system'.

SLAM describes the process of building a map of an unknown environment and computing at the same time the robot position with the constructed map. Both steps depend on each other. A good map is necessary to compute the robot position and on the other hand, just an accurate position estimate yields to a correct map Creating maps of the environment is important for two reasons: Allowing first responders to both perform situation assessment and localize themselves inside buildings While purely geometric maps such as occupancy grid maps are useful for navigation and obstacle avoidance, additional semantic information like the location of objects of interest is very important for first responders and required for intelligent high-level autonomous behaviour control This means the SLAM problem has to be solved to generate sufficiently accurate metric maps useful for navigation of first responders or a robot system.

Mapping
Occupancy grid maps aim to distinguish between empty and occupied space within the environment. In order to do so, the area to be explored is split into a grid of squares Each square is assigned a prior probability of occupancy, typically 0.5. Assigning such a prior probability indicates that there is no available information about the environment a priori since each square is assigned a prior probability of 0.5 signifying it is equally likely to be occupied or unoccupied. As new sensor data is collected the occupancy value of each square is recursively updated to incorporate this data. This type of representation is most commonly used to represent data collected by robots mounted with distance sensors such as ultrasound or laser range finders and many implementations of SLAM. The white areas have a high probability of being unoccupied, black areas a low probability of being unoccupied and grey areas have not been explored (so are equally likely to be unoccupied or occupied). The discrete nature of occupancy grid maps limits the precision that can be achieved and also does not allow the direct computation of interpolated values or derivatives. For this reason, an interpolation scheme allowing sub-grid cell accuracy through bilinear filtering is employed for both estimating occupancy probabilities and derivatives. Intuitively, the grid map cell values can be viewed as samples of an underlying continuous probability distribution

This representation has a number of advantages. Information is collected and represented for all map locations with an explicit representation of empty space. This representation also allows for different models of sensor uncertainty to be used when updating occupancy values This representation has a number of limitations. Depending on the size of the area to be mapped and the chosen grid resolution it can be computationally costly, having to update and maintain occupancy values for a large number of grid cells. A finer grid results in better map quality, however, so there is a trade-off between map accuracy and computational complexity To be able to represent arbitrary environments an occupancy grid map is used, which is a proven approach for mobile robot localization using LIDARs in real-world environments As the LIDAR platform might exhibit DOF motion, the scan has to be transformed into a local stabilized coordinate frame using the estimated attitude of the LIDAR system. Using the estimated platform orientation and joint values, the scan is converted into a point cloud of scan endpoints. Depending on the scenario, this point cloud can be preprocessed, for example by downsampling the number of points or removal of outliers. For the presented approach, only filtering based on the endpoint z coordinate is used, so that only endpoints within a threshold of the intended scan plane are used in the scan matching process.

Localization

Mobile robot localization is the problem of determining the pose of a robot relative to a given map of the environment. It is often called position estimation. Mobile robot localization is an instance of the general localization problem, which is the most basic perceptual problem in robotics. the robot is given a map of its environment and its goal is to determine its position relative to this map given the perceptions of the environment and its movements. Mobile robot localization can be seen as a problem of coordinate transformation. Maps are described in a global coordinate system, which is independent of a robot's pose. Localization is the process of establishing a correspondence between the map coordinate system and the robot's local coordinate system. Knowing this coordinate transformation enables the robot to express the location of objects of interest within its own coordinate frame,a necessary prerequisite for robot navigation.For estimating the pose of the platform hector slam use an Extended Kalman Filter (EKF) .

