بسم الله الرحمن الرحیم

ربات‌های متحرک خودگردان – دی‌ماه ۱۴۰۳

پروژه پایانی: جستجو و موقعیت‌یابی اشیاء در محیط

در این تمرین فرض شده است تخمین‌گر وضعیت و کنترل‌گر سطح پایین و بالا از قبل پیاده‌سازی شده و در هر لحظه دسترسی کامل به بردار حالت تخمین‌گر داریم. همچنین امکان جابجایی پرنده به هر نقطه و هر ارتفاع در محیط را داریم.

هدف اصلی پروژه جستجوی محیط برای یافتن تمامی خودروها و انسان‌ها در آن و ثبت موقعیت آن‌ها در یک فرودگاه است. همچنین فعالیت امتیازی این تمرین شامل یافتن دوچرخه‌ها و موتورسیکلت‌ها و هواپیماها در محیط می‌باشد. لازم به ذکر است که تمامی اشیاء که می‌بایست شناسایی شوند در فضای محدود به جاده پیرامونی فرودگاه واقع شده‌اند (تصویر زیر).

چالش‌ها:

- انجام کالیبراسیون دوربین با شیء مرجع محیط
- انجام تبدیلات مربوط به بازنمایی وضعیت پرنده و دوربین
- ناوبری پرنده و آشنایی بیشتر با موقعیت و زوایای دوربین نسبت به بدنه و زوایای پرنده نسبت به دستگاه مختصات جهانی
- انتخاب استراتژی جستجو و پیاده‌سازی جستجوی سریع و جامع برای یافتن اهداف
- شناسایی اشیاء در محیط مبتنی بر تصاویر هوایی
- محاسبه موقعیت سه‌بعدی اهداف تشخیص داده شده در دستگاه مختصات جهانی مبتنی بر مدل افکنش دوربین، موقعیت دوربین، موقعیت پرنده و موقعیت هدف تشخیص داده شده در تصویر و گزارش موقعیت اهداف در یک plot یا نقشه

تشریح پروژه:

هر یک از دانشجویان دارای شناسه مربوط به خود هستند. پرنده شما در یک نقطه از محیط واقع شده و سپس شما می‌بایست با توابع سطح بالای کنترلی به جستجو و شناسایی اهداف در محیط اقدام کنید. بدین منظور دسترسی به تصاویر بلادرنگ دوربین پرنده خود خواهید داشت و می‌بایست مبتنی بر مدل‌های پردازش تصاویر اقدام به بازشناسی هدف در تصویر دریافت شده کنید. اهداف اصلی شما افراد و خودروهای محیط هستند که می‌بایست آن‌ها را شناسایی و موقعیت آن‌ها را گزارش کنید. اهداف فرعی بخش امتیازی نیز هواپیما، هلیکوپتر، دوچرخه و موتورسیکلت می‌باشند.

بخش امتیازی:

در این پروژه از دو جنبه به صورت اختیاری برای کسب نمره بیشتر فعالیت امتیازی تعریف شده است:

۱. شناسایی افراد و خودرو به دلیل پر استفاده بودن، دارای مدل‌های از قبل آموزش داده شده عمومی قابل دریافت نظیر VisDrone می‌باشند. در صورت علاقه می‌توانید کلاس هواپیما و هلیکوپتر را به مدل خود اضافه کنید. بدین منظور و برای آموزش مدل می‌توانید از تصاویر موجود در شبیه‌ساز و یا تصاویر عمومی موجود بر روی اینترنت استفاده کنید.

۲. چند دوچرخه و موتورسیکلت نیز در محیط به‌گونه‌ای قرار گرفته‌اند که صرفاً با پروازِ در ارتفاع نمی‌توانید آن‌ها را شناسایی کنید و برای یافتن آن‌ها می‌بایست اقدام به گشتن سوله‌ها و سایر فضاها داشته باشید. بدین منظور و برای جلوگیری از تصادف می‌توانید از تصاویر حاوی اطلاعات عمق استفاده کنید. دوچرخه و موتور سیکلت نیز از شیئ‌هایی هستند که در مدل‌های عمومی آموزش دیده شده وجود دارند و برای انجام این بخش امتیازی تمرین، نیازی به آموزش مدل جدید ندارید.

