IMPLEMENTATION OF TANGENT BUG ALGORITHM WITH A NON-HOLONOMIC WHEELED MOBILE ROBOT PIONEER 3-DX

Hasan İhsan Turhan
Mechatronics, Robotics and Control Laboratory
Electrical and Electronics Engineering Department
Middle East Technical University
Ankara/Turkey
hasan.turhan@metu.edu.tr

Abstract

In this study, tangent bug algorithm is implemented on a wheeled mobile robot (WMR) Pioneer 3-DX. Tangent bug algorithm is a kind of robot motion planning algorithm, which has the limited knowledge of environment. Also the used robot is a non-holonomic vehicle. With these constraints, implementation is made by using mobile robot simulator MobileSim for different obstacle configurations in the environment and the results are presented.

1. Introduction

According to the configuration space, and start and goal positions of the robot, generating a continuous path from start position to the goal position is known as a robot path planning or motion planning. It is a very popular research topic for especially wheeled mobile robots.

The robot path planning problem can be classified into two main categories: First one is path planning with complete information and the second one is the path planning with incomplete information [1].

At the first approach, robot has complete information about the environment. It knows start and goal configurations. Also, it knows where the obstacles and passages are. For example, piano movers problem.

At the second approach, robot has only limited-knowledge such as distance sensor or tactile sensor information in addition to the start and goal positions. With this limited information robot tries to find a continuous path from start to goal.

This paper presents the one of the limited information path planning algorithm "Tangent Bug". It contains the algorithm, implementation and constraints about implementation. This paper consist of the following sections: At the second part, tangent bug algorithm is briefly introduced. At the third part, problem definition and the implementation specifications are given with implementation constraints.. Then, the WMR Pioneer 3-DX robot and the simulation environment MobileSim are explained at the fourth part. After that, simulation results are presented at part five. Finally this paper ends up with conclusion and future work.

2. Tangent Bug Algorithm

The tangent bug algorithm is one of the limited information path planning algorithms. At tangent bug algorithm the robot only knows start and goal positions. It has no information about environment i.g. where the obstacles are. But, it can detect the obstacles with distance sensor; however, sensor has limited range.

2.1. Sensor Modeling

The distance sensor is modeled with the raw distance function $\rho = \mathbb{R}^1 x S^1 \rightarrow \mathbb{R}$. For $\theta \epsilon S^1$, the value $\rho(x,\theta)$ is the distance to the closest obstacle along the ray from x at an angle θ. [2]

$$\rho(x,\theta) = \min_{\lambda \epsilon [0,\infty]} d(x, x + \lambda[cos\theta, sin\theta]^T) \quad (1)$$
$$such\ that\ x + \lambda[cos\theta, sin\theta]^T \epsilon \bigcup_i WO_i$$

The robot can detect the obstacles by detecting the discontinuities in the raw distance function i.e. sensor model.

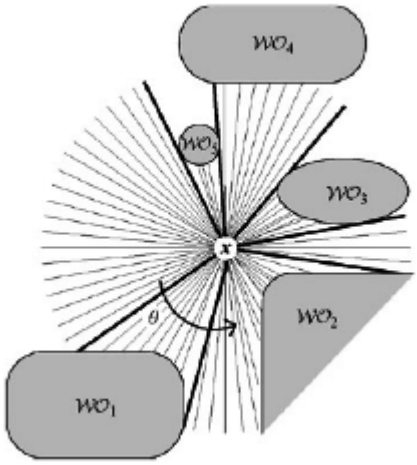

Figure 1 - Range data acquisition of a point robot

If sensor rays don't intersect an obstacle raw distance function gives the maximum value. But if it intersects an obstacle it gives distance to that intersection point. For example, if we examine WO_1 obstacle at the figure 1, we see that sensor rays gives maximum value until they intersect the boundary, after passing the obstacle they again gives the maximum value. Hence, discontinuities exist in the raw distance function and robot detects the obstacles according to this continuities.

