American University of Sharjah

ELE 494-08: Autonomous Robotic Systems

Project Report: CTE Document 1

Robot Way-Point Navigation using Ackermann Steering with Obstacle Avoidance

Submitted By:
Taha Ameen

Student ID:
@00066555

Submitted To:
Dr. Shayok Mukhopadhyay

November 29, 2019

CONTENTS

LIST OF FIGURES

Robot Way-Point Navigation using Ackermann Steering with Obstacle Avoidance

Taha Ameen ur Rahman

Abstract—**This document is a continuous time evolution (CTE) document submitted to course instructor Dr. Shayok Mukhopadhyay, in partial fulfillment of the project requirement for the course ELE 49408 - Autonomous Robotic Systems. The objective of this report is to present a proposal along with information on the expected outline of the project. This project is a practical and hardware oriented project, which deals with way-point navigation of robots. The objective of the project is to develop a robot that is capable of navigating from one point to another, given GPS coordinates when selected from a map, and doing so using the Ackermann steering method, with obstacle detection and avoidance. This report outlines the objectives and expected procedural methodology that will be followed over the course of the semester to achieve this goal.**

I. INITIAL GOAL STATEMENT

The project is a hardware-oriented endeavor in building an autonomous robot that is capable of navigating between two points that are specified as GPS coordinates on a map. The goal is to engineer a user-friendly robot that achieves the navigation objective specified above, with obstacle detection and avoidance. The robot is expected to be capable of sensing its environment for potential obstacles, and re-routing its path to avoid any accidental collisions. Lastly, an additional goal that will be pursued if time permits is a continuous extension of the way-point navigation in order to allow the robot to follow a path as marked by the user on a graphical interface.

II. TEAM FORMATION

The team formation process has been done and submitted as part of HW0. The team consists of three undergradaute students, with details provided below:

Taha Ameen ur Rahman, @*00066555*
Hazem Sharaf, @*00062852*
Ahmed Yasser, @*00063487*

III. TEAM STRENGTHS AND WEAKNESSES ASSESSMENT

The team has identified the following as strengths and weaknesses of the individual memebrs:

A. Individual Strength and Weakness Assessment

1) Taha Ameen:

Strengths

- Strong mathematical background for analysis of performance and theoretical modeling.
- Solid background in MATLAB coding and numerical techniques.

Weaknesses

- Able to only allocate limited time for project.
- Not very experienced with hardware-oriented projects.

2) Hazem Sharaf:

Strengths

- Good at developing and debugging algorithms.
- Sound experience with programming.

Weaknesses

- Unorganized at documentation.
- Limited communication skills.

3) Ahmed Yasser:

Strengths

- Very experienced with hardware oriented projects.
- Good programming skills.

Weaknesses

- Unorganized at documentation.
- Time constraints.

B. Team Strength and Weakness Assessment

Strengths

- Complementary skills in hardware, software and interfacing.
- Motivated to persevere towards project completion.
- Reasonable experience with using Arduino / Infrared Camera.

Weaknesses

- Difficult to find common meeting time due to very different schedules.
- Limited experience with using MATLAB on Raspberry Pi, and interfacing robotic systems.

The team formation process was a result of extensive deliberation to ensure a balance between the strengths and weaknesses of individual members. It is expected that the skill-sets of each member complements the others. The roles of each member were accordingly delineated after a team meeting, and are listed in Section V.

IV. BROAD OBJECTIVES

This section discusses the objectives of the project, and provides an insight into the team plan to be implemented in order to achieve the goal. The following list outlines the expected objectives to be fulfilled over the course of the two month venture.

- To build the physical structure and framework of a general navigation robot.
- To translate the selection of a point on a map on a mobile phone to workable GPS coordinates that are communicated to the robot.