گزارش خروجی:

پروژه هر دانشجو جداگانه می‌بایست انجام شده و دارای یک گزارش شامل تشریح در مورد روش انجام کار و خروجی‌ها می‌باشد. لازم به ذکر است صرف گزارش موقعیت‌ها از جمله اهداف پروژه نیست و در روز ارائه می‌بایست یک مرتبه الگوریتم شما اجرا شده و موقعیت‌ها در همان لحظه نیز استخراج شود. ایده مربوط به روش جستجو، مدل‌های مربوط به بازشناسی، محاسبات مربوط به موقعیت و البته بخش‌های امتیازی نظیر اجتناب از موانع مبتنی بر تصاویر دوربین عمق و بازشناسی اشیاء جدید از جمله فعالیت‌هایی است که مجموع آن‌ها نمره نهایی شما را تشکیل می‌دهد.

نکته مهم: به عنوان یک قانون توجه کنید که تنها می‌توانید از موقعیت از قبل دانسته ورودی سوله‌ها و پد کالیبراسیون استفاده کنید و جستجوی دستی و دانستنِ از قبلِ موقعیت سایر اشیاء در انجام این پروژه مجاز نیست.

راهنمایی‌ها:

- موقعیت‌های گزارش شده در بردار حالات مبتنی بر دستگاه مختصات کارتزین جهانی است.
- در حین پرواز ممکن است شیء هدف را از زاویه‌ای دیگر مجدد ببینید که این موضوع هم می‌تواند چالش و هم اعتبار سنجی موقعیت ثبت شده قبلی باشد.
- دوربین اصلی نسبت به پرنده در موقعیت $t = [0, 0, 0.24]^T$ و در ابتدا زاویه pitch ۲۵ درجه رو به پایین دارد. این وضعیت دوربین را باید در کنار وضعیت پرنده درنظر بگیرید تا وضعیت نهایی دوربین را در دستگاه مختصات جهانی بتوانید برای انجام محاسبات موقعیت اشیاء استفاده کنید.
- دوربین عمق مشابه دوربین اصلی می‌باشد. درجه روشنایی معنای عمق بیشتر دارد. بیشترین سفیدی معادل ۸۰ متر فاصله و بیشتر و سیاهی معادل فاصله صفر متر است. پارامترهای داخلی دوربین عمق و دوربین اصلی یکی خواهند بود.

نمونه‌ای از تصاویر دو دوربین در ادامه آمده است:

- با استفاده از توابع فراهم شده می‌توانید زاویه دوربین‌ها را [مستقلاً] نسبت به بدنه تغییر بدهید.
- مشخصات زوج دوربین هر پرنده نسبت به پرنده دیگر تفاوت دارد و نسبت به هر پرنده یکتا است. لذا می‌بایست هر یک از دانشجویان کالیبراسیون دوربین خود را مبتنی بر شیء مرجع جداگانه انجام بدهند.
- برای کالیبراسیون می‌توانید از شیء مرجع که هر یک از مربع‌های صفحه شطرنجی یک متر در یک متر می‌باشند استفاده کنید. برای داده‌برداری از شیء مرجع می‌توانید موقعیت تقریبی آن را بدست آورده و سپس پرواز و داده‌برداری خودکار انجام بدهید[1] و هم با ابزارک flight_with_keyboard پرواز بر روی شیء مرجع انجام داده و تعداد مناسبی تصویر از شیء مرجع با کلید space در پوشه snapshots ذخیره و کالیبراسیون را انجام بدهید (تصویر زیر). البته مناسب است زاویه دوربین را در هنگام داده‌برداری از شیء مرجع تغییر بدهید تا بتوانید از شیء مرجع از زوایای مختلف تصاویر متعدد داشته باشید.

[1] صفحه شطرنجی به عنوان شیء مرجع [تقریباً] در موقعیت (8- ,30- ,285-) واقع شده است.

