Navigation Algorithms

Bengisu YÜCEL[1]★
[1] *Paranavision*

09/07/2020

ABSTRACT
Since 1932, lots of people use navigation applications or devices to get one point to another point. In the beginning, people used that device on cars for going a place. Nowadays, navigation has been recently used on machines and robots for going somewhere another point autonomously. This paper shows how these navigation techniques and algorithms are advanced.

Key words: navigation – algorithm

1 INTRODUCTION

Navigation is using different areas, autonomously find the place where it is in. Detecting areas then it goes another point but how it can be possible which algorithms that we need for robots are doing this by people Navigation can be defined as the combination of the three fundamental competences.

The navigation ROS assumes that the mobile depended on desired velocity commands to achieve in the form of x velocity, y velocity, and theta velocity. It requires a planar laser mounted somewhere on the mobile base. The navigation stack was developed on a square robot, so its performance will be best on robots that are nearly square or circular.

Velocity and Acceleration In ROS navigation, know the translational and rotational velocity and acceleration.

Global Planner There global planners that ad here to nov core : : BaseGlobal Planner interface :: carrot planner, navfn and global planner. **Carrot Planner** The check given goal has an obstacle, and if so it picks on an alternative goal close to the original one.

Navfn and Global Planner Navfn uses Dijkstra's algorithm to find a global path with a minimum cost between the start-end point. A global planner is built as a more flexible replacement of navfn with more options. These options include support for A*, toggling quadratic approximation, toggling grid path.
Navigation algorithm consist of three main topics.

(i) Self Localization
(ii) Path Planning
(iii) Map Building and Map Interpretation

Vision - Based Navigation

- Representations of the environment
- Sensing Models
- Localization Algorithms

Trajectory Design and Motion Control In IpaBD mapping algorithm,using for path planning, scanning pixels that presents the surrounding environment of the robot can be described as a set of line segments as in where line segments are obstacles.
VHF algorithm is used for obstacle avoidence.

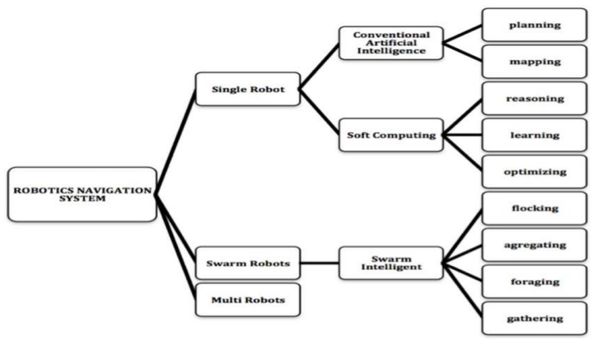


Figure 1. Intelligent robotics navigation system algorithms

2 LOCAL NAVIGATION METHODS

In local navigation techniques,sensors are usually employed to control the orientation and position of robot.

2.1 LIDAR

LIDAR works independently as compared to GPS system; therefore, it has the capability of mapping the environment. LIDAR can be used independently but when coupled with other sensors like GPS, Inertial navigation system, and camera, it gives improved results.

2.2 Vector Field Histogram

Vector Field Histogram (VFH) is another local navigation method used to solve the path planning problem for mobile robots. Obstacles are detected at a specific distance from the vehicle will apply repulsive force on a vehicle to move away from obstacles, and to draw the device toward goal point an attractive force is used. Generally uses a certainty grid-like radar screen, where obstacle found by the sensor

★ yucelbengisu@gmail.com

Figure 2. AGV System

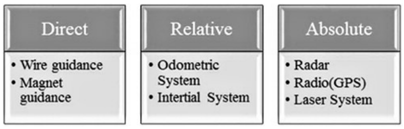


Figure 3. Division of navigation system.

will count up sum grid. This method is suitable for sparse moving objects.

2.3 Autonomous Guided Vehicles System

The AGV can be categorized into two methods, depending upon the number of load units it can carry simultaneously, that is, Single Unit Load or Multiple Unit Load. A load is considered as single unit, which carries vehicle from its pick-up point to drop-off point. In single unit load system, an empty, idle vehicle is chosen for task to perform, that is, to deliver load to assigned destination. Then vehicle travels from its initial position to pick-up position to acquire the load and then travels back to its drop-off destination. During the load drop-off assignment task, the vehicle is not interrupted by any other task. In multiple unit load system, the loaded vehicles are interrupted during their ongoing task to pick up additional loads.

