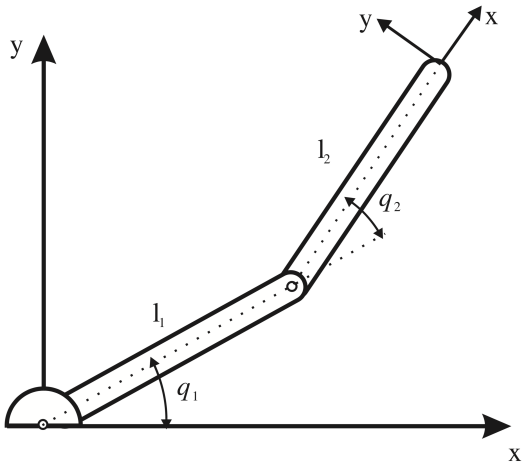
PRÁCTICA 2:
Análisis cinemático de un manipulador RR

En esta práctica vamos a analizar un robot RR (con 2 articulaciones de rotación) como el siguiente:



Estudiaremos por separado el problema cinemático directo y el problema cinemático inverso, haciendo uso, en ambos casos, de varias funciones escritas en Python.

1. Problema cinemático directo

En primer lugar vamos a trazar la trayectoria que realiza el extremo del manipulador dado un valor de sus variables de articulación. Esto implica resolver el problema cinemático directo.

Ejercicio 1

Implemente una función que resuelva el problema cinemático directo para un manipulador RR, esto es, que calcule las coordenadas de la posición del extremo del robot dadas las variables de articulación. Use el siguiente esquema:

```
def pcd(q1, q2, l1, l2):
  <cálculo de x e y>
  return (x, y)
```

donde:

- `q1` es el ángulo de giro de la primera articulación (en radianes), q_1.
- `q2` es el ángulo de giro de la segunda articulación (en radianes), q_2.
- `l1` es la longitud del primer eslabón, l_1.
- `l2` es la longitud del segundo eslabón, l_2.
- `x` es la coordenada x del extremo del robot.
- `y` es la coordenada y del extremo del robot.

Ejercicio 2

Implemente una nueva función que, haciendo uso de la función `pcd`, dibuje la trayectoria seguida por el extremo del robot dadas unas variables de articulación concretas. La función debe tener la siguiente sintaxis:

```
def dibujar_trayectoria_pcd(q1s, q2s, l1, l2):
  <código de la función>
```

donde:

- `q1s` es un vector con los ángulos de giro (en radianes) que debe tomar la primera articulación a lo largo de la trayectoria.
- `q2s` es un vector con los ángulos de giro (en radianes) que debe tomar la segunda articulación a lo largo de la trayectoria.
- `l1` es la longitud del primer eslabón, l_1.
- `l2` es la longitud del segundo eslabón, l_2.

Para evitar que la trayectoria se vea distorsionada, centre el origen de coordenadas en la figura estableciendo una longitud máxima para cada eje de $\pm(l_1 + l_2 + 1)$.

Ejercicio 3

Al analizar la trayectoria, resulta de ayuda visualizar los eslabones del robot. Para conseguirlo, implemente una tercera función que los dibuje junto a la trayectoria. Puede usar el siguiente código:

```
def dibujar_robot(q1, q2, l1, l2):
  x0, y0 = 0, 0;                    # Pos. de la articulación 1
  x1, y1 = pcd(q1, 0, l1, 0);       # Pos. de la articulación 2
  x2, y2 = pcd(q1, q2, l1, l2);     # Pos. del extremo del robot

  x = [x0,x1,x2]; y = [y0,y1,y2];   # Coordenadas de la trayec.
  plt.plot(x, y, 'k');              # Traza la trayectoria
  plt.plot(x0, y0, 'k.');           # Dibuja la articulación 1
  plt.plot(x1, y1, 'k.');           # Dibuja la articulación 2
```

Esta función debe ser llamada al final de la función `dibujar_trayectoria_pcd`, introduciendo en `q1` y `q2` los últimos ángulos en `q1s` y `q2s` respectivamente.

Ejercicio 4

Compruebe el correcto funcionamiento de las funciones creadas en los ejercicios anteriores trazando una trayectoria con 100 puntos en los que q_1 vale siempre $\pi/4$ y q_2 toma valores entre 0 y $\pi/2$. Suponga que $l_1 = l_2 = 2$. Represente el resultado. Debe ser similar al mostrado en la figura 1. Adicionalmente pruebe con otros valores para q_1 y q_2 que generen trayectorias distintas.

La función `dibujar_trayectoria_pcd` resulta útil para analizar cuál ha sido la trayectoria del extremo del manipulador, pero, al generar únicamente una figura estática, no permite analizar el movimiento en sí. Para esto necesitaríamos ver el movimiento "paso a paso" por la trayectoria.