2.2. The Algorithm

The tangent bug algorithm consists of two main behaviors in distinct from the other bug algorithms:

1-Motion to goal behavior
2-Boundary following behavior

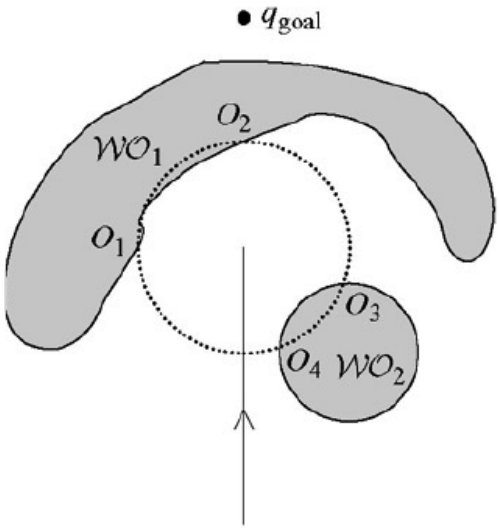

Figure 2 – Motion to goal behavior

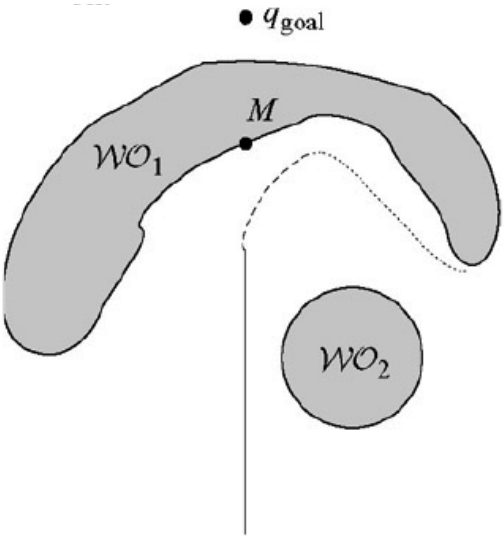

Figure 3 - Boundary following behavior

Firstly robot starts to go towards to the goal. When an obstacle is sensed, it starts to boundary following behavior, until it can go to the goal again. The whole tangent bug algorithm and definition of the terms of this algorithm are given below.

x : current position of the robot.

q_{start} : start position.

q_{goal} : goal position.

O_i : i[th] discontinuity (tangent) point as in shown in the figure 2.

$d(x, O_i)$: distance to the i[th] tangent point from robot's current position

$d(O_i, q_{goal})$: distance to the goal from i[th] tangent point

$d_{reached}$: shortest distance between *blocking* obstacle and goal (or my distance to goal if no blocking obstacle visible)