The Kalman filter is a Bayesian filter framework in which a Gaussian random variable with mean and covariance matrix is estimated through periodic measurements. The filter is typically comprised of two main steps: prediction and update. These are repeated over a series of time steps, delineated by the arrival measurements. The prediction step propagates the state estimate through time. The Kalman filter uses a linear dynamic model, characterized by a linear operator and is subject to zero-mean Gaussian-distributed noise, with covariance matrix. The update step uses the measurement to update the state estimate. The Kalman filter uses a linear measurement model, characterized by a linear operator, and is subject to zero-mean Gaussian-distributed noise, with covariance matrix. Using a quantity called the optimal Kalman gain, the Kalman filter update yields the minimum mean squared error This implies that for linear dynamic and measurement system models, it is possible to construct perfect relative map using the Kalman filter approach, and therefore solve the SLAM problem The Extended Kalman Filter (EKF) is a generalization of the Kalman filter to non-linear models. For estimating the pose of the platform hector slam use an Extended Kalman Filter (EKF). To improve performance of the scan matching process, the pose estimate of the EKF is projected on the xy-plane and is used as start estimate for the optimization process of the scan matcher. Alternatively, the estimated velocities and angular rates can be integrated to provide a scan matching start estimate In the opposite direction, covariance intersection (CI) is used to fuse the SLAM pose with the full belief state. A simple Kalman measurement update would lead to overconfident estimates because it assumes statistically independent measurement errors.

3 Working of SLAM

SLAM is constructing a map using mobile robot (mapping), then determining the position of this robot relative to this map. First, data about the entire environment is collected. Secondly, the data (observation) is assigned to the landmarks of the environment. These landmarks are either artificial or natural. The advantages of artificial landmarks are that they are designed to be detected optimally. In addition, the artificial method is more reliable than the natural one. On the other hand, the detection of natural landmarks is more difficult, but they are more efficient for large maps. There should be three or more landmarks to manage us pose the robot. The error of landmark positioning is bounded as it doesn't depend on previous readings. The collected data is used also for building maps for the surroundings. In this task, the data is collected from different sensors, each one is oriented in a different direction. The view obtained from each sensor is called the sensor view, and then these views from each sensor are integrated to compose the local map (a map for a single location at a time). After that, the robot moves to another location and repeats the previous steps to get another local map for a new location. Finally, the local maps are integrated to obtain the global view. There are different kinds of maps. First, the metric (grid) map. In this map the environment is divided into square cells of the same size. The obstacles are defined with black cells and a free space is defined with a blank cell. This map requires relatively huge memory size and extensive computational time to generate it. The second kind is the topological map (graph-based), it is relatively better than the metric map in terms of computational time consumed and memory consumption but on the other hand, it is more difficult in constructions; also, the position of the robot cannot be located specifically on it. Therefore, it is not useful for SLAM. The third map is metric topological map; it has the same problem of the topological map. Therefore, the most suitable map for SLAM is the metric map.

Applications of SLAM
1.Evaluation of a proposed facility for producing a rocket launch system
2.Design and analysis of a pharmaceutical manufacturing facility
3.Analysis of methods for distributing oil product
4.At home: vacuum cleaner, lawn mower
5.Air: surveillance with unmanned air vehicles
6.Underwater: reef monitoring
7.Underground: exploration of mines
8.Space: terrain mapping for localization

Chapter 2

Motivation

Most intelligent robot control systems begin with the goal of creating a robot to carry out human- issued tasks. These tasks vary in difficulty but must, by their very nature, involve abstract concepts. Modern industrial society needs increasingly accurate and reliable methods of navigation to compensate for its decreasing tolerance of failures, delays, and costs in various systems Autonomous Construction machinery must dig in precisely the right place, lest they cause a fire or a widespread blackout. Modern farm machinery now varies the quantities of seed and soil conditioner dispensed on each furrow to optimize crop yields across the entire field. The economic success of mine depends on the ability of its machines to locate the ore and avoid the gangue. All these applications and more demand better navigation technology as time goes by. GPS has been extensively used in land vehicle navigation applications.

Mobile robots are finding a way more and more in our daily life from vacuum cleaners to autonomous driving vehicles. Furthermore, navigate-able industrial robots are not very far away. Mobile robots can be used at warehouse, autonomous cargo handling at seaports, or to operate in a hazardous environment for humans such as the site of nuclear power plant for debris removal or used underwater environment. Mobile robots offer a multitude of tasks that could be performed by autonomous robots. They could for example monitor reefs, examine offshore oil rigs, explore subsea caves or collect data for environmental studies. The appeal for using autonomous robots for these tasks is that they can generally reduce the costs and the risks involved when performing them manually. For a robot to be able to work autonomously it needs to be able to localize itself, sometimes in surroundings which have never been mapped before. Since GPS is not available underwater, localization needs to be performed in a different way. This could, for example, be solved by so-called underwater GPS based on acoustic localization. The problem with these techniques is that they are expensive and only work in the local area where they have been installed. A location-independent and low-cost solution to this problem could be the usage of SLAM.

