PRÁCTICA DOBOT

TEMA: Programación online con robot DOBOT Magician

1. Objetivos

- Identificar los componentes del brazo robótico
- Programar un brazo robótico para tareas repetitivas
- Programar tarea de pick and place

2. Marco teórico

Dobot Magician.- Espacio de trabajo

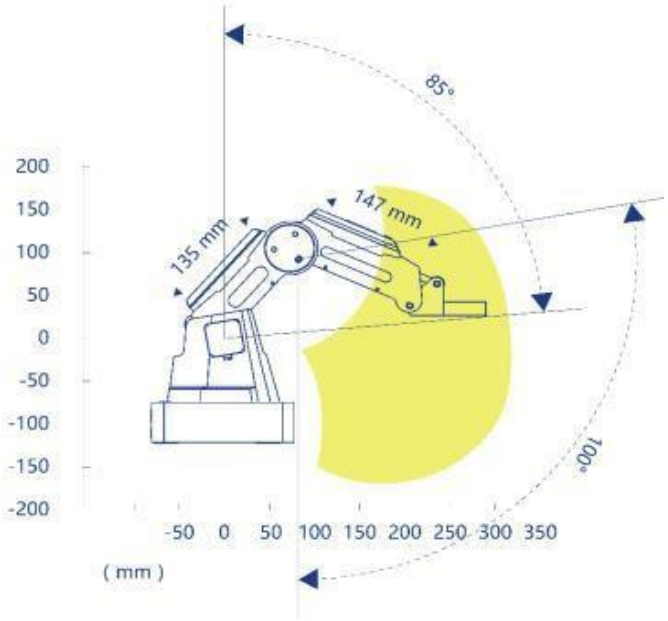


Figure 3.2 Workspace of Dobot Magician (1)

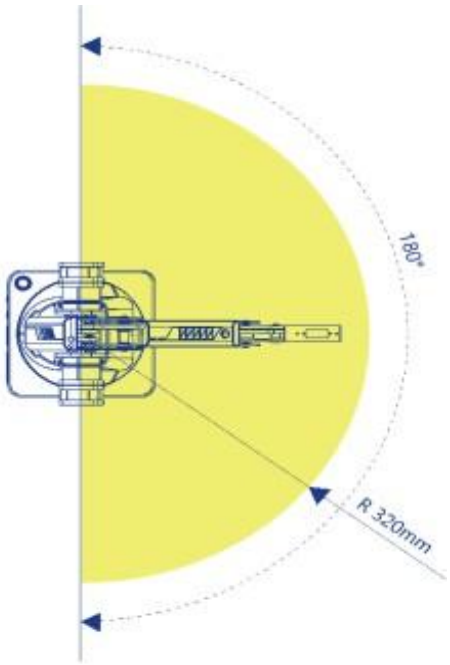


Figure 3.3 Workspace of Dobot Magician (2)

Sistemas de coordenadas

Tiene dos sistemas de coordenadas, coordenada Joint y coordenada cartesiana.

