Laboratorio de Robots Móviles
Practica No. 2

Comportamientos Reactivos

Objetivo:

*Familiarizar al alumno con los comportamientos reactivos

Duración: Dos semanas

1.- Descargue de la página: http://biorobotics.fi-p.unam.mx/robotics-courses/robots-moviles

el archivo denominado catkin_ws.tar.gz, el cual contiene el sistema que se utilizará para simular el robot móvil usando ROS, código en C++ y Python.

Descomprimir archivo rosRobotics.tar.gz en /home/<usuario>, antes de hacer esto deberá asegurarse de que no exista algún folder llamado catkin_ws, si existiera, cambiar su nombre, posteriormente:

- Colocarse en el directorio *catkin_ws con:* $ cd ~/catkin_ws
- Compilar el proyecto con: $ catkin_make install
- *Dar permisos de ejecución al archivo* start_ros.sh, *si se necesitase, con:*
  $ sudo chmod +x start_ros.sh
- *Dar permisos de ejecución al archivo* setup.bash, *si se necesitase,* con:
  $ sudo chmod +x devel/setup.bash
- Dar permisos de ejecución al archivo *GUI_ROS.py, si se necesitase,* con:
  $ sudo chmod +x src/svg_ros/src/GUI/GUI_ROS.py
- Iniciar al rosmaster, colocarse en /home/<usuario> y ejecutar: $. ~/catkin_ws/devel/setup.bash después: $ roscore
- Abrir una xterm y colocarse en el directorio ~/catkin_ws y ejecutar el archivo con: $ ./start_ros.sh

  Aquí se abrirán 7 terminales las cuales contienen los siguientes nodos: base_node, GUI_ROS, sensor_node, motion_planner_node, light_node, inputs. Además se abrirá la interfaz gráfica en donde se introducen datos y la interfaz donde navega el robot.

Esta interfaz gráfica muestra el resultado de la simulación del código en motion_planner_ROS.cpp que se encuentra en el directorio ~/catkin_ws/src/svg_ros/src/motion_planner, este código usa algoritmos de máquinas de estados que se encuentrán en ~/catkin_ws/src/svg_ros/src/state_machines. Seleccione con el botón izquierdo del mouse la posición inicial del robot y con el derecho la final, después observe el comportamiento del robot. Seleccionando de nuevo el botón derecho del mouse el robot tomará su última posición como origen y el destino la posición del botón derecho. Las coordenadas (0,0) del mapa se encuentran en el lado inferior izquierdo de la figura. Familiarícese con el funcionamiento de la interfaz gráfica con diferentes configuraciones y comportamientos ejecutados por el nodo motion_planner_ROS.

En el campo “Behaviour Selection” de la interfaz ROS_GUI_ROBOTS se puede seleccionar el algoritmo que ejecuta el robot para evadir obstáculos, como el que se muestra en la figura, y dirigirse a una fuente luminosa, modifique la selección y observe el comportamiento del robot, entienda el funcionamiento de este código.

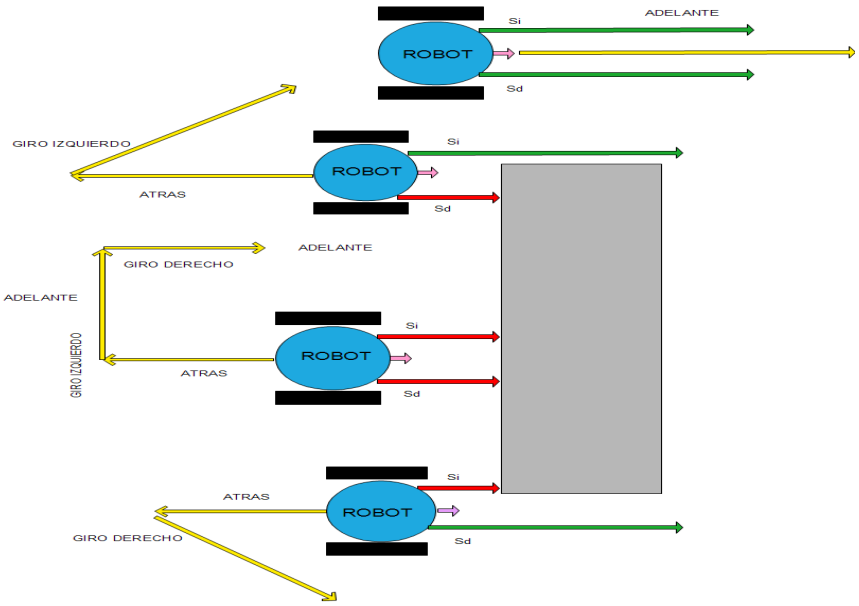

Figura 1. Robot Móvil que evade obstáculos

2.- En el apéndice A se muestra el mapa simbólico en donde el robot navega, los objetos se representan con polígonos. Este tipo de archivos tienen terminación *.wrl y se encuentran en el directorio: ~/catkin_ws/src/svg_ros/src/data

En este apéndice se muestra las siguientes directivas:

2.1. “dimensions” indica las dimensiones del medio ambiente, las cuales están indicadas en el sistema métrico decimal. En el siguiente ejemplo se presentan las dimensiones del medio ambiente “room” de 1m x 1m:

```
( dimensions room 1.000 1.000 )
```

2.2. “polygon” indica los vértices de un polígono, se indica si es de tipo obstáculo o pared, después viene el nombre del obstáculo y a continuación los vértices. Los vértices de los polígonos se indican con las coordenadas Xi y Yi, ordenados hacía el sentido horario de las manecillas del reloj:

```
( polygon obstacle obs1  .40 .55 .60 .55 .60 .35 .40 .35 )

( polygon wall wall1  0.0 0.0 0.0 1.0 0.01 1.0 0.01 0.0 )
```

2.3. Se pueden incluir comentarios o comentar una linea si se coloca un “;” al principo del renglón. Ejemplo:

```
; * File: room.wrl       *
```

En el apéndice B se muestra una parte del archivo de datos generado por la simulación. Este archivo de datos es utilizado por la interfaz gráfica para mostrar su resultado, este tipo de archivos tienen terminación *.raw

En este apéndice se muestra las siguientes directivas:

1. “radio_robot” indica el radio del robot, en el ejemplo siguiente se indica que el radio del robot es de 3 cm:

```
( radio_robot 0.030000 )
```

2. “robot” indica la posición del robot, que en este ejemplo se llama Justina, con la posición X, Y y el angulo del robot con respecto al eje x, el cual se encuentra en la parte inferior de la figura.

```
( robot Justina 0.080000 0.332500 0.000000 )
```

3. “sensor” indica el tipo de sensor utilizado, así como el número de sensores, el rango angular del sensor, la posición de la primera lectura con respecto al centro del robot, después vienen los valores obtenidos por los sensores.

En el siguiente ejemplo se obtuvieron las lecturas de 2 sensores láser, con un rango angular de 0.400 radianes, posición inicial -0.2000 (lado derecho del robot) y las lecturas de 30 y 28.5697 centímetros:

```
( sensor laser 2 0.400000 -0.200000 0.300000 0.285697 )
```

Los archivos *.wrl y *.raw se encuentran en el directorio ~/catkin_ws/src/svg_ros/src/data

El tamaño del robot, números de sensores, tipo de sensores, posición de estos, etc, se incluyen como parámetros cuando se ejecuta el nodo motion_planner_ROS y los cuales son colocados por la interfaz gráfica.

3. Modifique el código de ~/catkin_ws/src/svg_ros/src/motion_planner_ROS.cpp para que el robot tenga un comportamiento que cuando encuentre un obstáculo lo rodee completamente, pruebe su algoritmo con el archivo room.raw. Para compilar su código hacer lo siguiente:

- *Colocarse en el directorio* ~/catkin_ws/build
- *Compilar de nuevo el proyecto con:* $ make, aquí solo se compilaran los últimos archivos modificados.

Para probar el código, sin inicializar todos los nodos :

- Elimine la terminal con el nodo del motion_planner_ROS
- Abra una nueva terminal y ejecutar:

  $ export PYTHONPATH=$PYTHONPATH:/usr/lib/python2.7/dist-packages
  $. ~/catkin_ws/devel/setup.bash
  $ roscd svg_ros/src/data

  Ejecutar de nuevo el nodo con: $ rosrun svg_ros motion_planner_ROS

Si es necesario terminar la ejecución de este nodo para hacer más pruebas, teclee $ps para saber el número PID-Number del proceso de motion_planner_ROS y terminarlo con $kill -kill PID-Number

4. Haga un comportamiento para que el robot siga paredes.

APENDICE A
Mapa Simbólico room.wrl

```
; ********************************************************************
; * File: room.wrl                                                   *
; *       Definition of the forbidden and allowed areas in the Robot's *
; *       world. These areas are derivated from the objects in the   *
; *       world.                                                     *
;********************************************************************
( dimensions room 1.000 1.000 )
( polygon obstacle obs1  .40 .55 .60 .55 .60 .35 .40 .35 )
( polygon wall wall1  0.0 0.0 0.0 1.0 0.01 1.0 0.01 0.0 )
( polygon wall wall2  0.0 0.99 0.0 1.0 1.0 1.0 1.0 0.99 )
( polygon wall wall2  0.99 1.0 1.0 1.0 1.0 0.0 0.99 0.0 )
( polygon wall wall2  1.0 0.01 1.0 0.0 0.0 0.0 0.0 0.01 )
```

APENDICE B
Parte del archivo room.raw generado por la simulación

```
( radio_robot 0.030000 )
( robot Justina 0.080000 0.332500 0.000000 )
( sensor laser 2 0.400000 -0.200000 0.300000 0.300000 )
( robot Justina 0.120000 0.332500 0.000000 )
( sensor laser 2 0.400000 -0.200000 0.300000 0.285697 )
( robot Justina 0.160000 0.332500 0.000000 )
( sensor laser 2 0.400000 -0.200000 0.300000 0.244883 )
...
…
...
```