Chapter 3

Work Done

1 Quick View Of Robot

Fire Bird VI platforms set the standard for intelligent mobile robots by containing all the components for sensing and navigation in real-world environment. This has become reference platforms in a wide variety of research projects. It comes in fully assembled and ready to use form. It has high precision DC gear motors with the high resolution position encoders. Motors are driven by smart motion control unit with velocity and acceleration control. By selecting correct locomotion drive, robot can be used in indoor and outdoor environment. Robot has ultrasonic range sensors with range of 6 meters, IR distance range sensors and line sensors and many more expandable interfaces. Robot is powered by high capacity battery pack. Fire Bird VI robot has two wheel differential drive and a caster wheel at the front side for the support. It is the most recommended configuration for accurate locomotion. This configuration is only used in the indoor environment. It has on board pc with external gps and imu, Servo Pod interfacing board.
HARDWARE SPECIFICATIONS:-
Weight of the robot 4.8 Kg
Power:-
System Power: 4 Cell, 16.8V Lithium Polymer battery
PC Power : 4 Cell, 16.8V Lithium Polymer battery
Sensors :-
8 x Ultrasonic sonar sensors with 6 meter range
8 x Infrared distance range sensors.
Motion control :- DC geared
Communication :- 1.USB serial communication.
2).wireless 2.4 GHz Wireless communication module(rf module)

2 Method Of Communication

Communication between or among the robots is done by using is LCM- **Lightweight Communications and Marshalling (LCM)** module. The robots do not speak a language like English. Rather, they speak a computer language that allows them to receive and transmit commands.

We describe the Lightweight Communications and Marshalling (LCM) library for message passing and data marshalling. The primary goal of LCM is to simplify the development of low-latency message passing systems, especially for real-time robotics research applications. Messages can be transmitted between different processes using LCM's publish/subscribe message-passing system. A platform and language-independent type specification language separates message description from implementation. Message specifications are automatically compiled into language-specific bindings, eliminating the need for users to implement marshalling code while guaranteeing run-time type safety. LCM is notable in providing a real-time deep traffic inspection tool that can decode and display message traffic with minimal user effort and no impact on overall system performance. This and other features emphasize LCM's focus on simplifying both the development and debugging of message passing systems. In this paper, we explain the design of LCM, evaluate its performance, and describe its application to a number of autonomous land, underwater, and aerial robots.
Supported platforms:- GNU/Linux; OS X; Windows; Any POSIX-1.2001 system (e.g., Cygwin, Solaris, BSD, etc.)
Supported languages:-o C C++; C; Java; Lua; MATLAB; Python;
We divide our description of LCM into several sections: type specification, marshalling,communications, and tools. Type specification refers to the method and syntax for defining compound data types, the sole means of data interchange between processes using LCM. Marshalling is the process of encoding and decoding such a message into binary data for the communications system which is responsible for actually transmitting it.

LCM message consists of a channel name and a payload. The channel name is a string identifying the semantic grouping of the message. The payload is a binary blob of data that corresponds to the contents of the message. Usually, the payload is the binary encoding of a marshalled LCM type. Each LCM message is transmitted using one or more UDP datagrams to a UDP multicast group and port. The address and port must have been agreed on beforehand − LCM does not have an automatic discovery mechanism. To publish a message on a channel, an LCM client packages it up as described in the next two sections, and then transmits it directly to the multicast group. If the size of the LCM message is small, then it can be transmitted using a single UDP datagram. Otherwise, the message must be fragmented and transmitted in several parts.To subscribe to a messaging channel, an LCM client joins the multicast group and inspects each message transmitted to the multicast group, retaining messages of interest and discarding the remainder.

3 Distance Calculation

The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles. The first table of haversines in English was published by James Andrew in 1805, but Florian Cajori credits an earlier use by José de Mendoza y Ríos in 1801. The term haversine was coined in 1835 by James Inman.