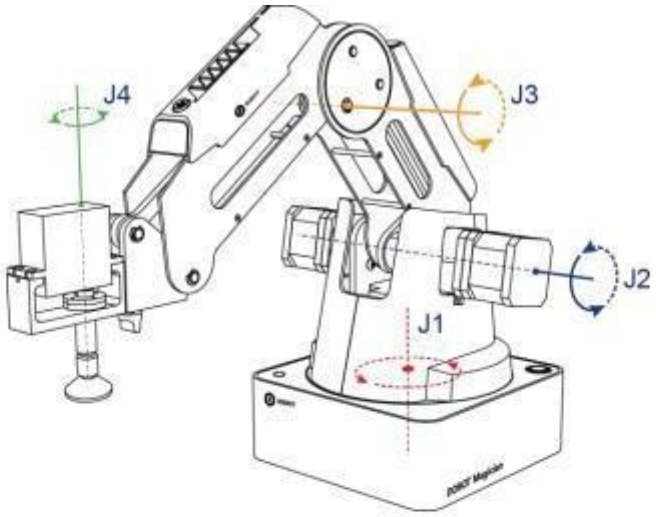


Figure 3.4 Joint coordinate system

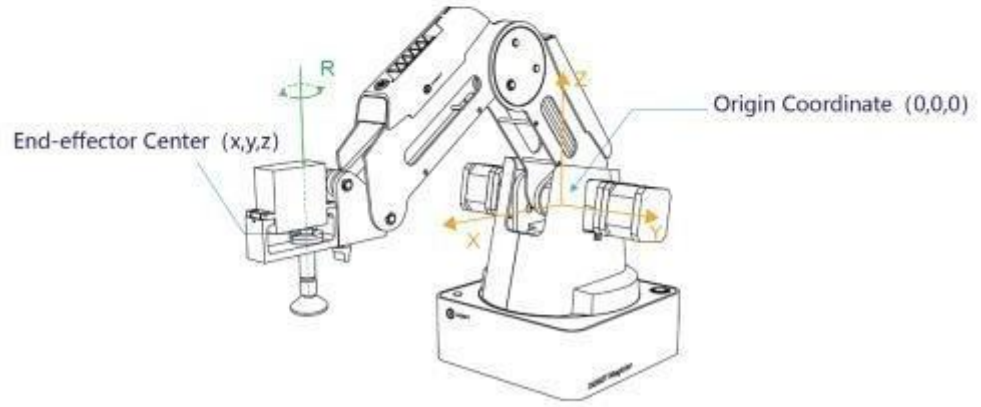


Figure 3.5 Cartesian coordinate system

- Si no se encuentra instalado ninguna herramienta, el robot cuenta con 3 ejes (J1, J2 y J3) y la dirección positiva para el movimiento es en sentido contrario de las manecillas del reloj.
- Si se encuentra instalado alguna herramienta el robot cuenta con 4 ejes (J1, J2 J3 y J4) y la dirección positiva para el movimiento es en sentido contrario de las manecillas del reloj.
- En el **sistema cartesiano**, sus coordenadas están determinadas por la base.
- El eje X se encuentra saliendo de la base, el eje Y hacia la derecha, y siguiendo la regla de la mano derecha el eje Z se encuentra hacia arriba.
- El eje R es la actitud del servo centro con respecto al origen del brazo robótico, de los cuales la dirección positiva es en sentido antihorario.

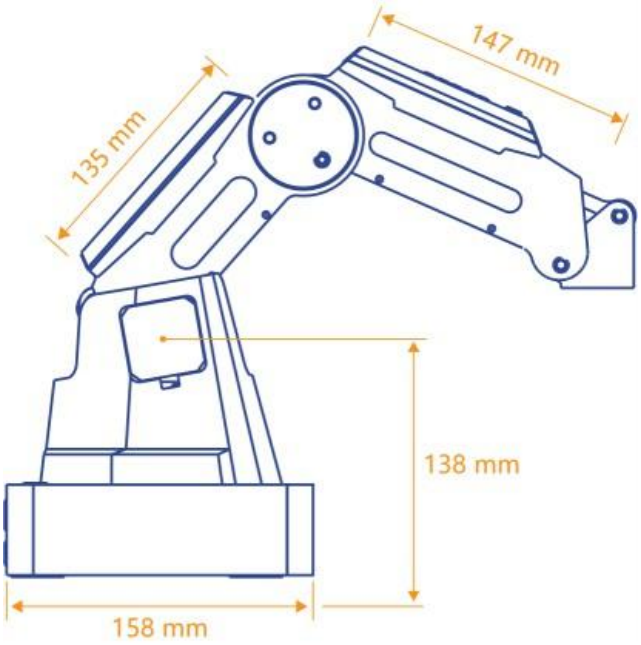


Figure 3.9 Size of Dobot Magician

Name	Dobot Magician	
Maximum payload	500g	
Maximum reach	320mm	
Motion range	Base	- 90 °~+90 °
	Rear Arm	0 °~85 °
	Forearm	- 10 °~+90 °
	End-effector rotation	- 90 °~+90 °

Maximum speed (with 250g payload)	Rotational speed of Rear arm, Forearm and base	320 °/s
	Rotational speed of servo	480 °/s
Repeated positioning accuracy	0.2mm	
Power supply	100V-240V AC, 50/60Hz	
Power in	12V/7A DC	
Communication	USB, WIFI, Bluetooth	
I/O	20 extensible I/O interfaces	
Software	DobotStudio	
Working temperature	-10 ℃~60 ℃	

