Multi Agent Drone Simulation Project

The concept of Companion Computer

Companion Computers can be used to interface and communicate with ArduPilot on a flight controller using the MAVLink protocol. By doing this your companion computer gets all the MAVLink data produced by the autopilot (including GPS data) and can use it to make intelligent decisions during flight.

There are two major parts to Companion Computers - **hardware** and **software**.

Hardware is typically a small ARM-based Single Board Computer (Arduino, RaspberryPi, etc).

Software refers to the programs and tools that run on the Companion Computer. They will take in MAVLink telemetry from the Flight Controller and can route and process the telemetry data (ROS, DroneKit, etc).

Simulation Setup: ROS with Ardupilot

What is ROS (Robot Operating System)?

The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

ROS provides libraries, tools, hardware abstraction, device drivers, visualizers, message-passing, package management, and more to help software developers create robot applications. In the future we expect ROS will be replaced by ROS2

What is MAVROS?

MAVROS is a ROS “node” that can convert between ROS topics and MAVLink messages allowing ArduPilot vehicles to communicate with ROS. The MAVROS code can be found here.

A node is a process that performs computation. Nodes are combined together into a graph and communicate with one another using streaming topics, RPC services, and the Parameter Server. These nodes are meant to operate at a fine-grained scale; a robot control system will usually comprise many nodes. For example, one node controls a laser range-finder, one Node controls the robot's wheel motors, one node performs localization, one node performs path planning, one node provides a graphical view of the system, and so on.

What is MAVLink?

MAVLink is a very lightweight messaging protocol for communicating with drones (and between onboard drone components).

MAVLink follows a modern hybrid publish-subscribe and point-to-point design pattern: Data streams are sent / published as topics while configuration sub-protocols such as the mission protocol or parameter protocol are point-to-point with retransmission. Messages are defined within XML files.

Installation of ROS Noetic for Ubuntu 20.04

ROS Noetic is the recommended ROS distribution for Ubuntu 20.04. The disadvantage of this is that ROS Kinetic is the best documented. The instalation instruction are found here: http://wiki.ros.org/noetic/Installation/Ubuntu (http://wiki.ros.org/noetic/Installation/Ubuntu)

Picked: Desktop-Full Install - Everything in Desktop plus 2D/3D simulators and 2D/3D perception packages

You can always install a specific package found @ https://index.ros.org/packages/page/1/time/#noetic (https://index.ros.org/packages/page/1/time/#noetic) directly using:

```
sudo apt install ros-noetic-PACKAGE
```

Finished ROS Installation - 1/26/2021

Next: Install MAVROS: https://github.com/mavlink/mavros/tree/master/mavros#installation (https://github.com/mavlink/mavros/tree/master/mavros#installation

Started 1/31/2021

Command to Instal MAVROS for ROS Noetic:

```
sudo apt-get install ros-noetic-mavros ros-noetic-mavros-extras
```

Then, install GeographicLib datasets:

```
wget https://raw.githubusercontent.com/mavlink/mavros/master/mavros/scripts/
install_geographiclib_datasets.sh
sudo chmod +x ./install_geographiclib_datasets.sh
```

As recommended by Ardupilot (https://ardupilot.org/dev/docs/ros-install.html#installing-mavros (https://ardupilot.org/dev/docs/ros-install.html#installing-mavros))

For ease of use on a desktop computer, please also install RQT:

```
sudo apt-get install ros-noetic-rqt ros-noetic-rqt-common-plugins ros-noetic
-rqt-robot-plugins
```

We recommend using caktin_tools instead of the default catkin_make as it is more powerful:

```
sudo apt-get install python3-catkin-tools
```

Next: Instal Gazebo

What is Gazebo? (https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl (https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl))

Gazebo is a well-known and respected robotics simulator, and is also the official DARPA Virtual Robotics Simulator.

"A well-designed simulator makes it possible to rapidly test algorithms, design robots, perform regression testing, and train AI system using realistic scenarios. Gazebo offers the ability to accurately and efficiently simulate populations of robots in complex indoor and outdoor environments. At your fingertips is a robust physics engine, high-quality graphics, and convenient programmatic and graphical interfaces. Best of all, Gazebo is free with a vibrant community."

What is SITL Simulation? (https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html (https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html))

SITL allows you to run ArduPilot on your PC directly, without any special hardware. It takes advantage of the fact that ArduPilot is a portable autopilot that can run on a very wide variety of platforms. Your PC is just another platform that ArduPilot can be built and run on.

