به نام خدا

نام: امیرمسعود

نام خانوادگی: شاکر

شماره دانشجویی: 97243081

تمرین پنجم

.1

a) در این بخش سرعت چرخ چپ برابر 1- و سرعت چرخ راست برابر 1 است.

با توجه به اینکه سرعت ها قرینه هستند، ربات به دور خودش میچرخد.

توضیح بخش های تکراری در کنترلرها:

کتابخانه های لازم را import میکنیم. در این تمرین از کتابخانه های Robot, Motor, math, numpy, matplotlib استفاده شده است.

مقادیر TIME_STEP و MAX_SPEED را مشخص میکنیم.

یک شی جدید از ربات میسازیم.

چرخ های چپ و راست را ساخته و موقعیت و سرعت اولیه آنها را تعیین میکنیم.

ابتدا GPS, Compass را به children ربات اضافه میکنیم.

سپس از آنها شی ساخته و آنها را فعال میکنیم.

متغیر های c, t برای مدت زمان اجرای شبیه سازی تا رسم نمودار و گام های زمانی طی شده توسط ربات هستند.

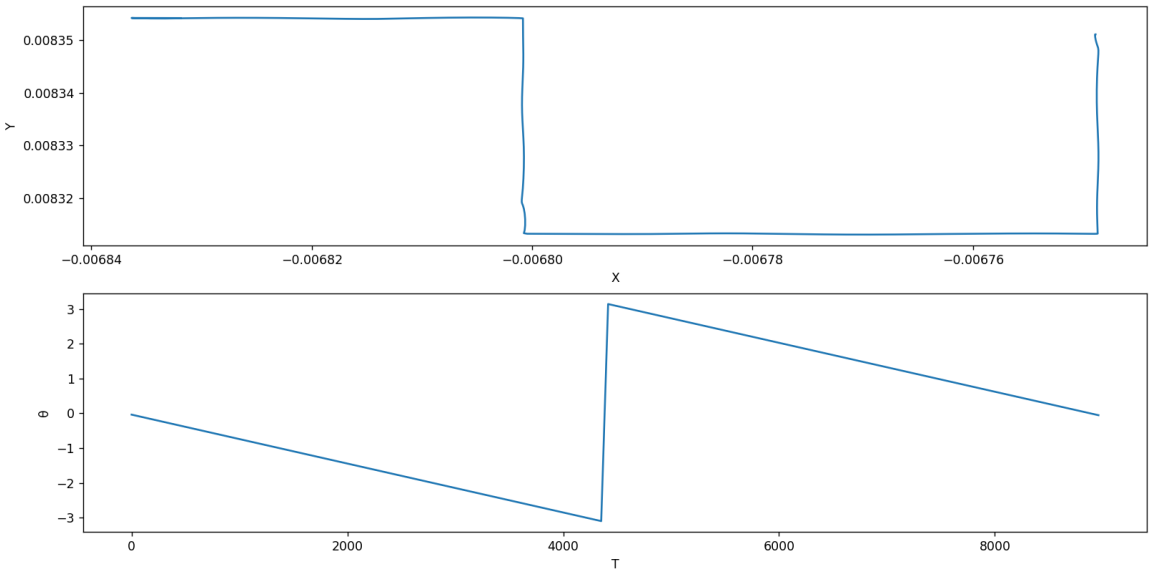
چهار لیست برای ذخیره کردن مختصات x, y, theta, time میسازیم.

در حلقه while مقادیر اندازه گیری شده gps, compass (که به صورت بردار هستند) را گرفته و مولفه های آنها را به لیست های مربوطه برای ذخیره سازی مختصات اضافه میکنیم.

یک شرط برای مقدار c داریم که مشخص کننده مدت زمان اجرای شبیه سازی تا رسم نمودار است.

مقادیر c, t را در هر اجرای حلقه آپدیت میکنیم.