- Wired Type
- Guide Type
- Gyro Based
- Vision Based

3 GLOBAL NAVIGATION METHODS

3.1 APF

Artificial Potential Field is to fill the robot environment with the artificial potential field in which obstacles are repelled using repulsive force and robot is attracted toward the goal using attractive force. The Potential function can be written for attractive and repulsive forces, as shown in equation

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{1}$$

Commonly used attractive potential used is,

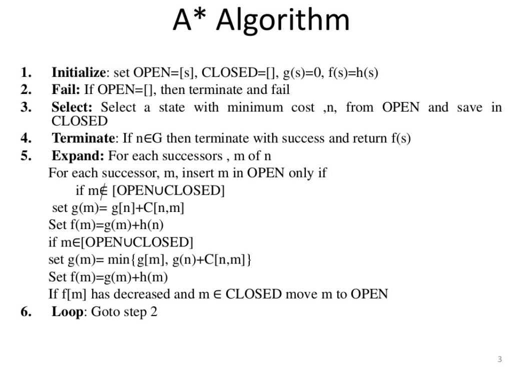
$$U_{att}(q) = \frac{1}{2}\zeta\rho(q.q_{goal}). \tag{2}$$

where ζ is gain, and (q,q_{goal}) is the distance between robot and the goal.
The attractive force is given as the negative gradient of attractive potential and the attractive force will be considered as zero as it approaches goal.

$$F_{att}(q) = -\nabla U_{att}(q) = \zeta(q_{goal} - q). \tag{3}$$

Following repulsive function can be used.

$$U_{rep}(q) = \frac{1}{2}\eta.(\frac{1}{\rho(q.q_{obs})} - \frac{1}{\rho_0})^2 \tag{4}$$


A* Algorithm

1. **Initialize**: set OPEN=[s], CLOSED=[], g(s)=0, f(s)=h(s)
2. **Fail:** If OPEN=[], then terminate and fail
3. **Select:** Select a state with minimum cost ,n, from OPEN and save in CLOSED
4. **Terminate**: If n∈G then terminate with success and return f(s)
5. **Expand:** For each successors , m of n
   For each successor, m, insert m in OPEN only if
   if m∉ [OPEN∪CLOSED]
   set g(m)= g[n]+C[n,m]
   Set f(m)=g(m)+h(n)
   if m∈[OPEN∪CLOSED]
   set g(m)= min{g[m], g(n)+C[n,m]}
   Set f(m)=g(m)+h(m)
   If f[m] has decreased and m ∈ CLOSED move m to OPEN
6. **Loop**: Goto step 2

Figure 4. A* Algorithm

In potential field approach an attractive field is created to reach the goal. In each time step, potential field is calculated at the robot above mentioned forces. The major problem with APF is that robot may trap local/global minima problem.

3.2 DIJKSTRA

The process continues until a better and optimal solution is achieved from one node to another. When the robot reaches the desired target, the robot is allowed to proceed to new location. Dijkstra algorithm is considered as graph searching method that solves the optimal path problem with non-negative edge path costs producing shortest path. It is used to find path cost from single point to single destination. Dijkstra is the smallest cost of each step it usually covers a large area of the graph.

3.3 A STAR

A* is a search algorithm that can also be used to path finding, seaches for unexplored location in graph.To sum up, A* is simply an informed variation of Dijsktra. A* is considered a " best first search" because it greedily chooses which vertex to explore next,

$$f(v)[f(v) = h(v) + g(v)] \tag{5}$$

Where h is heuristic, g is cost. A* more faster than Dijkstra. However, Dijkstra does not need to know the target node before hand. For this reason it’s optimal in cases where we can not estimate the idstance between each node and the target.
A* can be manipulated into other algorithms.

