گزارش تیم کنتروتیک

در تمام سناریوهایی که ما در این پروژه داریم، قلب ماجرا یکسان هست. قلب ماجرا همان بخشی است که کنترل گروهی پهپادها رو به دست گرفته و سعی می‌کند تا آرایش یا همان فرمیشن را حفظ کند. فرق اصلی سناریوها چگونگی کنار آمدن با بحث لیدر و فالوور است و اینکه هر کدام چه قابلیت‌های اضافه‌ای روی بخش اصلی سوار می‌کنند.

نکته: روند نصب و اجرای نرم افزارها و پکیج های مورد نیاز در repository ها (مخزن_سناریوی_سه - مخزن_سناریوی_چهار) توضیح داده شده و در اینجا بیشتر سعی شده تا به الگوریتم ها پرداخته شود.

پایه‌ی این پروژه، یک الگوریتم کنترل آرایش توزیع‌شده است که به سبک لیدر–فالوور کار می‌کند و توپولوژی وزن‌دار دارد.
این الگوریتم از مقاله‌ی Distributed leader-follower formation control for multiple quadrotors with weighted topology نوشته‌ی Zhicheng Hou و Isabelle Fantoni الهام گرفته شده که توسط artastier پیاده سازی شده و به صورت متن باز منتشر شده. برای دیدن کد ها به لینک مراجعه کنید.

اجزای مهم هسته اصلی (core)

پیدا کردن همسایه‌ها – NearestNeighbors

یکی از کلاس های پایه هست و وظیفه آن این است که متوجه شود هر پهپاد چه همسایه‌هایی دارد.
به طور کلی:

- هر پهپاد یک Subscriber دارد که موقعیت محلی خودش رو از تاپیک fmu/out/vehicle_local_position/ می‌گیرد.
- یک Publisher هم دارد که اطلاعات همسایه‌ها رو روی fmu/out/nearest_neighbors/ منتشر می‌کند.
- فاصله‌ی هر پهپاد از بقیه محاسبه می‌شود و هرکس که درون محدوده‌ی neighbor_distance (تعریف شده در control_config.json) باشد، همسایه محسوب میشود.
- قبل از این محاسبه، موقعیت‌ها با استفاده از آفست اولیه (x_init, y_init) به مختصات کلی تبدیل می‌شوند.
- این کلاس چندین تابع مجازی مثل process_neighborhood، process_neighbor_position و enrich_neighborhood دارد که کلاس‌های پیشرفته‌تر (مثل WeightedTopologyNeighbors) آن‌ها رو پیاده‌سازی می‌کنند.

همسایه‌های وزن‌دار – WeightedTopologyNeighbors

این کلاس از کلاس NearestNeighbors مشتق شده با این حال توپولوژی وزن‌دار را پیاده سازی می‌کند.

- پارامترهای آرایش: موقعیت دلخواه هر پهپاد نسبت به لیدر (یا مرکز) از فایل‌های پیکربندی (x_formation, y_formation, z_formation) خوانده می‌شود.
- مقایسه موقعیت واقعی با آرایش دلخواه: برای هر همسایه این اختلاف محاسبه می‌شود و بعد در کنترل استفاده می‌شود.
- ضریب اولویت (PRC):
  - برای لیدر همیشه ۱ (بیشترین اولویت هست).
  - برای فالوور، کمترین PRC همسایه‌ها + ۱. این باعث می‌شه یک ساختار سلسله‌مراتبی طبیعی بین پهپادها شکل بگیره.
- وزن‌دهی بدین صورت کار میکنه که هرچقدر PRC کمتر ← وزن بیشتر ← تأثیر تو فرمان کنترل بیشتر.

کنترل توپولوژی وزن‌دار – WeightedTopologyController

وظیفه این بخش تولید فرمان‌های حرکتی بر اساس اطلاعات همسایه‌هاست.

- گین‌های PID از control_config.json برای سه محور X, Y, Z (مجموعاً ۹ گین) خوانده می‌شوند.
- سپس داده‌های موقعیت و سرعت نسبی همسایه‌ها، ضرب در وزنشون، به شکل یک ماتریس درمیایند.
- در نهایت جمع وزن‌دار ستون‌های ماتریس محاسبه می‌شود و از سه کنترل‌کننده‌ی PID جداگانه برای تولید شتاب استفاده می‌شود.
- اگر همسایه‌ای پیدا نشود، پهپاد از حالت کنترل شتاب خارج خواهد شد و به یک نقطه‌ی پیش‌فرض (۰، ۰، -۵) برمی‌گردد. موقعیت و سرعت نیز در پیام خروجی روی NaN می‌رود که PX4 متوجه شود فقط باید به شتاب گوش بدهد.

مسلح کردن و آماده‌سازی پرواز – Arming

این گره پهپادها را مسلح می‌کند و در حالت Offboard می‌برد.

- فرمان‌ها از طریق سرویس vehicle_command ارسال می‌شوند.
- یک تایمر وضعیت همه پهپادها را چک می‌کند و وقتی همه آماده شدند، خودش خاموش میشود.

تفاوت سناریوها

سناریوی سوم – تغییر آرایش در زمان پرواز

تمرکز این سناریو روی تغییر شکل آرایش حین پرواز و دنبال کردن مسیر از پیش تعیین‌شده توسط لیدر هست.

- فایل ChangeWaypoint مسیر رو از فایل YAML خوانده و لیدر را میان waypointها حرکت میدهد. وقتی به اندازه‌ی کافی به یک نقطه رسید (به اندازه آستانه ای که در فایل YAML تعریف شده است)، به سراغ بعدی میرود.
- فالوورها با الگوریتم وزن‌دار جای خودشان را نسبت به لیدر حفظ میکنند.
- تغییر فرمیشن در زمان اجرا با سرویس simulation/change_formation/ اتفاق می افتد. (line, triangle, row).

سناریوی چهارم – تغییر دینامیک لیدر

این سناریو بر روی انتقال خودکار لیدر وقتی که لیدر از دسترس خارج میشود تمرکز دارد که به دو نحو برای دو سناریوی مختلف توسعه داده شده.
۱- زمانی که لیدر به صورت خود به خود از دسترس خارج بشه (شارژ لیدر تموم بشه یا لای شاخه درختا گیر کنه یا مورد هدف قرار بگیره)
۲- به صورت دستی و دلخواه

- فایل SwarmAgent برای هر پهپاد اجرا می‌شود و می‌تواند دو حالت لیدر یا فالوور باشد.
- لیدر مسیر رو دنبال می‌کند و مرکز آرایش رو منتشر می‌کند، فالوورها نیز با اضافه کردن آفست خودشان نقطه هدفشان رو پیدا می‌کنند.
- برای سناریوی اول فایل LeaderMonitor همیشه چک می‌کند که آیا لیدر فعال هست یا نه. اگر نباشد، نزدیک‌ترین پهپاد رو انتخاب و معرفی می‌کنه.
- برای سناریوی دوم سرویس disarm_leader هست برای زمانی که لازم است تا لیدر به صورت دستی عوض شود. بعد از عوض شدن لیدر، لیدر قبلی به فالوور لیدر جدید تبدیل میشود.