Group 44 - ARC

Tech Review

Senior Capstone Project

Oregon State University

Fall 2016

Tao Chen, Cierra Shawe, Daniel Stoyer

Abstract

CONTENTS

1 Vision System Options - Cierra

For autonomous operation, vision systems are critical. The three main options include stereoscopic cameras, Infrared (IR) based systems such as Microsoft's Kinect, and Light Detection And Ranging (LiDAR) vision systems. With the exception of some forms of LiDar, all of these methods require what is called a disparity map, which creates a ?3D image? of the surface, that can be used for telling which objects in an image are closest or farthest away.

1.1 Stereo Vision

Stereo-vision uses two different cameras to create disparity maps in order to create a sense of depth. This is similar to how our eyes work. The biggest benefit to a stereoscopic camera system is the ability to detect objects outdoors, as the cameras are able to function with vast amounts of ultraviolet (UV) light. IR LEDs can also be used in order to illuminate an area at night, also allowing for nighttime navigation. One of the challenges of stereo vision is the computational power required. Another is clarity of the disparity map without post processing of images, which makes real-time operation more difficult. [**?**] OpenCV [**?**] contains many examples of how to configure and process stereoscopic images, and is one of the largest vision resources.

1.2 IR Camera's such as Kinect and RealSense

Using an infrared point map, these cameras are able to tell the disparity between the points, which helps in creating disparity maps. The most popular example of an IR camera system, is the Microsoft Kinect. A big advantage to IR camera?s, is the ability to function in low and non-natural lighting conditions, due to using the infrared spectrum, rather than only using the visible spectrum. The biggest problem with IR cameras, the functionality is greatly reduced outdoors, due to massive amount of infrared waves from the sun. IR cameras don?t meet our requirement of being able to use the vision system reliably outdoors.

1.3 Lidar

LiDAR works by using LAZERSSSS

1.4 Our choice

Due to the need to be able to navigate in outdoor environments, our team will start out attempting to use stereoscopic imaging as our primary vision system. We will do this using the OpenCV library to analyze the images, and create the disparity map that can be used for other purposes. If we have the computational power to post-process disparity maps in real time, we will attempt to do so.

2 SENSORS - CIERRA

2.1 GPS units

2.2 Encoders

2.3 Depth Sensors

3 SYSTEM CONTROL AND SYNCHRONIZATION - CIERRA

3.1 PXFmini vs Other Options

3.2 Synchronizing Sensors

4 Image Analysis Software - Dan

Image analysis, for the ARC project, is the processing of visual data received from cameras into deterministic information, such as pathfinding, or spacial awareness. This is the primary means for our autonomous vehicle to assess its surroundings and find its way to a given waypoint while avoiding obstacles. We require software that is freely available for use (via fairly liberal open source licensing), known to be correct (works well) with little modification needed, and has relatively easy to use API libraries.

4.1 DroneKit-Python

DroneKit-Python (http://python.dronekit.io/) is part of the DroneKit ecosystem (dronekit.io). DroneKit is compatible with all vehicles using the Micro Air Vehicle Communication Protocol (MAVLink). DroneKit's image analysis is integrated into DroneKit-Python.

more to follow...

4.2 ArduPilot

ArduPilot (http://ardupilot.org/rover/index.html)

more to follow...

4.3 Image analysis choice

ArduPilot is our choice for image analysis software. The documentation for the image analysis software researched was rather vague across the board when it comes to information on path-finding and image analysis capabilities. So, while we are going with ArduPilot to start with, we will not really know its effective capability at obstacle avoidance and pathfinding until we have tested it.

4.4 References

5 Telemetry Radio Communication - Dan

In this section we will examine three different telemetry radios, comparing and contrasting them and making a choice on which radio we will use for ARC. Telemetry is simply the transmition of measurement data (velocity, angle, rotation, etc.) by radio to some other place. [1] This data allows the user to know the current state of the vehicle. This is especially important for autonomous operation, as the vehicle may not be operating within line of sight. Telemetry transmission is well-established, so we will not be comparing vastly different transmission technologies, such as long range (MHz radio frequencies) versus short-range (bluetooth) where the advantages of ranges of 2-15+ kilometers obviously outweigh ranges of 20-100 meters.

The main criteria for consideration are:

- Cost
  One of our main goals with ARC is to keep the costs low.
- Power consumption
  We have limited power available, therefore we need power consumption to be low.
- Ease of use
  The radio needs to be easily integrated into the autopilot system. This means it needs to have a developed API with little no modification required.
- Form factor
  The size and weight needs to be small and light. If it is too bulky, we might not have space on the vehicle. If it is too heavy, more power will be required to operate the drive system and will drain the battery faster.