- در شناخت جهت مثبت x در دستگاه مختصات جهانیِ ، دانستن جهت پرنده در بازنمایی NED مهم است. در بازنمایی NED هنگامی که دوربین دارای yaw=0 نسبت به بدنه و خود پرنده دارای yaw=0 در دستگاه مختصات ثابت خود باشد، سمت جلوی پرنده، سمت مثبت x و سمت راست مثبت y است. و البته مثبت z سمت پایین خواهد بود (قانون دست راست).
- در این تمرین نیاز به در اختیار داشتن اینترنت با تاخیر خیلی پایین نیست و صرفاً در اختیار داشتن پهنای باند ۲ تا ۴ مگابیت برثانیه کافی است.

راهنمای واسط برنامه‌نویسی:

- پیش‌نیازهای نرم‌افزاری ماژول tools بسته‌های redis-py، numpy و open-cv می‌باشد.
- برای انجام تبدیلات زوایا برخی توابع scipy می‌تواند به شما کمک کند.
- برای نمایش تصاویر و پردازش تصاویر ممکن است نیازمند نصب بسته‌های opencv و بسته‌های پردازش تصاویر مبتنی بر شبکه‌های عمیق نظیر pytorch و سیستم‌های بازشناسی اشیاء نظیر yolo یا vis-drone باشید.
- هر یک از دانشجویان دارای یک هواگرد و رمز ورود هستند که جداگانه در اختیار آن‌ها قرار می‌گیرد.
- مجموعه توابع زیر می‌تواند در ناوبری هواگرد و تصویربرداری به شما کمک کند.

function	description
immediate_stop	Cancel last command and hover immediately
reset	Teleport your drone to the home
rotate_yaw_rate	Rotate drone to by a specific yaw rate (relative rotation – degrees/s)
rotate_to_yaw	Rotate drone to a specified yaw (Absolute NED)
move_by_body_vels	Move by body cartesian speeds
move_to_position	Move by absolute NED cartesian speeds
teleport_to_position	Intentionally disabled
get_cam_image	Fetch latest drone main cam image
get_depth_image	Fetch latest drone depth cam image
get_drone_state	Fetch latest drone state
set_main_cam_pose	Set main camera pose (wrt body)
set_depth_cam_pose	Set depth camera pose (wrt body)

- به جهت‌های مثبت و منفی در موقعیت‌ها و بردار سرعت‌ها توجه کنید. این موضوع به‌ویژه در تنظیم ارتفاع ممکن است خطا برانگیز باشد. همچنین سرعت‌ها و موقعیت‌ها متریک می‌باشند.
- محدوده فضای قابل جستجو (گوشه بالا سمت چپ و گوشه پایین سمت راست به صورت زیر است:

$[-500, -180]^T$

$[300, 240]^T$

- به غیر از تابع reset و immediate_stop، تمامی توابع نیازمند ورودی duration یا timeout برحسب ثانیه هستند. بدین معنا که شما مثلاً می‌خواهید با سرعت ۲ متر برثانیه پرنده به سمت جلو (جهت مثبت x فریم بدنه) حرکت کند و این حرکت ۱۰ ثانیه به‌طول انجامد. یعنی تقریباً پرنده ۲۰ متر به سمت جلو حرکت خواهد کرد. هنگامی که این فرمان صادر شود، تا طی شدن duration (یا timeout در سایر فرامین) تمامی فرمان‌های دیگر شما به غیر از immediate_stop نادیده گرفته خواهد شد. یعنی اگر خواستید فرمان را تغییر بدهید ابتدا باید immediate_stop و سپس فرمان جدید صادر کنید.
- با استفاده از تابع get_cam_image و get_depth_image تصویر بلادرنگ فعلی دوربین‌های پرنده را می‌توانید دریافت کنید که می‌توانید این تصاویر را به زیرسیستم پردازش تصویر خود ارسال کنید. خروجی این توابع مقادیر برچسب زمانی تصاویر و موقعیت دوربین‌ها [نسبت به بدنه پرنده] می‌باشد. واحد برچسب زمانی مشابه واحد برچسب زمانی در بردار حالت می‌باشد.

