① $[x(0),\ y(0),\ \theta(0)]^T = [10, 20, 0.5]^T$

$[x(T),\ y(T),\ \theta(T)]^T = [0, 0, 0]^T$

$T = 10$

At $t = 0$

$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$

$y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3$

$x(0) = a_0 = 10 \quad -①$

$y(0) = b_0 = 20 \quad -②$

At $t = 10$

$x(t) = a_0 + 10a_1 + 100a_2 + 1000a_3 \quad -③$

$y(t) = b_0 + 10b_1 + 100b_2 + 1000b_3 \quad -④$

For initial state

$\dot{x}\cos\theta + \dot{y}\sin\theta = v$

At $v = 0 \Rightarrow t = 10$

$\dot{x}\cos\theta + \dot{y}\sin\theta = 0 \quad -⑤$

At $v = 10 \Rightarrow t = 0$

$\dot{x}\cos\theta + \dot{y}\sin\theta = 10 \quad -⑥$

Nonholonomic constraint

$\dot{x}\sin\theta - \dot{y}\cos\theta = 0$

$\cancel{x\sin(0.5) - 20\cos(0.5) = 0} \quad \cancel{④}$

$-⑦$

$\dot{x}_0\sin\theta_0 - \dot{y}_0\cos\theta = 0 \quad -⑧$

$\dot{x}_f\sin\theta_0 - \dot{y}_f\cos\theta = 0$

Solve these 8 equations

(2) $z_1 = x$

$z_2 = y$

$z_3 = \dot{x} = v\cos\theta$

$z_4 = \dot{y} = v\sin\theta$

Differentiating

$\dot{z}_1 = \dot{x} = z_3$

$\dot{z}_2 = \dot{y} = z_4$

$\dot{z}_3 = \ddot{x} = \dot{v}.\cos\theta - v\sin\theta.\dot{\theta}$

$\dot{z}_4 = \ddot{y} = \dot{v}.\sin\theta + v\cos\theta.\dot{\theta}$

(3) $u_1 = \dot{z}_3$ $\qquad$ $\dot{z}_1 = z_3$

$u_2 = \dot{z}_4$ $\qquad$ $\dot{z}_2 = z_4$

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \\ \dot{z}_4 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Reachability check: calculate $[A^3B \quad A^2B \quad AB \quad B]$

From MATLAB

$$\begin{bmatrix} 0 & 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

The system is reachable.

Rank = 4

Full rank matrix = No of state variables

Scanned by CamScanner

(4) From the previous question

$$\dot{z} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

error dynamics

$$e = z - z\_d$$

$$k_p = \begin{bmatrix} k_p & 0 \\ 0 & k_p \end{bmatrix} \qquad k_D = \begin{bmatrix} k_D & 0 \\ 0 & k_D \end{bmatrix}$$

$$k = [k_P, \; k_D]$$

$$u = -k[z - z_d] + a$$

$$\dot{z} = Az + Bu$$

(5) $x_a = [x, y, \theta, v]$

$\dot{v} = [a, \omega]$

$$\dot{x}_a = f(x_a, a, \omega) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \omega \\ a \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \\ a \end{bmatrix}$$

$$= \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \end{bmatrix}$$

(6) $u_1 = \ddot{x}$

$u_2 = \ddot{y}$

$\dot{x} = v\cos\theta$

$\ddot{x} = \dot{v}\cos\theta - v\sin\theta . \dot{\theta}$

$u_1 = a\cos\theta \quad - v\sin . \omega \qquad -(1)$

$\dot{y} = v\sin\theta$

$\ddot{y} = \dot{v}\sin\theta \quad + v\cos\theta . \dot{\theta}$

$u_2 = a\sin\theta + v\cos\theta . \omega \qquad -(2)$

solve (1) & (2)

$$
\begin{aligned}
u_1\sin\theta &= a\cos\theta\sin\theta \quad - v\sin^2\theta\,\omega \qquad (+) \\
-u_2\cos\theta &= -a\cos\theta\sin\theta - v\cos^2\theta\,\omega
\end{aligned}
$$

$u_1\sin\theta - u_2\cos\theta = -v\omega$

$$\boxed{\omega = \frac{u_2\cos\theta - u_1\sin\theta}{v}}$$

similarly

$$
\begin{aligned}
\ddot{x}\cos\theta &= a\cos^2\theta - v\cos\theta . \sin\theta . \omega \qquad (+) \\
\ddot{y}\sin\theta &= a\sin^2\theta \quad + v\cos\theta . \sin\theta . \omega
\end{aligned}
$$

