ELEC6530 Homework 2 - Differential Drive Kinematics

M. Carroll and N. Jha

1 Model Derivation

1.1 Kinematics - Body Frame

First, we must derive the kinematics of the model in the body frame. The coordinate frame of the robot is x forward, y left, and z up. Theta represents the rotation of the body about the z-axis. The axes form a right-handed coordinate system, so positive theta indicates counter-clockwise rotation.

The velocity of the robot can be represented as a pair of vectors. $\vec{v}$ represents the linear velocity (forwards and backwards) of the robot, and $\vec{w}$ represents the angular velocity (rotation) of the robot.

Given ω_R and ω_L, the wheel speeds of the right and left wheels, we can represent the linear and angular velocity of the differential drive robot as shown in Equation 1.

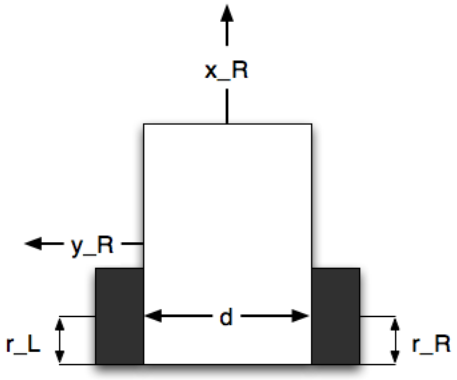

Figure 1: Physical Configuration of the Robot

$$\begin{aligned} v &= \frac{r_R}{2}\omega_R + \frac{r_L}{2}\omega_L \\ w &= \frac{r_R}{d}\omega_R - \frac{r_L}{d}\omega_L \end{aligned} \tag{1}$$

Where r_R and r_L are the wheel radii of the left and right wheels, and d is the width of the wheelbase as shown in Figure 1.

1.2 Kinematics - Inertial Frame

The next step is to derive the kinematics of the robot in the global coordinate frame. The global coordinate frame is represented in Figure 2.

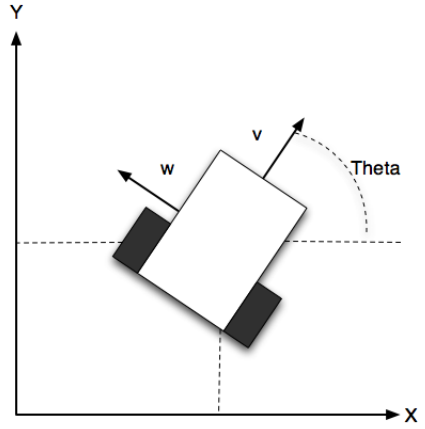

Figure 2: Robot in the Inertial Coordinate Frame

Since v and w are velocities, it is possible to derive the velocity and angular velocity of the robot in the inertial coordinate frame. The equations of motion are derived in Equation 2.

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{2}$$

To find the position of the robot, then these time-differentials must be integrated from a beginning time T_0 to the current time T. Additionally, a constant offset may be added if the robot does not start at the origin of the intertial coordinate frame. These integrals are performed in Equation 3.

$$\begin{aligned} x(t) &= \int_{T_0}^{T} v(t)\cos\left(\theta(t)\right)dt + X_0 \\ y(t) &= \int_{T_0}^{T} v(t)\sin\left(\theta(t)\right)dt + Y_0 \\ \theta(t) &= \int_{T_0}^{T} w(t)dt + \theta_0 \end{aligned} \tag{3}$$

1.3 Discrete Time Model

Often, it is more useful to represent the kinematics of the robot via a discrete-time model. This model assumes that the continuous values of the motor speeds are sampled at a constant rate $1/\Delta t$ and represented as v_k and w_k. This makes the equations of motion as in Equation 4.