When working with GPS, it is sometimes helpful to calculate distances between points. But simple Euclidean distance doesn't cut it since we have to deal with a sphere, or an oblate spheroid to be exact. So we have to take a look at geodesic distances. There are various ways to handle this calculation problem. For example there is the Great-circle distance, which is the shortest distance between two points on the surface of a sphere. Another similar way to measure distances is by using the Haversine formula, which takes the equation

a=hav($\Delta\varphi$) $+ cos(\varphi 1) \cdot cos(\varphi 2) \cdot hav(\Delta\lambda)$

with haversine function

hav(θ) $= sin^2(\theta/2)$

and takes this to calculate the geodesic distance

distance=2$\cdot R \cdot arctan(\sqrt{a}, \sqrt{1-a})$

where,the latitude is denote as φ,

the longitude is denoted asλ, and

R corresponds to Earths mean radius in kilometers (6371).

Encoder Distance

Robot is moving forward or backward

Wheel diameter: d (mm) = 1000

Wheel circumference (C): d * π

No. of counts in 1 rotation of output shaft: N = 3840

Number of counts per 1 mm distance traveled (Y) = N / C

Or

Distance traveled per count = C / N (mm) OR = (C / N) * 1000 (microns)

Velocity consensus

The basic idea for consensus is that each constituent agent updates its control input based on the information states of its local neighbours in such a way that the states of all agents converge to a common value of certain physical quantities. It was implemented last semester Here the master robot sends its transmit its GPS Location and also it transfer its velocity so that velocity consensus can be achieved

