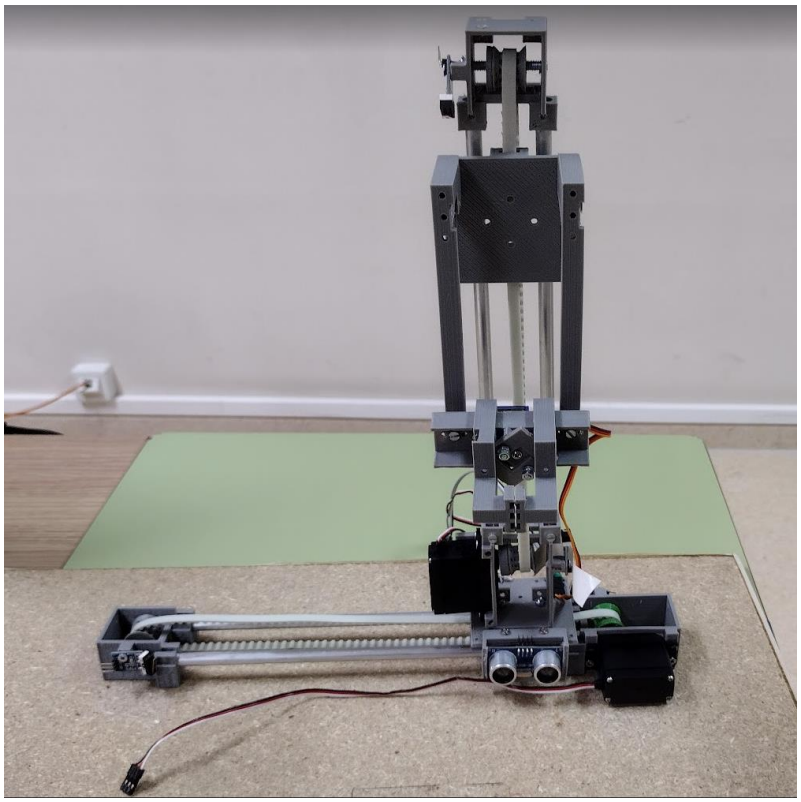
Robots manipuladores



Práctica 5 – Actividades (v1.6.1 octubre 2022)

Software para robots

Cristian González García

gonzalezcristian@uniovi.es

Basado en el material original de Jordán Pascual Espada

Índice

El total de las actividades principales tienen un valor de 1,8 puntos y 4,25 puntos de ampliaciones dentro del **bloque 3**.

Actividades principales

(Grabación5.1) Grabación de tareas sobre un actuador lineal (1,5 Puntos)

Material necesario: 1 actuador lineal, 2 sensores de colisión y 1 joystick.

Se requiere tenerlo implementado para el ejercicio optativo 5.7.

Se recomienda tener implementado el ejercicio 4.4 [y tal vez también el 4.6].

Se recomienda ***encarecidamente*** hacer primero el ejercicio de teoría de esta práctica, elegir una librería y ver los posibles problemas.

Implementar un sistema de teleoperación para permitir grabar tareas. Una tecla del PC debe servir para activar y desactivar el modo grabación y la otra para reproducir de forma automática la última tarea grabada. No hace falta hacer una aplicación gráfica, basta con ejecutarlo desde el propio IDE que se utilice. El lenguaje de programación es libre.

El **ejemplo** de comunicación de un Arduino por el puerto USB utilizando Java y con toda la información sobre la configuración de las librerías está **explicado en el PDF de la teoría de prácticas 5.1**. **Se puede realizar en Python**.

Una vez comience el modo grabación, el usuario podrá mover el actuador lineal de forma libre utilizando el joystick. Todas las acciones que va realizando quedan registradas, hasta que se vuelve a pulsar la tecla de activar/detener el modo grabación. Una buena idea puede ser que el actuador comience siempre en la posición 0, pues no disponemos de sensores físicos para saber su posición inicial.

Hay varias formas de hacer la grabación:

- Velocidad normal de funcionamiento.
- Velocidad más lenta: esto ocurre con muchos robots reales, en los que después se puede ajustar la velocidad de reproducción. Así pues, si se graba a una velocidad menor, hay que tener en cuenta cuál es la reducción de velocidad.
- Grabación por pasos (usando las coordenadas) e ir verificando estos en Java/Python. No obstante, hay que tener en cuenta, que, si se hacen pausas, este deberá de hacerlas cuando se reproduzca, pues puede ser que esa pausa sea importante y necesaria en la reproducción ya que puede ser la espera para coger el siguiente objeto o esperar a que algo suceda. Si se graba por pasos, tiene que existir uno que sea la pausa (se puede utilizar un botón).

Consideraciones:

- **En caso de no tener el sistema de coordenadas implementado**, el actuador lineal deberá de comenzar en un lateral para así evitar problemas. Claramente, siempre

deberá comenzar desde el mismo punto. El ejercicio puede ser más sencillo si se tiene el sistema de coordenadas implementado.

