Simulación de clasificación de piezas de lego con el robot UR3e

Andrea Casal Gutiérrez

Resumen

Utilizando el programa RoboDK se ha realizado una simulación en la que el robot UR3e, situado en una mesa, recoge objetos situados en la misma mesa y los clasifica en una estantería. En dicha estantería habrá 9 posiciones distintas para la clasificación. El robot situará el objeto en cada una en función de lo que le indique el usuario a través del HMI realizado.

1. Introducción

En este trabajo se pretende que un robot colaborativo clasifique piezas de lego situadas en una mesa en diferentes apartados de una estantería. El robot es el disponible en el laboratorio de la universidad: UR3e de Universal Robots. También se encuentra en el laboratorio el entorno de trabajo con la mesa, la estantería y las piezas de lego de diferentes colores.

Aunque el robot trae incorporado un software que facilita notablemente su programación, se ha decidido utilizar el programa RoboDK para realizar el programa que utilizará el robot para la clasificación. Se ha decidido así para aprovechar la ocasión y explorar las múltiples posibilidades que ofrece este programa.

Primeramente, se hizo un estudio de las múltiples posibilidades que ofrece RoboDK, seguramente sin llegar a descubrirlas todas. Consta de librerías de múltiples objetos y robots, entre ellos el UR3e utilizado para este trabajo, y librerías de programación de robots propias que facilitan mucho el desarrollo software. También permite la programación de múltiples tipos de robots, así como serían ABB, Kuka, Universal Robots,… Por último destacar que también permite la programación en varios lenguajes, como por ejemplo MATLAB o Python.

Aprovechando esta información y como segundo paso del proyecto, se han realizado los modelos 3D de la estantería y mesa a medida, utilizando Autodesk Fusion 360. También se ha encontrado un modelo de lego como los del laboratorio en la librería de objetos de RoboDK. He importado todos estos objetos, en la posición que se encuentran en el laboratorio, al entorno de simulación en una carpeta llamada “Entorno”, todos como archivos tipo STL. Además, también he importado al entorno el robot desde la librería, situándolo en el centro de la mesa para que alcance todas las posiciones de clasificación y los legos a clasificar. Dicho robot debe constar también de una herramienta para atrapar los lego, por lo que se le ha incorporado también el modelo de la presente en el laboratorio para dicho robot, la herramienta RGI-14 de DH-Robotics. Se puede ver todo ello junto en la Fig. 1, con los números que se han asignado para la clasificación.

