Instituto Tecnológico de Estudios Superiores de Monterrey

Campus Puebla

Implementación de robótica inteligente (Gpo 501)

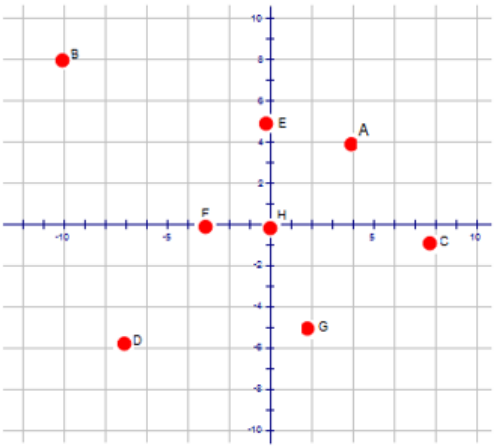
Actividad 1.9: Landmarks

Alumno
Victor Manuel Vázquez Morales A01736352

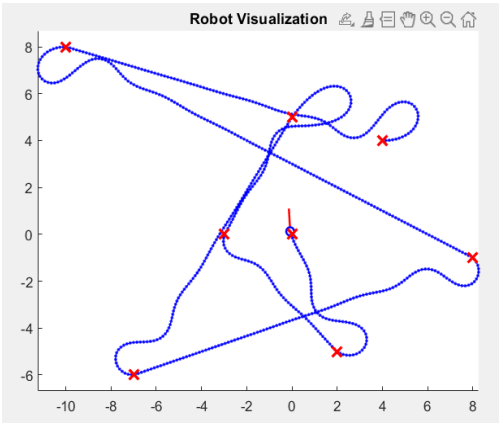
Fecha de entrega
Lunes 06 de Mayo de 2024

Implementar el código requerido para generar el seguimiento de los siguientes waypoints (puntos de referencia), ajustando el tiempo de muestreo: “sampleTime”, vector de tiempo: “tVec”, pose inicial “initPose”, y los waypoints: “waypoints”.

Ejercicio 1:

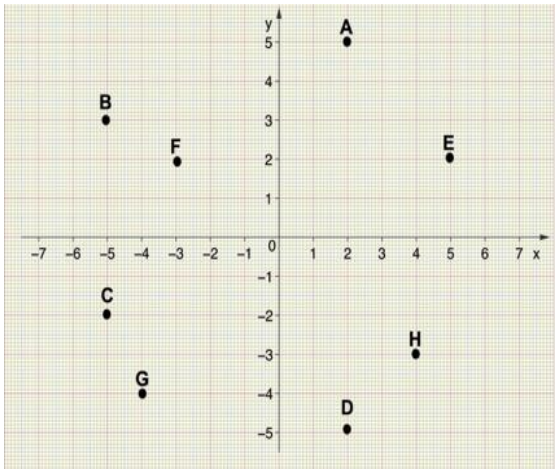



```
sampleTime = 0.05;
tVec = 0:sampleTime:160;
initPose = [4;4;0];

waypoints = [4,4; -10,8; 8,-1; -7,-6; 0,5;
-3,0,; 2,-5; 0,0];
controller.LookaheadDistance = 0.4;

controller.DesiredLinearVelocity = 3;

controller.MaxAngularVelocity = 5;
```

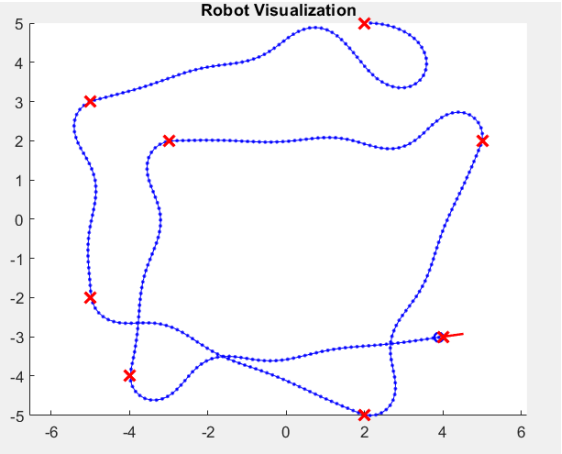


Análisis: Para este primer ejercicio podemos observar como nuestro robot diferencial sigue cada uno de los puntos marcados o declarados en el vector waypoints. Notemos de igual forma, que los cambios de orientación (o ángulo) en nuestro robot se hacen de una forma relativamente lenta. Si bien para este caso no requerimos de un cambio de orientación rápido, es importante resaltar esto para identificar la razón ya que más adelante necesitaremos una respuesta más rápida.