Puertos E/S

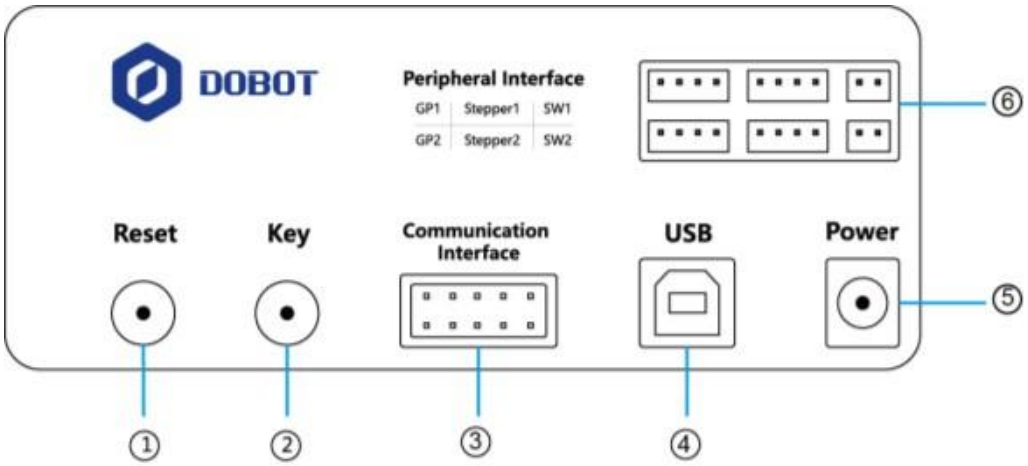
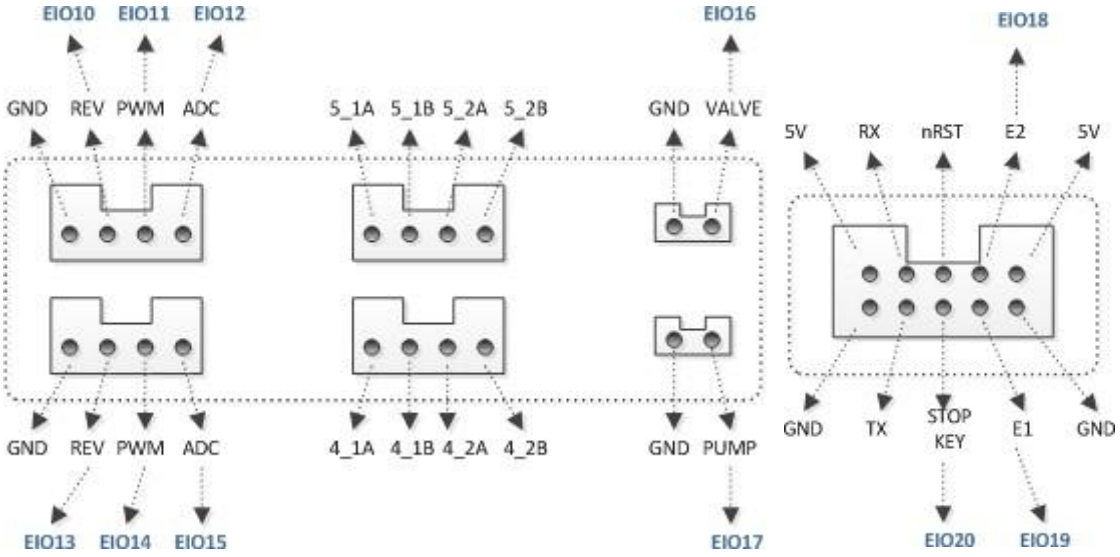
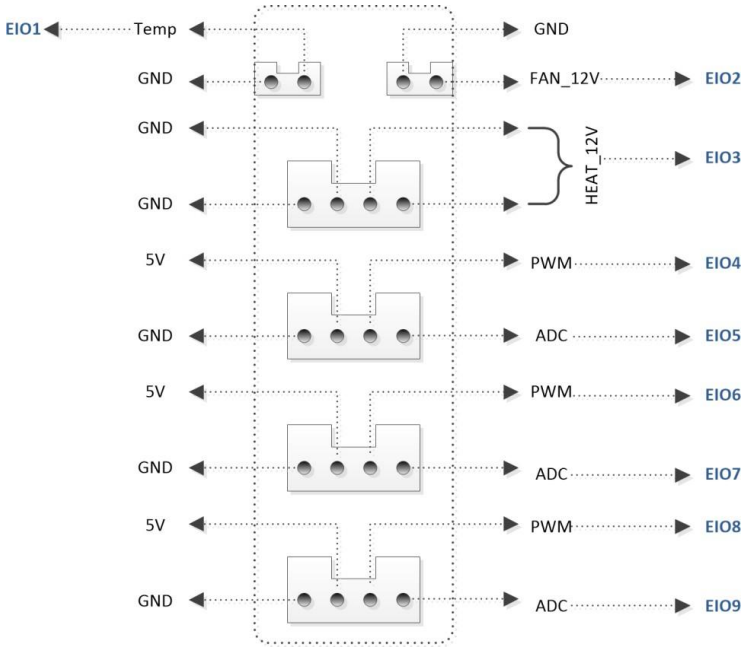