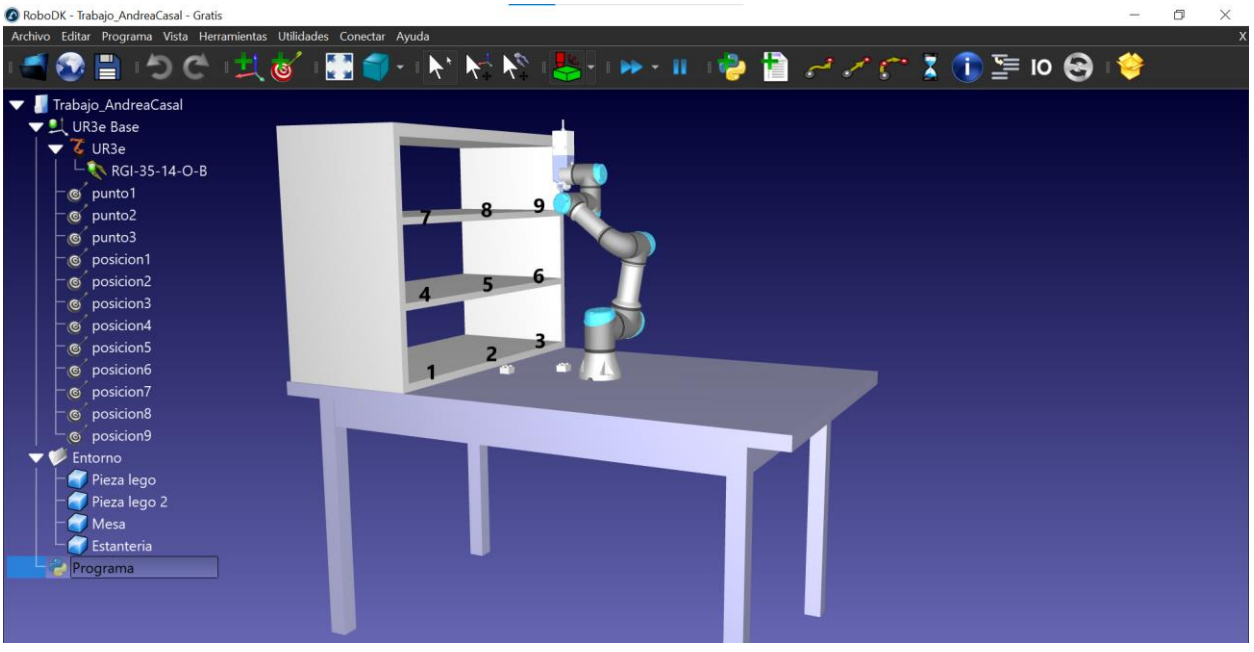

Fig. 1. Entorno de simulación con las posiciones de clasificación de los lego indicadas con el número correspondiente.

1.1. Programa de la simulación

Una vez creado el entorno de trabajo, se han explorado las posibilidades que ofrecía la librería para programación en Python y se ha utilizado para crear el programa. Con RoboDK se pueden crear *Targets*, que son puntos objetivo a los que desplazar el robot. Se han creado entonces targets llamados *posicionI*, siendo *I* el número que representa en la Fig. 1 la posición a la que se debe mover el robot para clasificar. Para evitar la colisión con la estantería al moverse a dichos puntos, existen también unos puntos de paso (*punto1*, *punto2* y *punto3*) a los que el robot se debe de desplazar antes de dirigirse a cada una de las posiciones de clasificación. Por ejemplo, si se quiere desplazar al target *posicion1*, primero debe ir al *punto1*, luego a *posicion1* para soltar el lego y luego a *punto1* de nuevo. En la Tabla 1 se recogen las posiciones que hay en cada balda de la estantería y el punto de paso para cada balda.

Tabla 1. Tabla con las posiciones de clasificación y puntos de paso para cada balda de la estantería.

Balda de la estantería	Punto de paso	Posiciones		
Balda baja	*punto1*	*posicion1*	*posicion2*	*posicion3*
Balda intermedia	*punto2*	*posicion4*	*posicion5*	*posicion6*
Balda alta	*punto3*	*posicion7*	*posicion8*	*posicion9*

En la realidad, sería ideal que el robot tuviese incorporada una cámara que, mediante procesamiento de imagen, determinase dónde se debe clasificar el lego en función de su color o forma. Ya que esto no entra dentro del ámbito de la asignatura, se ha realizado con la librería Tkinter un HMI que permita al usuario determinar la clasificación de la pieza. Al lanzar el programa de Python, aparecerá dicho HMI (ver Fig. 2). El usuario deberá especificar entonces el número de la clasificación y, posteriormente, pulsar el botón *Clasificar* para que el robot coja el lego y lo clasifique en la posición seleccionada.

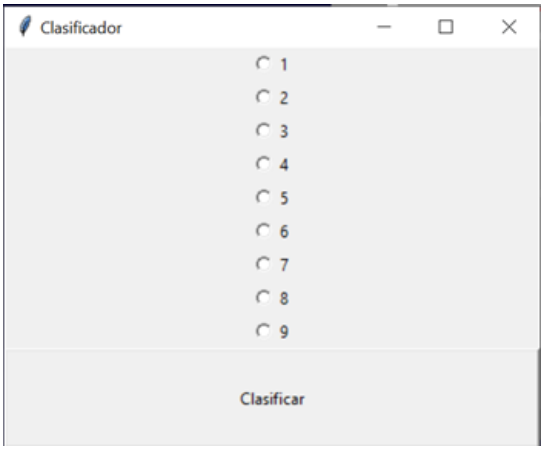

Fig. 2. HMI que aparece al lanzar el programa Python de la simulación para que el usuario seleccione la posición en la que se debe clasificar el lego y ordene el comienzo de la clasificación.

