بسمه تعالی

نوید رزاقی

810199424

مینی پروژه 4 مکاترونیک

با توجه به اسم من (Navid) و 3 شماره آخر دانشجویی ام (424) ، عبارتی که توسط لاکپشت ها باید نوشته شود :

NIVIIIV

بدین منظور من برای هر حرف از حروف بالا ، یک نود ایجاد کردم و آن را در فایل drawing.launch صدا زدم تا اجرا شود . همچنین برای هر حرف یک نود جداگانه به نام "turtle_name of the letter_trace_color.py"

ایجاد کردم تا هر حرف که نوشته میشود ، همزمان رنگش هم به رنگ دلخواه من شود .

اول از همه در لانچ فایل ، باید نود turtlesim_node را وارد کنیم و این نود را صدا بزنیم تا پنجره ی لاک پشت باز شود سپس دستور های لازم را در ادامه میدهیم .

در اولین گام می آییم لاکپشت اولیه که در هنگام باز کردن turtlesim در وسط پنجره دیده می شود را kill میکنیم. سپس در هر کدام از نود ها ، لاکپشتی را در مختصات مشخصی از پنجره بازشده ، spawn میکنیم . سپس دستور حرکت را با استفاده از تابعی که خودمان تعریف کردیم(move_turtle) میدهیم .

همانطور که اشاره شد، برای حرکت لاکپشت ها یک تابع نوشته ام به نام move_turtle با تعریف زیر:

```
def move_turtle(turtle_name, linear_vel,vx,vy,desired_distance)
```

این تابع اینگونه عمل میکند که نام لاکپشتی که قرار است حرکتی را انجام دهد ، سرعت خطی حرکت ، سرعت در راستای x ، سرعت در راستای y و در نهایت مسافت مطلوب را به عنوان ورودی به این تابع میدهیم. من طبق الگوریتمی که در فیلم های آموزشی برای حرکت لاکپشت توضیح داده شد، رفتم . بدین گونه که تا وقتی لاکپشت مسافت مطلوب (desired_distance)

را طی نکرده ، پیام twist را منتشر(publish) کند. این موضوع در قطعه کد زیر نشان داده شده است :

```
t0 = rospy.Time().now().to_sec()
while current_distance <= desired_distance :
    turtle_pub.publish(twist_msg)
    t1 = rospy.Time().now().to_sec() - t0
    current_distance = linear_velocity * t1
    rate.sleep()
```

برای رنگ حرف ها هم اینگونه عمل کردم که از سرویس set pen از سرویس های آماده turtlesim استفاده کردم . سپس با تعیین مقادیر r و g و b میتوانیم رنگ مطلوب را ایجاد کنیم .

نتیجه ی نهایی به صورت زیر می شود :