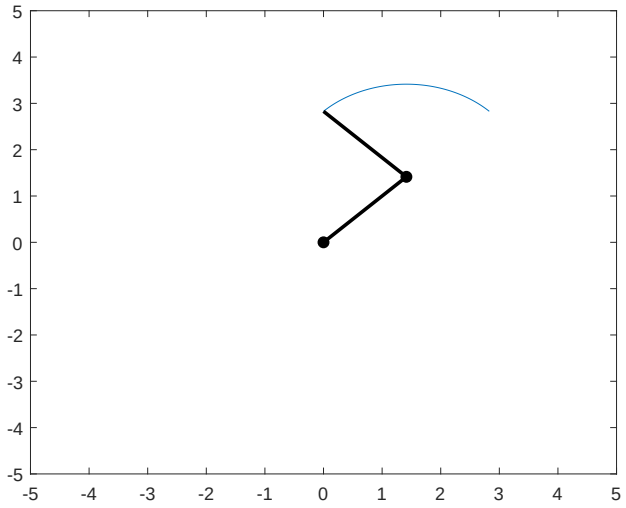


Figura 1: *En color azul:* Trayectoria seguida por el extremo de un robot RR cuando $q_1 = \pi/4$ y q_2 toma valores entre 0 y $\pi/2$. *En color negro:* Eslabones de un robot RR cuando $q_1 = \pi/4$ y $q_2 = \pi/2$.

Ejercicio 5

Implemente una función que permita analizar el movimiento "paso a paso". Puede usar el siguiente código:

```
def animacion_trayectoria_pcd(q1s, q2s, l1, l2):
  n = min(len(q1s), len(q2s))
  for i in range(1,n):
    plt.clf()
    dibujar_trayectoria_pcd(q1s[0:i], q2s[0:i], l1, l2)
    plt.pause(0.001)
```

Ejercicio 6

Utilice la función `animacion_trayectoria_pcd` para analizar la trayectoria realizada por el extremo del robot cuando q_1 y q_2 toman los siguientes valores:
$q_1 \rightarrow 0, \pi/100, 2\pi/100, ..., \pi$
$q_2 \rightarrow 0, \pi/100, 2\pi/100, ..., 50\pi/100, 49\pi/100, 48\pi/100, ..., 0$

2. Problema cinemático inverso

En este apartado vamos a analizar el cambio en las variables de articulación cuando el extremo del robot realiza una trayectoria definida por unas coordenadas concretas. Esto implica resolver el problema cinemático inverso.

Ejercicio 7

Implemente una función que resuelva el problema cinemático inverso para un manipulador RR, esto es, que calcule las variables de articulación dadas las coordenadas de la posición del extremo del robot. Use el siguiente esquema:

```
def pci(x, y, l1, l2):
  <cálculo de q1 y q2>
  return (q1, q2)
```

donde:

- `x` es la coordenada x del extremo del robot.
- `y` es la coordenada y del extremo del robot.
- `l1` es la longitud del primer eslabón, l_1.
- `l2` es la longitud del segundo eslabón, l_2.
- `q1` es el ángulo de giro de la primera articulación (en radianes), q_1.
- `q2` es el ángulo de giro de la segunda articulación (en radianes), q_2.

Observe que, al contrario de lo que ocurre en la función que resuelve el problema cinemático directo, en esta ocasión los argumentos de entrada son las coordenadas del extremo del robot y las longitudes de los eslabones, mientras que los argumentos de salida son las variables de articulación.

Ejercicio 8

Implemente las funciones equivalentes a `dibujar_trayectoria_pcd` y `animacion_trayectoria_pcd` para el problema cinemático inverso. La sintaxis de las mismas se muestra a continuación:

```
def dibujar_trayectoria_pci(xs, ys, l1, l2):
  <código de la función>

def animacion_trayectoria_pci(xs, ys, l1, l2):
  <código de la función>
```

Ejercicio 9

Compruebe que `dibujar_trayectoria_pci` y `animacion_trayectoria_pci` funcionan correctamente analizando el valor de las variables articulares cuando el extremo del robot se mueve:

- Desde el punto $(2, 0)$ hasta el punto $(0, 2)$ en línea recta.
- Desde el punto $(1, 1)$ hasta el punto $(-2, 1)$ en línea recta.
- Desde el punto $(1, 0)$ hasta el punto $(-1, 0)$ en línea recta.

Si al resolver el problema cinemático inverso ha calculado el arcotangente usando la función `arctan` de `numpy`, el robot se comportará de modo extraño al trazar las trayectorias anteriores. Para evitarlo use la función `arctan2` y compare su funcionamiento con el de la función `arctan`. ¿A qué se debe la diferencia?

Tanto usando `arctan2` como `arctan` es posible que el robot se comporte de forma extraña al representar la tercera trayectoria del ejercicio anterior. Esto se debe a

que el el problema cinemático inverso del robot RR tiene más de una solución y no siempre la primera que obtenemos es la más adecuada. Si el problema cinemático inverso tiene más de una solución, debemos elegir la "mejor" solución posible y para ello debemos decidir previamente qué entendemos por una "mejor" solución.

Si analizamos la trayectoria del extremo del robot al ir del punto $(1, 0)$ al punto $(-1, 0)$ en línea recta, podemos ver que el movimiento brusco del brazo robótico se debe a que el ángulo del primer eslabón (q_1) va haciéndose cada vez más negativo hasta llegar a $-\pi/2$ y en ese momento pasa a tener un valor $\pi/2$ y comienza a crecer. Para evitar movimientos bruscos del robot, siempre que tengamos 2 o más soluciones posibles debemos elegir aquélla que suponga un menor cambio respecto al valor anterior de las variables de articulación. Es decir, si en el instante de tiempo t, q_i vale $-\pi/2$ y para el instante de tiempo $t+1$ tenemos 2 posibles valores para q_i debemos elegir aquél que sea más próximo a $-\pi/2$.

Ejercicio 10

Modifique la función `pci` para que devuelva todas las soluciones posibles y a continuación adapte la función `dibuajar_trayectoria_pci` para que escoja la "mejor" solución de todas las que devuelva `pci`.