به نام خدا

اعضای گروه: امیرحسین نوری - کیمیا ایمنی - امیرمسعود شاکر

تمرین اول

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.

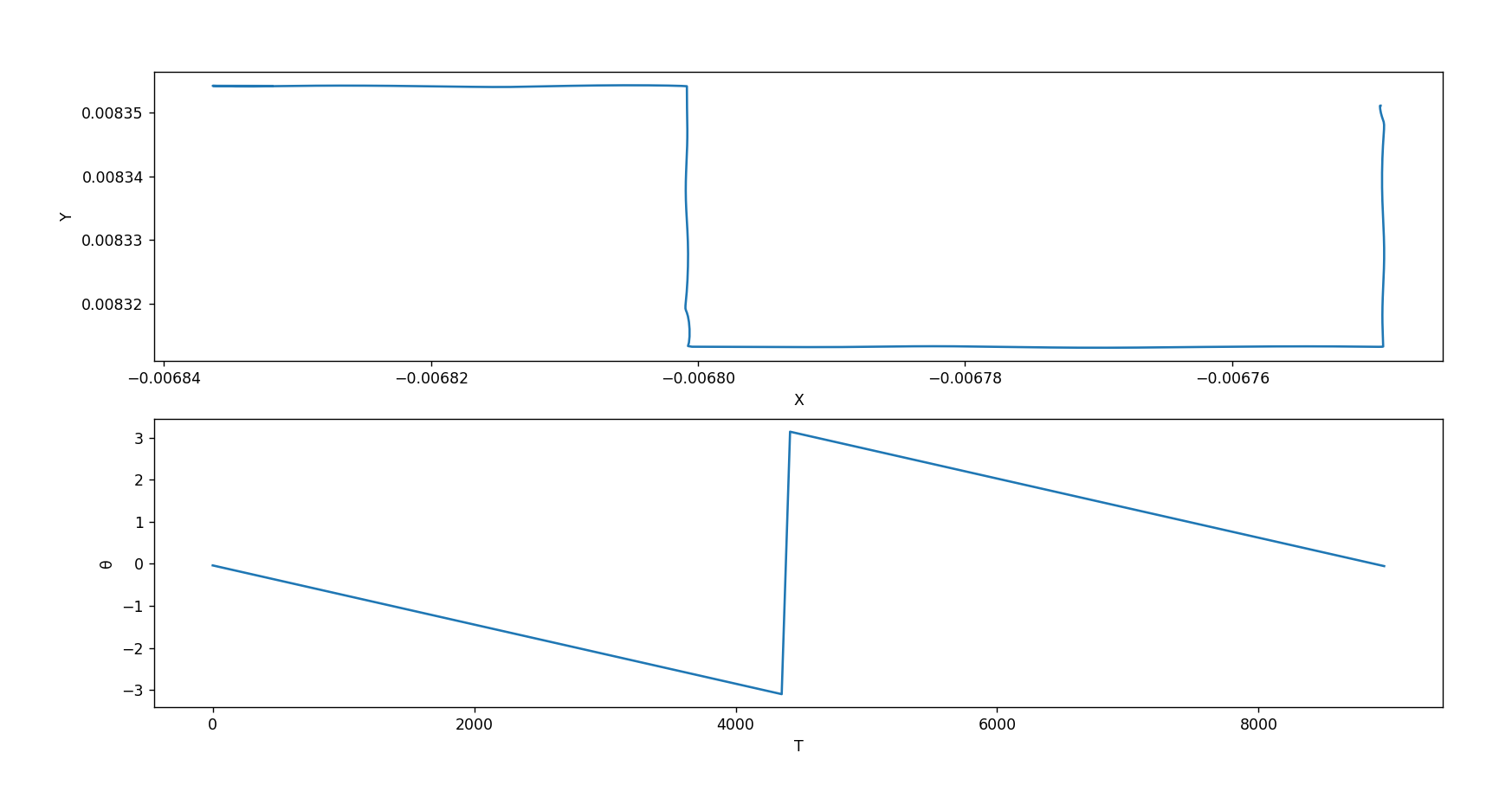
چهار لیست برای ذخیره کردن مختصات x, y, theta, time میسازیم.

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.