در انتها با استفاده از کتابخانه matplotlib، نمودار های x-y و theta-time را رسم میکنیم.




```python
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

# set up the motor speeds
leftMotor.setVelocity(-1)
rightMotor.setVelocity(1)
```

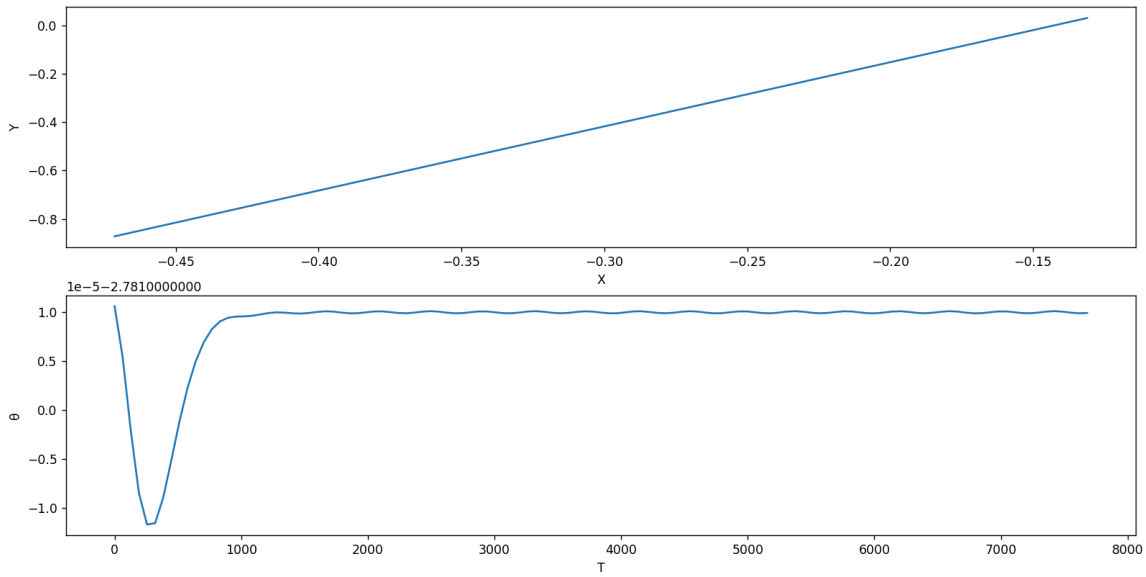
```
# GPS
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 140):
        break
    c += 1
    t += TIME_STEP

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
```

```
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

b) در این بخش دو مقدار circle radius = 80cm و chassis length = 52mm = 5.2cm را داریم.

با استفاده از رابطه داده شده در صورت سوال، سرعت چرخ چپ برابر 82.6 و سرعت چرخ راست برابر 77.4 به دست می آید.

ربات یک مسیر تقریبا مستقیم به سمت جنوب را طی میکند.




```
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
```

```
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

circle_radius = 80
axle_length = 5.2

# set up the motor speeds
leftMotor.setVelocity(circle_radius + (axle_length/2))
rightMotor.setVelocity(circle_radius - (axle_length/2))


