USER MANUAL
for
FOLLOW ME DRONES

COS301, University of Pretoria, South Africa

Five Guys, One Branch

July 19, 2019

Contents

System Overview

This document outlines how each subsystem can be operated independently, this is not a recommended way to use the system as it is designed to work in its entirety.

For the system to work entirely, follow the methods for setting up each subsystem and then follow the jetson and mobile application user manuals.

Getting Started

2.1 Requirements

2.1.1 Software Requirements

Communcation Server

- npm
- nodeJS

Jetson Nano

- Minimum 64GB SD card
- Wi-fi adapter
- DC Power adapter (*See jetson spec sheet*)
- Bridge to complete DC power circuit

Object Detection [1]

- cmake
- gcc
- knowledge on how to use a terminal

Drone Controls

- Git
- Python 3 (or Better)
- Pip
- drone-silt (for simulation) [2]
- mavproxy (to transfer commands to drone) [3]

Mobile Application

- Android v4.4+ mobile phone.
- GPS capable device

2.1.2 User Requirements

Object Detection

- Knowledge on how to use a terminal

Jeston Nano

- Knowledge on linux and how to set up a distribution
- Networking knowledge
- SSH

Mobile Application

- Knowledge on how to use a mobile phone

System Installation

3.1 Object Recognition

- **Clone the repository (If not already done)**

  $ *git clone* `https://github.com/cos301-2019-se/Follow-Me-Drones.git`

- **Install OpenCV**

  $ *sudo pacman -S opencv*

- **Install Cuda and Cudnn**

  $ *sudo pacman -S cuda cudnn*

3.2 Jetson Nano

- **Follow the Nvidia guide to setting up your jetson**
  - *Nvidia Guide*

3.3 Communcation Server

- **Clone the repository (If not already done)**

  $ *git clone* `https://github.com/cos301-2019-se/Follow-Me-Drones.git`

- **Navigate into the server directory**

  $ *cd Follow-Me-Drones/server/*

- **Install the depenecies**

  $ *npm install*

3.4 Drone Controls

- **Pip installations**

  $ *pip install dronekit-silt*

  $ *pip install mavproxy*

  After installing you can just run the interface.

3.5 Mobile Application

- **APK installation**
  - Download APK (Google Play Store)
  - Install via Android package manager

System Configuration

4.1 Object Recognition

- **Configure Makefile**

  In the darknet_ directory, locate the Makefile and change the following lines need to be changed:

  # 89 - */path/to/cuda/include*

  # 92 - */path/to/cuda/lib*

  # 94 - */path/to/cuda/lib64*

  # 101 - */path/to/usr/local/cuda/include*

  # 102 - */path/to/usr/local/cuda/lib*

  # 104 - */path/to/usr/local/cudnn/include*

  # 105 - */path/to/usr/local/cudnn/lib64*

  These lines must reflect the installation folder for cuda and cudnn specific to your device and it's installation folders.

- **Configure config file for training**

  In the darknet_/cfg directory, get 300IQ, locate the animals.cfg and change the following lines:

  # 15 - *max_batches = (classes * 2000)*

  # 17 - *steps = (classes*0.8), (classes*0.9)*

  # 224 - *filters = (classes + 5)*5*

  # 230 - *classes = no. classes*

  * Note - Don't actually type in a formula like (classes * 2000). Only type in the value once you've calculated it

4.2 Communication Server & Jetson Nano

- **Configure commserver.service file**

  In the home directory on your jetson nano create a service file:

  $ *touch commserver.service*

  Paste the following into the file:

```
[Unit] Description=LTPS NodeJS Test Application
After=network-online.target

[Service]
Restart=on-failure
WorkingDirectory=/home/jetson/Follow-Me-Drones/server
ExecStart=/usr/bin/node /home/jetson/Follow-Me-Drones/server/comms.js

[Install]
WantedBy=multi-user.target
```

In the home directory on your jetson nano create a service file:

$ *touch commserver.service*

System Usage

5.1 Communication Server

Once the configuration has been completed, navigate to the Follow-Me-Drones/server/ directory and run the following command to run the server:

$ npm start

5.2 Object Detection

Once the configuration has been completed, navigate to the Follow-Me-Drones/object-recognition/interface/ directory and run the following commands to open the interface:

$ *g++ interface.cpp*

$ *./a.out*

5.3 Drone Controls

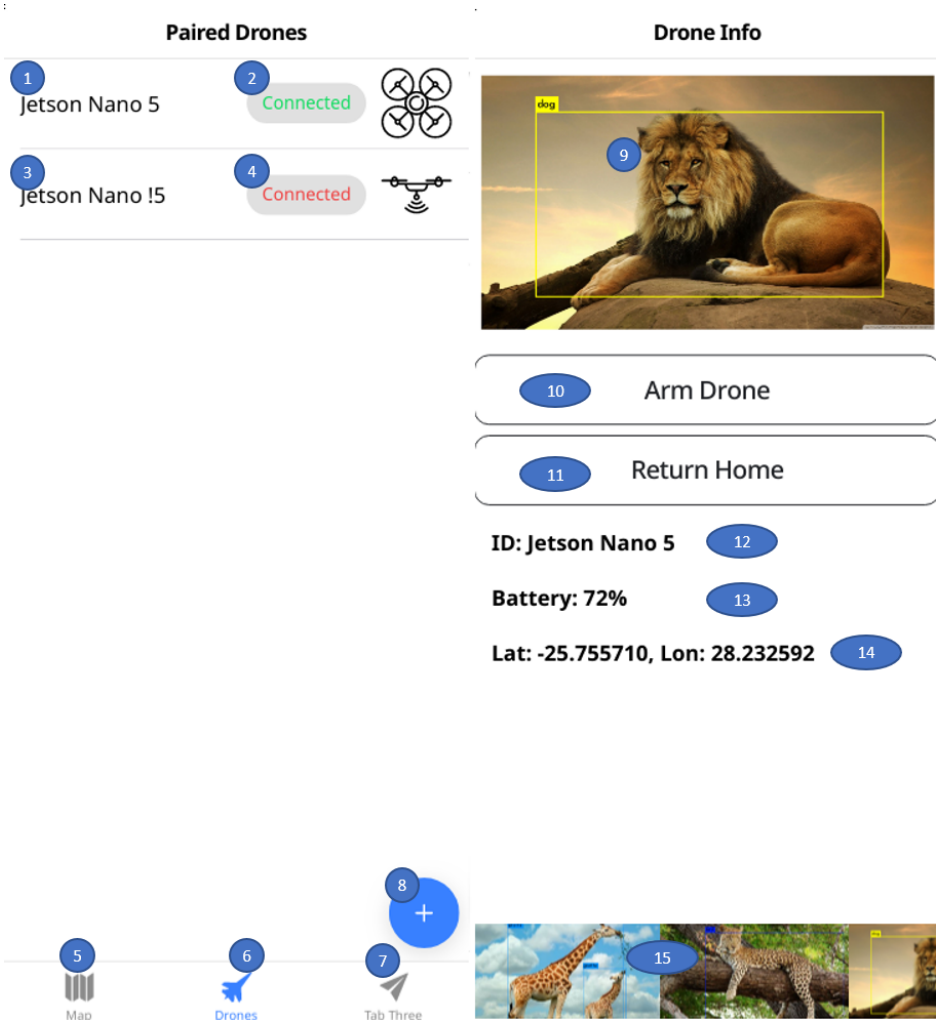
If not running a simulation step 1 can be ommited

- Optional Start Dronekit with the command
  $ dronekit-sitl copter –home=-lat-coords,long-coords,584,353
- Start MavProxy with the command
  $ mavproxy.py –master tcp:127.0.0.1:5760 –out udp:127.0.0.1:14551 –out udp:10.1.2.100:14550
  –master specifies the IP that the drone commands are supposed to be forward
  –out specifies the ports that can be accessed
- Drone Commands after mavproxy is running
  - This is the command to connect to the dronekit.
    `CONNECT_UDP,UDP_PORT`
  - Disconnects the interface from the drone.
    `DISCONNECT_UDP`
  - Sets the destination for the drone to fly to.
    `GO_TO_POINT,lat-coords, long-coords`
  - Cancels the current flight and sets the drone to return to base.
    `CANCEL_FLIGHT`

5.4 User Application

See figure below:

- 1. Device name.
- 2. Connection status when drone is connected



- 3. Connection status when drone is disconnected
- 4. Tab to view map.
- 5. Tab to view paired drones
- 6. Tab to view current active drone sessions
- 7. Pair a new drone.
- 8. Currently selected image in carousel below
- 9. Arm the drone remotely. (Start object detecttion and lift off)
- 10. Return home remotely. (End of session)
- 11. Current session. (End of session)
- 12. Battery percentage.
- 13. Current latitude and longitude coordinates of the drone.
- 14. Carousel of previously detected animals.

Troubleshooting

- Connectivity Issues:
  - Make sure the mobile device is wifi enabled.
  - Make sure the application and drone is on the same network.
  - Make sure the mobile device's antivirus is non blocking.
  - Make sure the server is set up correctly and running.
- Performance Issues:
  - Make sure the Nvidia Jetson is connected with an adapter that can provide sufficient power.
  - Running camera at lower resolution.
  - Decrease the input video's height and width properties in the cfg/animals.cfg file.
  - Ensure that heatsink is properly attached, there is sufficient airflow and the Nvidia Jetson is not in direct sunlight.
- Server Issues:
  - Ensure that the correct IP address and port are specified.

Bibliography

[1] "Darknet." https://github.com/AlexeyAB/darknet. Accessed: 2019-04-30.

[2] "Drone-Kit SILT." https://github.com/dronekit/dronekit-sitl. Accessed: 2019-04-30.

[3] "Mavproxy." https://github.com/ArduPilot/MAVProxy. Accessed: 2019-04-30.