­­­­­­­

*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)

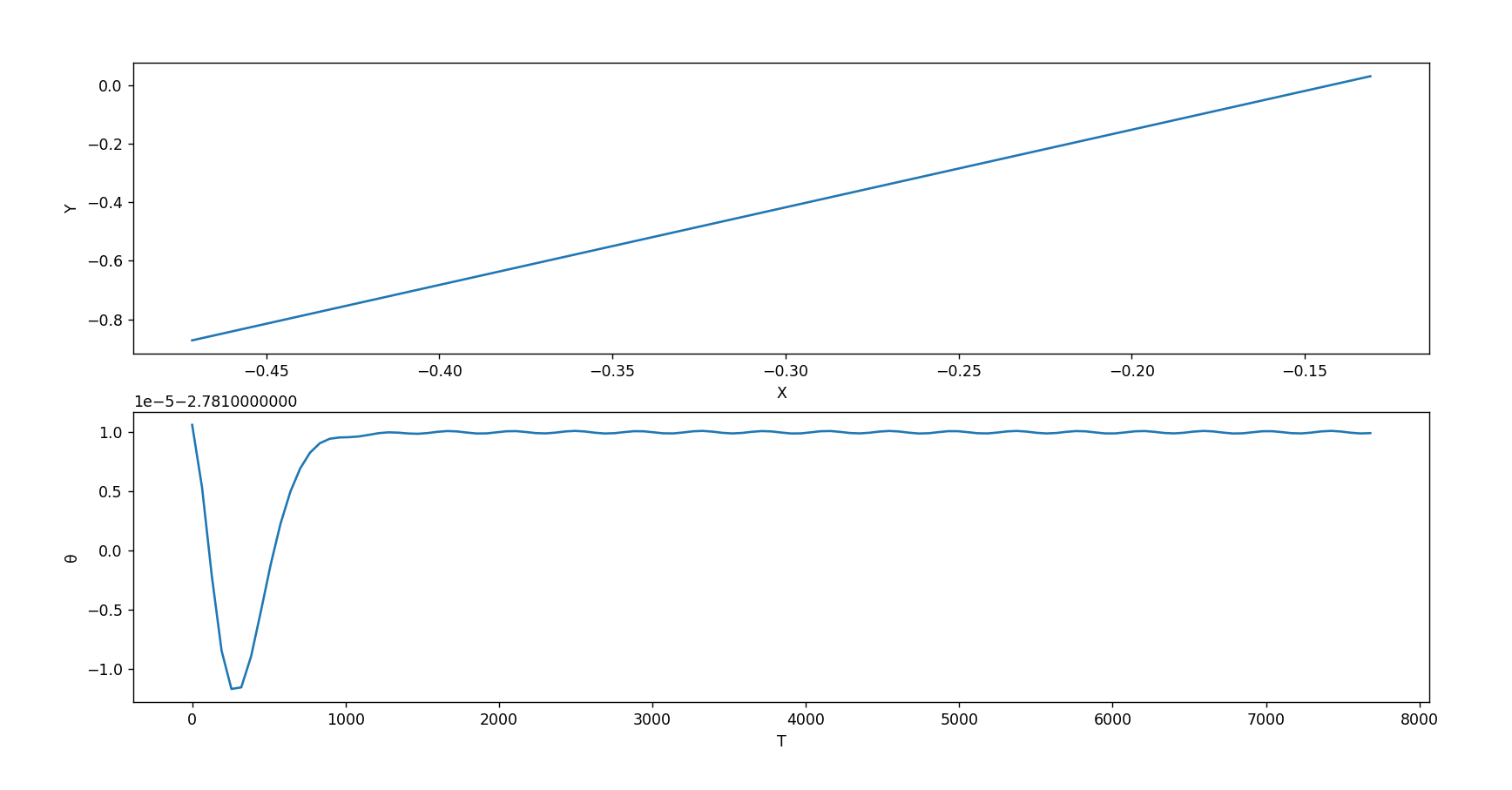
ax[1].set(xlabel='T', ylabel='θ')

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()