- **Los movimientos se almacenarán como variables**: ADELANTE, ATRÁS, en que coordenada se inicia el movimiento y en cual se finaliza. Para las pausas PARADO hace falta saber cuánto tiempo se han mantenido en este estado.
- Se puede establecer un **máximo de instrucciones** que serán grabadas, por ejemplo 1000.
- El actuador solo almacena una tarea: la última que se ha grabado.

Notas:

No se puede tener la consola del IDE (Eclipse, PyCharm, …) y la del IDE de Arduino funcionando a la vez, pues no funcionarán. En ese caso, o no funcionarán u os lanzara «UnkownException» en el Eclipse.

Si el IDE dice que el puerto COM está en uso, cerrad el Eclipse y el IDE de Arduino y volved a abrirlos ya que puede que el Eclipse, o vosotros por código, dejarais el puerto abierto.

Si queréis actualizar el código del Arduino, deberías cerrar el programa en el IDE del lenguaje de programación utilizado en el ordenador (Eclipse, Pycharm, …), pies está conectado al serial y no permitirá subir esa actualización de código al Arduino.

Cuidado con el tipo de parámetro que se envía y con cual se recibe a través del puerto COM. Si se envía un entero (int) y se recibe como si fuera un carácter (char) o una cadena de texto (String), pueden salir caracteres raros. Lo que hay que hacer es enviar un «print» y recibir un «readline» para así enviarlo y recibirlo como «string». También puede haber problemas si se trata de imprimir dos tipos diferentes de datos en el Arduino en una misma línea: String+Int en vez de String+String(Int).

Deberéis utilizar hilos en Java. Esto es debido a que, si no los utilizáis, entonces, cuando se pida algo por consola, el programa se bloqueará esperando esa entrada. Utilizando hilos, el programa podrá hacer varias cosas a la vez.

(Cartesiano5.2) Manipulador cartesiano teleoperado (0,3 Puntos)

Material necesario: 2 actuadores lineales y 1 pinza (montados formarán el cartesiano XY), 4 sensores de colisión y 2 joysticks.

Construir un robot manipulador cartesiano utilizando dos actuadores lineales y una pinza. La explicación y montaje se encuentran en el PDF de teoría de las prácticas 5.1.

Ampliaciones

(Grabación5.3) Grabación de tareas en un manipulador cartesiano (0,75 Puntos)

Material necesario: 2 actuadores lineales y 1 pinza (montados formarán el cartesiano XY), 4 sensores de colisión y 2 joysticks.

Requiere tener implementados los ejercicios obligatorios 5.1 y 5.2

Ampliar la actividad 5.1 para que función con el manipulador cartesiano X, Y.

- La **grabación** de tareas debe registrar el **movimiento de los dos actuadores y de la pinza**.

El robot solo almacena una tarea, la última que se ha grabado.

(Cajas5.4) Apilar cajas con un manipulador cartesiano (1 Punto)

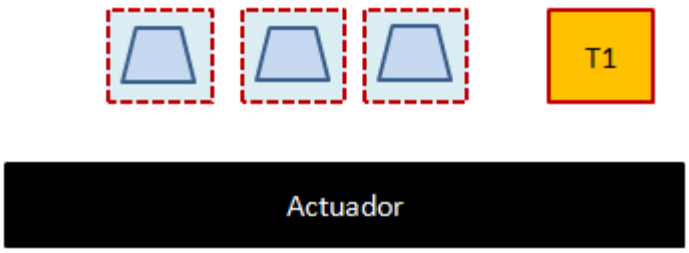
Material necesario: 2 actuadores lineales y 1 pinza (montados formarán el cartesiano XY), 4 sensores de colisión, 2 joysticks, y 3 cubos.

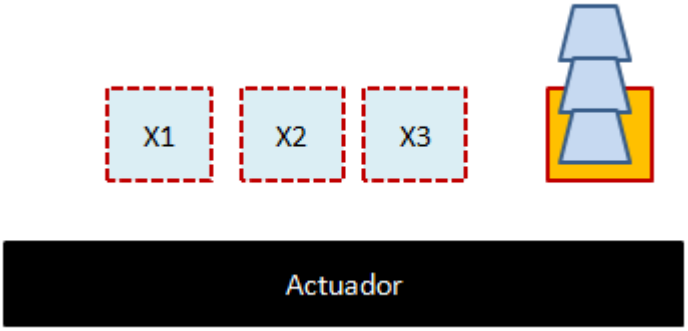
Requiere tener implementados los ejercicios obligatorios y optativos 4.3, 5.1, 5.2 y 5.3 [y tal vez también el 4.6]

Desarrollar un programa que tenga pregrabadas en variables las coordenadas donde se encuentran **tres** cajas y las coordenadas del punto donde debe depositarlas en forma de torre. También, en vez de utilizar coordenadas, se podría hacer midiendo tiempos, siempre que funcionará bien este otro sistema, aunque es más recomendable usando coordenadas (mirar ejercicio 4.5).