# GPS
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 120):
```

```
        break
    c += 1
    t += TIME_STEP

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

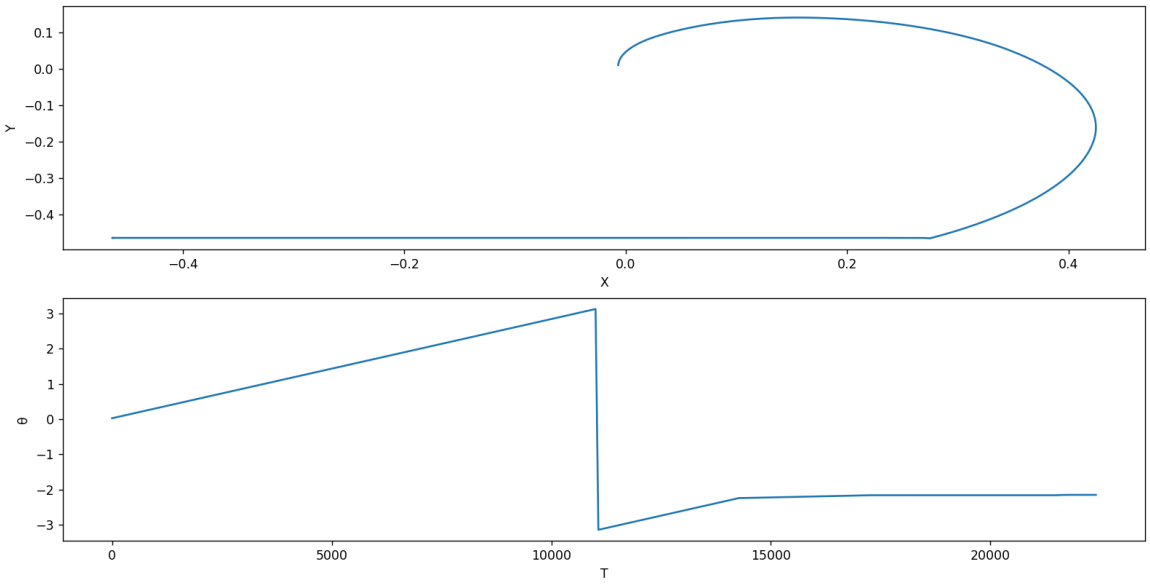
c) در این بخش ابتدا سرعت های اولیه چرخ ها را تعیین میکنیم.

سپس در حلقه while، با توجه به فرمول صورت سوال، شرط میگذاریم که اگر t % 180 برابر صفر باشد، سرعت هر دو چرخ برابر سرعت قبلی به اضافه یک میشود.

در غیر این صورت نیز سرعت های چرخ ها تغییر نمیکنند.

سپس چک میکنیم که اگر سرعت هر کدام از چرخ ها از MAX_SPEED بیشتر شد، سرعت هر چرخ برابر MAX_SPEED شود.

در شبیه سازی مشاهده میکنیم که سرعت ربات هر چند وقت یک بار زیاد میشود و مجددا ثابت میشود.



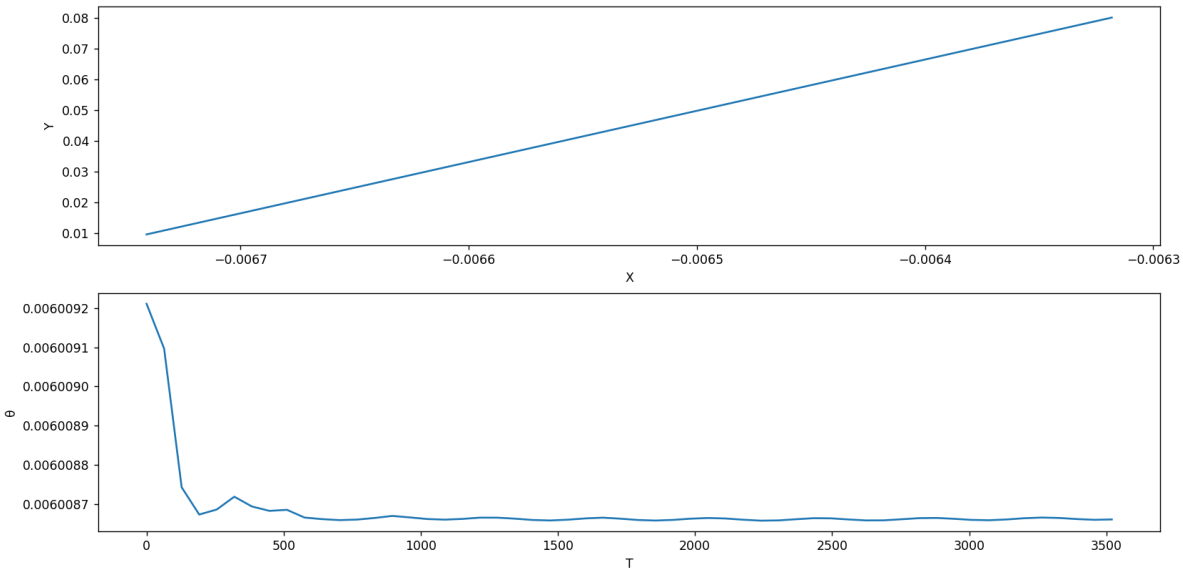

```python
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

# set up the motor initial speeds

left_motor_velocity = 2
right_motor_velocity = 1.2
```

```
leftMotor.setVelocity(left_motor_velocity)
rightMotor.setVelocity(right_motor_velocity)

# GPS
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 350):
        break
    c += 1
    t += TIME_STEP

    if t % 180 == 0:
        left_motor_velocity += 1
        right_motor_velocity += 1

        if left_motor_velocity > MAX_SPEED:
```

```
            left_motor_velocity = MAX_SPEED

        if right_motor_velocity > MAX_SPEED:
            right_motor_velocity = MAX_SPEED

    leftMotor.setVelocity(left_motor_velocity)
    rightMotor.setVelocity(right_motor_velocity)

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

d) در این بخش سرعت هر دو چرخ را برابر 1 قرار میدهیم.

ربات یک مسیر مستقیم رو به جلو را طی میکند.