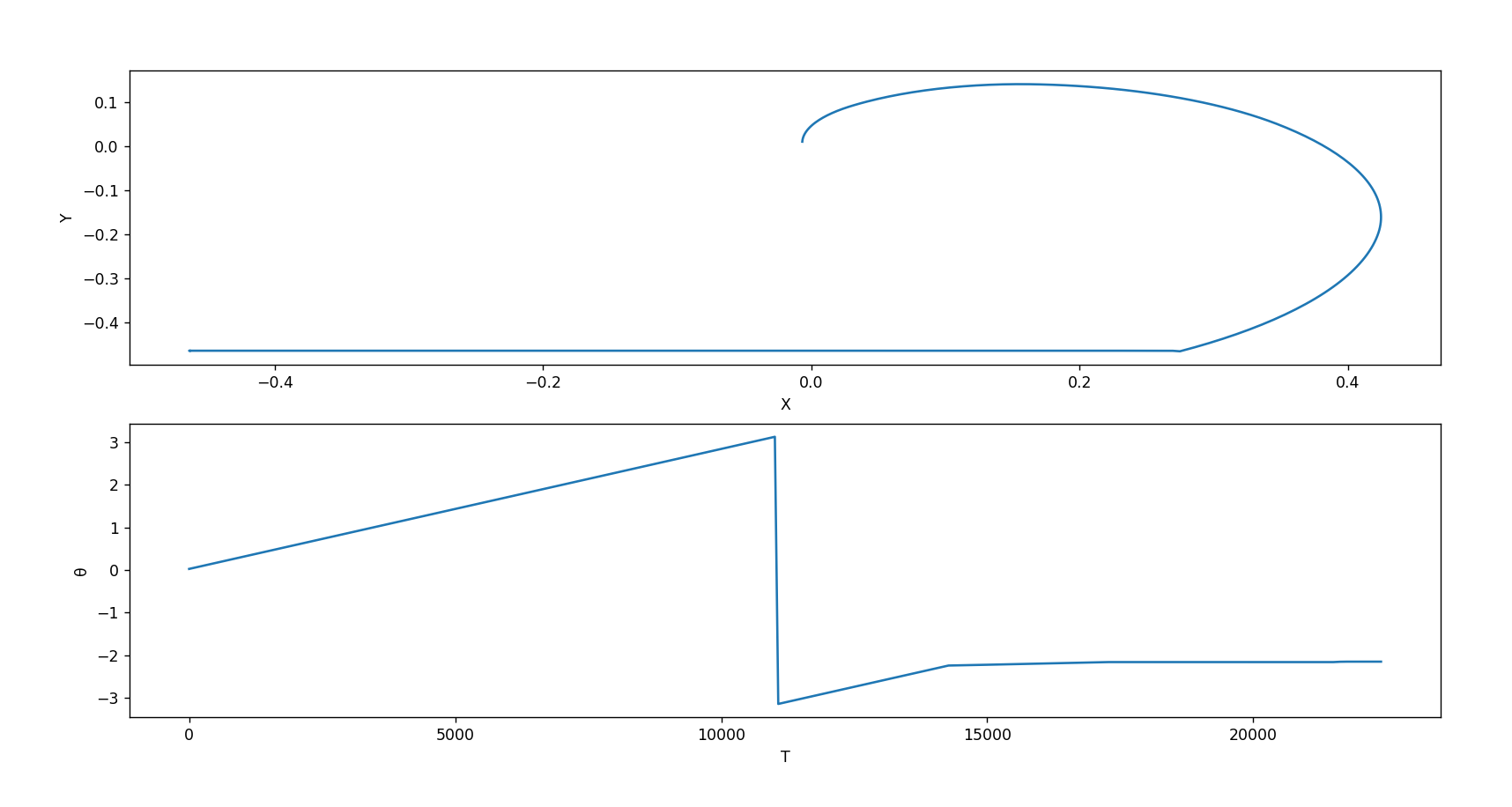
c) در این بخش ابتدا سرعت های اولیه چرخ ها را تعیین میکنیم.

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.



*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)

