IDK something about controlling autonomous fleets

Kyle Montemayor, Ian Wilmoth, and Willian Zheng

***Abstract*—Controlling fleets of autonomous vehicles presents a challenge in that overall all of the vehicles represent a very large system, of which a control law is hard to develop. However, given our increasing propensity towards autonomy, being able to control and model a fleet of such systems is crucial to our society.**

I. Introduction

Vehicles with a differential drive are very common in a small form factor. Due to their ability to have a very low turning radius, they are very maneuverable. Unfortunately they are not well suited to larger tasks as they require additional drive motors to be most efficient.

For this paper, a procedure for controlling a fleet of the autonomous differential drive vehicles(ADDV) such that they can locate each other, form a single file, and follow a path in the formation. This will be accomplished in 4 parts, simulating a ADDV, controlling an ADDV to a desired end point, controlling an ADDV along a desired path, having two ADDVs follow each other, and finally, having a fleet of ADDVs follow each other.

II. Autonomous Differential Drive Vehicles

The vehicle's that are studied in this paper have a drive mechanism as shown below.

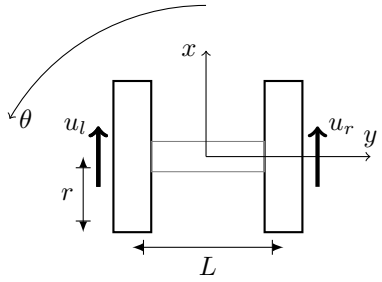

Fig. 1. Diagram of ADDV drive mechanism

Where the controls, given by u_r, u_l, are for the left and right motors, r is the radius of the wheels, L is the distance between the wheels, (x, y) are Cartesian coordinates of the center of the ADDV, and θ is the angle between r and x. All together, the state vector $(x(t), y(t), \theta(t))$ is the pose of the ADDV at time t.

A. Control equations

With the given definitions we can define the equations of motion of the as follows

$$\dot{x} = \frac{r}{2}(u_r + u_l)cos(\theta) \tag{1}$$

$$\dot{y} = \frac{r}{2}(u_r + u_l)sin(\theta) \tag{2}$$

$$\dot{\theta} = \frac{r}{L}(u_r - u_l) \tag{3}$$

Clearly, the equations for $(\dot{x}, \dot{y})$ are non-linear, but if we define

$$\dot{r} = \sqrt{\dot{x}^2 + \dot{y}^2} = \frac{r}{2}(u_r + u_l) \tag{4}$$

and use a polar coordinate system originating from the origin of the ADDV then we have a completely linear system of equations given by 3, 4.

This can be a useful coordinate system as we can define all desired end points as polar coordinates originating from ADDV. In order to additionally simplify our controls, we will define our translational and rotational movement as follows

$$\dot{r} = \frac{r}{2}(u_r + u_l), u_r = u_l \tag{5}$$

$$\dot{\theta} = \frac{r}{L}(u_r - u_l), u_r = -u_l \tag{6}$$

We will also only employ one form of movement at a time, either translational (5), or rotational (6).

WE SHOULD PROBABLY GO INTO DETAIL ABOUT CONTROLLABILTY AND HOW WE CAN USE THE CONTROL LAWS STRAIGHT UP

III. Procedure

A. Modelling/Simulation

B. Control of One ADDV

If we want to move our ADDV from (x_0, y_0, θ_0) to some $(\bar{x}, \bar{y}, \bar{\theta})$ then a simple, if inelegant solution is to break our movement into 3 steps

1) Rotate until θ is pointed towards our desired location
2) Translate until the ADDV has arrived at the destination
3) Rotate again to match θ with $\bar{\theta}$

The matlab code for this fucntion is given in the appendix.

C. Control of One ADDV Along Path

D. Pair of ADDV

E. Fleet of ADDV

IV. Discussion

APPENDIX

```
% returns controls and positions to move from the
% current pose (x_0, y_0, o_0) to the desired end pose (x_f, y_f, z_f)
% currently DOES NOT account for max speed/errors/etc
% or RETURN x, y, o VALUES (im lazy)
function [u_l, u_r, x, y, o] = move_addv(x_0, y_0, o_0, x_f, y_f, o_f)
    r = 1; % radius of wheels
    L = 1; % radius of car

    % first rotate to face x_f, y_f
    theta = o_f - atan(y_o/x_o); % Need to have cases for atan
    u_l(1) = L*2/r * theta;
    u_r(1) = - u_l(1);

    % now move to x_f, y_f
    d = sqrt((x_f-x_0)^2 + (y_f-y_0)^2);
    u_l(2) = 4/r * d;
    u_r(2) = u_l(2);

    % now face to o_f

    theta = o_f - theta;
    u_l(3) = L*2/r * theta;
    u_r(3) = - u_l(3);
end
```