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{k+1} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{k} + \begin{bmatrix} \Delta t v_k \cos\left(\theta_k + \Delta t w_k/2\right) \\ \Delta t v_k \sin\left(\theta_k + \Delta t w_k/2\right) \\ \Delta t w_k \end{bmatrix} \tag{4}$$

2 Simple Tests of Kinematic Model

2.1 Equal, Contant Wheel Velocities

Basic intuition tells us that if both wheels are at the same speed, then the robot will drive in a straight line in the direction that it started in. This can be seen from our simulation results below.

The robot was started heading towards 0 degrees (the x-axis), and then each wheel was commanded to the same speed. The robot travels in the x-direction throughout the entire simulation.

Figure 3 shows the x,y, and theta values of the pose of the robot over time, while Figure 4 shows the top-down view of the robot's motion.

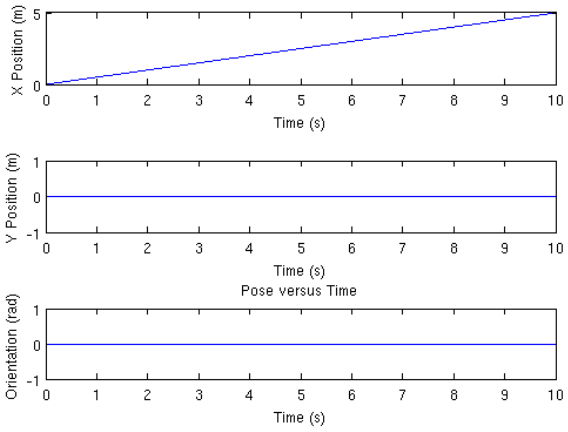

Figure 3: Pose of the Robot vs Time with Equal, Constant Wheel Speeds

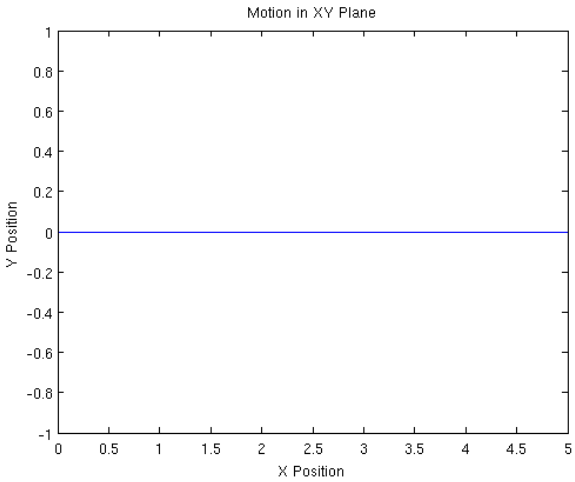

Figure 4: Top-Down view of Robot Motion with Equal, Constant Wheel Speeds

2.2 Unequal, Constant Wheel Velocities

Once again, intuition tells us that if one wheel rotates faster than the other, the robot will drive in a large arc, and given enough time, a circle. This can bee seen from our simulation results below.

The robot was started heading towards 0 degrees (the x-axis), and the right wheel was commanded to faster than the left wheel by a factor of 5. The robot travels in a circle.

Figure 5 shows the x,y, and theta values of the pose of the robot over time, while Figure 6 shows the top-down view of the robot's motion.

