AI FOR FUTURE

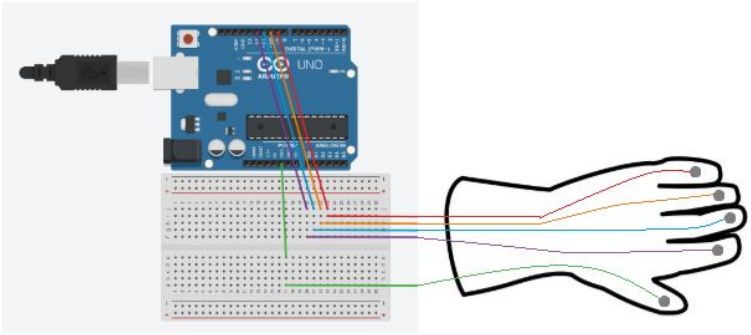
Nombre de los integrantes:

José Ricardo Romo Gutierrez
Fabián Esteban Beltrán Gómez
Sergio Andrés Robles Serrano

1. **Sistema controlador de interacción externo al computador.**

La solución a este reto se divide en dos partes, una es el dispositivo físico externo al computador y otra es el nodo implementado en el paquete de pacman_controller.
Para la primera parte se decidió fabricar un guante con contactos metálicos en la yema de los dedos como dispositivo controlador. Este guante se conecta a través del computador con un arduino que se comunica por puerto serial. El dedo pulgar está conectado a tierra (GND) y los otros 4 dedos a un pin digital del arduino. La lógica implementada en la programación del microcontrolador, detecta los cambios de estado de los pines digitales, estos se mantendrán en HIGH (ALTO) siempre que no hagan contacto con el dedo pulgar. Este cambio genera una respuesta del arduino, mandado un caracter numerico al puerto serial (0,1,2,3,4).
La segunda parte se lleva a cabo dentro del paquete de pacman_controller, este estará leyendo constantemente el puerto serial en busca de nuevos cambios del dispositivo externo. Una vez detectado el carácter numérico adecuado, se procede a publicarlo al topic de /pacman_Actions0 para ser interpretado por /pacman_world (en caso de jugar con un pacman, /pacman_Actions1 en caso de jugar con los dos).
La manera en que están distribuidos los controles en el guante se realiza de la siguiente manera:

Dedo índice = arriba.
Dedo medio = derecha.
Dedo anular = izquierda.
Dedo meñique = abajo.



Ejecución del controlador manual:
Con el simulador de pacman_world abierto y corriendo, la manera de ejecutar el controlador será la siguiente:

- dirigirse a la carpeta Reto 1 suministrada en los archivos para la competencia
- abrir una terminal y ejecutar los siguientes comandos:
- rm -rf ~/catkin_ws/src/pacman-controller && cp -r pacman-controller/ ~/catkin_ws/src/
- cd ~/catkin_ws
- source devel/setup.bash
- rosrun pacman-controller pacman_controller_py.py

Una vez ejecutado el último comando, la consola mostrará una lista de los dispositivos conectados a algún puerto serial del computador. Cuando encuentre el del arduino lanzará desde la consola con un mensaje de esta manera “Controller finded: /dev/ttyACM0”, informando que ya se puede utilizar el controlador manual.

2. Agente controlador autónomo:

Para la implementación del agente controlador se intenta separar los estados del juego en campos generales: adversarial (enemigos encerrados, enemigos libres), muchas galletas o pocas galletas, cantidad de enemigos, etc. En cada campo o estado de juego se realiza un comportamiento determinado.

Cuando el mapa es un laberinto con una sola galleta y sin enemigos o con enemigos encerrados, se ejecuta un algoritmo de búsqueda por profundidad para cada combinación de los movimientos disponibles (arriba, abajo, izquierda, derecha).
Cuando el mapa es un laberinto y tiene más de una galletas pero no demasiadas, se genera un grafo de todas las galletas y el pacman y se utiliza algoritmos genéticos para encontrar la ruta más óptima.

Cuando se tiene un estado de juego adversarial, el pacman se mueve dependiente de un agente automada. Este agente se basa en un algoritmo minmax. Este algoritmo supone que los enemigos se mueven de manera óptima, tratando de minimizar mi score (puntaje). El algoritmo lo que busca es obtener el mejor puntaje suponiendo que los fantasmas se moverán (luego del pacman moverse) buscando minimizar mi puntaje. Es decir, el pacman busca el mejor puntaje que puede obtener cuando los fantasmas se muevan de manera óptima. Nota: el puntaje es dado por una funcion de evaluacion creada por el grupo.

Debilidades:
Cuando el mapa tiene pocas galletas,y estado del juego es adversarial el pacman intenta huir de los fantasmas, y a veces no ve la galletas cercanas por darle mucho peso a la huida de los fantasmas.
Cuando se tiene un mapa de dimensiones grandes, se encuentran menos cantidades galletas y el estado del juego no es adversarial, el sistema autónomo es menos eficiente.

Algunos trabajos en los que se basaron las soluciones:

- ALGORITMO GENETICO: https://github.com/CodingTrain/website/tree/master/CodingChallenges/CC_035_TSP/CC_35.4_TSP_GA

- AGENTE AUTOMATA: https://github.com/douglaschan32167/multiagent/blob/master/multiAgents.py

Con el simulador de pacman_world abierto y corriendo, la manera de ejecutar el controlador será la siguiente:

Ejecución del controlador autónomo:

- dirigirse a la carpeta Reto 2 suministrada en los archivos para la competencia
- abrir una terminal y ejecutar los siguientes comandos:
- rm -rf ~/catkin_ws/src/pacman-controller && cp -r pacman-controller/ ~/catkin_ws/src/
- cd ~/catkin_ws
- source devel/setup.bash
- rosrun pacman-controller pacman_controller_py.py
-

3. Solución de cooperación humano/máquina:

Para la ejecución del controlador

Con el simulador de pacman_world en modo challenge abierto y corriendo un mapa diseñado para jugar con ambos pacman, la manera de ejecutar los controladores será la siguiente:

Para este reto se deberán abrir dos terminales y en cada una ejecutar los siguientes comandos

Ejecución del controlador autónomo:

- dirigirse a la carpeta Reto 3 suministrada en los archivos para la competencia
- abrir una terminal y ejecutar los siguientes comandos:
- rm -rf ~/catkin_ws/src/pacman-controller && cp -r pacman-controller/ ~/catkin_ws/src/
- cd ~/catkin_ws
- source devel/setup.bash
- rosrun pacman-controller pacman_controller_py.py

Ejecución del controlador manual:

- cambiar a la segunda terminal y ejecutar los siguientes comandos:
- cd ~/catkin_ws
- source devel/setup.bash
- rosrun pacman-controller pacman_controller2_py.py