El programa debe hacer que el robot manipulador cartesiano recoja cada una de las tres cajas y las vaya apilando en el punto indicado. Debe apilar las cajas unas encima de otras sin que se caigan. En caso de no conseguirlo, subir el ejercicio igualmente, pues algo se puntuará.

Se debe ir calculando internamente la altura de la torre de cajas para que la pinza no tire ninguna de las cajas por error.



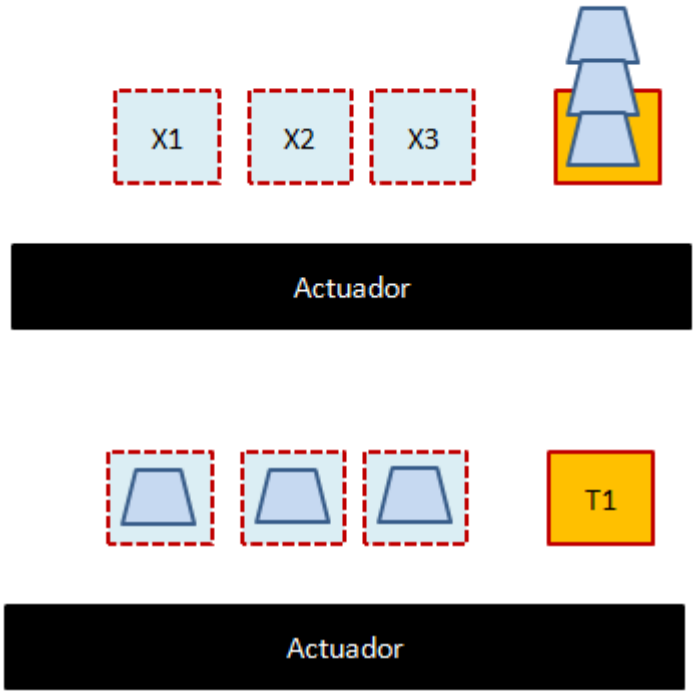


(Cajas5.5) «Desapilar» cajas con manipulador cartesiano (0,5 Puntos)

Material necesario: 2 actuadores lineales y 1 pinza (montados formarán el cartesiano XY), 4 sensores de colisión y 2 joysticks.

Requiere tener implementado el ejercicio opcional 5.4

«Desapilar» las cajas apiladas y volver al estado inicial, es decir, exactamente lo contrario al ejercicio 5.4.



(Instrucciones5.6) Instrucciones de alto nivel (1 Puntos)

Material necesario: 2 actuadores lineales y 1 pinza (montados formarán el cartesiano XY) y 4 sensores de colisión.

Reutilización de parte de los ejercicio obligatorios y optativos 5.1 y 5.3

Crear un intérprete de instrucciones de alto nivel que permita enviar instrucciones al robot manipulador a través del puerto COM desde el PC.

MOVX(X) - se mueve hacia un punto X, manteniendo la misma Y en la que se encontraba.

MOVY(Y) - se mueve hacia un punto, primero avanzando en el eje Y (hasta llegar a la posición) y luego en el eje X.

MOVXY(X, Y) - se mueve hacia un punto, primero avanzando en el eje X (hasta llegar a la posición) y luego en el eje Y.

MOVYX(Y, X) - se mueve hacia un punto, primero avanzando en el eje Y (hasta llegar a la posición) y luego en el eje X.

OP - abre la pinza. Los grados aplicados al motor para abrir la pinza son conocidos por el robot.

CP - cierra la pinza, los grados aplicados al motor para abrir la pinza son conocidos por el motor. No tiene que cerrarse completamente, pero si lo necesario para coger un cubo.

WAIT(milisegundos) - realiza una pausa del número de milisegundos indicado.

Los comandos se deben poder introducir por el puerto COM, separados por «;»

Ejemplo de conjunto de instrucciones enviadas al robot:

OP;MOVXY(10,20);MOVYX(10,0);CP;WAIT(1000);MOVXY(24,24);

(Modificar5.7) Modificar grabación (1 Puntos)

Material necesario: 1 actuador lineal, 2 sensores de colisión y 1 joystick.

Requiere tener implementado el ejercicio obligatorio 5.1

Usando como base el ejercicio 5.1, modificarlo para que admita la modificación de las instrucciones desde Java antes de que estas se envíen.

Las instrucciones tendrán que estar guardadas en memoria, después de la grabación se mostrarán al usuario, y este podrá modificarlas:

- Borrar una instrucción.
- Añadir una nueva instrucción.
- Repetir alguna de las existentes.
- Mover orden de las tareas.
- Cambiar tiempo de las tareas y de espera.