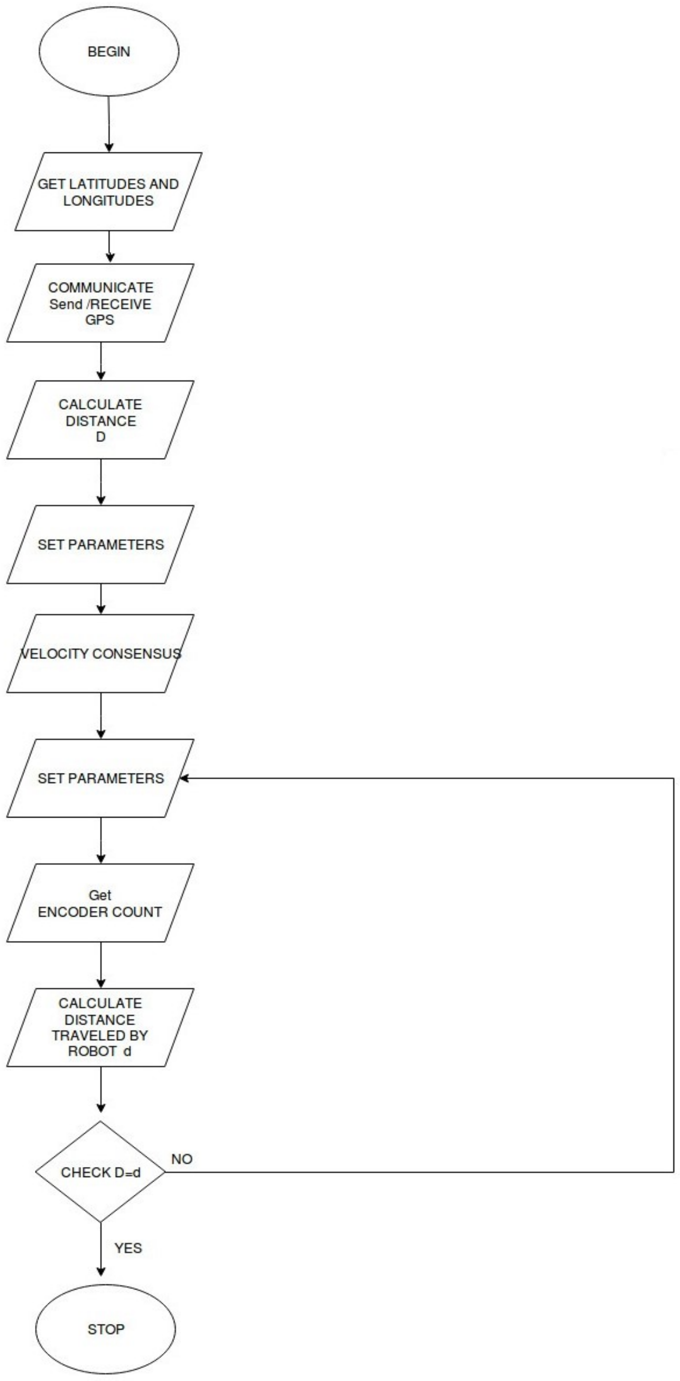


Figure 1: FLOW CHART OF GPS NAVIGATION PROGRAM

4 Implementation Of SLAM

The SLAM algorithm is supposed to simultaneously create a map of the vehicle's environment as well as calculating the position of the vehicle within this map. This data is to be broadcast to another ROS node on a PC, which will show the map and the vehicle's position in real-time in a GUI. Simultaneous localization and mapping is a problem where a moving object needs to build a map of an unknown environment, while simultaneously calculating its position within this map.

Robot Operating System

The ROS robotics framework, including affiliated open source autonomy libraries, all integrated into a ground robot equipped with sensors, (essentially called a work platform), is sufficient to present robot behaviors, robot manipulation, and robot (P2P) communication as part of our research conducted in the plant. With basic robotic knowledge now we realize how quickly robots may be used in an industrial environment, to help improve production flow and cost efficiency.

Operating System used : Ubuntu 14.04
Ros Version : INDIGO
LIDAR : RP LIDAR

LIDAR

LiDAR is a remote sensing technology which uses the pulse from a laser to collect measurements which can then be used to create 3D models and maps of objects and environments. LIDAR commonly stands for "light detection and ranging" or some variation of that. Like sonar and radar, LIDAR uses echolocation to develop an "image" of the devices surroundings. In order to create this image, LIDAR uses lasers to send out light pulses and tracks how long it takes for the lights to bounce back. LIDAR works well in both long range and short range, making it a great tool for self driving cars, which need to map obstacles and objects on the road far and near.where as Radar and sonar are both great tools that for creating a map of local surroundings, but both have limitations. Radar uses radio waves but has limitations in short distances. Sonar, using sound waves, is limited in long range uses. LIDAR works well in both long range and short range,

LiDAR mapping uses a laser scanning system can be integrated with Inertial Measurement Unit (IMU) or GPS which allows each measurement, or point in the resulting point cloud, to be georeferenced. Each 'point' combines to create a 3D representation of the target object or area. LiDAR maps can be used to give positional accuracy – both absolute and relative, to allow viewers of the data to know where in the world the data was collected and how each point relates to objects terms of distance. LiDAR data, in the form or a point cloud, can be used to map entire cities, enabling decision makers to accurately pinpoint structures or areas of interest in millimetre perfect detail. Features and objects such as road networks, bridges, street furniture and vegetation can be classified and extracted. LiDAR systems are most commonly used for surveying tasks.

Hector SLAM
Hector SLAM is an open source algorithm used for building a 2D grid map for the surrounding environment based on laser scan sensor (LIDAR). This algorithm locates the position of the robot based on scan matching and doesn't use the odometryor can use the odometry of the wheel which is the common method in other SLAM algorithms. The LIDAR manages it to perform the scan matching task to locate the robot fast and accurately due to its high update rate. Hector algorithm uses Gaussian- Newton minimization method witch is considered as update for Newton method, it has the advantages that second derivatives needn't to be computed. This method is used to find the optimum end point of laser scan relative to the build map by getting the transformation.However, this algorithm doesn't have closed loop, it closes its loop in the real world. The advantages of it are that it requires low energy to avoid the changes in the dynamic environments

For this task hector slam uses hector slam package, consisting of "hector mapping, hector map server , hector geotiff and hector trajectory server modules".