Figure 1.1 Interfaces in the base





3. Descripción

En esta práctica se tendrá como objetivo principal conocer comandos básicos para poder manipular el robot mediante programación Python, sin hacer uso de su software.

Se realizarán tareas de movimiento en los ejes cordenados x,y,z, para realizar una tarea de pick and place sencilla con el uso de la ventosa o de la garra.

4. Procedimiento

4.1. Programación usando Python.

a) Descargar la carpeta comprimida subida al Aula Virtual.

b) Descomprimir la carpeta y guardarla en un lugar fácil de recordar.

c) Abrir Visual Studio Code

d) Seleccionamos Open Folder y abrimos la carpeta descomprimida.

e) Creamos un nuevo archivo dando click en  dentro de la carpeta “dobots” y le colocamos un nombre seguido de .**ipynb** para crear segmentos de códigos de programación. Por ejemplo: *Practica.ipynb*

f) Con la opción +code agregamos un nuevo segmento de código y pegamos el siguiente código para conectarnos con el robot.

```
import threading
import DobotDllType as dType
#se crean definiciones para determinar errores en la comunicacion
CON_STR = {
    dType.DobotConnect.DobotConnect_NoError:  "DobotConnect_NoError",
    dType.DobotConnect.DobotConnect_NotFound: "DobotConnect_NotFound",
    dType.DobotConnect.DobotConnect_Occupied: "DobotConnect_Occupied"
    }

#Load Dll and get the CDLL object
api = dType.load()
#Connect Dobot
state = dType.ConnectDobot(api, "", 115200)
print("datos de retorno de conexion ",[state])
print("Id Robot ",[state[1]])
print("Connect status:",CON_STR[state[0]])
```

Para esta práctica usaremos la librería DobotdllType para manipular el robot.
El archivo *DobotDllType.py* contiene todas las funciones que podemos utilizar. Para ello, debemos llamarlas en el script que creamos configurando sus respectivos parámetros.

El siguiente código nos permite configurar los parámetros de inicio.

```
# Establecer posición HOME (de inicio)
dType.SetHOMEParams(api, x, y, z, r) #Coordenadas (X, Y, Z, R)
dType.SetHOMECmd(api, temp=0, isQueued=1)
dType.SetQueuedCmdStartExec(api)
print("Robot configurado y en posición inicial.")
```

Los parámetros de SetHOMEParams son coordenadas cartesianas x, y, z, r donde r es el ángulo de giro del elemento terminal.

Movimiento Lineal en Coordenadas Cartesianas

```
SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, x, y, z, r)
```

Donde x, y, z se encuentran en milímetros y r en grados

Movimiento Angular en Coordenadas Articulares

```
SetPTPCmd(api, dType.PTPMode.PTPMOVJANGLEMode, J1,J2,J3,J4)
```

Donde J1, J2, J3 y J4 se encuentran en grados.

Elementos Terminales

```
Gripper: SetEndEffectorGripperEx()
Ventosa: SetEndEffectorSuctionCupEx()
```

En el archivo Ejemplo.ipynb se encuentran algunos ejemplos de cómo utilizar las funciones de la librería para realizar los movimientos del robot, configurar sus elementos terminales, activar o desactivar la banda transportadora y sensores.

4.2. Actividad

Realizar una tarea de *Pick and Place* con una caja de color utilizando los comandos de la información revisada en este documento y en los documentos adjuntos.

Extra: Modificar el código para que la tarea de Pick and Place se inicie únicamente después de que un sensor de presencia detecte el objeto.