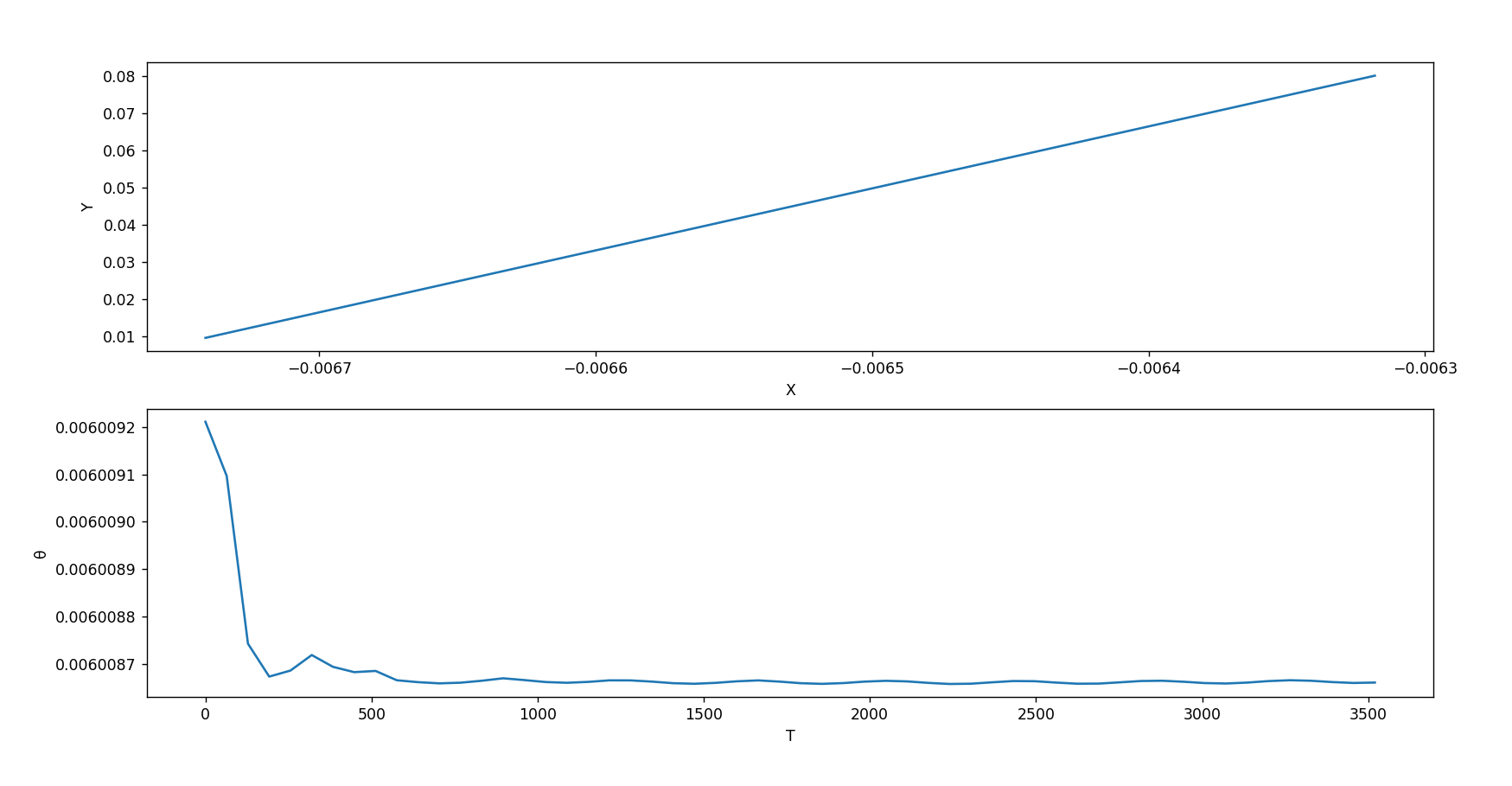
ax[1].plot(time, theta)

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.



*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) توضیح تابع سینماتیک معکوس:

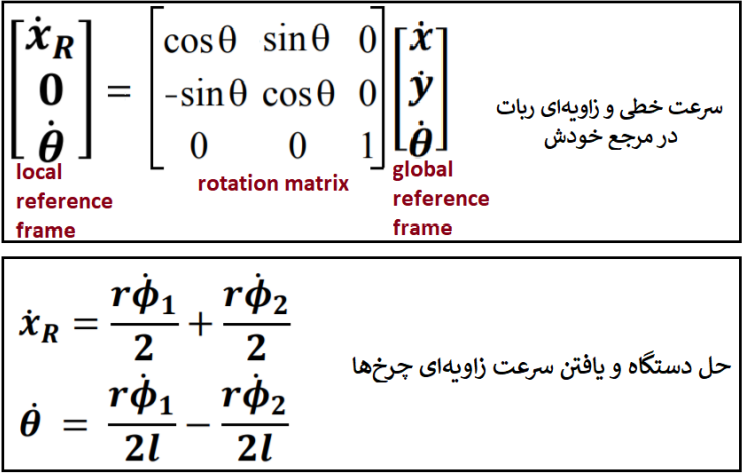
تابع ما در ورودی سرعت خطی، سرعت زاویه ای، جهت سر ربات، طول محور و شعاع چرخ را میگیرد.

سپس وکتور intertial frame را میسازیم که شامل مولفه های x\_dot, y\_dot, theta\_dot است.

ماتریس R(theta) را میسازیم.

