Sheet 3 Topic: Locomotion, Differential Drive Kinematics

Name: Berkay

Surname: Gülen

No: 20170613017

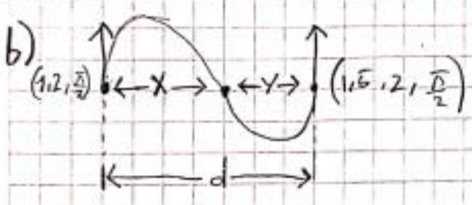
Exercise 1: Locomotion

Berkay Gülen

Exercise 1

a) we need $\boxed{2}$ commands to arrive desired target location

b)



(S_1) First circle arc lenght: $\frac{\pi \cdot x}{2}$

(S_2) Second circle arc lenght: $\frac{\pi \cdot y}{2}$

$x+y=d$ / $d=0.5$

The lenght of the trajectory $\Rightarrow S_1+S_2 = \frac{\pi \cdot x}{2} + \frac{\pi \cdot y}{2} = \frac{\pi(x+y)}{2} = \frac{d\pi}{2} = \boxed{\frac{\pi}{4}}$

c) The easiest way is first rotation in place (RIP) then go straight line until arrived the goal point. then another RIP. In summary, we need 3 commands.

(I) First command

- we need to rotate in place.
- we need to go right so left wheel: v, right wheel: −v

$\boxed{\theta' = \theta + \frac{V_L - V_R}{\ell} \cdot \Delta t}$ $\quad \Delta\theta = \frac{\pi}{2}$

$\frac{(\theta' - \theta) \cdot \ell}{V_L - V_r} = \Delta t \Rightarrow \boxed{\Delta t = \frac{\pi \cdot \ell}{4V}}$

$\boxed{\text{First command: } \left(V, -V, \frac{\pi \cdot \ell}{4V}\right)}$

(II) Second command

- we need to go straight
- left wheel: V, right wheel: V

$\Delta\theta = 0$ $\quad \boxed{x' - x = d}$

$x' = x + V \cdot \cos(\theta) \cdot \Delta t$

$y' = y + V \cdot \sin(\theta) \cdot \Delta t$

$\Rightarrow \Delta t = \frac{x' - x}{V \cdot \cos(0)} = \boxed{\frac{d}{V}}$

$\boxed{\text{Second command: } \left(V, V, \frac{d}{V}\right)}$

(III) Third command

- it is like first command. However, wheel speed are inverse.

$\boxed{\text{third command: } \left(-V, V, \frac{\pi \cdot \ell}{4V}\right)}$

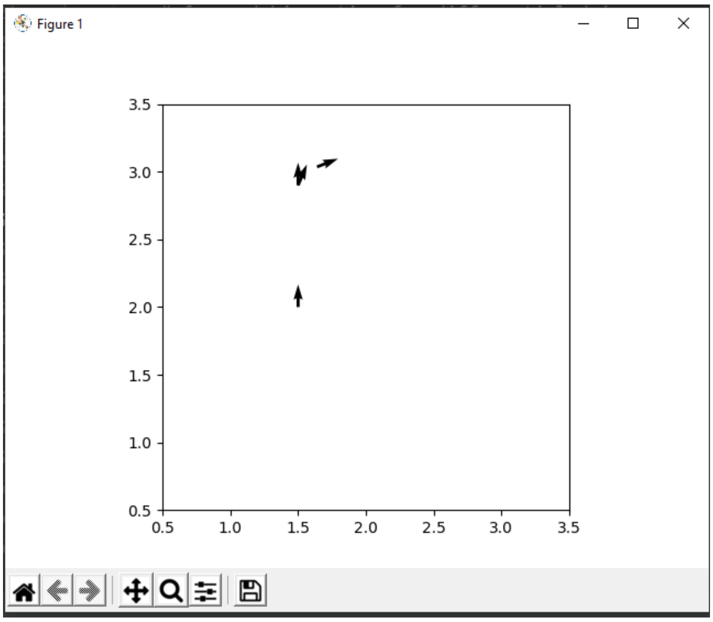
D) The lenght of the trajectory is distance between two poses. Which is $\boxed{0.5\text{ m}}$