Ejercicio 2:

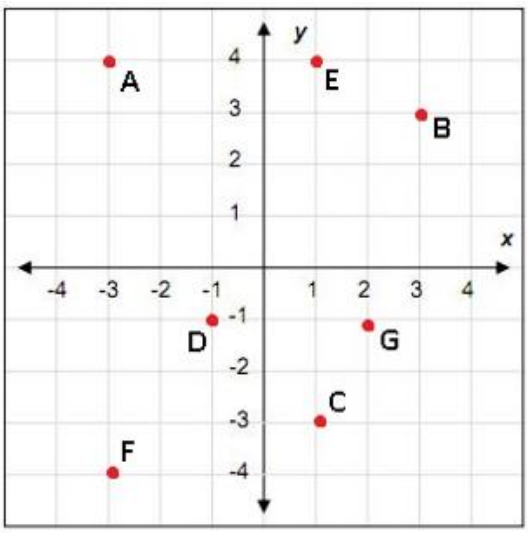



```
sampleTime = 0.05;

tVec = 0:sampleTime:19;

initPose = [2;5;0];

waypoints = [2,5; -5,3; -5,-2; 2,-5;

5,2; -3,2,; -4,-4; 4,-3];

controller.LookaheadDistance = 0.4;

controller.DesiredLinearVelocity = 3;

controller.MaxAngularVelocity = 5
```

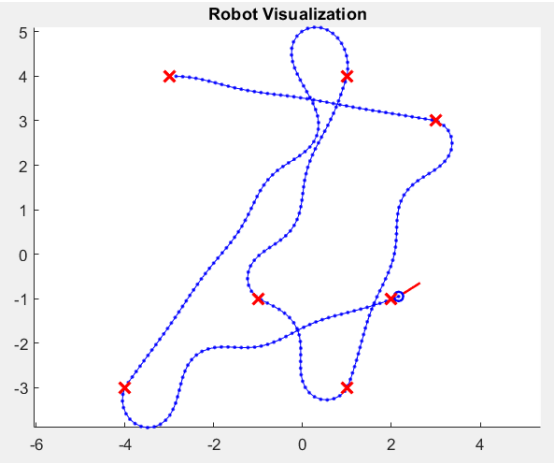


Análisis: En este segundo ejercicio, mantenemos los parámetros de velocidad y de *LookAheadDistance*, modificando únicamente los waypoints y la posición inicial y, podemos observar que el comportamiento o la trayectoria del robot parece ser más suave o estable, esto debido a que entre puntos no hay un cambio tan radical en cuanto a orientación, por lo que al robot diferencial se le facilita más realizar esta trayectoria.

Ejercicio 3:



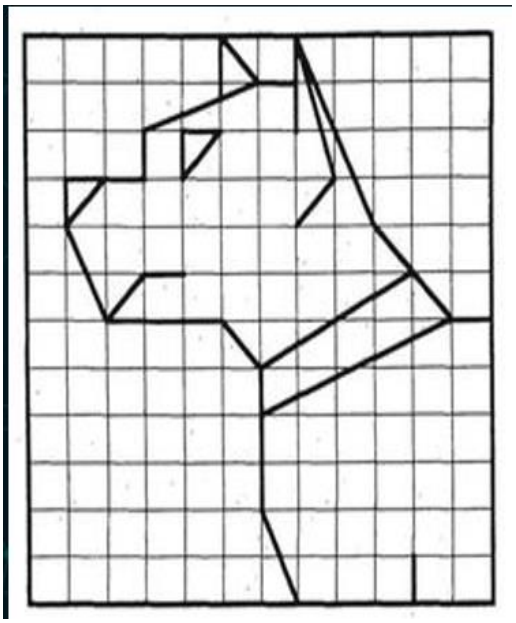

```
sampleTime = 0.05;

tVec = 0:sampleTime:14;

initPose = [-3;4;0];

waypoints = [-3,4; 3,3; 1,-3; -1,-1;

1,4; -4,-3,; 2,-1];

controller.LookaheadDistance = 0.35;

controller.DesiredLinearVelocity = 3;

controller.MaxAngularVelocity = 8;
```