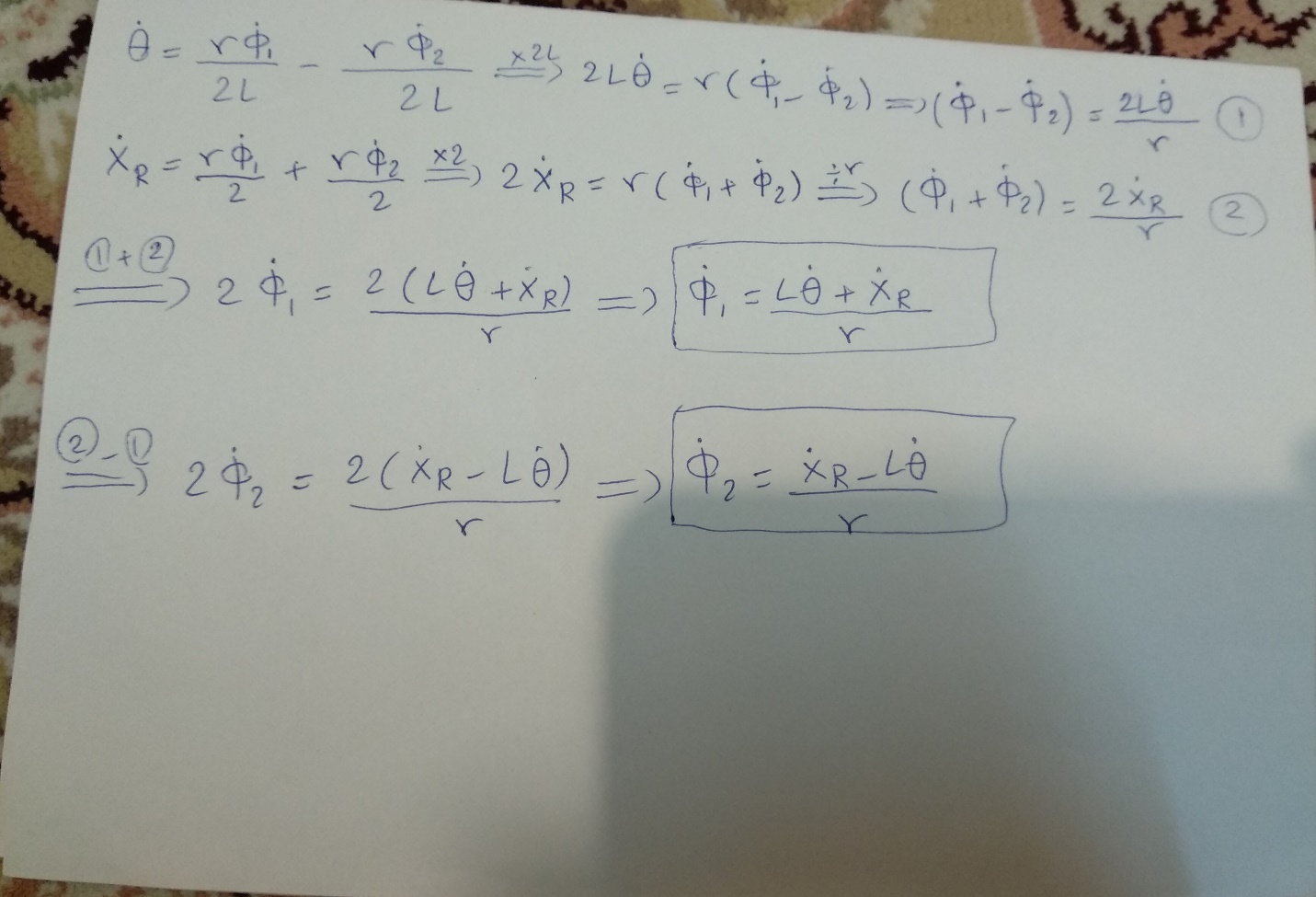
حاصل ضرب ماتریس R(theta) و inertial frame را برابر robot\_frame قرار میدهیم.

