1 Task 1

Task 1 required us to first find out the velocities $v_x(t), v_y(t)$ as derivatives w.r.t time of the functions $x(t), y(t)$. This was done using MATLAB's symbolic math toolbox by taking the derivatives as shown

1.1 Finding velocities

```
syms x y v_x v_y a_x a_y t;

x = 8*(sin(t)).^3;
y = 8*(sin(2*t)).^3;

v_x = diff(x,t);
v_y = diff(y,t);
```

1.2 Finding accelerations

Similarly, for accelerations

```
a_x = diff(v_x,t);
a_y = diff(v_y,t);
```

1.3 Finding linear and angular velocities

Finally, the linear and angular velocities are calculated as follows

```
v_t = sqrt((v_x)^2 + (v_y)^2);
omega_t = (v_x*a_y - v_y*a_x)/(v_x^2 + v_y^2);
```

1.4 Plotting

It is important to note that the values calculated are still symbolic. These are plotted using MATLAB's *fplot()* function as shown below

```
%Plotting velocities
hold on;
subplot(211),fplot(v_t,[-pi pi],'linewidth',2)
title('Linear Velocities'),
```

```
xlabel("time"),ylabel("v_t");

subplot(212),fplot(omega_t,[-pi pi],'linewidth',2)
title('Angular Velocities')
xlabel("time"),ylabel("\omega_t");
```

The plots can be viewed in Figure 1.