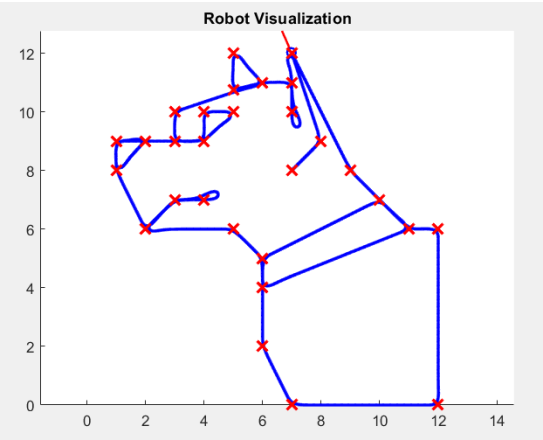
Análisis: Para este tercer ejercicio se realizaron algunas modificaciones a los parámetros del robot, ya que con los valores anteriores el robot no pasaba exactamente por todos los landmarks. Observé que el parámetro que corrigió este comportamiento fue el de *LookAheadDistance*, ya que al disminuirlo de 0.4 a 0.35 unidades el robot pudo ajustar con mayor precisión su trayectoria pasando así por cada uno de los landmarks de manera efectiva.

Generar los waypoints (puntos de referencia) necesarios para obtener las siguientes trayectorias, ajustando el tiempo de muestreo: “sampleTime”, vector de tiempo: “tVec”, posee inicial: “initPose”, y los waypoints: “waypoints”.

Ejercicio - Perro:

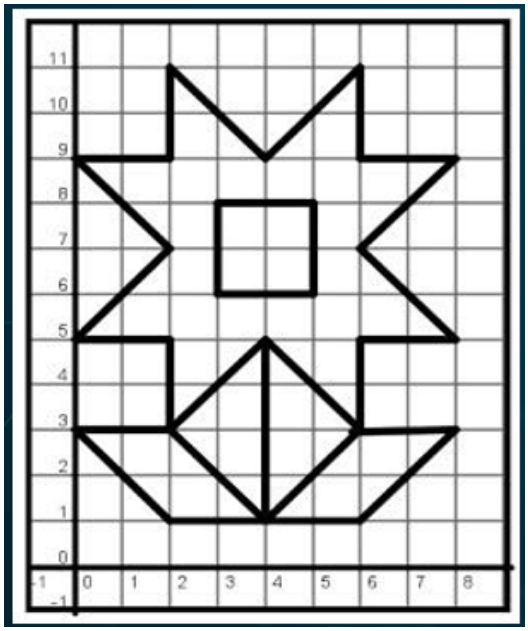


```
sampleTime = 0.05;

tVec = 0:sampleTime:143;

initPose = [7;8;pi/4];

waypoints = [7,8; 8,9; 7,12; 7,10; 7,11;
6,11; 5,10.75; 5,12; 6,11; 3,10; 3,9; 4,9;
4,10; 5,10; 4,9; 2,9; 1,9; 1,8; 2,9; 1,9;
1,8; 2,6; 3,7; 4,7; 3,7; 2,6; 5,6; 6,5;
6,4; 11,6; 10,7; 6,5; 6,2; 7,0; 12,0;
12,6; 11,6; 10,7; 9,8; 7,12];

controller.LookaheadDistance = 0.35;

controller.DesiredLinearVelocity = 0.53;

controller.MaxAngularVelocity = 10;
```

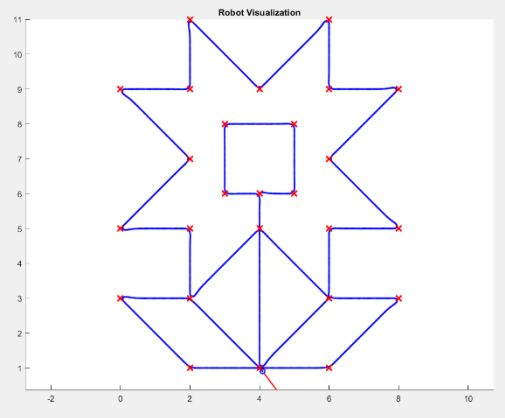


Análisis: En este ejercicio, observemos que para realizar el “trazado” de la figura necesitamos realizar trayectorias más rectas, evitando tanto como sea posible dibujar curvas que puedan afectar en la visualización de la figura. Para lograr esto, fue necesario hacer que el robot pudiera realizar giros o cambios de orientación de una manera rápida. Esto se logró aumentando la velocidad angular de tal forma que fuera por mucha mayor a la lineal.