- To implement an efficient navigation mechanism such as Ackermann Steering to allow way-point navigation of the robot.
- To equip the robot with a working obstacle detection and avoidance module, so that the robot can alter its route to avoid collisions with obstacles.
- To embed microcomputers/controllers in the structure to handle the software and hardware interface of the robot.
- To serially iterate the way-point navigation procedure to obtain a discrete string which can be interpolated to match a predefined path that the robot is required to follow.

V. TEAM MEMBER ROLES

This section deliberates on the roles of each team member in the project, based on the strengths and weaknesses discussed previously.

A. *Taha Ameen*

- Develop the MATLAB based code for Ackermann Steering.
- Implement MATLAB-based simulations to verify and test the system for a wide variety of cases.

B. *Hazem Sharaf*

- Integrating the obstacle avoidance module by developing the code to interface the sensing with the decision making.
- Develop the algorithmic basis for translating and communicating the GPS coordinates from the phone to the MATLAB-based program running on the Raspberry-Pi on the robot.

C. *Ahmed Yasser*

- Procure and construct the structure and framework of the robot body, with emphasis on hardware.
- Establish a means of communication between the sensors, micro-controllers and microcomputer on board.

VI. THEORETICAL BACKGROUND

This section presents the theoretical background behind Ackermann steering in order to better motivate the choice of the navigation mechanism. We begin by considering a robot with center of mass (x, y), such that the objective is to move (x, y) to the point (x_r, y_r). Ackermann steering procedure introduces another point (x_h, y_h) such that $\lim_{t \to \infty} (x_h, y_h) = (x_r, y_r)$. Let us also introduce an inertial frame of reference, with origin at (x, y) and a mutually orthogonal basis set, such that one of the basis vectors is collinear with the direction of heading of the robot. We then introduce θ as the angle made by the heading with respect to the horizontal, and θ_d as the angle made by the horizontal with respect to the vector that points from the current position to the final desired position. Hence we have

$$\theta_d = \arctan \frac{y_r - y}{x_r - x} \tag{1}$$

Let v represent the linear velocity of the robot, and ω represent its angular velocity. Robot navigation is governed by the following set of non-holonomic equations:

$$\dot{x} = v \cos(\theta) \tag{2a}$$

$$\dot{y} = v \sin(\theta) \tag{2b}$$

$$\dot{\theta} = \omega \tag{2c}$$

If we were interested in Heading control, our objective would be to make θ approach θ_d so that the robot can then follow a straight line to reach $(x_r, y_r$. However, the process of controlling the angular velocity, followed by the linear velocity requires the tuning of many gains assuming the utilization of a PID controller. The Ackermann steering method provides an efficient method to overcome this problem, albeit with consequences of its own.

Our control strategy is to ensure that

$$\lim_{t \to \infty} (x_h, y_h) = (x_r, y_r) \tag{3}$$

Observe the following

$$x_h = x + h \cos(\theta) \tag{4a}$$

$$y_h = y + h \sin(\theta) \tag{4b}$$

Letting $\dot{x}, \dot{y}$ represent the time derivative of x, y respectively, we have

$$\dot{x_h} = \dot{x} - h \sin(\theta) \dot{\theta} \tag{5a}$$

$$\dot{y_h} = \dot{y} + h \cos(\theta) \dot{\theta} \tag{5b}$$

By substituting the non-holonomic constraints, we get

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -h \sin(\theta) \\ \sin(\theta) & h \cos(\theta) \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{6}$$

Let e represent the error signal, which is a measure of the difference between the actual value and the desired value. Hence,

$$e = \begin{bmatrix} x_h - x_r \\ y_h - y_r \end{bmatrix}, \quad \dot{e} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{7}$$

Next, we let R to be the matrix in (6) and write

$$\dot{e} = R \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{8}$$

Lastly, we pick

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = -R^{-1} K e, \quad K = \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix}$$

Here $K_1, K_2 > 0$, and mathematical manipulation yields

$$\dot{e} = -K e \tag{9}$$

Since $-K$ is a diagonal matrix with negative entries, it is rest assured that the eigenvalues of this matrix are negative and real, and hence the system is LTI stable. This allows us to conclude that $\lim_{t \to \infty} e = 0$. This implies that $(x_h, y_h) \to (x_r, y_r)$. It is also worth noting that the R matrix is invertible when $h \neq 0$. This requirement is a necessary condition for Ackermann steering. However, the choice of (x_h, y_h) is entirely up to the user, and it can be chosen sufficiently close to (x, y) without affecting the invertibility.