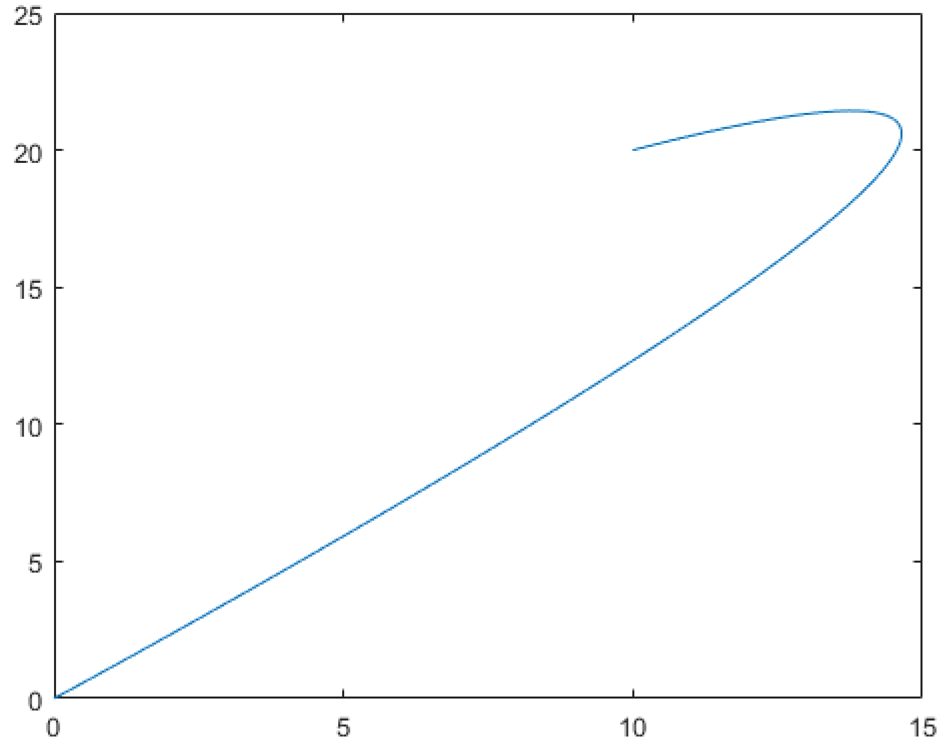
$u_1\cos\theta + u_2\sin\theta = a$

$$\boxed{a = u_1\cos\theta + u_2\sin\theta}$$

Quintic polynomial trajectory

```
syms a0 a1 a2 a3
syms b0 b1 b2 b3

% Initial and Final velocity
v0=5;
vt=0;

%Initial State vector values
x0=10;
y0=20;
th0=0.5;

%Final State vector values
xt=0;
yt=0;
tht=0;

%Initial and Final Values
t0=0;
tf=10;

% Trajectory Initial
xd_0=a1+2*a2*t0+3*a3*t0^2;
yd_0=b1+2*b2*t0+3*b3*t0^2;
%thd_0=w

xd_t=a1+2*a2*tf+3*a3*tf^2;
yd_t=b1+2*b2*tf+3*b3*tf^2;
%thd_t=w


% Solving the equations
eqns=[x0-a0,xt-a0-a1*tf-a2*tf^2-a3*tf^3,y0-b0,yt-b0-b1*tf-b2*tf^2-
b3*tf^3,xd_0*cos(th0)+yd_0*sin(th0)-v0,xd_t*cos(tht)+yd_t*sin(tht)-
vt,xd_0*sin(th0)-yd_0*cos(th0),xd_t*sin(tht)-yd_t*cos(tht)];
vars = [a0 a1 a2 a3 b0 b1 b2 b3];
sol = solve(eqns, vars);


% Extracting the coeffiecients
sola=double([sol.a0 sol.a1 sol.a2 sol.a3]);
solb=double([sol.b0 sol.b1 sol.b2 sol.b3]);

% Defining Time interval
t = linspace(t0,tf,100);

% Calculating the x and y trajectory
x=sola(1)+sola(2)*t+sola(3)*t.^2+sola(4)*t.^3;
y=solb(1)+solb(2)*t+solb(3)*t.^2+solb(4)*t.^3;
```