Hector pose estimation For estimating the pose of the platform hector slam use an Extended Kalman Filter (EKF) . To improve performance of the scan matching process, the pose estimate of the EKF is projected on the xy-plane and is used as start estimate for the optimization process of the scan matcher. Alternatively, the estimated velocities and angular rates can be integrated to provide a scan matching start estimate In the opposite direction, covariance intersection (CI) is used to fuse the SLAM pose with the full belief state. A simple Kalman measurement update would lead to overconfident estimates because it assumes statistically independent measurement errors Hector To achieve comparability between environment maps generated by different approaches, the GeoTIFF format is used as standard map. Using geo-reference and scale information, maps can be overlaid over each other using existing tools and accuracy can be compared. The hector geotiff package allows generating Ros 0x robot (Fire bird vi) compliant GeoTIFF maps which can be annotated through a plugin interface. Plugins for adding the path travelled by the robot. The node can run onboard a robot system and save maps to permanent storage based on a timer, reducing the likelihood of map loss in case of connectivity problems. using the hector geotiff node. Hector mapping is a SLAM approach that exhibit roll/pitch motion (of the sensor, the platform or both). It leverages the high update rate of modern LIDAR systems like the RP LIDAR and provides 2D pose estimates at scan rate of the sensors. While the system does not provide explicit loop closing ability, it is sufficiently accurate for many real world scenarios.

5 Overview of Frame Work

ROS is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic Platforms. A node is an executable that uses ROS to communicate with other nodes

ros0xrobot node :-The ros0xrobot provides ROS interface for most Nex Robotics robot bases in 0xseries (pronounced as Ox) supported by 0xRobotCpp library. Information from the robot base, and velocity and acceleration control, is implemented via a ros0xrobot node, which publishes topics providing data received from the robot's embedded controller by 0xRobotCpp library, and sets desired velocity, acceleration and other commands in robot when new commands are received from command topics.

rprplidar _ros :-The rplidar _ros is the ROS driver library for the rplidar sensor used on the robot to scan the environment. This enables using the LiDAR as an ROS node together with the other tools needed for the project.

Hector _slam :-This is a package including the hector _mapping node to create a 2D map based on the "high update rate of modern LIDAR systems" developed for ROS. It is used to create the 2D map of the environment in which the robot has driven. It is a bunch of launch files and configuration files to use the LiDAR sensor to create a SLAM map. Using these files makes it more easier and faster to create the map

Steps to follow for testing FIRE BIRD VI and Hector slam in real time

The system is installed with ROS and catkin_make the rplidar package, ros0xrobot, hector slam.After sucessful running of roscore

1.Launching the fire bird vi node

```
cd catkin _ws
devel/setup.bash
roslaunch ros0xrobot ros0xrobot _minimal.launch
```

2.Open new terminal and lauch the rplidar node

```
cd catkin _ws
. devel/setup.bash
roslaunch rplidar_ros rplidar.launch
```

3. On working pc open new terminal

```
cd catkin_ws
. devel/setup.bash
roslaunch ros0xrobot ros0xrobot.launch
```

4.Open new terminal and Launch the hector slam node

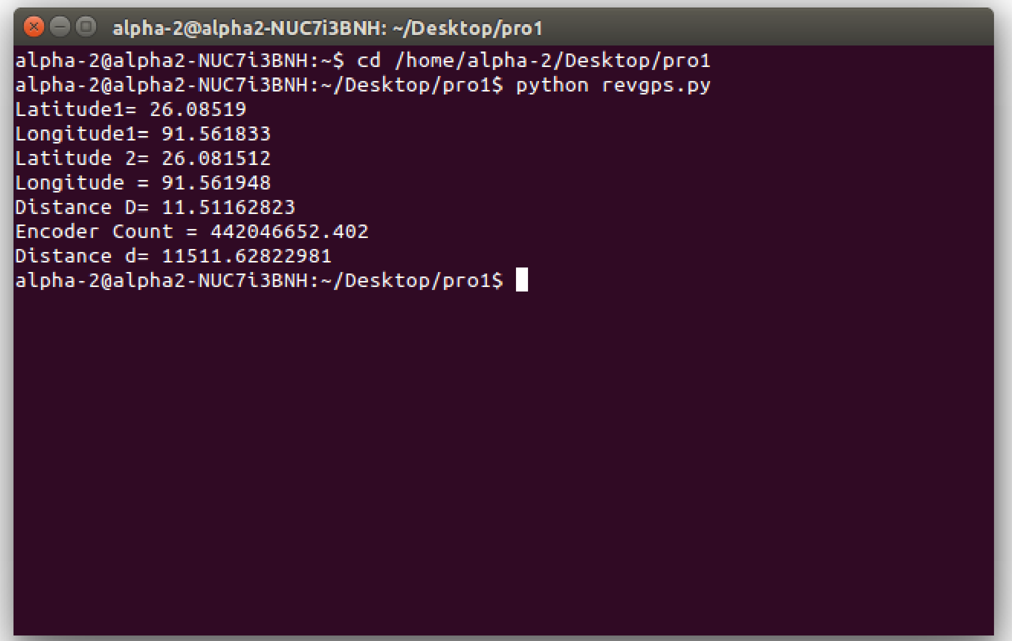
```
cd catkin_ws
. devel/setup.bash
roslaunch hector_slam _launch tutorial.launch
```