The trajectory is shown in Figure 2

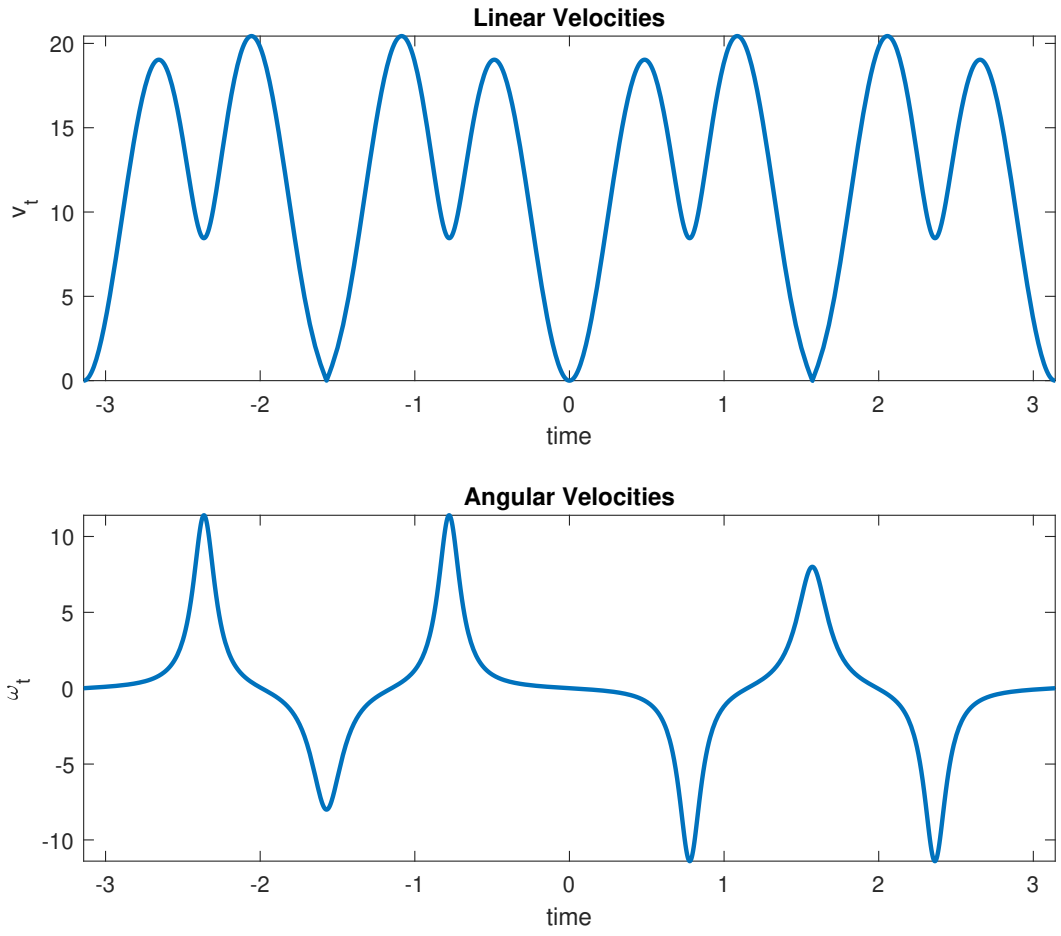

Figure 1: Linear and angular velocities of the robot

The animation can be viewed in the gif file called *"anim.gif"* when the script is run

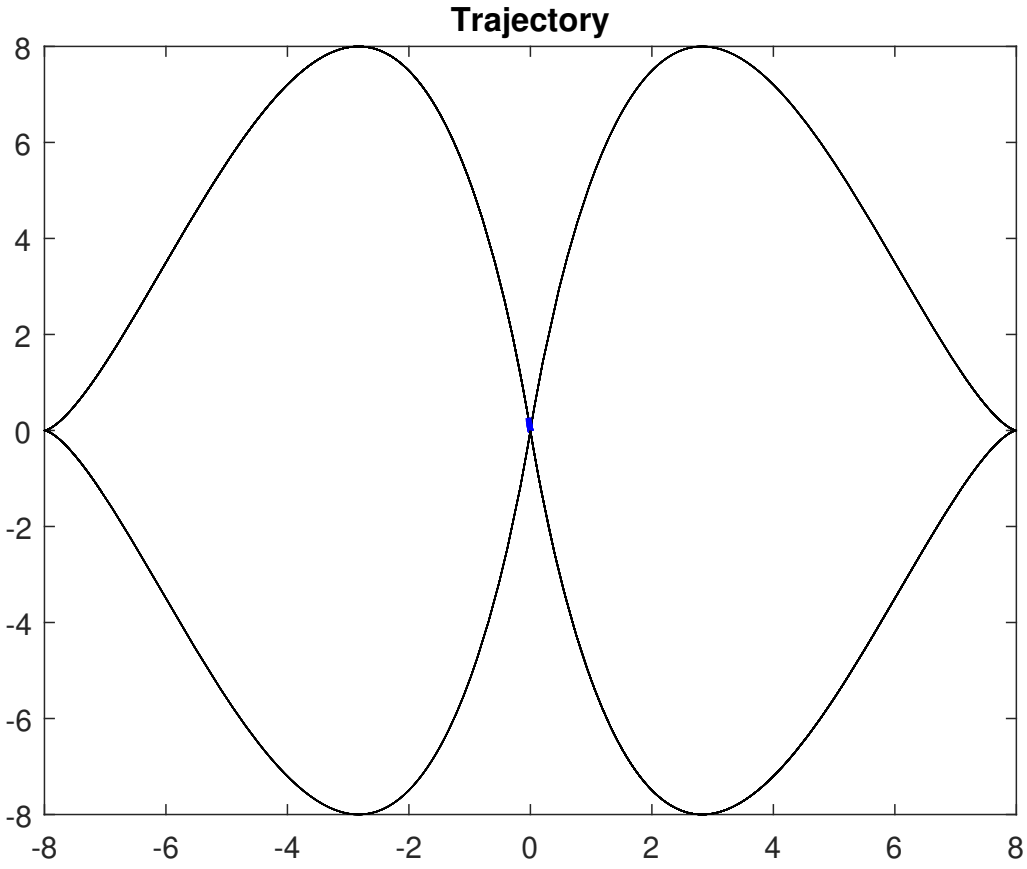

Figure 2: velocities

2 Task 2

2.1 Converting symbolic expressions to arrays

So far in Task 1, we had all values calculated as symbolic variables and expressions. This is converted to numeric arrays using the following snippet of code shown as examples.

```
v_y = eval(subs(v_y,t,t_num));
v_x = eval(subs(v_x,t,t_num));
```

We substituted the expressions with the numeric array t_num and used the *eval* function.