Manhattan Distance

$$h = abs(currentcell.x - goal.x) + abs(currentcell.y - goal.y). \tag{6}$$

Diagonal Distance

$$h = abs(currentcell.x - goal.x), abs(currentcell.y - goal.y). \tag{7}$$

Euclidean Distance

$$h = \sqrt{(cc.x - g.x)^2 - (cc.y - g.y)^2}. \tag{8}$$

4 FAST LANDMARK TRACKING LOCALIZATION ALGORITHM FOR THE MOBILE ROBOT SELF LOCALIZATION

It is include 5 directors.

(i) Localization
(ii) Mapping
(iii) SLAM
(iv) Path Planning
(v) Path Tracking

Localization Ability of a robot to know its poisiton and orientation with sensors.
Bayesian Filters :
- Kalman
- Histogram
- Particle filters
Mapping Mapping is the ability of a robot to understand its surrounding with external sensors such as LIDAR and camera. The grid mapping is 2D ray casting. 2D object clustering K-means algorithm.
SLAM (Simultaneous localization and mapping)
Simultaneous Localization And Mapping (SLAM) is the process by which a mobile robot can build a map of the environment and, at the same time, use this map to compute its location. which no accurate map is available.
+Kalman Filters +FastSLAM2.0
Path Planning Robot to search feasible and efficient path to the goal. The path has to satisfy some constraints based on the robot's motion model and obstacle positions.
Path Tracking Control to account for modeling error and other forms of uncertainty.

LIDAR SENSOR SMALL FIELD ROBOT

An evaluation of three different infield navigation algorithms LIDAR systems take accurate measurements of the scene, by reading distance from the sensor to the first obstacles and repeating this for the whole range. There are three different additional sensors:

- two digital cameras
- an IMU unit
- a LIDAR sensor

The data from the sensors are processed by two onboard computers that rely on the distance measurements from the LIDAR sensor. The experimental robot is equipped with the SICKTIM310 LIDAR, with a 270° area at a 1° angular resolution.
Minimal Row - Offset based algorithm Robot standing in the middle of the row. Each double-sided arrow represents a measurement of the sensor, where the actual number of readings is 30 on each side. The sensor returns the distance for each degree it measures.

$$Offset = dr - dl. \tag{9}$$

$$Orientation = \frac{Offset}{dr - dl}. \tag{10}$$

By the new value orientation which also corresponds directly to how much the wheel should turn in radians. All three, were implemented as part of robotic operating system (ROS).
Least- Square Fit Approach Algorithm
Designed to navigate the robot between two walls that are parallel to

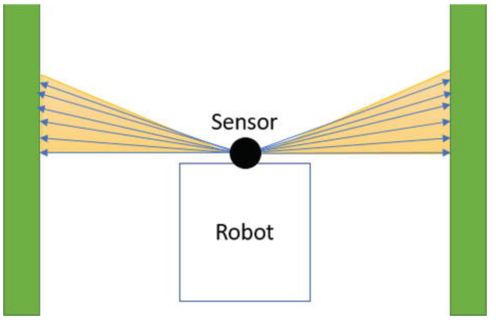


Figure 5. Minimal Row Offset- Based Algorithm

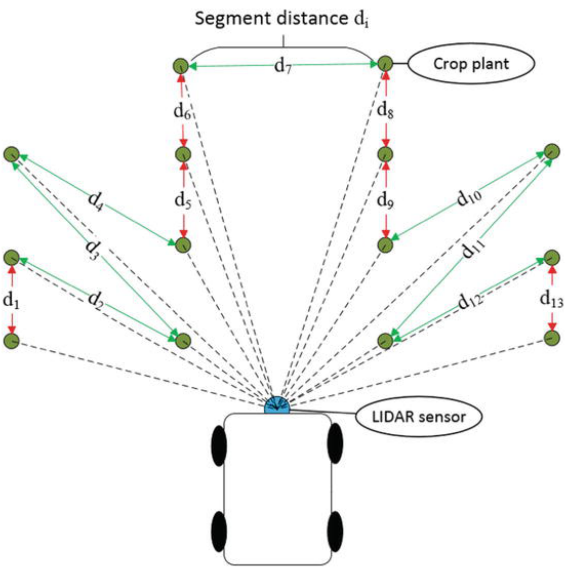


Figure 6. The way the robot calculates each segment of distances between every two sequential points from LIDAR sensor.