Por otra parte, para clasificar el objeto tiene que detectarlo y cogerlo primero. Sin embargo, el robot no puede detectar los lego ya que, una vez más, no se ha incorporado una cámara al robot. Por ello, se han establecido una serie de puntos en el otro extremo de la mesa, que es donde se encontrarían los lego. Cada vez que el usuario pulse el botón *Clasificar*, el robot seleccionará aleatoriamente uno de esos puntos y creará un target llamado *objeto* para desplazarse a dicho punto y simular el movimiento que haría para recoger el lego. Una vez que el robot se haya desplazado al punto, se moverá de nuevo al *punto3* y se eliminará el target *objeto*. Finalmente, para terminar de utilizar el programa no hay más que cerrar la ventana del HMI situada en el extremo superior derecho de la ventana.

2. Componentes mecánicos

Como componentes mecánicos se podrían considerar la mesa, la estantería y las piezas lego, aunque ninguna de estos se mueva. Se enumeran a continuación las medidas generales de cada uno de los componentes:

- Mesa: 1350x90x750 cm.
- Estantería: 400x900x720 cm.
- Lego: lego cuadrado 2x2x2 cm.

3. Componentes eléctricos y de control

Como componentes de control se identifican el robot UR3e y la herramienta RGI-14 de DH-Robotics. Se pueden ver ambos juntos en la Fig. 3.

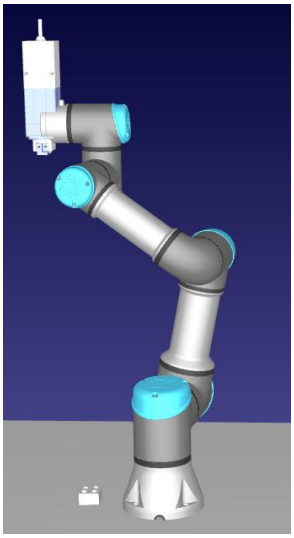

Fig. 3. Robot UR3e con la herramienta RGI-14 de DH-Robotics.

4. Explotación y conclusiones

Se ha creado entonces un programa que clasifica, en función de lo que desee el usuario, piezas de lego. Sin embargo, han quedado algunas tareas que se podrían realizar siguiendo con este trabajo. Como la herramienta no es más que un modelo STL, y no una herramienta con movimiento, se podría investigar cómo adaptarla para que agarrase los objetos en la simulación. Sin embargo no se ha hecho principalmente porque no tiene el suficiente tamaño para agarrar las piezas de lego del laboratorio. Se deberían conseguir piezas más pequeñas o una herramienta más grande. Además, aunque se realizaron pruebas cargando al robot otros programas Python más sencillos a través de RoboDK, no se llegó a cargar este programa concreto, aunque no debería ser complicado. Colocando al robot en la posición correcta de la mesa, cargar este programa y reproducir la simulación en el laboratorio, como ya se hizo durante el curso con otros programas, debería ser sencillo.

Como conclusión, cabe destacar que RoboDK es un software tremendamente útil y con un gran abanico de posibilidades. No solo por sus librerías, sino por su flexibilidad en cuanto a tipos de robot que maneja, tipos de archivos que acepta y lenguajes de programación con los que se puede trabajar.

References

[1] DH-Robotics. https://en.dh-robotics.com/service/download###.
[2] Librería Python RoboDK. https://robodk.com/doc/en/PythonAPI/robodk.html.
[3] Librería robot y objetos RoboDK. https://robodk.com/library?RDK&utm_source=RoboDK&utm_medium=Software&Lang=es&PC_OS=WIN64&ProgName=RoboDK&ProgVer=5.4.1&ProgDate=2022-02-25&PCID=387328b524ac79d0536897e9b772f867&SH=376361373735636138633264323433643265366634663239613139643132316 5&LNH=477261746973.