2.2 Finding the necessary values

1. We calculated μ by first finding x_m, y_m, as well as ϕ_n, using the *arctan2* function.
2. We then calculated ICR as x_*, y_*, using x_m,y_m, μ and y_{n+1}. From this we were able to get $R(t)$.
3. We were also able to find $\Delta\phi$ from which we got ω_t. We could have used the ω_t from Task 1 but evaluating it caused division by zero error. Thus we recalculated ω_t using $\Delta\phi$
4. The final angular velocities are found using the radius $r = 0.25$ of the robot, the width $a = 0.25$ and ω_t

   The graphs for angular velocities of each wheel are present in Figure 3

3 Scripts

The complete scripts are attached below and the animation can be found at my Github link

3.1 Task_1.m

```
clear;
clc;

N = 500;
t_1 = linspace(-pi,pi,N);


syms x y v_x v_y a_x a_y t;

x = 8*(sin(t)).^3;
y = 8*(sin(2*t)).^3;

v_x = diff(x,t);
v_y = diff(y,t);

a_x = diff(v_x,t);
a_y = diff(v_y,t);

v_t = sqrt((v_x)^2 + (v_y)^2);
omega_t = (v_x*a_y - v_y*a_x)/(v_x^2 + v_y^2);

%Plotting velocities
```

```
hold on;
subplot(211),fplot(v_t,[-pi pi],'linewidth',2)
title('Linear Velocities'),
xlabel("time"),ylabel("v_t");

subplot(212),fplot(omega_t,[-pi pi],'linewidth',2)
title('Angular Velocities')
xlabel("time"),ylabel("\omega_t");

% Creating Animated gif
T_s = 2*pi/N;


h = figure;
axis tight manual % this ensures that getframe() returns a consistent size
filename = 'anim.gif';
for n = 1:N
    fplot(x,y,'k');
    hold on;
    fplot(x,y,[-pi -pi+T_s*n],'b','linewidth',2),title("Trajectory");
    drawnow
      % Capture the plot as an image
      frame = getframe(h);
      im = frame2im(frame);
      [imind,cm] = rgb2ind(im,256);
      % Write to the GIF File
      if n == 1
          imwrite(imind,cm,filename,'gif', 'Loopcount',inf);
      else
          imwrite(imind,cm,filename,'gif','WriteMode','append');
      end
end
```

3.2 Task_2.m

```
%Task 2
clear;
clc;
Task_1;

%Evaluating ICR

r = 0.25;
L = 0.5;
a = 0.25;

t_num = linspace(-pi,pi,N);
```

```
%Finding phi_n
v_y = eval(subs(v_y,t,t_num));
v_x = eval(subs(v_x,t,t_num));
phi_n = atan2(v_y,v_x);

%Finding mu
x = eval(subs(x,t,t_num));
y = eval(subs(y,t,t_num));

y1 = [0 y(1:499)];
x1 = [0 x(1:499)];

y_np1 = [y(2:500) 0];
x_np1 = [x(2:500) 0];

x_m = 0.5*(x1+x);
y_m = 0.5*(y1+y);

mu = (sin(phi_n).*(y_m-y)-cos(phi_n).*(x-x_m))./...
(cos(phi_n).*(y_np1-y)-sin(phi_n).*(x_np1-x));

x_star = x_m - mu.*(y_np1-y);
y_star = y_m - mu.*(x_np1-x);

radii = sqrt((x-x_star).^2+(y-y_star).^2);

theta_1 = atan2(y-y_star,x-x_star);
theta_2 = atan2(y_np1 - y_star,x_np1 - x_star);

delta_phi = theta_1 - theta_2;
delta_phi = wrapToPi(delta_phi);

T = 2*pi/N;
omega_t = delta_phi./(T);
% omega_t = eval(subs(omega_t,t,t_num));

v_R = (radii + a).*omega_t;
v_L = (radii - a).*(omega_t);

omega_L = v_L./r;
omega_R = v_R./r;

figure;

subplot(211),plot(t_num(2:end),omega_L(2:end),'linewidth',2),
title("\omega_L"),xlabel("time"),ylabel("\omega");

subplot(212),plot(t_num(2:end),omega_R(2:end),'linewidth',2),
title("\omega_R"),xlabel("time"),ylabel("\omega");
```