each other this being either an artificial barrier e.g. walls or real crop lines, such as maze plants.
Performed Steps in Each Cycle
A. Sensor reads the data.
B. The algorithm makes it possible to set how many points should be included for each count for each side that corresponds to how many degrees will use in the subsequent steps of the algorithm.
C. A linear fit is used. We need to know some parameters xsum, xsquaresum, ysum, ysquaresum,yxsum. Once these parameters are known. The slope (k) and y-intercept (n) can easily be calculated.
D. With the calculated slopes and y intercepts a cross point is calculated where those two lines cross each other.
K_L= Slope of the left side n_L = y intercept of the left side k_D = slope of the right side n_D = y intercept of the right side

$$xL = n2 - \frac{n}{K1} - k2. \tag{11}$$

$$yL = K1 * xL + n1. \tag{12}$$

Wrap up,

- Receive data from the sensor
- Filter the data and divide into two sides (lines).
- Calculate the slopes and y- intercepts of both lines
- Calculate the line intersection

Table 1 Summary of typical control problems and cost functions [15, 31]

Problem	Cost function
Minimum-time: To transfer a system from an arbitrary initial state to a specified final state in minimum time	$J = \int_{t_0}^{t_f} dt = t_f - t_0$
Terminal control: To minimize the deviation of the final state of a system , $x(t_f)$, from its desired value, $r(t_f)$	$J = \lVert x(t_f) - r(t_f) \rVert_H^2$
Minimum-control-effort: To transfer a system from an arbitrary initial state to a specified final state with minimum expenditure of control effort	$J = \int_{t_0}^{t_f} \sum_{i=1}^{m} \beta_i \lvert u_i(t) \rvert dt$ $J = \int_{t_0}^{t_f} \lVert u(t) \rVert_R^2 \, dt$
Tracking: To maintain the system state, $x(t)$, as close to the desired state, $r(t)$, in the interval $[t_0, t_f]$	$J = \lVert x(t_f) - r(t_f) \rVert_H^2 + \int_{t_0}^{t_f} \left(\lVert x(t) - r(t) \rVert_{Q(t)}^2 + \lVert u(t) \rVert_{R(t)}^2 \right) dt$

Figure 7. Example Cost Functions

- Calculate the rotational direction
- Once the information about the intersection from the two lines is known, the position of the robot is calculated.

5 SIMULATED ENVIRONMENT USING ROS

The role of a navigation algorithm is to generate safe paths through terrain while achieving specific goals. The navigator uses sensor data to evaluate terrain and locomotion components to execute items.

5.1 Terrain Traversability Analysis

Sensor data is converted into a model. Terrain traversability analysis that is employed at a preceding stage as a means to effectively and efficiently guide the task of motion planning.

5.2 Cost Functions

Transform these models into a form that can be used for planning. Uses for performence measure.

5.3 Action Selectors

Using this planning space to determine how the robot should move.

6 CONCLUSIONS

Navigation algorithms and methods are changeable to using areas. According to these algorithms we can use the most fruitful algorithm.

REFERENCES

https://www.tandfonline.com/doi/full/10.1080/23311916.2019.1632046
http://robots.stanford.edu/papers/Montemerlo03a.pdf
https://www.sciencedirect.com/science/article/pii/S1474667017347031
https://atsushisakai.github.io/PythonRobotics/
https://arxiv.org/pdf/1808.10703.pdf
https://benalexkeen.com/implementing-djikstras-shortest-path-algorithm-with-python/
https://www.geeksforgeeks.org/a-search-algorithm/
https://en.wikipedia.org/wiki/A*searchalgorithm
https://www.researchgate.net/figure/Intelligent-robotics-navigation-system-algorithms-fig1-320083259
https://dergipark.org.tr/tr/download/article-file/848182
://planning.cs.uiuc.edu/node615.html
https://en.wikipedia.org/wiki/Robot-navigation
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7070314/
http://robots.stanford.edu/papers/Thrun03g.pdf
https://www.slant.co/versus/11584/11585/ dijkstra-s-algorithmvsa-algorithm
https://ieeexplore.ieee.org/abstract/document/1678144
https://ieeexplore.ieee.org/document/1638022

This paper has been typeset from a TEX/LATEX file prepared by the author.