$d_{followed}$: shortest distance between the sensed boundary and the goal

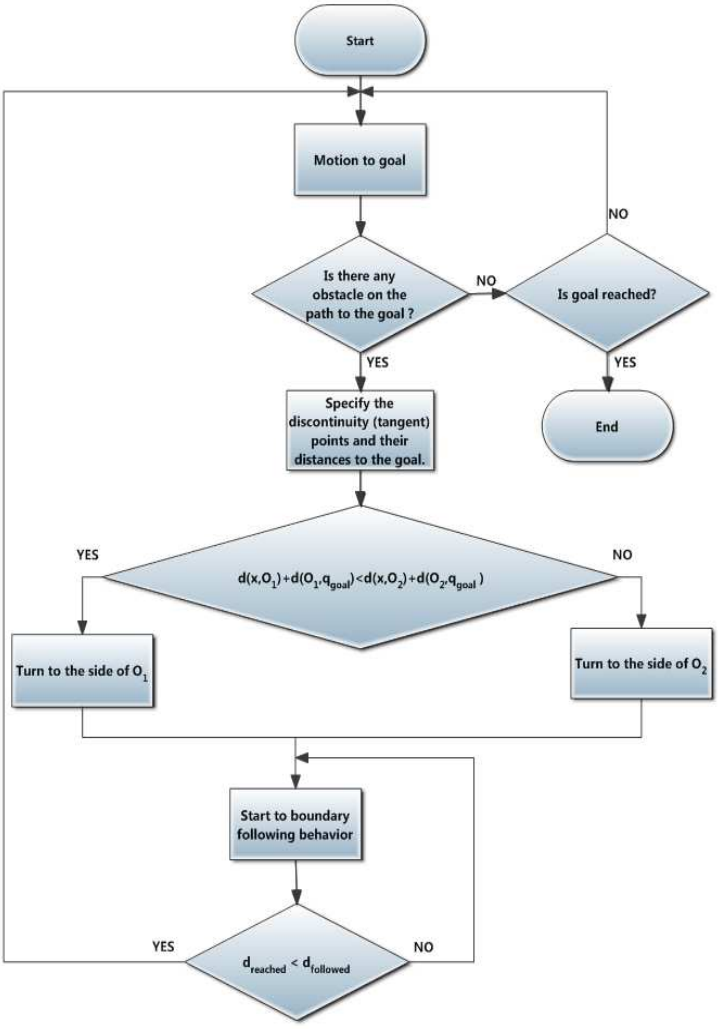

Figure 4 - Flowchart of the Tangent Bug Algorithm

3. WMR Pioneer 3-DX

Pioneer 3-DX robot, is a research robot, which is produced by Mobile Robots Inc. Pioneer P3-DX robot has an internal computer, which has Pentium III processor and operates with Linux operating system. And this computer can be reached and controlled with wireless connection from an external computer. For sensing, it has 16 ultrasonic and 1 laser sensors, and a camera. For collision avoidance it has bumpers. Also it has 2D gripper for grabbing objects. Besides these, it has 2 drive wheel and 1 free wheel, in order to realize movements.

The devices available for Pioneer 3 robots are listed in Table-1 and illustrated in Figure-5 and Figure-6 below, which show two hardware configurations. The devices include a SICK LMS200 laser range finder, sonar range finding sensors, a Canon VC-C4 pan-tilt-zoom camera, grippers, bumpers, and five-degree-of-freedom (5D) arm. Table-1 shows that which devices may be controlled by each of the programming interfaces, ARIA and Player, and considers both real and simulated robots. (MobileSim is the 2D simulator for ARIA and Stage is the 2D simulator for Player) ACTS is a blob finding software package compatible with ARIA.

Device	Description	ARIA (Real robot)	Mobile Sim 0.5.0
Sick LMS200 Laser (front 180°)	Range finding laser sensor	√	√
Pioneer Sonar Ring (8 front, 8 rear)	Range finding sonar sensors	√	√
Canon VC-C4 ptz camera	Pan-tilt-zoom camera	√	
Simple blob finding device	For tracking colour		
ACTS blob finder	For tracking colour	√	
Pioneer bumper pads (5 rear)	Collision detection	√	√
Pioneer 2D gripper	2 degrees freedom	√	
Pioneer 5D arm	5 degrees freedom	√	

Table 1 - Available Devices for Pioneer 3 Robots

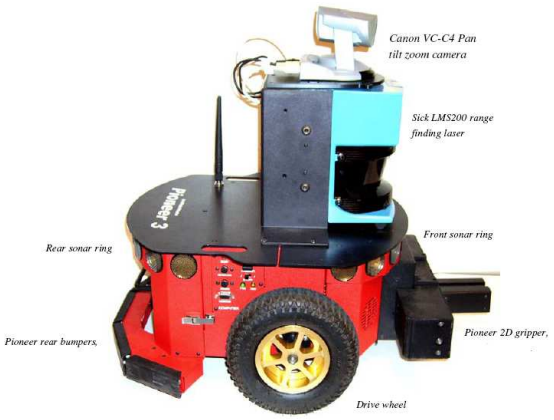

Figure 5 - Pioneer 3-DX Robot

Figure 6 - Pioneer P3-DX Robot with 5D Arm

The configuration of the robot is illustrated below:

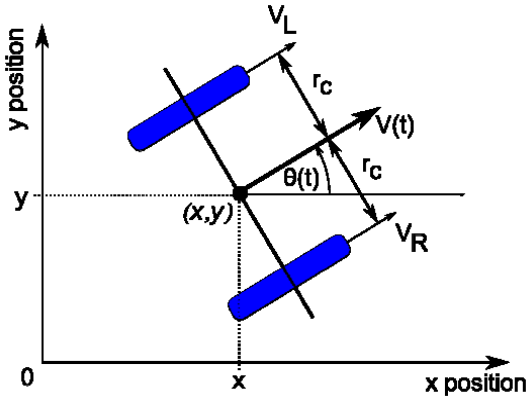

Figure 7 - Configuration of Pioneer 3-DX

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

$$v_L = r_w . \omega_L$$

$$v_R = r_w . \omega_R$$

$$v(t) = \frac{v_L(t) + v_R(t)}{2}$$

$$\omega(t) = \frac{v_R(t) - v_L(t)}{r_c}$$

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos(\theta(t)) . v(t) \\ \sin(\theta(t)) . v(t) \\ \omega(t) \end{bmatrix}$$

MobilSim is a simulation environment for Pioneer robots. The simulated pioneer robot has laser range finding sensor, sonar range finding sensor and bumpers, as the real pioneer robot has. And it can be programmed same as the real robot. Furthermore, you can create a simulation environment with Mapper3 drawing program for MobileSim simulator.

4. Problem Definition and the Implementation Specifications

Problem definition is given below:

- A bounded, 2D rectangular workspace modeled as two continuous coordinate variables,
- A configurable set of polygonal obstacles, not necessarily convex. In fact, you should include non-convex obstacles in your examples,
- "Virtual" sensors for measuring the positions of the robot and the goal
- A laser range sensor with a predetermined range limit (initialized before starting the simulation), returning an array of distance values for a discrete set of angles (for example, 180 equally spaced samples spanning 180 degrees) from a limited, 180 degree field of view centered in the direction of the goal,
- The ability to accept either velocity, or discrete displacement commands from a "controller", that can be used to implement a particular planning algorithm.

According to the problem definition the algorithm is implemented with WMR Pioneer 3-DX in the MobileSim robot simulator. There are some constraints, which is caused by both non-holonomic robot and the problem definition.

First of all the robot is non-holonomic i.g. it can't moves towards all direction. In order to go one direction, robot has to set its heading to that direction.

Secondly, robot has limited range sensor. The range of the laser sensor for Pioneer is 20m. ; but, it is set to the 1m., in order to limit the information for robot. Also, laser range finder only can scan 180 degree front view. This sometimes causes problems. For example, for sharp corners robot slog to find the obstacle boundary again; because, the boundary can be left behind the robot.

5. Simulation Results

The algorithm is tested with Pioneer 3-DX in MobileSim simulator. For two different configuration spaces.

The video of first configuration can be reached from the link below

http://www.youtube.com/watch?v=clQIAJ_Da-s&feature=youtu.be

The video of second configuration space can be reached from the link below:

6. Conclusion

With this work, the tangent bug algorithm was implemented on a non-holonomic WMR and the results are presented. The implementation of algorithm works for the rectangular configuration spaces.

Because of the non-holonomic constraints of the robot, I only can implemented and tested the algorithm for rectangular obstacles.

It is seen that the tangent bug algorithm can also be implemented on non-holonomic robots.

7. Future Work

For future work, I will improve and test the implementation of tangent bug algorithm. For more complex configuration spaces like 2D map of the real room of a home.

After that step, I will implement the algorithm on a real Pioneer 3-DX robot and present the results as conference paper.

References

[1] V. J. Lumelsky and A. A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst obstacles of arbitrary shape. Algoritmica, 2:403-430, 1987

[2] Choset H., Lynch K. M., Hutchinson S. Kantor G., Burgard W., Kavraki L. E., Thrun Sebastian. Principles of Robot Motion. MIT Press