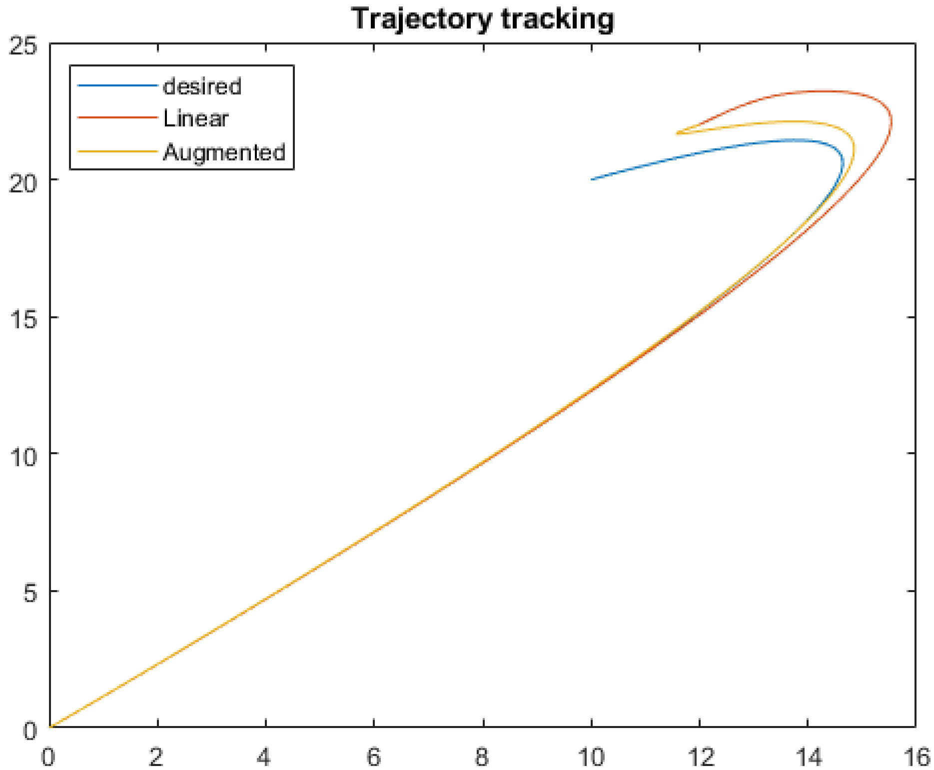
```
figure
plot(x,y);
hold on;

% Starting with initial error
z1=x0+2;
z2=y0+2;
z3=10;
z4=10;

% Store it in a single vector
z0=[z1;z2;z3;z4];
```



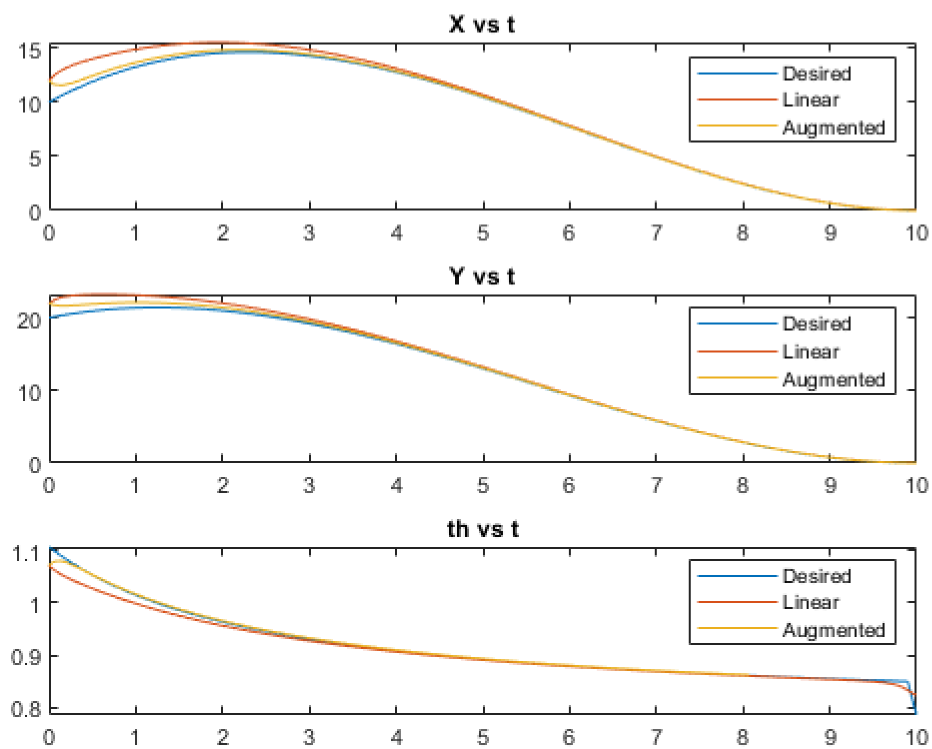
Implement the trajectory tracking with ODE function

```
x0=[z1;z2;z3;z4];
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4, 1e-4]);

% Trajectory tracking using linear system
[T1,X_L] = ode45(@(t,x) DublinTrajectoryTracking(t,x,[sola;solb]),[t0
 tf],z0, options);

% Trajectory tracking using augemented state vector
```

```
[T,X_NL] = ode45(@(t,x) DublinTrajectoryTracking_nonlinear(t,x,
[sola;solb]),[t0 tf],z0, options);

%Plotting the results of linear trajectory tracing controller
plot(X_L(:,1),X_L(:,2));
hold on

%Plotting the result of augmented state vector trajectory tracking
plot(X_NL(:,1),X_NL(:,2));
legend({'desired','Linear','Augmented'},'Location','northwest');
title('Trajectory tracking')


%Plotting the x,y,z vs t
figure
subplot(3,1,1)
plot(t,x)
hold on
plot(T1,X_L(:,1))
hold on
plot(T,X_NL(:,1))
title('X vs t')
legend('Desired','Linear','Augmented');


subplot(3,1,2)
plot(t,y)
hold on
plot(T1,X_L(:,2))
hold on
plot(T,X_NL(:,2))
title('Y vs t')
legend('Desired','Linear','Augmented');


subplot(3,1,3)
plot(t,atan2(y,x))
hold on
plot(T1,atan2(X_L(:,2),X_L(:,1)))
hold on
plot(T(1:250,1),atan2(X_NL(1:250,2),X_NL(1:250,1)))
title('th vs t')
legend('Desired','Linear','Augmented');
```



