Memoria Práctica 3

Integrate path-planning and task-planning

2020

David García Ortiz

UAH

25/05/2020

# Introducción

El objetivo de esta práctica es el de comparar los algoritmos de Dijkstra, A\* y Theta\*; aplicar heurística al algoritmo Theta\* para así comprobar los cambios y el impacto que tiene aplicar dicha heurística. Para las simulaciones se usará R2P2. Finalmente, realizará la integración de path-planning y task-planning.   
Esta práctica se ha realizado de manera individual, aunque si bien se ha intercambiado información con algunos compañeros para resolver dudas en ciertos puntos de esta.

# Instalación

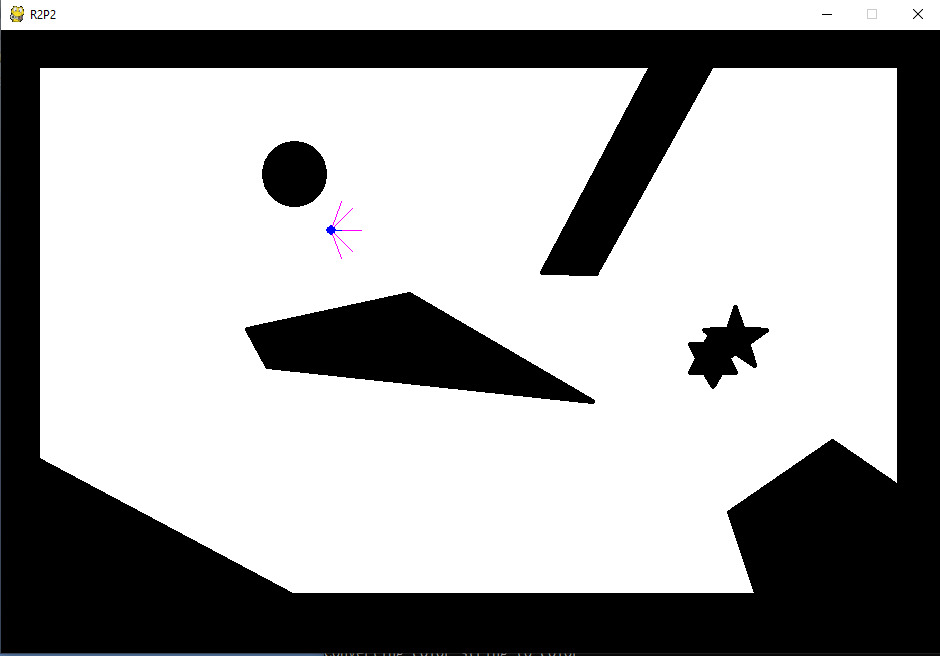
Una vez se ha instalado R2P2, Anaconda, y se ha creado el respectivo enviroment con Conda, y tras realizar las convenientes actualizaciones a las versiones de python y de pip que se indican en la guía, ya se tiene todo a punto para ejecutar el simulador.

# Ejercicio1

La primera prueba la haremos con el siguiente comando desde la consola de comandos de Conda en la carpeta de r2p2:



El resultado de ejecutar este comando será el siguiente:



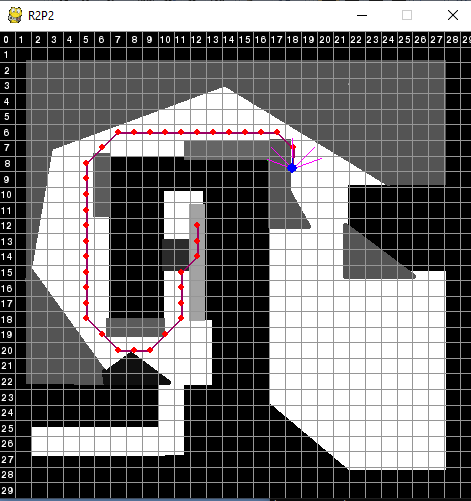
Se trata de un simulador de un robot con sensores, el cual podemos controlar con las flechas del teclado.

