

Atividade Fomartiva 02

Leonardo Santos - GRR20196154

A)

Para realizar essa simulação foi optado em utilizar o exemplo disponibilizado pelo Matlab.

Primeramente foram ajustados o waypoints, de forma a realizar a trajetória retangular conforme ilustrada pela Figura 1 a seguir

```
1 waypoints = [0 1; 0 10; 0 20; 10 20; 20 20; 20 10; 20 0; 10 0; 0 0];
2 sampleTime = 0.05; % Sample time [s]
3 tVec = 0:sampleTime:20; % Time array
4
5 initPose = [waypoints(1,:)' ; 0]; % Initial pose (x y theta)
6
7 goalPoints = [0 ; -1] ;
8 goalRadius = 1;
```

Figura 1: Codigo da trajetória retangular

Realizando a simulação chegou-se no seguinte resultado:

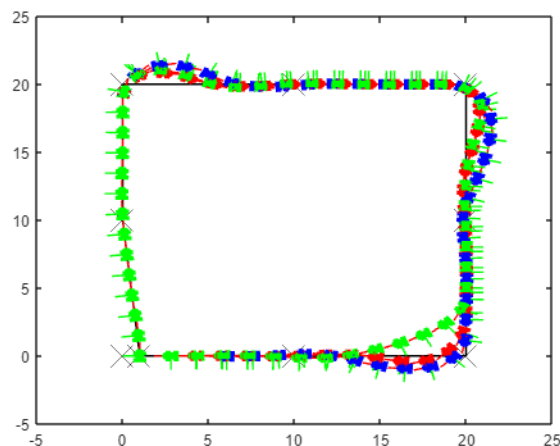


Figura 2: trajetória retangular

Após isso foi adaptado o código de forma a relizar a trajetória, o qual está ilustrado pela Figura 3 a seguir.

```
1 circle = 0:0.1:2*pi;
2 waypoints = [cos(circle') sin(circle')] * 10;
3 % Define the total time and the sample rate
4 sampleTime = 0.05; % Sample time [s]
5 tVec = 0:sampleTime:20; % Time array
6 initPose = [waypoints(1,:)' ; pi/2]; % Initial pose (x y theta)
7
8 goalPoints = [0 ; -5] ;
9 goalRadius = 1;
10
```

Figura 3: Codigo da trajetória retangular

Em seguida foi feito a trajetoria circular do o resultado esta ilustrado pela Figura 4 a seguir.

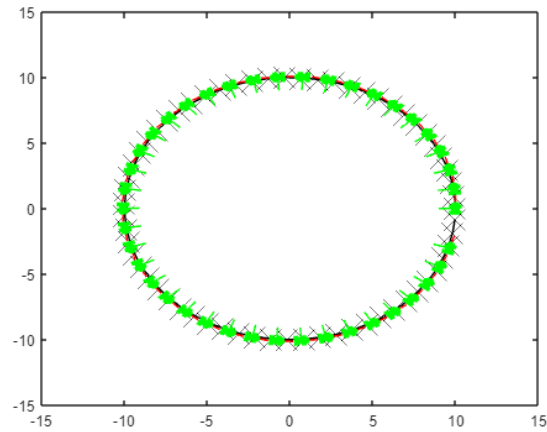


Figura 4: Trejetória Circular

B

Pode se dizer que o erro médio de cada modelo(MSE), é a distancia média de cada modelo para o polígono definido pelos waypoints. Sendo assim para fazer esse calculo foi feito um ajuste no codigo de forma o numero de amostras simuladas fosse que o mesmo numero de pontos presentes nos waypoints. Então foi calculado o erro médio. Para isso foi feito os seguintes trechos de código, que estão sendo ilustrados pelas Figura 5, Figura 6, Figura 7 a seguir,

```

1  i = 1
2  n = size(waypoints)[1]
3  MSEx = 0
4  MSEy = 0
5
6  while i < n
7      MSEx_unicycle(i) = (unicyclePose(i,1) - waypoints(i,1))^2;
8      MSEy_unicycle(i) = (unicyclePose(i,2) - waypoints(i,2))^2;
9      MSEx = MSEx + MSEx_unicycle(i);
10     MSEy = MSEy + MSEy_unicycle(i);
11     i = i+1;
12 end
13
14 MSE = (MSEx+MSEy)/n

```

Figura 5: MSE retangular unicycle

```

1  i = 1
2  n = size(waypoints)[1]
3  MSEx = 0
4  MSEy = 0
5
6  while i < n
7      MSEx_bicycle(i) = (bicyclePose(i,1) - waypoints(i,1))^2;
8      MSEy_bicycle(i) = (bicyclePose(i,2) - waypoints(i,2))^2;
9      MSEx = MSEx + MSEx_bicycle(i);
10     MSEy = MSEy + MSEy_bicycle(i);
11     i = i+1;
12 end
13
14 MSE = (MSEx+MSEy)/n

```

Figura 6: MSE retangular unicycle



```

1
2 i = 1
3 n = size(waypoints)[1]
4 MSEx = 0
5 MSEy = 0
6
7 while i < n
8     MSEx_diff(i) = (diffDrivePose(i,1) - waypoints(i,1))^2;
9     MSEy_diff(i) = (diffDrivePose(i,2) - waypoints(i,2))^2;
10    MSEx = MSEx + MSEx_diff(i);
11    MSEy = MSEy + MSEy_diff(i);
12    i = i+1;
13 end
14
15 MSE = (MSEx+MSEy)/n
16

```

Figura 7: MSE retangular uniciclo

E com isso chegou-se aos seguintes valores de MSE:

$$\text{MSE}_{\text{uni}} = 5.91$$

$$\text{MSE}_{\text{bi}} = 6.3$$

$$\text{MSE}_{\text{dif}} = 5.83$$

Portanto, podemos realizar uma suposição de que o movimento dos três modelos não estão sendo limitados pela velocidade angular máxima. Apesar dos resultados é visível que o modelo diferencial é o que apresenta o melhor comportamento em curvas perpendiculares, o que é observado na Figura 2.

Em seguida foi realizado o mesmo processo para a trajetória circular, onde foi calculado MSEs muito próximos de zero, o que era de se esperar visto que a variação da direção é constante. E portanto a velocidade angular também não influencia no desempenho de cada modelo.