پروژه رباتیک

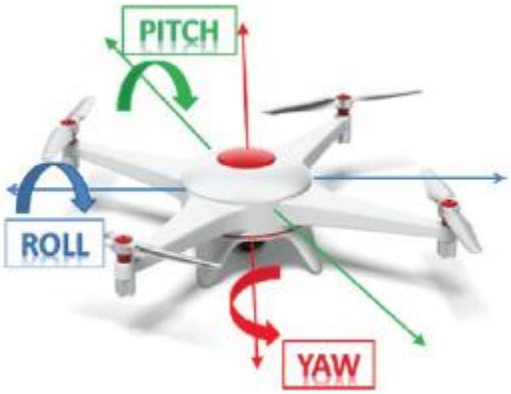
اعضای گروه:

زهرا استادمحمدی – ۹۸۲۴۳۱۰۷

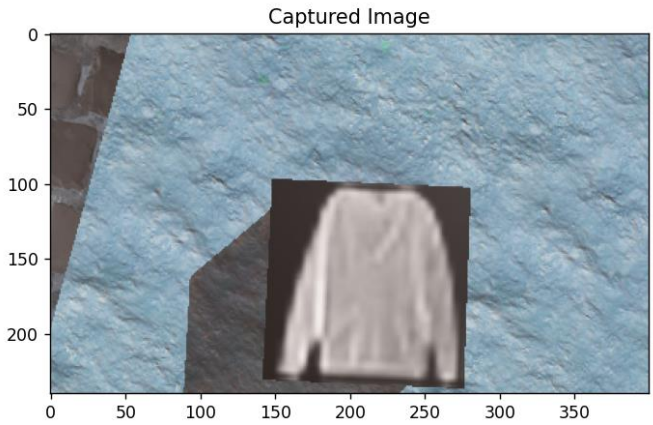
حمیدرضا حیدری – ۹۸۲۴۳۰۱۹

علی نوران – ۹۸۲۴۳۰۶۳

ابتدا برای اینکه ربات به سمت مقصد های مدنظر حرکت کند باید موقعیت‌های آن ها را داشته باشیم سپس باید با استفاده از سنسورهای ربات موقعیت ربات و pitch که بیانگر چرخش حول محور y است و roll که بیانگر چرخش حول محور x است و yaw که بیانگر چرخش حول محور z است را استفاده کنیم.



ما می‌توانیم با استفاده از سنسور gyro سرعت چرخش ربات در محور های مختلف را بدست آوریم. باید برحسب اختلاف فاصله در محورهای مختلف و اختلاف زاویه ای که با ربات داریم مقادیر pitch و yaw و roll را تغییر دهیم تا در جهت درست حرکت کنیم. همچنین باید از سنسور camera استفاده کنیم تا بتوانیم تصویر برداری از اشیا را انجام دهیم. برای اینکه زمانی که به شی مدنظر رسیدیم بتوانیم در زاویه مناسبی تصویر برداری را انجام بدیم باید کوادکوپتر صاف باشد و yaw را طوری تنظیم کنیم که عکس به در جهت مناسبی گرفته شود(وارانه نباشد).



در این پروژه ما از کنترلری استفاده کرده ایم که روی مقادیر yaw و roll و pitch تاثیر می‌گذارد و برحسب میزان خطا آن‌ها را اپدیت می‌کند. در کد زیر مقادیر ثابت کنترلر در مقدار ورودی ضرب شده سپس با ناهنجاری ایجاد شده و مقدار بدست آمده از سنسور gyro که سرعت چرخش یا انحراف ربات در زوایا مختلف است جمع می‌شود(برای yaw ,roll). سپس با استفاده از مقادیری که با استفاده از کنترلر بدست آمده اند سرعت زاویه ای چرخ ها را بدست می‌آوریم. نکته ای که وجود دارد این است که برای چرخ های جلو باید pitch تاثیر مثبت داشته باشد و برای چرخ های عقب تاثیر منفی، همچنین برای چرخ های سمت راست roll باید تاثیر مثبت داشته باشد و برای چرخ‌های سمت چپ منفی.

```
roll_in = self.ROOL_PROPORTIONAL * clamp(roll, -1, 1) + roll_acceleration + roll_deviation
pitch_in = self.PITCH_PROPORTIONAL * clamp(pitch, -1, 1) + pitch_acceleration + pitch_deviation
yaw_input = yaw_deviation
clamped_difference_height = clamp(self.target_height - height + self.VERTICAL_DISPLACEMENT_CONSTANT, -1, 1)
vertical_input = self.K_VERTICAL_P * np.power(clamped_difference_height, 3.0)

thrust = self.VERTICAL_CONSTANT + vertical_input

front_left_motor_input = thrust - yaw_input + pitch_in - roll_in
front_right_motor_input = thrust + yaw_input + pitch_in + roll_in
rear_left_motor_input = thrust + yaw_input - pitch_in - roll_in
rear_right_motor_input = thrust - yaw_input - pitch_in + roll_in

self.front_left_motor.setVelocity(front_left_motor_input)
self.front_right_motor.setVelocity(-front_right_motor_input)
self.rear_left_motor.setVelocity(-rear_left_motor_input)
self.rear_right_motor.setVelocity(rear_right_motor_input)
```

شبکه عصبی پیچشی:

در این بخش برای تشخیص نوع پوشاک موجود در تصویر، نیاز است تا به کمک شبکه عصبی پیچشی مدلی را آموزش دهیم. مدل ما از چهار لایه کانوولوشنی، دو لایه پولینگ و دو لایه fully connected تشکیل شده است. در لایه های پولینگ، ما از مکس پولینگ با پنجره ۲*۲ استفاده کردیم. در لایه های کانوولوشنی و لایه اول fully connected از تابع فعالیت RELU استفاده شده و در لایه آخر، از سافت مکس استفاده کردیم تا در خروجی احتمال تعلق عکس ورودی به هر کلاس را داشته باشیم.

دقت این مدل نیز بر روی مجموعه‌داده آموزشی داده شده برابر ۹۸ درصد بوده است.

برای استفاده از این مدل در ربات، ساختار مدل را در یک فایل json و وزن های آموخته شده را در فایل h5 ذخیره کردیم. بنابراین ربات نیازی به محاسبات سنگین برای یادگیری مدل ندارد و تنها کافیست این دو فایل را load کند.

لینک آپارات https://www.aparat.com/v/f0F1w