5.Saving the map The map can directly saved in RVIZ or by following the command rosrun map _server map _saver -f /home/nex/.../ros0xrobot/maps/map0

Chapter 4

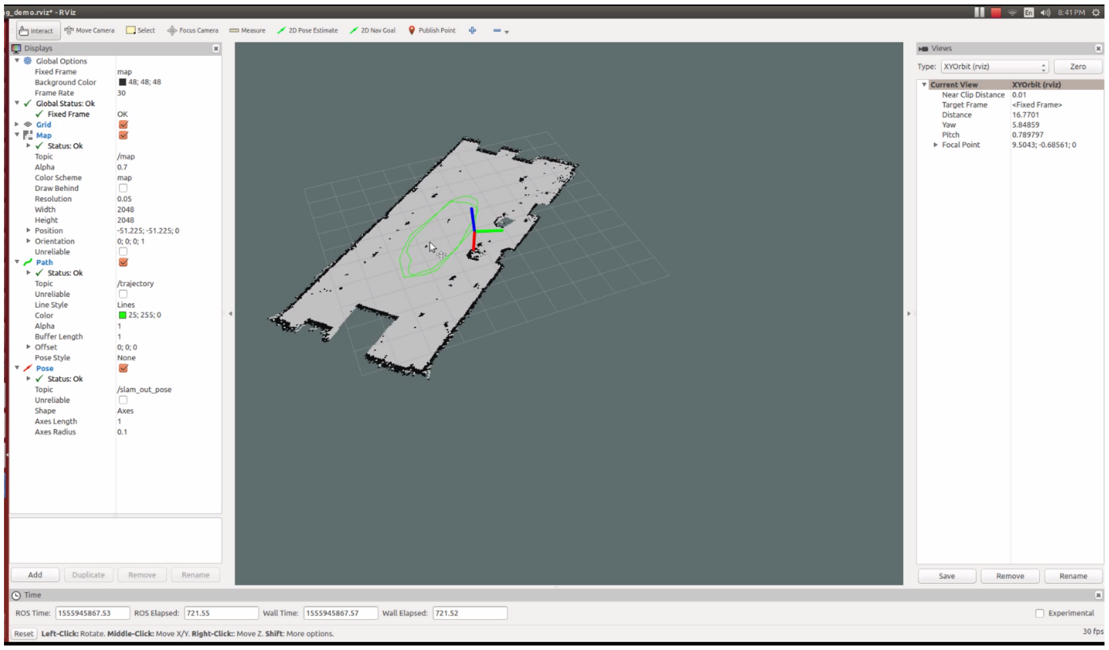
Results

Results for GPS based navigation is obtained as shown in the figure.successfully able to navigate the robot with the help of GPS and able to achieve velocity consensus.



```
alpha-2@alpha2-NUC7i3BNH: ~/Desktop/pro1
alpha-2@alpha2-NUC7i3BNH:~$ cd /home/alpha-2/Desktop/pro1
alpha-2@alpha2-NUC7i3BNH:~/Desktop/pro1$ python revgps.py
Latitude1= 26.08519
Longitude1= 91.561833
Latitude 2= 26.081512
Longitude = 91.561948
Distance D= 11.51162823
Encoder Count = 442046652.402
Distance d= 11511.62822981
alpha-2@alpha2-NUC7i3BNH:~/Desktop/pro1$
```