Ejercicio 2 - Flor:

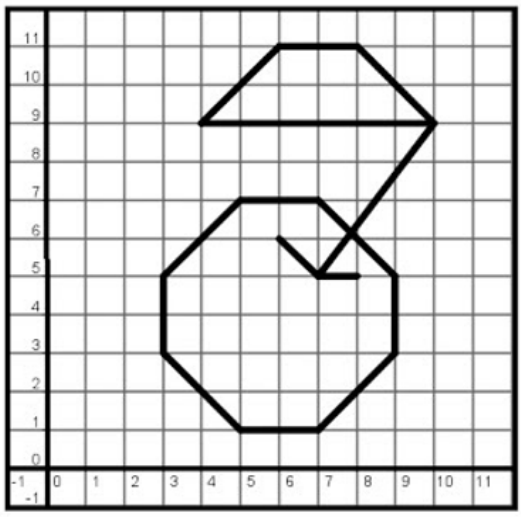


```
sampleTime = 0.05;

tVec = 0:sampleTime:131;

initPose = [4; 6; 0];

waypoints = [4,6; 5,6; 5,8; 3,8; 3,6;
4,6; 4,1; 6,3; 6,5; 8,5; 6,7; 8,9; 6,9;
6,11; 4,9; 2,11; 2,9; 0,9; 2,7; 0,5;
2,5; 2,3; 4,5; 6,3; 8,3; 6,1; 2,1; 0,3;
2,3; 4,1];

controller.LookaheadDistance = 0.2;

controller.DesiredLinearVelocity = 0.53;

controller.MaxAngularVelocity = 10;
```

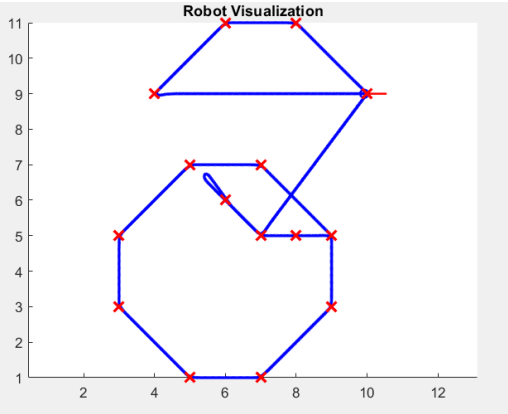


Análisis: Para este segundo ejercicio mantuvimos las velocidades en los mismos valores para lograr nuevamente un buen trazado de la figura. No obstante, además de esto, para esta figura se modificó el valor de *LookAheadDistance* para lograr que el robot pasara apropiadamente en cada uno de los landmarks.

Ejercicio 3 - Manzana:



```
sampleTime = 0.05;

tVec = 0:sampleTime:83.4;

initPose = [9; 5; 3*pi/4];

waypoints = [9,5; 7,7; 5,7; 3,5; 3,3;
5,1; 7,1; 9,3; 9,5; 8,5; 7,5; 6,6; 7,5;
10,9; 8,11; 6,11; 4,9; 10,9];

controller.LookaheadDistance = 0.2;

controller.DesiredLinearVelocity = 0.53;

controller.MaxAngularVelocity = 30;
```



Análisis: Para este último ejercicio inicialmente mantuvimos los valores del ejercicio anterior (a excepción de los waypoints e initpos) y observamos que el robot pasaba apropiadamente por cada uno de los landmarks. No obstante, podemos observar que en la raíz de la hoja de esta fruta no logramos que quedara totalmente como una línea recta, esto debido a la forma en la que trazamos la trayectoria. Con el fin de realizar esa corrección, intentamos aumentar el valor de la velocidad angular pero el resultado fue el mismo. Este comportamiento o dificultad para trazar ciertas partes de la figura se debe a un cambio brusco (de +180°) en la orientación del robot, por lo que independientemente de la velocidad es muy probable que no se logre demasiada exactitud. En su lugar, podríamos implementar una modificación en el código afectando en el movimiento del robot, de tal forma que si el ángulo es demasiado grande (acorde a un umbral), entonces priorice la “corrección” del ángulo antes de comenzar a avanzar (velocidad lineal).