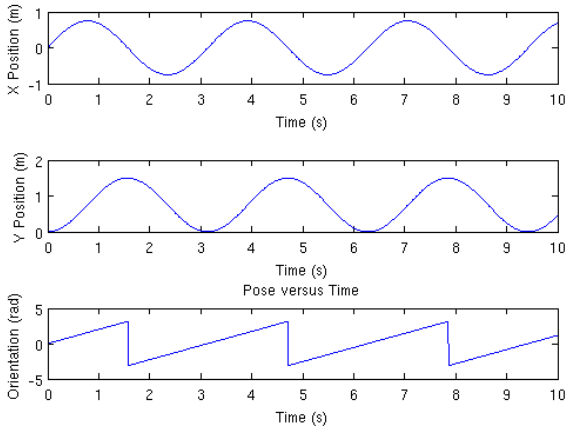

Figure 5: Pose of the Robot vs Time with Unequal, Constant Wheel Speeds

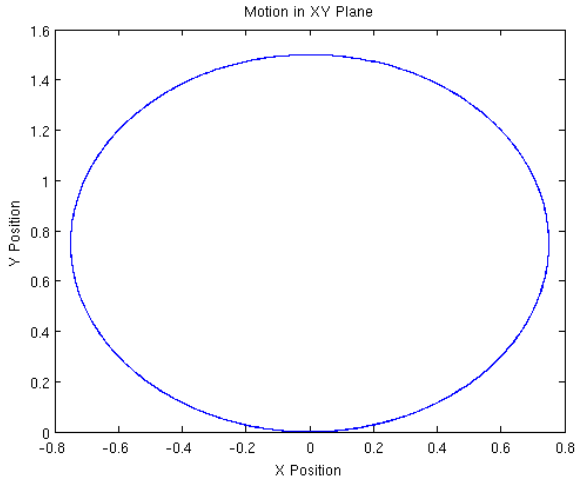

Figure 6: Top-Down view of Robot Motion with Unequal, Constant Wheel Speeds

2.3 Equal, Noisy Wheel Velocities

For a better test of the simulation, we also added in unit normally-distributed noise to the motor velocity commands to see the effect.

When the wheels were commanded to an equal velocity before the noise was applied, the robot "wandered" to the left and right of the x-axis line, but generally stayed around it.

Figure 7 shows the x,y, and theta values of the pose of the robot over time, while Figure 8 shows the top-down view of the robot's motion.

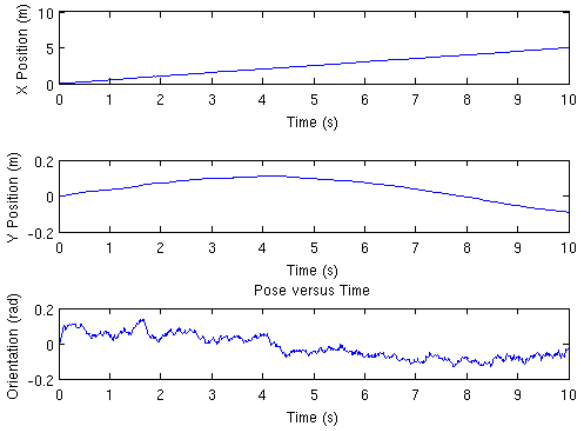

Figure 7: Pose of the Robot vs Time with Equal, Noisy Wheel Speeds

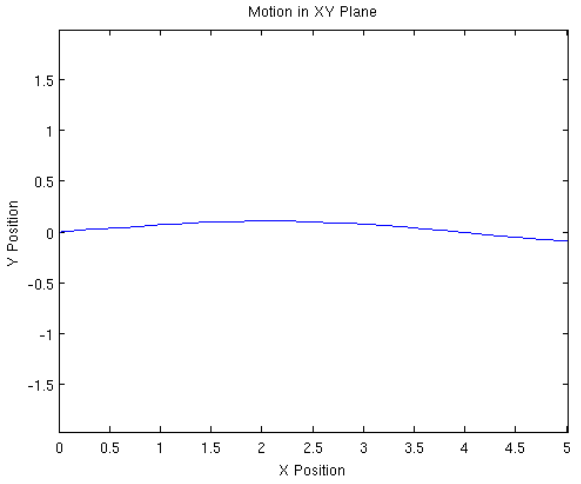

Figure 8: Top-Down view of Robot Motion with Equal, Noisy Wheel Speeds

2.4 Unequal, Noisy Wheel Velocities

When the wheels were commanded to an unequal velocity before the noise was applied, the robot "wandered" around a circular path, but couldn't repeat the path in multiple rotations.

Figure 9 shows the x,y, and theta values of the pose of the robot over time, while Figure 10 shows the top-down view of the robot's motion.