# Ejercicio2

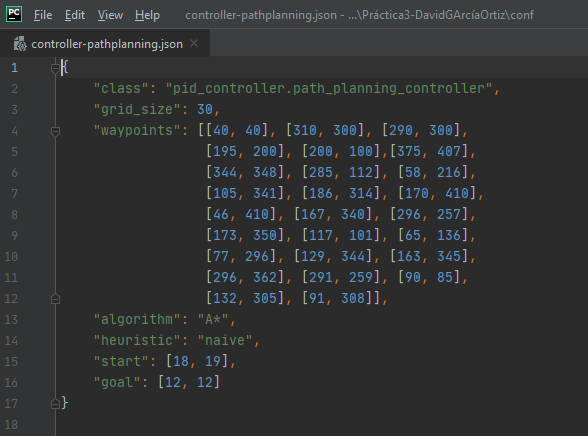
A continuación, se ejecutará el escenario Path-planning como se indica en la práctica, con el siguiente comando:



El resultado será el siguiente:



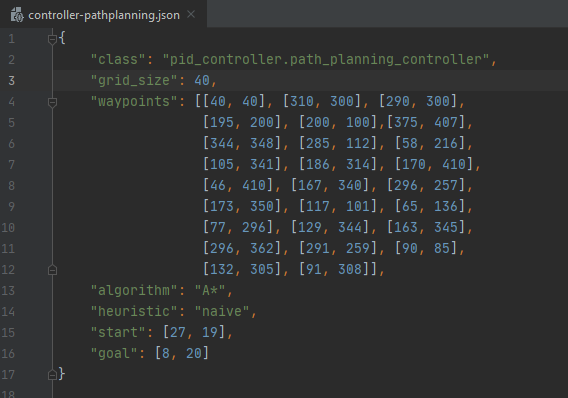
El robot tratará de seguir una ruta preestablecida desde el punto inicial hasta el final, sorteando los obstáculos. El punto de inicio y de fin viene establecido dentro de la clase controlador que hemos seleccionado:



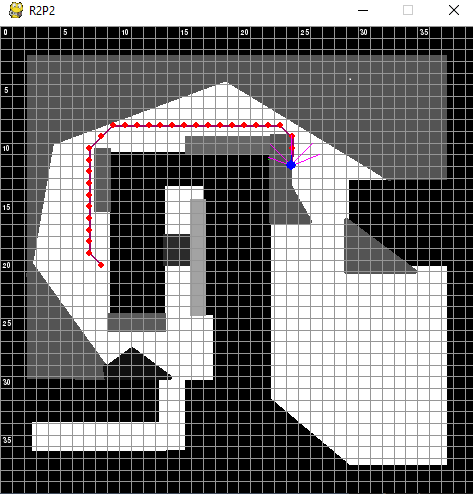
Para la visualización y modificación de los archivos de estará utilizando a lo largo de toda la práctica en entorno de Python: *PyCharm.*

# Ejercicio 3

Como se pide, se cambiará el punto de origen de (18,19) a (27,19), así como la meta se cambiará de (12,12) a (8,20). También se ampliará el tamaño de la rejilla a 40. El resultado del .json es el siguiente:



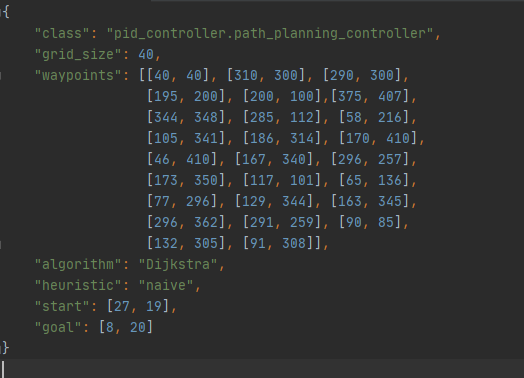
Al ejecutar de nuevo el Path-planning obtendremos lo siguiente:

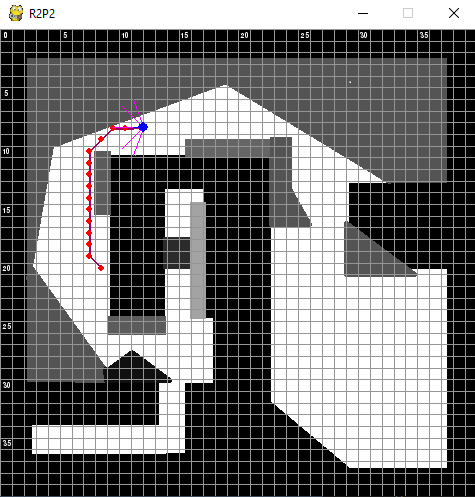


Como se puede apreciar en la imagen, se han producido los cambios indicados.