The Ackermann steering method is a powerful technique because it guarantees the stability of the system. It is clear that the system will always converge provided that the constraints on $K_1; K_2$ are met. Ofcourse, this comes with the drawback of losing control over the path being taken. However, with sufficiently small space step, it is possible to ensure that the robot does indeed follow a smooth path.

VII. Progress

This section presents the progress of the team until midterm I.

A. Software

In this section, we illustrate the implementation of the Ackermann Steering model. Note that the code provided below is an initial attempt at implementing the model, and requires significant improvements and testing. However, it is included in the report as a proof of concept and progress.

Listing 1: MATLAB Code: Ackermann Steering: Attempt 1.m

```matlab
% Script to implement Ackermann steering for Ground Robot Way-Point
% Navigation, Attempt 1

% Variables:
%% Parameters:
%   (x,y) : Initial position coordinates
%   theta : Orientation with respect to x axis measured ccw
%   h     : Rectilinear distance between (x,y) and arbitrary point

%% Inputs to Function
x = 1;      % Initial x Coordinate
y = 1;      % Initial y Coordinate
theta = 0; % Initial Orientation

xh = 1.1;   % Choice of Point close to x
yh = 1.1;   % Choice of Point close to y

xr = 5;     % Desired Output x coordinate
yr = 0.6;   % Desired Output y coordinate

%% Computing
h = 0.2;    % Step size in space
dt = 0.1;   % Step size in time

t = 0:dt:10; % Time Vector

v = 0.1;
omega = 0.1;

Y = [v; omega]; % Vector of Interest
K = [0.5 0; 0 0.5]; % Matrix as given in HW 3

%% Iterative Calculations for linear and angular speed
for i = 1:length(t)
xh(i) = x+h*cos(theta);
yh(i) = y+h*sin(theta);
tvec = t(1:i);

R = [cos(theta) -h*sin(theta); sin(theta) h*cos(theta)];
XF = R*Y;
xh_dot = XF(1);
yh_dot = XF(2);

Y = -inv(R)*K*e;
e = [xh(i) - xr; yh(i) - yr]
e_dot = -K*e;

% Updating the x and y coordinates
x = xh(i);
y = yh(i);
end
%% End of Code
```

B. Hardware

In this section, we present a basic overview of the proposed approach towards the hardware build-up, by illustrating a connection diagram of the components that will be used in obstacle avoidance. Figure 1 displays the connections between an Infrared (IR) sensor that senses obstacles and an Arduino that controls a H-Bridge to drive motors accordingly.

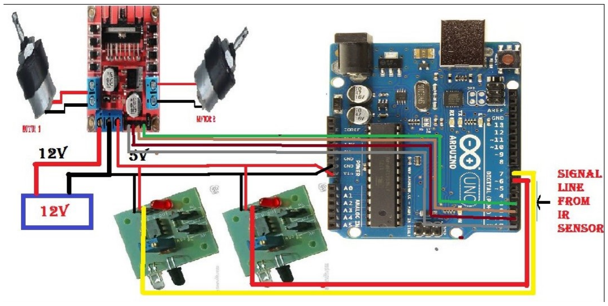

Figure 1. Hardware: Connection Diagram

The hardware and related equipment will be procured from the laboratories at AUS, in addition to utilizing a privately owned raspberry-pi module.

C. Team Dynamics

In this section, we present proof of team commitment and contribution through conversations and pictures of meetings.