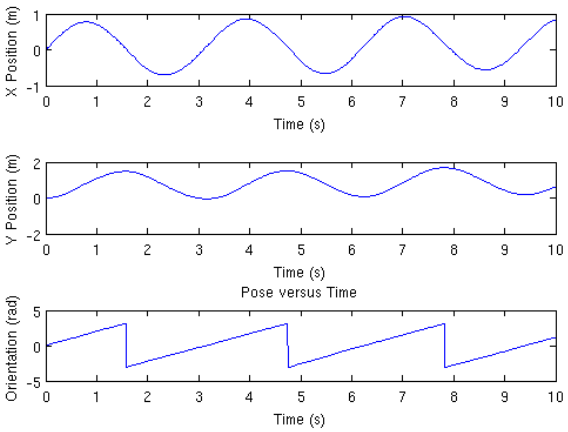

Figure 9: Pose of the Robot vs Time with Unequal, Noisy Wheel Speeds

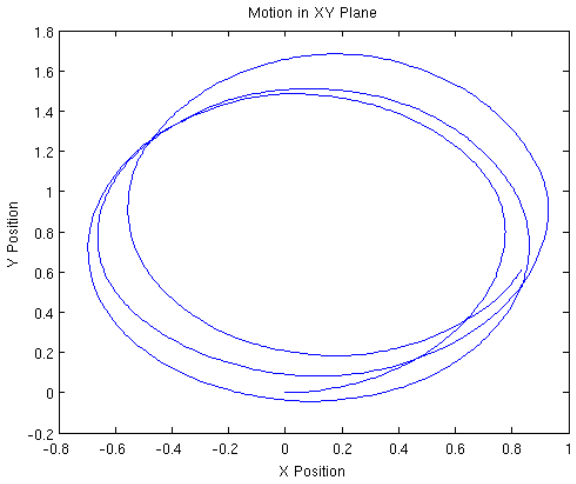

Figure 10: Top-Down view of Robot Motion with Unequal, Noisy Wheel Speeds

3 MATLAB Simulation Code

```
%%
clear; clc; close all;

%% Robot Parameters
right_radius = 0.5;
left_radius = 0.5;
wheel_base = 1.0;

%% Simulation Parameters

% Total simulation time is simLength * deltaT
simLength = 1000;
deltaT = 0.01;

% Origin is [time,x,y,theta]
origin = [0;0;0;0];

%wheelVels = repmat([1,1],simLength,1);
%wheelVels = repmat([1,5],simLength,1);
wheelVels = repmat([1,1],simLength,1) + randn(simLength,2);

%% Simulation

% Create an empty array of poses, and set the first pose to simulation
% origin.
pose = zeros(length(wheelVels),4);
pose(1,:) = origin;

% Calculate the pose of the robot throughout time.
for ii = 2:length(wheelVels),
    omegaL = wheelVels(ii,1);
    omegaR = wheelVels(ii,2);

    v = (right_radius * omegaR + left_radius * omegaL)/2;
    w = (right_radius * omegaR - left_radius * omegaL)/wheel_base;

    pose(ii,1) = pose(ii-1,1) + deltaT;
    pose(ii,2) = pose(ii-1,2) + (deltaT * v) * ...
        cos(pose(ii-1,4) + (deltaT * w/2));
    pose(ii,3) = pose(ii-1,3) + (deltaT * v) * ...
        sin(pose(ii-1,4) + (deltaT * w/2));
    pose(ii,4) = wrapToPi(pose(ii-1,4) + deltaT * w);
end

%% Plots
t = pose(:,1); x = pose(:,2); y = pose(:,3); theta = pose(:,4);

figure(1);
subplot(3,1,1); plot(t,x);
xlabel('Time (s)'); ylabel('X Position (m)');
subplot(3,1,2); plot(t,y);
xlabel('Time (s)'); ylabel('Y Position (m)');
```

```
subplot(3,1,3); plot(t,theta);
xlabel('Time (s)'); ylabel('Orientation (rad)');
title('Pose versus Time');

figure(2);
plot(x,y); xlabel('X Position'); ylabel('Y Position'); axis equal
title('Motion in XY Plane');
```