```python
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

# set up the motor speeds
leftMotor.setVelocity(1)
rightMotor.setVelocity(1)

# GPS
```

```
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 55):
        break
    c += 1
    t += TIME_STEP

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

.2

a) <u>توضیح تابع سینماتیک معکوس:</u>

تابع ما در ورودی سرعت خطی، سرعت زاویه ای، جهت سر ربات، طول محور و شعاع چرخ را میگیرد.

سپس وکتور intertial frame را میسازیم که شامل مولفه های x_dot, y_dot, theta_dot است.

ماتریس R(theta) را میسازیم.

حاصل ضرب ماتریس R(theta) و inertial frame را برابر robot_frame قرار میدهیم.

با استفاده از robot frame محاسبه شده، مقادیر سرعت خطی و زاویه ای ربات را مقداردهی میکنیم.

$$\begin{bmatrix} \dot{x}_R \\ 0 \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$$

local reference frame — rotation matrix — global reference frame

سرعت خطی و زاویه‌ای ربات در مرجع خودش

$$\dot{x}_R = \frac{r\dot{\phi}_1}{2} + \frac{r\dot{\phi}_2}{2}$$

$$\dot{\theta} = \frac{r\dot{\phi}_1}{2l} - \frac{r\dot{\phi}_2}{2l}$$

حل دستگاه و یافتن سرعت زاویه‌ای چرخ‌ها

برای یافتن خروجی های تابع که همان سرعت چرخ ها هستند، باید معادله بالا را حل کنیم تا به روابط نوشته شده در قسمت آخر تابع برسیم.

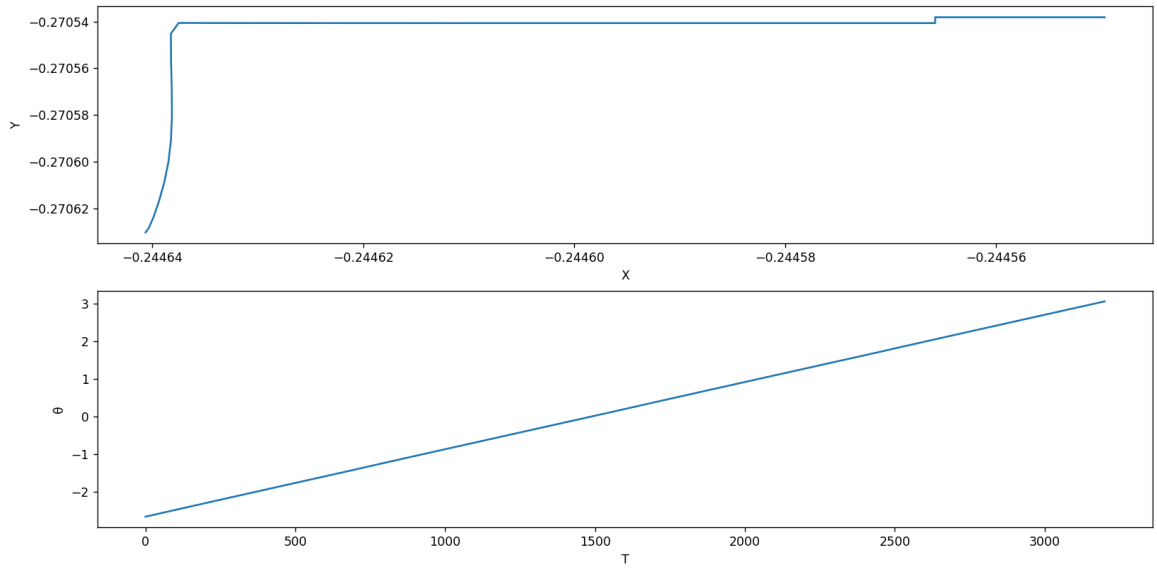
حل دستگاه:

$$\dot{\theta} = \frac{r\dot{\phi}_1}{2L} - \frac{r\dot{\phi}_2}{2L} \xrightarrow{\times 2L} 2L\dot{\theta} = r(\dot{\phi}_1 - \dot{\phi}_2) \Rightarrow (\dot{\phi}_1 - \dot{\phi}_2) = \frac{2L\dot{\theta}}{r} \quad (1)$$

$$\dot{X}_R = \frac{r\dot{\phi}_1}{2} + \frac{r\dot{\phi}_2}{2} \xrightarrow{\times 2} 2\dot{X}_R = r(\dot{\phi}_1 + \dot{\phi}_2) \xrightarrow{\div r} (\dot{\phi}_1 + \dot{\phi}_2) = \frac{2\dot{X}_R}{r} \quad (2)$$

$$(1)+(2) \Rightarrow 2\dot{\phi}_1 = \frac{2(L\dot{\theta} + \dot{X}_R)}{r} \Rightarrow \boxed{\dot{\phi}_1 = \frac{L\dot{\theta} + \dot{X}_R}{r}}$$

$$(2)-(1) \Rightarrow 2\dot{\phi}_2 = \frac{2(\dot{X}_R - L\dot{\theta})}{r} \Rightarrow \boxed{\dot{\phi}_2 = \frac{\dot{X}_R - L\dot{\theta}}{r}}$$

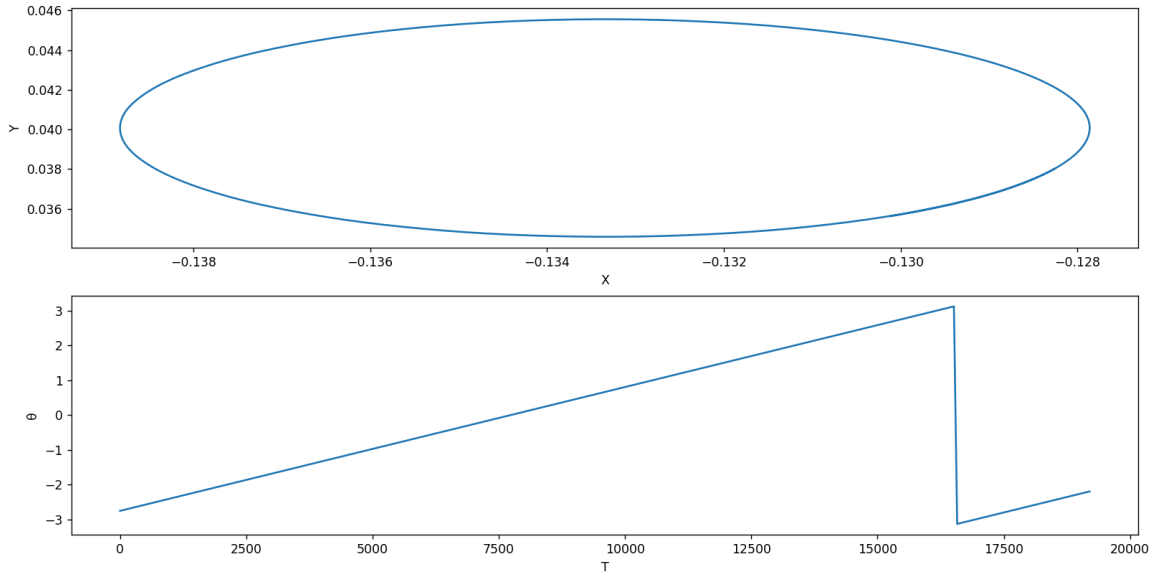


```python
from controller import Robot, Motor
import math
import numpy as np
import matplotlib.pyplot as plt

def inverse_kinematics(linear_velocity, angular_velocity,
heading_angle,
    axle_length, wheel_radius):

    inertial_frame = [linear_velocity[0],
                              linear_velocity[1],
                              angular_velocity]

    cos_theta = math.cos(heading_angle)
    sin_theta = math.sin(heading_angle)
    rotation_matrix = [[cos_theta, sin_theta, 0],
                       [-sin_theta, cos_theta, 0],
                       [0, 0, 1]]

    robot_frame = np.dot(rotation_matrix, inertial_frame)
    x_dot_r, y_dot_r, theta_dot_r = robot_frame

    phi_dot_1 = (x_dot_r + theta_dot_r * axle_length) / wheel_radius
    phi_dot_2 = (x_dot_r - theta_dot_r * axle_length) / wheel_radius

    return phi_dot_1, phi_dot_2


TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
```

```
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

# GPS
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

phi_dot_1, phi_dot_2 = inverse_kinematics([0, 0], 1, 0, 52, 20.5)

# # set up the motor speeds
leftMotor.setVelocity(phi_dot_1)
rightMotor.setVelocity(phi_dot_2)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 50):
        break
    c += 1
    t += TIME_STEP
```

```
# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

(b