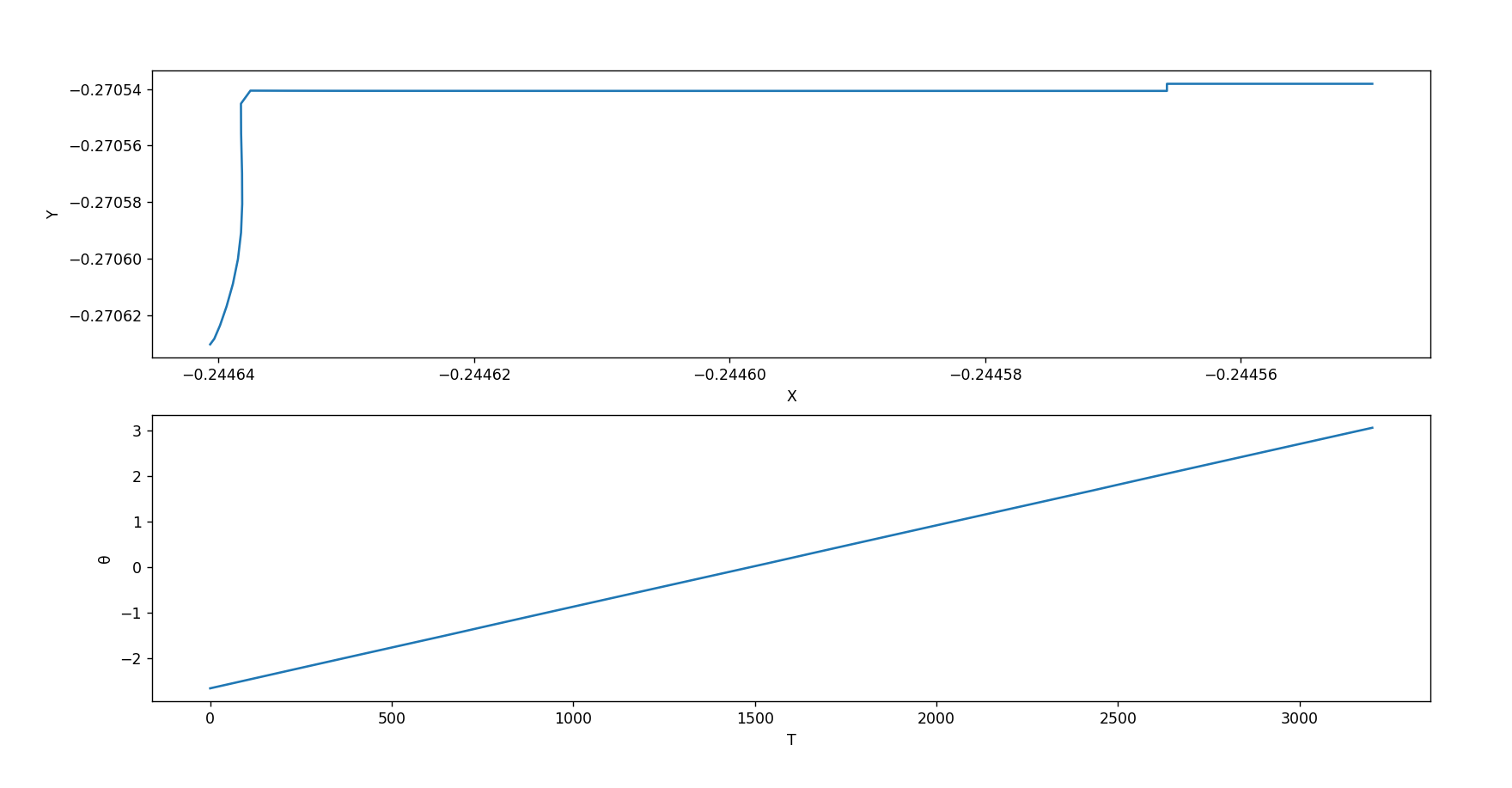
با استفاده از robot frame محاسبه شده، مقادیر سرعت خطی و زاویه ای ربات را مقداردهی میکنیم.



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:





*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')

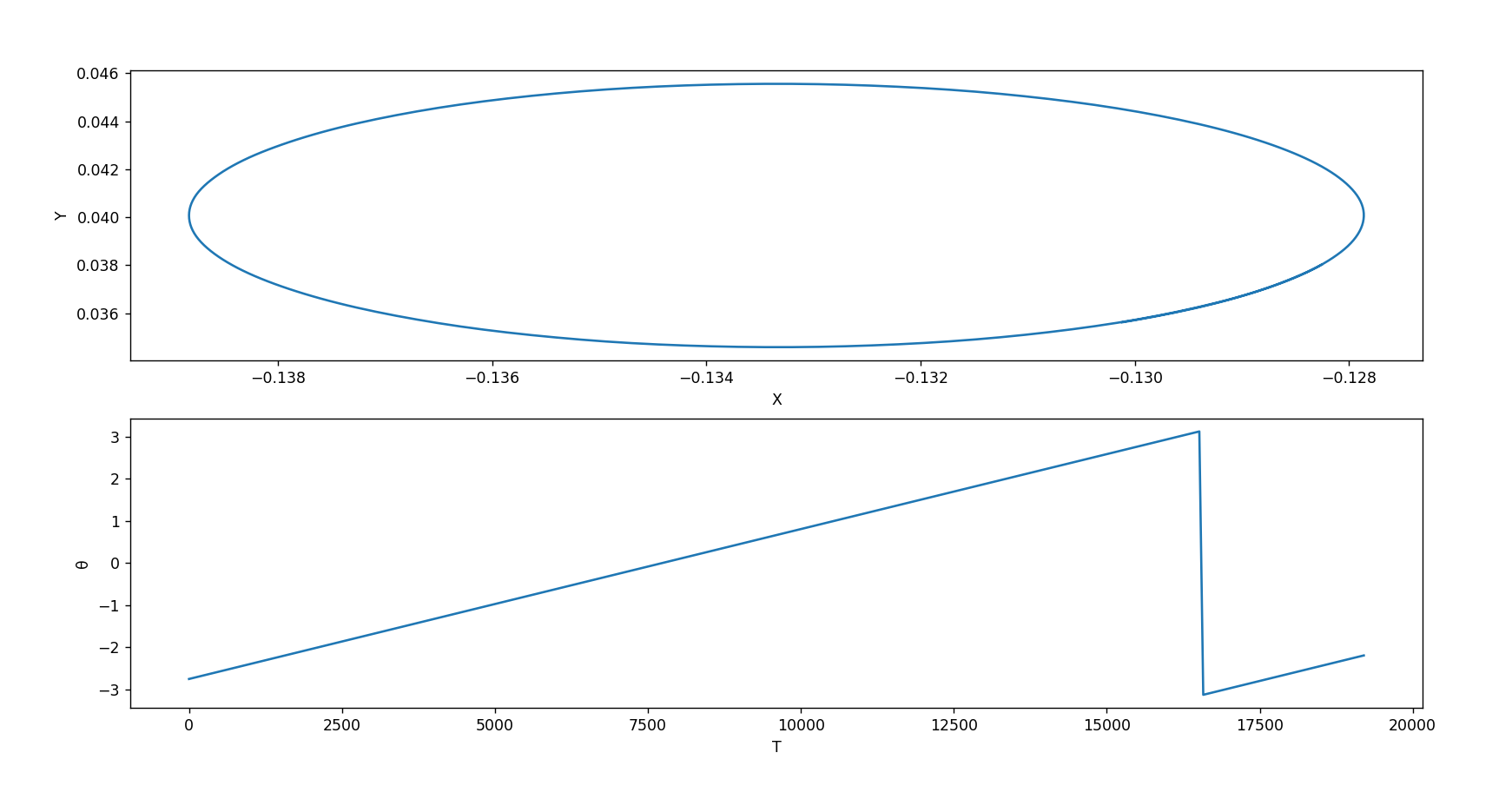
ax[0].plot(x, y)