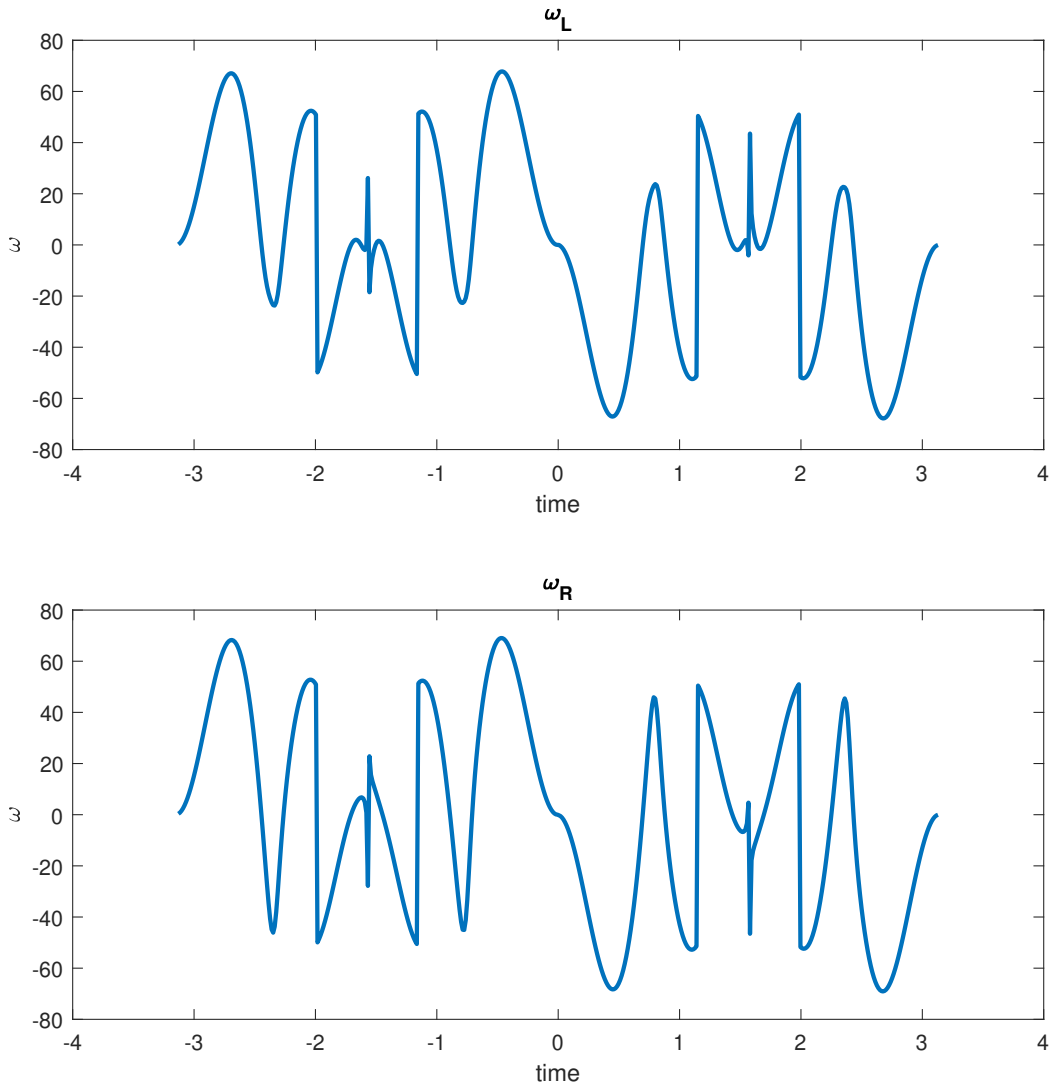

Figure 3: Wheel velocities