```
from controller import Robot, Motor
import math
import numpy as np
import matplotlib.pyplot as plt

def inverse_kinematics(linear_velocity, angular_velocity,
heading_angle,
    axle_length, wheel_radius):

    inertial_frame = [linear_velocity[0],
```

```
                                linear_velocity[1],
                                angular_velocity]

    cos_theta = math.cos(heading_angle)
    sin_theta = math.sin(heading_angle)
    rotation_matrix = [[cos_theta, sin_theta, 0],
                       [-sin_theta, cos_theta, 0],
                       [0, 0, 1]]

    robot_frame = np.dot(rotation_matrix, inertial_frame)
    x_dot_r, y_dot_r, theta_dot_r = robot_frame

    phi_dot_1 = (x_dot_r + theta_dot_r * axle_length) / wheel_radius
    phi_dot_2 = (x_dot_r - theta_dot_r * axle_length) / wheel_radius

    return phi_dot_1, phi_dot_2


TIME_STEP = 64

MAX_SPEED = 6.28

# create the Robot instance.
robot = Robot()

# get a handler to the motors and set target position to infinity
(speed control)
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

# GPS
gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

# Compass
compass = robot.getDevice('compass')
compass.enable(TIME_STEP)
```

```
phi_dot_1, phi_dot_2 = inverse_kinematics([2, 2], 0.2, 0, 52, 20.5)

# # set up the motor speeds
leftMotor.setVelocity(phi_dot_1)
rightMotor.setVelocity(phi_dot_2)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 300):
        break
    c += 1
    t += TIME_STEP