Exercise 2: Differential Drive Implementation

a)

```python
def diffdrive(x, y, theta, v_l, v_r, t, l):
    # straight line when left wheel speed = right wheel speed
    if (v_l == v_r):
        final_theta = theta
        final_x = x + v_l * t * np.cos(theta)
        final_y = y + v_l * t * np.sin(theta)

    # circular motion left wheel speed != right wheel speed
    else:
        # Calculate the radius
        R = l / 2.0 * ((v_l + v_r) / (v_r - v_l))

        # center of curvatures
        ICC_x = x - R * np.sin(theta)
        ICC_y = y + R * np.cos(theta)

        # computing theta prime
        theta_prime = ((v_r - v_l) * t) / l

        # forward kinematics for differential drive
        final_x = np.cos(theta_prime) * (x - ICC_x) - np.sin(theta_prime) * (y - ICC_y) + ICC_x
        final_y = np.sin(theta_prime) * (x - ICC_x) + np.cos(theta_prime) * (y - ICC_y) + ICC_y
        final_theta = theta + theta_prime

    return final_x, final_y, final_theta
```

Final Graph:



b)

```python
import numpy as np
import matplotlib.pyplot as plt


def diffdrive(x, y, theta, v_l, v_r, t, l):
    # straight line when left wheel speed = right wheel speed
    if (v_l == v_r):
        final_theta = theta
        final_x = x + v_l * t * np.cos(theta)
        final_y = y + v_l * t * np.sin(theta)

    # circular motion left wheel speed != right wheel speed
    else:
        # Calculate the radius
        R = l / 2.0 * ((v_l + v_r) / (v_r - v_l))

        # center of curvatures
        ICC_x = x - R * np.sin(theta)
        ICC_y = y + R * np.cos(theta)

        # computing theta prime
        theta_prime = ((v_r - v_l) * t) / l

        # forward kinematics for differential drive
        final_x = np.cos(theta_prime) * (x - ICC_x) - np.sin(theta_prime) * (y - ICC_y) + ICC_x
        final_y = np.sin(theta_prime) * (x - ICC_x) + np.cos(theta_prime) * (y - ICC_y) + ICC_y
        final_theta = theta + theta_prime

    return final_x, final_y, final_theta


plt.gca().set_aspect("equal")

# distance between the wheels and the initial robot position
x = 1.5
y = 2.0
l = 0.5
theta = (np.pi) / 2.0

# starting position
plt.quiver(x, y, np.cos(theta), np.sin(theta))
print(f"starting pose: x: {x}, y: {y}, theta:{theta}")

# first motion
v_l = 0.3
v_r = 0.3
t = 3
x, y, theta = diffdrive(x, y, theta, v_l, v_r, t, l)
plt.quiver(x, y, np.cos(theta), np.sin(theta))
print(f"after motion 1: x: {x}, y: {y}, theta:{theta}")

# second motion
v_l = 0.1
v_r = -0.1
t = 1
x, y, theta = diffdrive(x, y, theta, v_l, v_r, t, l)
plt.quiver(x, y, np.cos(theta), np.sin(theta))
print(f"after motion 2: x: {x}, y: {y}, theta:{theta}")

# third motion
v_l = 0.2
v_r = 0.0
t = 2
x, y, theta = diffdrive(x, y, theta, v_l, v_r, t, l)
plt.quiver(x, y, np.cos(theta), np.sin(theta))
print(f"after motion 3: x: {x}, y: {y}, theta:{theta}")

# plot the poses
plt.xlim([0.5, 3.5])
plt.ylim([0.5, 3.5])
plt.savefig("poses.png")
plt.show()
```