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()

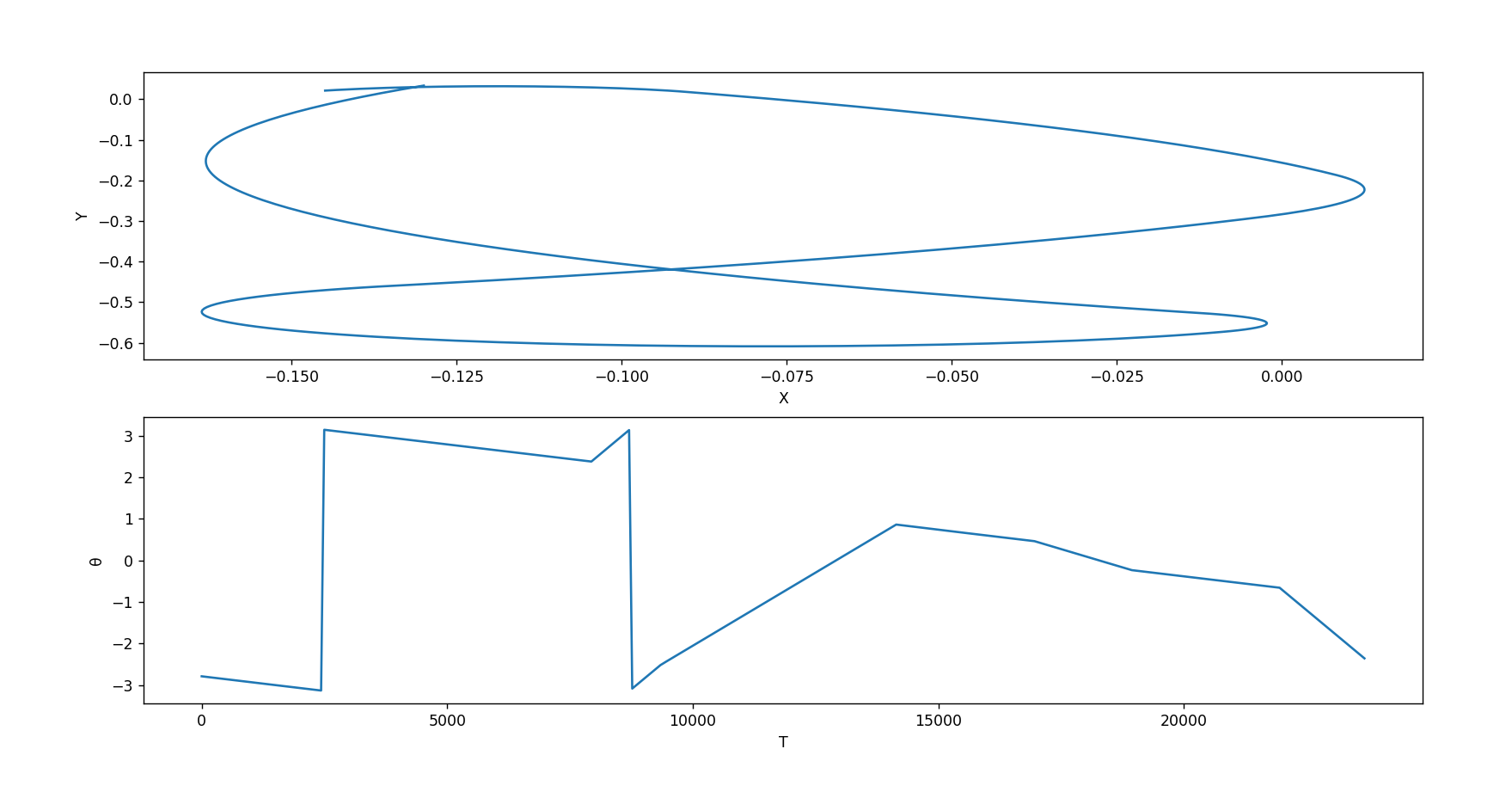
3.