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

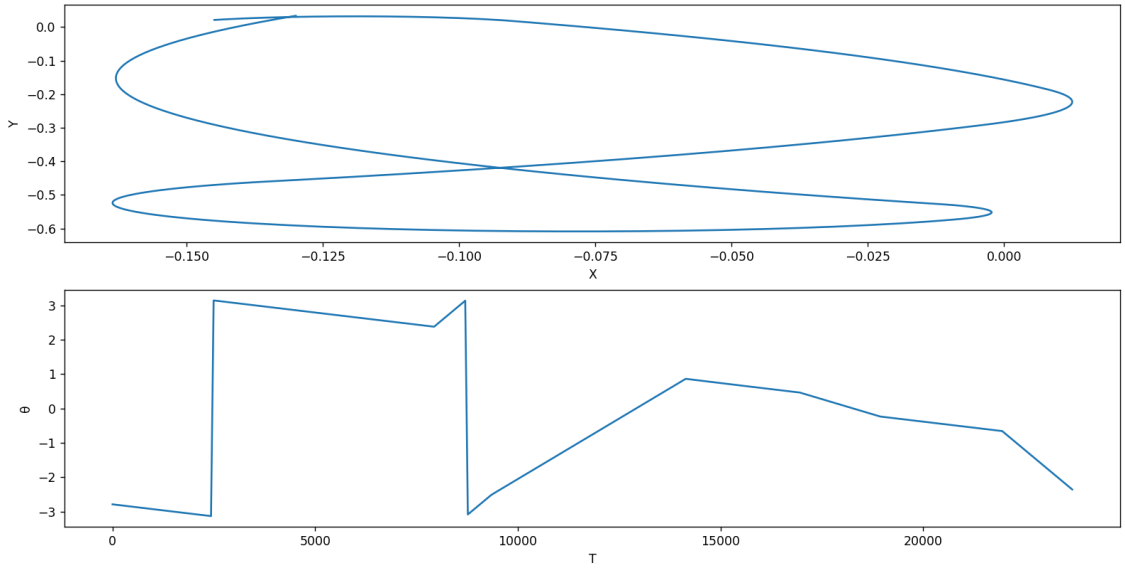
.3

a) در این بخش ابتدا سرعت های اولیه تعیین شده اند.

سرعت اولیه چرخ چپ برابر MAX_SPEED * 0.9 و سرعت اولیه چرخ راست برابر MAX_SPEED است.

سپس با استفاده از 6 شرط if، در زمان های لازم، تغییراتی در سرعت هر دو چرخ اعمال شده است تا مسیر مد نظر طی شود.

شکل کلی حرکت ربات تقریبا به صورت علامت بینهایت میشود.




```
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64
MAX_SPEED = 4
```

```
robot = Robot()
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

left_motor_velocity = 0.9 * MAX_SPEED
right_motor_velocity = MAX_SPEED

leftMotor.setVelocity(left_motor_velocity)
rightMotor.setVelocity(right_motor_velocity)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 370 ):
        break
    c += 1
```