I have tested the hector slam module with the help rp-lidar in our robotics lab and obtained the following result in rviz. The rviz is mainly a visualization tool which is able to provide a live update of the map generated from the SLAM algorithm. Moreover, the trajectory of the vehicle on the map can also be visualized.Black spots are obtained on the map when the robot finds an object. The video for this project link is available at [7]



Global Options	
Fixed Frame	map
Background Color	■ 48; 48; 48
Frame Rate	30
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	✓
Map	✓
✓ Status: Ok	
Topic	/map
Alpha	0.7
Color Scheme	map
Draw Behind	☐
Resolution	0.05
Width	2048
Height	2048
Position	-51.225; -51.225; 0
Orientation	0; 0; 0; 1
Unreliable	☐
Path	✓
✓ Status: Ok	
Topic	/trajectory
Unreliable	☐
Line Style	Lines
Color	🟩 25; 255; 0
Alpha	1
Buffer Length	1
Offset	0; 0; 0
Pose Style	None
Pose	✓
✓ Status: Ok	
Topic	/slam_out_pose
Unreliable	☐
Shape	Axes
Axes Length	1
Axes Radius	0.1

Type: XYOrbit (rviz)

Current View	XYOrbit (rviz)
Near Clip Distance	0.01
Target Frame	<Fixed Frame>
Distance	16.7701
Yaw	5.84859
Pitch	0.789797
Focal Point	9.5043; -0.68561; 0

ROS Time: 1555945867.53 ROS Elapsed: 721.55 Wall Time: 1555945867.57 Wall Elapsed: 721.52 ☐ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift**: More options. 30 fps

Chapter 5

Conclusion and Future Work

This project is an initial step for autonomous driving vehicles, which are growing rapidly; The aim is to investigate automated for mobile robots . This report presents details for a ROS implementation of hector slam and gps based navigation .

In recent years, commercially available mobile robots, that operate in indoor environments, and outdoor have been increasing. People would like to prefer a mobile robot which would work for both indoor and outdoor. Present trends regarding applications of mobile robots structured indoor and outdoor environments justify the demand for a robust system with fewer errors would like to work on the mechanization that combines data from a GPS receiver, an IMU, a monocular camera, and multiple laser scanners. Along with the development of a high-level GPS/Inertial/Optical filtering architecture, and to implement a real-time hierarchical Visual SLAM system focusing on the localization of a vehicle in large-scale outdoor urban environments. It is exclusively based on the visual information provided by both a low-cost wide-angle stereo camera and a low-cost GPS and lidar scan

List of References

[1] M.Z.Al-Faiz and G.E. Mahameda,"*GPS-based Navigated Autonomous Robot,*" International Journal of Emerging Trends in Engineering Research,Volume 3, No.4, April 2015

[2]K.R.Memon, S.Memom, B.Memon and B.Memon,*"Real Time Implementaion of Path Planning Algorthim with Obstacle Avoidance for Autonomous Vechile,"* International Conference on Computing for Substainable Global Development,2016

[3]Albert S. Huang,E.Olson, and D.C. Moore,*"LCM: Lightweight Communications and Marshalling,"* International Conference on Intelligent Robots and Systems,October 2010.

[4]J. Meyer, P. Schnitzspan, S. Kohlbrecher, K. Petersen, O. Schwahn, M. Andriluka, U. Klingauf, S. Roth, B. Schiele, and O. von Stryk, *"A Semantic World Model for Urban Search and Rescue based on Heterogeneous Sensors,"* in RoboCup 2010: Robot Soccer World Cup XIV, 2011, pp. 180–193.

[5] M Eliwa, A. Adham, I. Sami and M. Eldeeb,*"A critical comparison between Fast and Hector SLAM algorithms,"* REST Journal on Emerging trends in Modelling and Manufacturing,Vol:3(2),2017 [6] *Fire Bird VI Hard Ware Manual.*

[7] *https://www.youtube.com/watch?v=CeoEG9ljZ9o*

[8] https://lcm-proj.github.io/