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

سرعت اولیه چرخ چپ برابر 0.9 \* MAX\_SPEED 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')

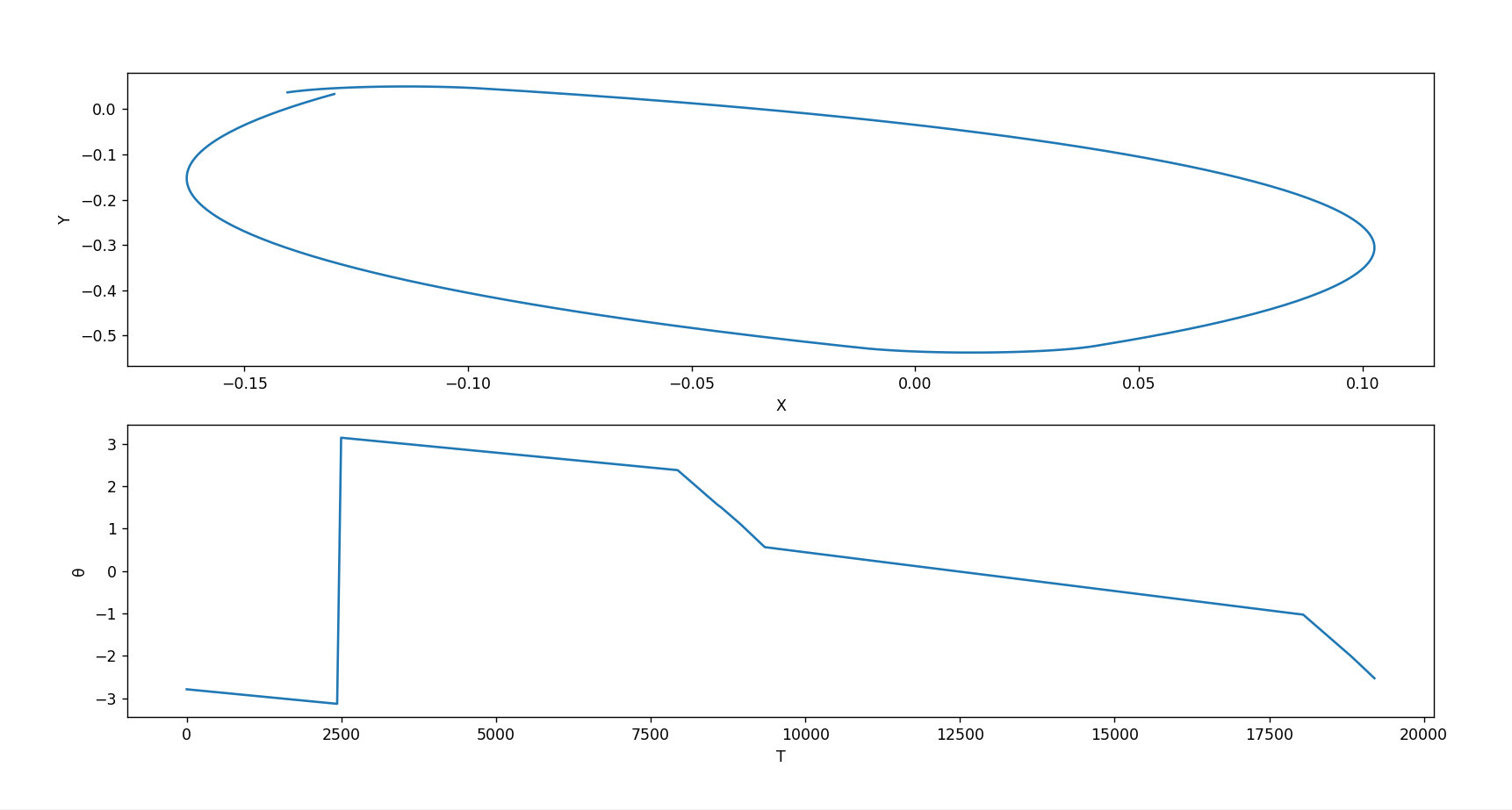
plt.show()

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

سرعت اولیه چرخ چپ برابر 0.9 \* MAX\_SPEED 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()