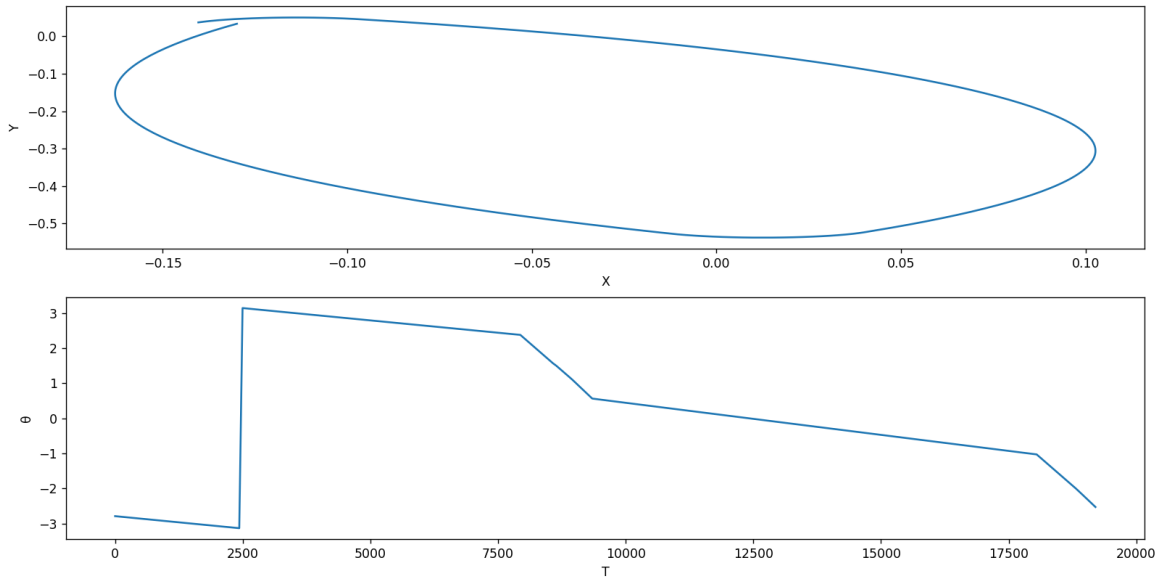
```
    t += TIME_STEP
    print(t)

    if t >= 8000 :
        left_motor_velocity = 0.8 * MAX_SPEED
        right_motor_velocity = 0.1 * MAX_SPEED

    if t >= 9200:
        left_motor_velocity = MAX_SPEED
        right_motor_velocity = 0.5 * MAX_SPEED

    if t >= 14000:
        left_motor_velocity = 0.9 * MAX_SPEED
        right_motor_velocity = MAX_SPEED

    if t >= 17000:
        left_motor_velocity = 0.7 * MAX_SPEED
        right_motor_velocity = 0.8 * MAX_SPEED

    if t >= 18000:
        left_motor_velocity = 0.9 * MAX_SPEED
        right_motor_velocity = MAX_SPEED

    if t >= 21000:
        left_motor_velocity = 0.1 * MAX_SPEED
        right_motor_velocity = 0.8 * MAX_SPEED

    leftMotor.setVelocity(left_motor_velocity)
    rightMotor.setVelocity(right_motor_velocity)

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```

b) در این بخش ابتدا سرعت های اولیه تعیین شده اند.

سرعت اولیه چرخ چپ برابر MAX_SPEED * 0.9 و سرعت اولیه چرخ راست برابر MAX_SPEED است.

سپس با استفاده از 3 شرط if، در زمان های لازم، تغییراتی در سرعت هر دو چرخ اعمال شده است تا مسیر مد نظر طی شود.

شکل کلی حرکت ربات تقریبا به صورت علامت بینهایت میشود.




```
from controller import Robot, Motor
import math
import matplotlib.pyplot as plt

TIME_STEP = 64
MAX_SPEED = 4

robot = Robot()
leftMotor = robot.getDevice('left wheel motor')
rightMotor = robot.getDevice('right wheel motor')
```

```
leftMotor.setPosition(float('inf'))
rightMotor.setPosition(float('inf'))

gps = robot.getDevice('gps')
gps.enable(TIME_STEP)

compass = robot.getDevice('compass')
compass.enable(TIME_STEP)

left_motor_velocity = 0.9 * MAX_SPEED
right_motor_velocity = MAX_SPEED

leftMotor.setVelocity(left_motor_velocity)
rightMotor.setVelocity(right_motor_velocity)

c = 1
t = 0

# create x, y, theta, time lists
x = []
y = []
theta = []
time = []

while robot.step(TIME_STEP) != -1:
    gps_value = gps.getValues()
    x.append(gps_value[0])
    y.append(gps_value[1])

    compass_value = compass.getValues()
    time.append(t)
    theta.append(math.atan2(compass_value[1], compass_value[0]))

    if (c > 300):
        break
    c += 1
    t += TIME_STEP
    print(t)
```

```
    if t >= 8000 :
        left_motor_velocity = 0.1 * MAX_SPEED
        right_motor_velocity = MAX_SPEED

    if t >= 9408:
        left_motor_velocity = 0.87 * MAX_SPEED
        right_motor_velocity = MAX_SPEED

    if t >= 18100:
        left_motor_velocity = 0.1 * MAX_SPEED
        right_motor_velocity = MAX_SPEED

    leftMotor.setVelocity(left_motor_velocity)
    rightMotor.setVelocity(right_motor_velocity)

# Plot
fig, ax = plt.subplots(2)

ax[0].set(xlabel='X', ylabel='Y')
ax[0].plot(x, y)

ax[1].plot(time, theta)
ax[1].set(xlabel='T', ylabel='θ')
plt.show()
```