Trajectory tracking
desired
Linear
Augmented
0
5
10
15
20
25
0
2
4
6
8
10
12
14
16



X vs t
Desired
Linear
Augmented
Y vs t
Desired
Linear
Augmented
th vs t
Desired
Linear
Augmented

Published with MATLAB® R2018b

Table of Contents


```
function [ dx ] = DublinTrajectoryTracking(t,x,param)
```

Implementation

```
        %Extracting the coefficients of the trajectory
        a1=param(1,:);
        a2=param(2,:);

        %Create the actual trajectory
        vec_t = [1; t; t^2; t^3];
        X_d= [a1*vec_t;a2*vec_t];  %position


        % compute the velocity and acceleration in both theta 1 and
 theta2.
        x_vel = [a1(2), 2*a1(3), 3*a1(4), 0];
        x_acc = [2*a1(3), 6*a1(4),0,0 ];
        y_vel = [a2(2), 2*a2(3), 3*a2(4), 0];
        y_acc = [2*a2(3), 6*a2(4),0,0 ];

        % compute the desired trajectory (assuming 3rd order
 polynomials for trajectories)
        dX_d =[x_vel*vec_t; y_vel* vec_t];  %Velocity
        ddX_d =[x_acc*vec_t; y_acc* vec_t];  %Acceleration
        X= x(1:2,1);
        dX= x(3:4,1);
```

```
Not enough input arguments.

Error in DublinTrajectoryTracking (line 5)
        a1=param(1,:);
```

PD controller

```
        KP=5;
        KD=10;
        K=[KP*eye(2), KD*eye(2)];
        U=-K*[X-X_d;dX-dX_d]+ddX_d;
        A=[0 0 1 0;0 0 0 1;0 0 0 0;0 0 0 0];
        B=[0 0;0 0;1 0;0 1];
        z=[X;dX];
```

Calculate dx for the dubins car

```
    dx=A*z+B*U;
end
```

Published with MATLAB® R2018b

Table of Contents


```
function [ dx ] = DublinTrajectoryTracking_nonlinear(t,x,param)
```

Implementation

```
        %Extracting the coefficients of the trajectory
        a1=param(1,:);
        a2=param(2,:);

        %Create the actual trajectory
        vec_t = [1; t; t^2; t^3];
        X_d= [a1*vec_t;a2*vec_t];  %position


        % compute the velocity and acceleration in both theta 1 and
 theta2.
        x_vel = [a1(2), 2*a1(3), 3*a1(4), 0];
        x_acc = [2*a1(3), 6*a1(4),0,0 ];
        y_vel = [a2(2), 2*a2(3), 3*a2(4), 0];
        y_acc = [2*a2(3), 6*a2(4),0,0 ];

        % compute the desired trajectory (assuming 3rd order
 polynomials for trajectories)
        dX_d =[x_vel*vec_t; y_vel* vec_t] ; %Velocity
        ddX_d =[x_acc*vec_t; y_acc* vec_t];  %Acceleration
        X= x(1:2,1);

        % Calculating the state variables
        th=x(3);
        v=x(4);
        dX=[v*cos(th);v*sin(th)];
```

```
Not enough input arguments.

Error in DublinTrajectoryTracking_nonlinear (line 5)
        a1=param(1,:);
```

PD controller

```
        KP=5;
        KD=10;
        K=[KP*eye(2), KD*eye(2)];
```

```
U=-K*[X-X_d;dX-dX_d]+ddX_d;


a=U(1)*cos(th)+ U(2)*sin(th);
w=(U(2)*cos(th)-U(1)*sin(th))/v;

A=[0 0 1 0;0 0 0 1;0 0 0 0;0 0 0 0];
B=[0 0;0 0;1 0;0 1];
z=[X;dX];
```

Calculate dx for the dubins car

```
dx=[dX;w;a];

end
```

Published with MATLAB® R2018b