



Tecnológico de Monterrey

Campus Puebla

Materia

Fundamentación de Robótica TE3001B

Tema

Examen

Alumno

Antonio Silva Martínez A01173663

Fecha

Mayo 17 2023

Instrucciones:

Utilizar el repositorio con el nombre: Evaluación 7.1 (Trayectorias en lazo abierto)

Obtener el mapa de puntos para generar la planeación de trayectorias a partir de tu rostro, marcando set-points con el software Geogebra.

Planear la trayectoria que debe seguir el algoritmo empleando el mapa de puntos obtenido, considerando la descripción mas próxima a los rasgos físicos de tu rostro (Forma de la cara, forma del cabello, ojos, cejas, nariz, boca y orejas), (barba en caso de que aplique).

Implementar el código requerido para generar la planeación de trayectorias empleado la técnica que consideres más eficiente para esta aplicación (Control en lazo abierto, control de posición, control de trayectorias ó PurePursuit (way-points)). Considerando el ajuste de los parámetros cinemáticos del robot móvil como: Velocidad angular (w), velocidad lineal (v), ancho de la muestra (t_s), tiempo de simulación (t), ganancias, etc.

Generar un reporte de la justificación de la estrategia de control elegida para poder obtener el recorrido de las trayectorias. Incluir el Código en MATLAB y el resultado final del seguimiento de trayectoria de tu rostro

Subir el link del repositorio en CANVAS para “Evaluación”

Explicación:

El algoritmo Pure Pursuit es una herramienta genial para trazar rutas de vehículos. Es fácil de usar y rápido, así que puedes aplicarlo en tiempo real sin complicaciones. La idea detrás del algoritmo es perseguir un punto objetivo según la posición y dirección actual del vehículo, como cuando sigues las indicaciones de un GPS. Lo interesante es que permite seguir rutas suaves y sin movimientos bruscos, adaptándose a diferentes velocidades. Además, es resistente a obstáculos y perturbaciones, por lo que no se deja intimidar fácilmente.


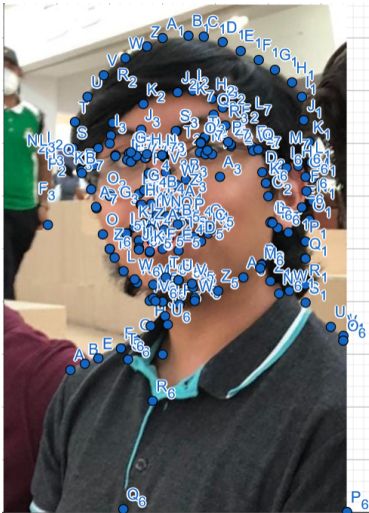
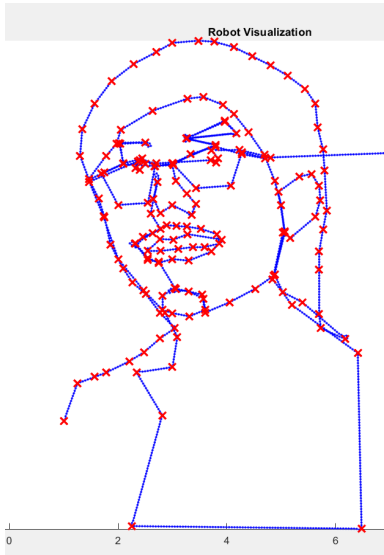
El funcionamiento de este código es similar al utilizado en el trazado de trayectorias del perro, siendo que las modificaciones realizadas están basadas en las coordenadas ubicadas en la foto.

Los parámetros que se tuvieron en cuenta para la realización de este modelo fueron los siguientes:

- LookaheadDistance
- DesiredLinearVelocity
- MaxAngularVelocity
- sampleTime

Siendo que estos son los que permiten que el modelo sea más preciso en donde el Lookahead permite el acercamiento a cierta distancia de un punto que nosotros asignamos, el DesiredLinear es la velocidad lineal y en este caso la velocidad lineal tiene que ser pequeña para que no exista un “sobretiro” el cual haga que el modelo no sea tan preciso, la MaxAngularVelocity permite que el giro sea más suave y se pueda generar un mejor modelo y por último el sampleTime el tiempo de muestreo permite registrar un cierto número de muestras en un tiempo determinado.

Resultados:

Imagen Original	Imagen con coordenadas	Modelo c
		

Código utilizado:

%% EXAMPLE: Differential drive vehicle following waypoints using the

% Pure Pursuit algorithm

```
%
```

```
% Copyright 2018-2019 The MathWorks, Inc.
```

```
%% Define Vehicle
```

```
R = 0.1;          % Wheel radius [m]
```

```
L = 0.5;          % Wheelbase [m]
```

```
dd = DifferentialDrive(R,L);
```

```
%% Simulation parameters
```

```
sampleTime = 0.1;    % Sample time [s]
```

```
tVec = 0:sampleTime:200;    % Time array
```

```
initPose = [1;2; 1];    % Initial pose (x y theta)
```

```
pose = zeros(3,numel(tVec));    % Pose matrix
```

```
pose(:,1) = initPose;
```

```
% Define waypoints
```

```
waypoints = [ 1,2; 1.25,2.70; 1.57,2.82; 1.78,2.9; 2.21,3.11; 2.48,3.28; 2.77,3.53; 3.04,3.72;  
2.77,4.01; 2.51,4.35; 2.26,4.57; 2.09, 4.83; 2,5; 1.86,5.27; 1.75,5.77; 1.64,6.12; 1.46,6.48;  
1.30,6.91; 1.34,7.40; 1.57,7.88; 1.88, 8.29; 2.29,8.61; 2.70,8.83; 3,9; 3.48,9.04; 3.77,9.02;  
4.13,8.92; 4.47,8.75; 4.81,8.58; 5.12,8.39; 5.44,8.15; 5.63,7.88; 5.67,7.44; 5.77,7.03;  
5.80,6.64; 5.84,5.90; 5.77,5.55; 5.70,5.14; 5.70, 4.81; 5.70,3.98; 5.73,3.72; 6.18,3.52;  
5.39,4.20; 5.03,4.39; 4.88,4.71; 5.06,5.51; 5,6; 4.88,6.45; 4.69,6.93; 4.40,7.35; 4.13,7.68;  
3.93,7.85; 3.57,8; 3.28,7.97; 2.63,7.74; 2.05,7.39; 1.97,7.13; 1.78,6.91; 1.46,6.48; 1.75,6.60;  
2.10,6.77; 2.44,6.86; 2.67,6.72; 3.02,6.74; 3.33,6.94; 3.79,7.08; 4.28,6.94; 4.71,6.86;  
4.28,6.98; 4.08,6.36; 3.43,6.31; 3.01,6.76; 2.68,6.72; 2.55,6.04; 2,6; 1.69,6.58; 2.10,6.76;  
2.02,7.15; 2.50,7.16; 2.04,7.13; 2.09,6.76; 2.67,6.72; 3.01,6.77; 3.79,7.11; 3.26,7.23;  
3.96,7.56; 4.18,7.33; 3.25,7.23; 3.96,7.54; 4.18,7.33; 3.25,7.23; 3.79,7.11; 2.99,6.77;  
3.06,6.45; 3.26,6.21; 3.43,6.02; 3.35,5.82; 3,6; 2.78,5.85; 2.65,6.11; 2.73,6.43; 2.65,6.11;  
2.60,5.84; 2.78,5.56; 2.63,5.48; 2.46,5.41; 2.32,5.29; 2.89,5.63; 3.07,5.61; 3.26,5.60;
```

```

3.52,5.56; 3.82,5.46; 3.86,5.29; 3.59,5.22; 3.38,5.22; 3.11,5.19; 2.78,5.15; 2.55,5.15;
2.77,5.37; 3.02,5.36; 3.28,5.46; 3.91,5.36; 3.72,5.14; 3.30,4.97; 3,5; 2.75,4.93; 2.55,5.02;
2.31,5.29; 2.55,4.98; 2.75,4.95; 3.04,4.47; 2.82,4.32; 2.82,4.06; 3,4; 3.31,3.94; 3.60,4.06;
3.57,4.27; 3.30,4.40; 3.06,4.45; 3.30,4.40; 3.55,4.34; 3.60,4.01; 4.05,4.20; 4.52,4.45;
4.86,4.69; 5.03,5.49; 5.17,5.39; 5.63,5.78; 5.72,6.09; 5.70,6.36; 5.56,6.57; 5.34,6.53;
4.95,6.24; 4.95,6.24; 5.07,5.49; 4.84,4.63; 5.20,4.15; 6.41,3.26; 6.48,0.01; 2.25,0.06;
2.81,2.11; 2.34,2.90; 3,3; 3.09,3.56; 2.86,4.01; 2.47,4.42; 2,5; 1.73,5.79; 1.47,6.43; 2.36,6.81;
2.32,6.68; 2.40,6.65; 2.50,6.76; 2.40,6.83; 2.70,6.77; 3.70,6.83; 3.81,6.79; 3.84,6.86;
3.72,7.01; 4.23,7.02; 4.81,6.88; 8,7;];

```

```

% Create visualizer

```

```

viz = Visualizer2D;

```

```

viz.hasWaypoints = true;

```

```

%% Pure Pursuit Controller

```

```

controller = controllerPurePursuit;

```

```

controller.Waypoints = waypoints;

```

```

controller.LookaheadDistance = 0.1;

```

```

controller.DesiredLinearVelocity = 0.5;

```

```

controller.MaxAngularVelocity = 20;

```

```

%% Simulation loop

```

```

close all

```

```

r = rateControl(1/sampleTime);

```

```

for idx = 2:numel(tVec)

```

```

    % Run the Pure Pursuit controller and convert output to wheel speeds

```

```

    [vRef,wRef] = controller(pose(:,idx-1));

```

```

[wL,wR] = inverseKinematics(dd,vRef,wRef);

% Compute the velocities

[v,w] = forwardKinematics(dd,wL,wR);

velB = [v;0;w]; % Body velocities [vx;vy;w]

vel = bodyToWorld(velB,pose(:,idx-1)); % Convert from body to world

% Perform forward discrete integration step

pose(:,idx) = pose(:,idx-1) + vel*sampleTime;

% Update visualization

viz(pose(:,idx),waypoints)

waitfor(r);

end

```