When running in SITL the sensor data comes from a flight dynamics model in a flight simulator. ArduPilot has a wide range of vehicle simulators built in, and can interface to several external simulators. This allows ArduPilot to be tested on a very wide variety of vehicle types. For example, SITL can simulate:

multi-rotor aircraft fixed wing aircraft ground vehicles underwater vehicles camera gimbals antenna trackers a wide variety of optional sensors, such as Lidars and optical flow sensors Adding new simulated vehicle types or sensor types is straightforward.

A big advantage of ArduPilot on SITL is it gives you access to the full range of development tools available to desktop C++ development, such as interactive debuggers, static analyzers and dynamic analysis tools. This makes developing and testing new features in ArduPilot much simpler.

Documentation followed to get Gazebo running: https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl (https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl)

finished 1/31/2021

Clone Ardupilot Firmware

First, fork the ardupilot repo, then clone. https://ardupilot.org/dev/docs/building-setup-linux.html (https://ardupilot.org/dev/docs/building-setup-linux.html)

Then install the required packages using:

```
Tools/environment_install/install-prereqs-ubuntu.sh -y
```

Reload the path (log-out and log-in to make permanent):

```
. ~/.profile
```

All installations finished, ROS, ardupilot, MAVROS, Gazebo. Now try to run SITL simulation.

Using Gazebo Simulator with SITL

https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl (https://ardupilot.org/dev/docs/using-gazebo-simulator-with-sitl.html#using-gazebo-simulator-with-sitl)

Start the Simulator ¶

In a terminal window start Gazebo:

```
gazebo --verbose ~/ardupilot_gazebo/worlds/iris_arducopter_demo.world
```

In another terminal window, enter the ArduCopter directory and start the SITL simulation:

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -f gazebo-iris -D --console --map
```

problem, I cannot see the drone when the simulator starts 2/5/2021

^Solved by running different commands listed above 2/9/2021 from:
https://github.com/ArduPilot/ardupilot/issues/9531 (https://github.com/ArduPilot/ardupilot/issues/9531)

Another very useful resource: https://github.com/Texas-Aerial-Robotics/Controls-Other (https://github.com/Texas-Aerial-Robotics/Controls-Other)

Connecting Ardupilot with ROS: stopped 2/9/2021

https://github.com/AS4SR/general_info/wiki/ArduPilot:-Instructions-to-set-up-and-run-an-autopilot-using-SITL-and-Gazebo-simulator#Connecting_with_ArduPilot_with_ROS (https://github.com/AS4SR/general_info/wiki/ArduPilot:-Instructions-to-set-up-and-run-an-autopilot-using-SITL-and-Gazebo-simulator#Connecting_with_ArduPilot_with_ROS)

Launch an SITL instance:

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -f gazebo-iris -D --console --ma
p
```

The next step is to create a new directory for a launch file. In a new terminal, we run these:

```
cd ardupilot
mkdir launch
cd launch
```

We then copy mavros default launch file for ardupilot and then open to edit it:

```
roscp mavros apm.launch apm.launch
gedit apm.launch
```

To connect to SITL we modify the first line to

```
<arg name="fcu_url" default="udp://127.0.0.1:14551@14555" />
```

We save the file and launch it with:

```
roslaunch apm.launch
```

The connection is now complete.

Troubleshooting: Since GeographicLib requires certain datasets (mainly the geoid dataset) so to fulfill certain calculations, these may be needed to be installed manually by the user. If you run into GeographicLib error, run these:

```
wget https://raw.githubusercontent.com/mavlink/mavros/master/mavros/scrip
ts/install_geographiclib_datasets.sh
chmod +x install_geographiclib_datasets.sh

sudo ./install_geographiclib_datasets.sh
```

Additionally, we can use RQT to see all the topics that mavros has to create from ardupilot information. In a new terminal, type:

```
rqt
```

We can now see all the topics that mavros has to create from ardupilot information.

2/11/2021

Doing the above gave me the following error when doing apm.launch:

[ERROR] [1613085883.312215878]: udp0: sendto: Invalid argument terminate called after throwing an instance of 'std::system_error' what(): Resource deadlock avoided ==========REQUIR process [mavros-2] has died! process has died [pid 92224, exit code -6, cmd /opt/ros/noetic/lib/mavros/mavros_node **name:=mavros** log:=/home/assistlab/.ros/log/51c2a4d2-6cc0-11eb-844d-f1790fe7c3f6/mavros-2.log]. log file: /home/assistlab/.ros/log/51c2a4d2-6cc0-11eb-844d-f1790fe7c3f6/mavros-2*.log Initiating shutdown!

Found a fix at: https://discuss.ardupilot.org/t/problem-calling-ros-service-for-arducopter-ros-sitl-tutorial/18992/13 (https://discuss.ardupilot.org/t/problem-calling-ros-service-for-arducopter-ros-sitl-tutorial/18992/13)

Succesfull. Connection with MAVROS established 11/2/2021.