img, ts, x, y, z, qx, qy, qz, qw

- با استفاده از تابع get_drone_state می‌توانید حالت هواگرد خود را دریافت کنید.
- در پروازهای خود به‌ویژه جستجو در فضای دارای مانع می‌توانید برخورد پرنده به موانع را با محتوای کلید ’has_collided‘ درون دیکشنری خروجی تابع get_drone_state باخبر شوید. تعداد collisionهای شما در سرور ثبت می‌شود و تعداد collisionهای اجرای نهایی شما نمره منفی خواهد داشت. البته collisionهای اجراهای آزمایشی شما مهم نیست و می‌توانید در اجراهای آزمایشی پرواز خود را به‌گونه‌ای که حداقل برخورد صورت گیرد انجام بدهید. اطلاعات دوربین عمق می‌تواند در این زمینه به شما کمک کند.
- زمان شبیه‌ساز تحت عنوان ts است که زمان را برحسب نانوثانیه بازنمایی می‌کند. مقدار زمان‌ها را با کم کردن زمان‌ها از اولین ts نرمالیزه از صفر کنید و در نمودارها بر حسب ثانیه یا میلی‌ثانیه ترسیم نمایید.

یک کد پروازی نمونه و توضیح عملکرد آن:

```
# In the name of Allah
import time
from tools.sim_tools import SimConnector
from tools.viewer import LiveViewer
from tools.credentials import robot_id, password

sim = SimConnector(robot_id, password, is_local=False)
viewer = LiveViewer(sim, 5, show_osd=True)
viewer.start_view(with_depth=False)
epsilon = 0.1  # seconds
sim.reset()
# Go Upward - 5 m/s for four seconds
duration = 6  # seconds
sim.move_by_body_vels(0, 0, -5, duration)
time.sleep(duration + epsilon)
while viewer.is_live:
    duration = 15 # seconds
    print("Move forward 5 m/s for 15 secs")
    sim.move_by_body_vels(5, 0, 0, duration)
    time.sleep(duration + epsilon)
    if not viewer.is_live:
        break
    duration = 3
    print("Rotate +90 degrees (3sx30d/s)")
    sim.rotate_yaw_rate(30, duration)
    time.sleep(duration + epsilon)
    viewer.save_images()
```

۱. ابتدا کلاس SimConnector با نام کاربری و شناسه هواگرد ایجاد می‌شود.

۲. کلاس LiveViewer نیز نمونه‌برداری می‌شود.

۳. توسط sim.reset پرنده به خانه می‌رود.

۴. توسط sim.move_by_body_vels(0, 0, -5, duration) پرنده ۶ ثانیه با سرعت ۵ متر بر ثانیه بالا می‌رود.

۵. در یک حلقه:

a. توسط sim.move_by_body_vels(5, 0, 0, duration) پرنده به اندازه ۱۵ ثانیه با سرعت ۵ متر برثانیه در دستگاه مختصات بدنه در جهت محور x پیش می‌رود.

b. توسط time.sleep(duration + epsilon) به اندازه مدت زمان دستور قبلی، دستور جدیدی صادر نمی‌شود.

c. توسط sim.rotate_yaw_rate(30, duration) پرنده ۹۰ درجه ساعت‌گرد می‌چرخد.

d. توسط time.sleep(duration + epsilon) به اندازه مدت زمان دستور قبلی، دستور جدیدی صادر نمی‌شود.

توجه داشته باشید کد فوق نمونه‌ای کاربردی از توابع API مربوط به شبیه‌ساز و تمرین است و شما می‌بایست برای پرواز در محیط از توابعی که بهتر نیاز شما را برآورده می‌کند و در مسیر پروازی که مناسب شرایط مساله است استفاده کنید. به‌عنوان مثال چرخش ۹۰ درجه‌ای، حرکت ۷۵ متری رو به جلو یا حرکت ۳۰ متری رو به بالا از نوع کنترل forward feed بوده و می‌تواند اندازه ایده‌آل شما اعمال نشود. این موضوع به سادگی با ملاحظه state vector قابل بررسی است. شاید بهتر باشد از توابع rotate_to_yaw و/یا move_to_position استفاده کنید.