# Ejercicio 4

Si en el campo de “algorithm” del controlador cambiamos “A\*” por “Dijkstra” el simulador se ejecutará teniendo en cuenta este algoritmo:





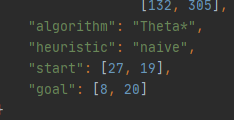
# Ejercicio 5

Como el algoritmo Theta\* no está implementado, se implementará sirviéndonos del pseudocódigo que se nos ofrece en la práctica. Ya que es muy similar al algoritmo de A\*, se copiará ese archivo y se realizarán las respectivas modificaciones sobre este, como ha sido añadir la función *LineOfSight*, modificar la función *aStar* y adecuarla a la nueva función *tStar*, donde se usa *LineOfSight*, además de añadir al final la función de *check*, para devolver el color del grid.

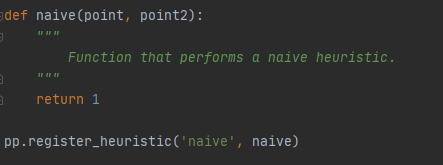
Este archivo se encontrará en la carpeta de r2p2 bajo el nombre de *thetaStar*.

# Ejercicio 6

La heurística elegida por defecto la podemos observar de nuevo en el .json del controlador de Path-planning. En este caso la heurística por defecto es “Naive”.



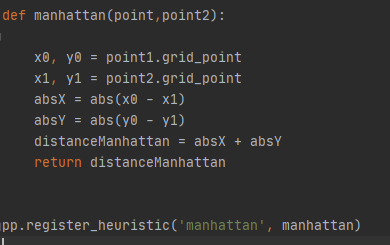
Podemos ver en qué consiste esta heurística si accedemos al archivo “heuristics.py” de la carpeta r2p2:



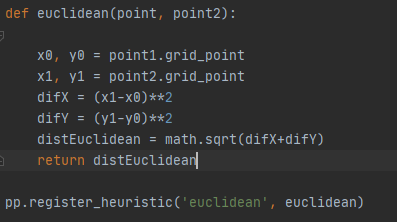
Como se aprecia, devuelve como coste entre dos puntos siempre un 1.

# Ejercicio 7

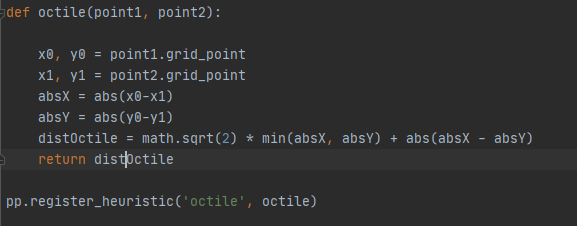
En este ejercicio se implementarán las heurísticas de *Manhattan*, *Octile* y *Euclidean* para los algoritmos *A\** y *Theta\**. Para ello modificaremos el archivo de heurísticas indicado en el ejercicio anterior.  
Se empezará implementando la heurística de *Manhattan*, cuyo resultado será el siguiente:



La siguiente en implementarse será la heurística *Euclidean*, cuyo resultado es el siguiente:



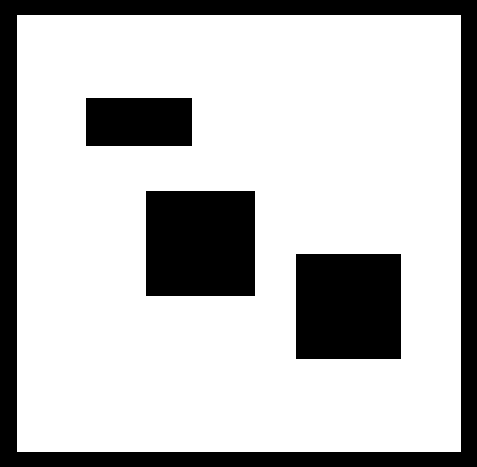
La ultima en ser implementada, y la cual se deberá añadir a dicho archivo, será la heurística *Octile*. El resultado de su implementación será el siguiente:



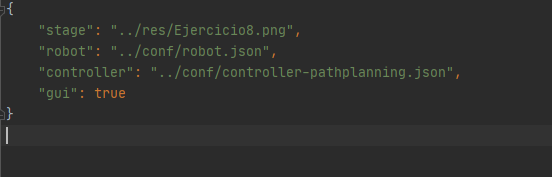
No se adjuntarán capturas de su funcionamiento en este ejercicio, ya que se utilizarán todas y se adjuntaran las imágenes correspondientes en el siguiente ejercicio:

# Ejercicio8

Se pide que se cree un escenario en concreto. En mi caso, lo he realizado a paint de manera aproximada, y el resultado ha sido el siguiente:



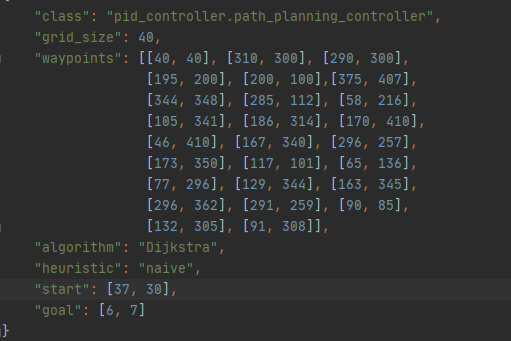
Se puede encontrar en la carpeta “res” bajo el nombre de “Ejercicio8”. Para utilizarlo, cambiaremos el fichero “scenario-pathplanning.json” y elegiremos nuestra imagen:



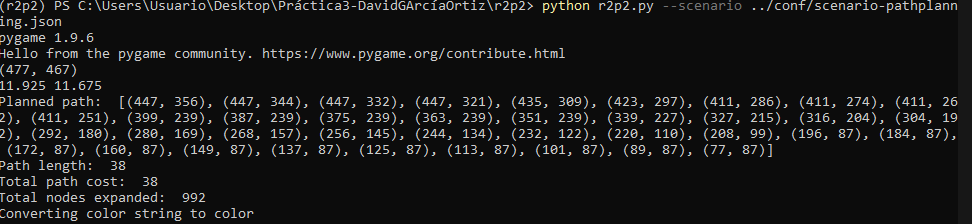
A continuación se procederá aprobar las diferentes heurísticas con los diferentes algoritmos en el nuevo escenario que acabamos de crear.

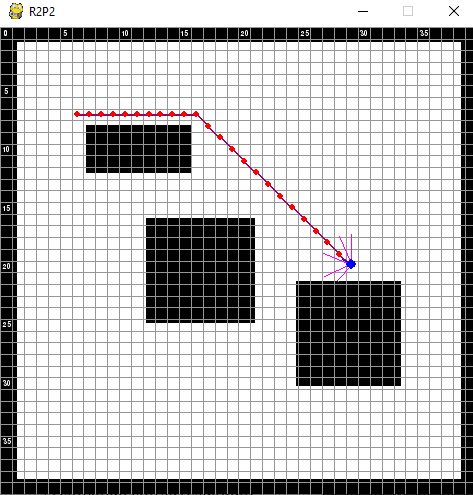
## Algoritmo de Dijkstra heurística naive

Para esta ejecución, dejamos el controller tal que así:

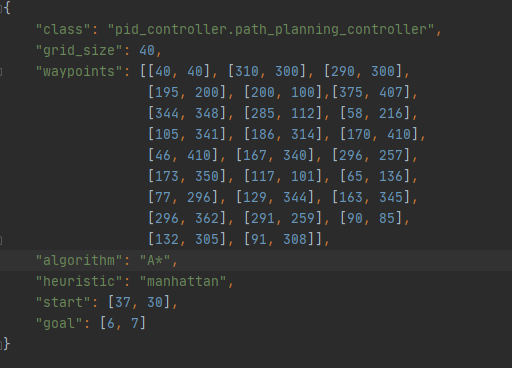


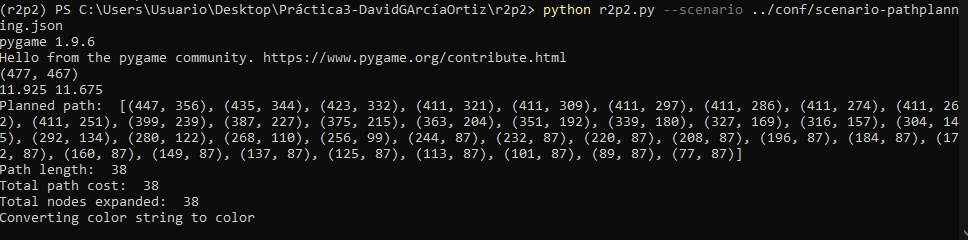
El comando con su resultado:

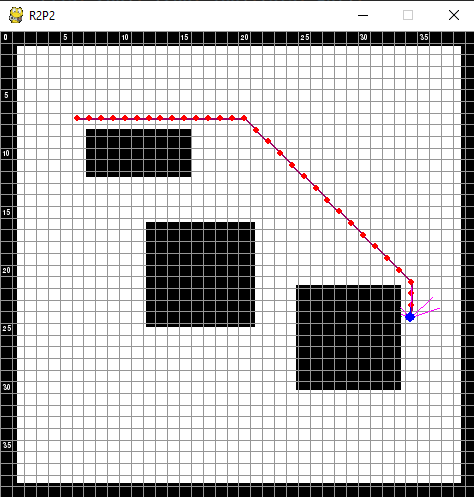




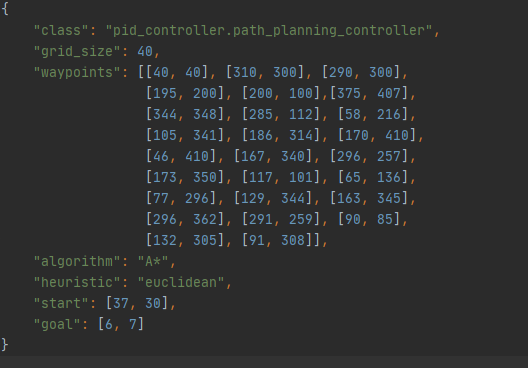
## Algoritmo A\* heurística Manhattan

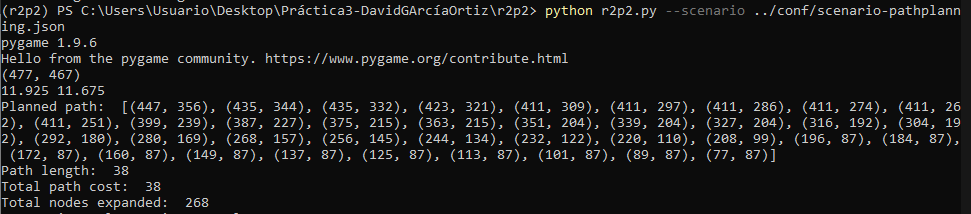


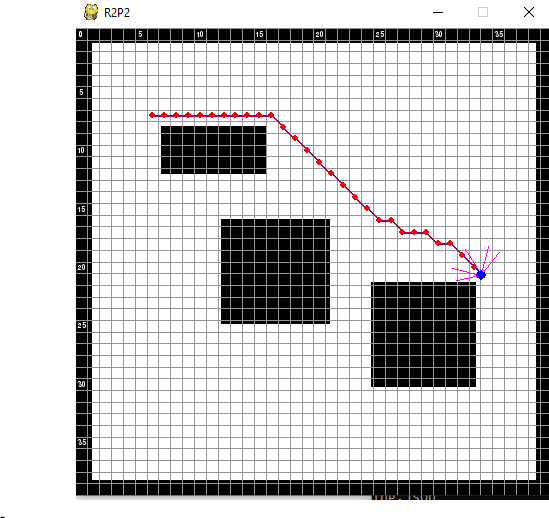




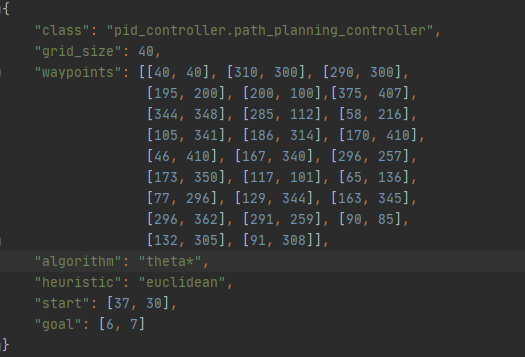
## Algoritmo A\* heurística Euclidean

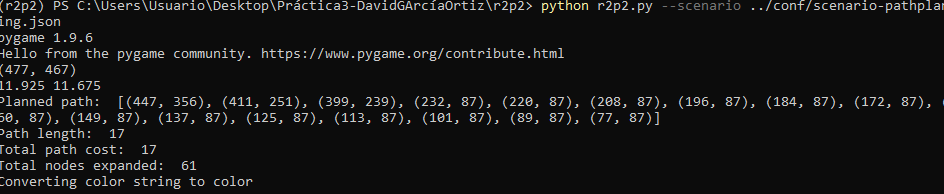


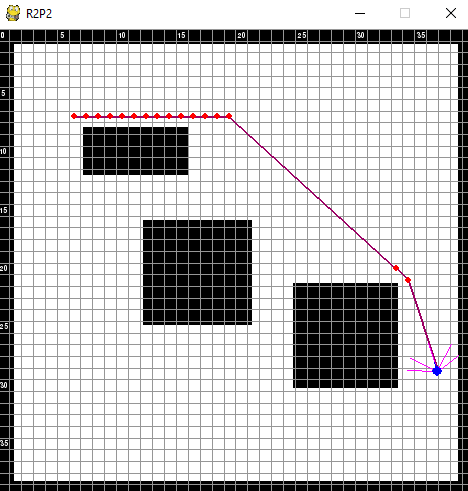




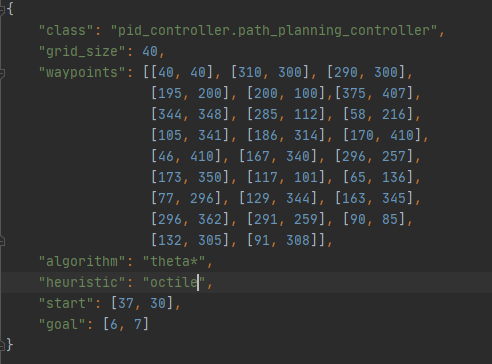
## Algoritmo Theta\* heurística Euclidean

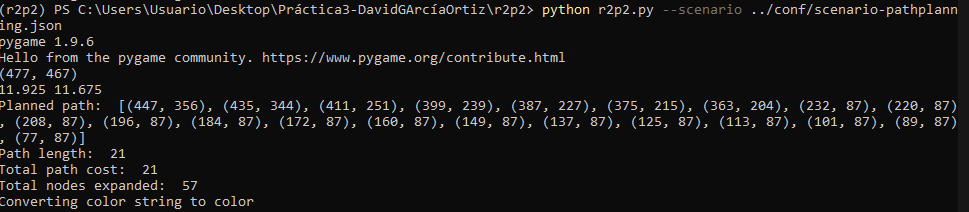


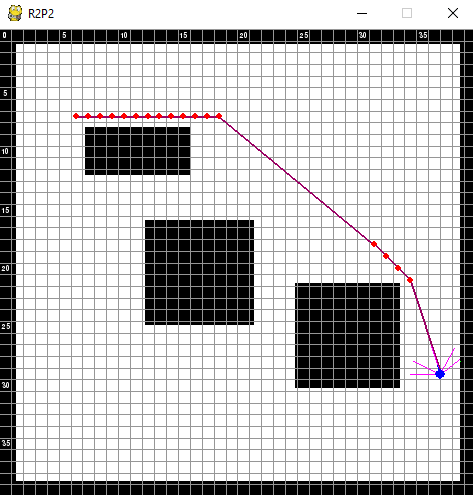




## Algoritmo Theta\* heurística Octile







Con este número de pruebas creo que es suficiente para tener unas ideas básicas de cómu funciona cada algoritmo con cada heurística y cuál es el rendimiento de estos.

A continuación se pasará con la segunda parte de la práctica, la cual se trata de la integración.

# Integración