Next, Setting up Adithya's Sim:

Github repo: https://github.com/adithya-tp/uppaal2ros (https://github.com/adithya-tp/uppaal2ros)

Evironment setup Guide: https://www.youtube.com/playlist?list=PLy9nLDKxDN683GqAiJ4IVLquYBod_2oA6 (https://www.youtube.com/playlist?list=PLy9nLDKxDN683GqAiJ4IVLquYBod_2oA6)

Swarming Using Ardupilot Turotial: https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md#launching-ardupilot-sitl-instances-with-unique-parameters (https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md#launching-ardupilot-sitl-instances-with-unique-parameters)

2/20/2021 Some tutorials with ROS and Gazebo to get familiar with file placements. Following youtube playlist link above.

Problem?? Gazebo topics not appearing in rostopic list

3/10/2021

Configured catkin workspace as my ardupilot_ws folder following the instructions in the link:

https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/installing_ros.md (https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/installing_ros.md)

Configuration appears to be correct because the following meassage appeared:

```
Workspace configuration appears valid.
```

Error when trying to run the sample autonomous scripts:

```
assistlab@assistlab:~/ardupilot_ws/iq_gnc/src$ rosrun iq_gnc square
                    [rospack] Error: package 'iq_gnc' not found
```

3/10/2021 finish

3/11/2021 Start, try to get scripts for autonomous flight running

Simulation for tutorial square.cpp running fine:

Steps:

```
roslaunch iq_sim runway.launch
```

New Terminal

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -f gazebo-iris -D --console --map
```

New Terminal

```
roslaunch iq_sim apm.launch
```

New Terminal ardupilot_ws

```
rosrun iq_gnc square
```

Then in mavlink terminal type commmand:

```
mode GUIDED
```

Now try with Adithya's file

Copied all of his repo files into my repo. Added neccessary files and configs.

Tried to replicate this video: https://drive.google.com/drive/folders/1x-4HOdpqfACrNvDQHiLv2c-BnYev2SYM (https://drive.google.com/drive/folders/1x-4HOdpqfACrNvDQHiLv2c-BnYev2SYM)

Steps to launch multi drone sim:

Launch the world with the three drones spawned:

```
roslaunch iq_sim multi_drone.launch
```

Launch the three SITL instances, one for each drone

New Terminal

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -v ArduCopter -f gazebo-drone1 --co
nsole -I0
```

New Terminal

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -v ArduCopter -f gazebo-drone2 --co
nsole -I1
```

New Terminal

```
cd ~/ardupilot/ArduCopter sim_vehicle.py -v ArduCopter -f gazebo-drone3 --co
nsole -I2
```

ROS Connection

New Terminal

```
cd ~/ardupilot_ws roslaunch iq_sim multi-apm.launch
```

Run the script:

New Terminal

```
cd ~/ardupilot_ws roslaunch iq_gnc multi_square_sol.launch
```

Possible Solution:

Follow steps in section "Connecting Multiple Drones to a Ground Station"

from https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md (https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md)

^ try next time.

3/11/2021 finished.

3/30/2021 start: try to get four drones moving at the same time with square.cpp file

Added steps for launching sim above.

I can manually make each of the three drones takeoff which means that the SITL instances are correct. Now I will check the solution file.

Problem was I had to add gazebo-drone models to ardupilot default folders. Process described here: https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md (https://github.com/Intelligent-Quads/iq_tutorials/blob/master/docs/swarming_ardupilot.md)

Success! Square script works with three drones moving independently. Only problem is that I have to manually set each drone to GUIDED mode.

Fixed guided mode issue with the following edit to the square.cpp file:

Replace wait4start(); with set_mode("GUIDED");

Reminder, removed adithya's stuff from gnc folder so I could run tutorial, also commented out cmakelist lines

finished 3/30/2021 simulation starts correctly but drones crash because of sensor failures...

start 4/14/2021 fix drones crashing

First try, simulation ran without drone crashes.

Now try setting up Adithya's stuff

All files built. Succesfully replicated. Run the same commands shown above but when running the SITL instances do:

```
sim_vehicle.py -v ArduCopter -f gazebo-drone1 -I0 --out=tcpin:0.0.0.0:8000
sim_vehicle.py -v ArduCopter -f gazebo-drone2 -I1 --out=tcpin:0.0.0.0:8100
sim_vehicle.py -v ArduCopter -f gazebo-drone3 -I2 --out=tcpin:0.0.0.0:8200
sim_vehicle.py -v ArduCopter -f gazebo-drone4 -I3 --out=tcpin:0.0.0.0:8300
```