5.1 3DR 915 MHz Transceiver

The 3DR 915 MHz telemetry radio has a cost of $39.99 USD for two radios. It is powered by the autopilot telemetry port (+5v) which means it has low power consumption. This radio transceiver uses open source firmware, has a robust API, and is fully compatible with PX4 Pro, DroneKit, and ArduPilot, using the MAVLink protocol. These features will allow us to implement telemetry transmission with little to no modification of the API, should we use one of those autopilot systems. The form factor has dimensions of 25.5 x 53 x 11 mm (including case, but not antenna) at 11.5 grams (without antenna). [2]

The range of this transceiver is from 300 meters to several kilometers, depending on the antenna arrangement.

Pros: inexpensive, small form factor, low power consumption.

Cons: range out of the box could be as low as 300 meters.

5.2 RFD900 Radio Modem

The RFD900 Radio Modem has a cost of $259.99 USD for two radios. [3] It requires seperate +5v power for operation which means that it has high power consumption. This radio has open source firmware, a robust API, and is fully compatible with PX4 Pro, DroneKit, and ArduPilot, using the MAVLink protocol, which will allow us to implement telemetry transmission with little to no modification of the API, should we use one of those autopilot systems. [4]

The form factor has dimensions of 70 x 40 x 23mm (including case, but not antenna) at 14.5 grams (without antenna). The range of this transceiver is 25+ kilometers.

Pros: ultra long range.

Cons: expensive, large size.

5.3 Openpilot OPLink Mini Ground and Air Station 433 MHz

(https://hobbyking.com/en_us/openpilot-oplink-mini-ground-station-433-mhz.html) (http://www.banggood.com/Openpilot-OPLINK-Mini-Radio-Telemetry-AIR-And-Ground-For-MINI-CC3D-Revolution-p-1018904.html)

The OPLink Mini Ground Station has a cost of $26.59 USD for two radios. [5] It requires input voltage of +5v and can be powered off the autopilot telemetry port which means that it has low power consumption. This radio has open source firmware but is only compatible with the OpenPilot RC control system. The form factor has dimensions of 38 x 23 x mm (including case, but not antenna) at 4 grams (without antenna). [6] The range of this radio is not known, but based on the power requirements and frequency it likely has less range than the 3DR 915 MHz radio.

Pros: smallest size and weight (only 4 grams), lowest cost ($26.59 USD)

Cons: Only works with the LibrePilot control system.

5.4 Telemetry radio choice

The 3DR 915 MHz Transceiver is our selection for the telemetry radio. While the OPLink Mini Ground Station was significantly smaller, lighter, and cheaper than the other two, its implementation being tied solely to LibrePilot was a deal breaker (more information on LibrePilot can be found in the User Interface evaluation). The RF900 Radio Modem would have been a good choice, it has fantastic range and all the API options we were looking for. But it had a significantly larger form factor, required a separate power supply, and was quite expensive at $259.99. Put together, these facts eliminated the RF900 as a viable option. The 3DR 915 MHz Transceiver is a good balance of cost, performance, and size. The cost of $39.99 for two radios, the ability to power the autopilot off the telemetry port, and the portability of its APIs and their ease of use, puts the 3DR 915 MHz at the top of our list and the clear choice for the telemetry radio going forward.

5.5 References

[1] Merriam-Webster.com, 'telemetry', 2016. [Online]. Available: http://www.merriam-webster.com/dictionary/telemetry. [Accessed: 15- Nov- 2016].

[2] 3DR, '915 MHz (American) Telemetry Radio Set', 2016. [Online]. Available: https://store.3dr.com/products/915-mhz-telemetry-radio. [Accessed: 15- Nov- 2016].

[3] jDrones.com, 'jD-RF900Plus Longrange', 2016. [Online]. Available: http://store.jdrones.com/jD_RD900Plus_Telemetry_Bundle_p/ [Accessed: 15- Nov- 2016].

[4] ArduPilot Dev Team, 'RFD900 Radio Modem', 2016. [Online]. Available: http://ardupilot.org/copter/docs/common-rfd900.html. [Accessed: 15- Nov- 2016].

[5] Banggood.com, 'Openpilot OPLINK Mini Radio Telemetry', 2016. [Online]. Available: http://www.banggood.com/Openpilot-OPLINK-Mini-Radio-Telemetry-AIR-And-Ground-For-MINI-CC3D-Revolution-p-1018904.html [Accessed: 15- Nov-2016].

[6] HobbyKing.com, 'Openpilot OPLink Mini Ground Station 433 MHz', 2016. [Online]. Available: https://hobbyking.com/en_us/op oplink-mini-ground-station-433-mhz.html [Accessed: 15- Nov-2016].

6 User Interface - Dan

In this section we will examine three user interfaces, comparing and contrasting them and making a decision on which one we will use with ARC. A user interface (UI) is required to allow the user to command the vehicle. The UI must be open source and have easy-to-implement API libraries. We are looking for a UI package that will work with both the control station (the user computer) and the companion computer (the computer on board the vehicle). It is preferrable that the UI be a combination of graphical UI (GUI) and command line UI (CLI). Note that though our project is a land vehicle (rover) the following software is primarily used for UAV flight control and is referenced in such a way. If possible, we would like to use software that can be configured to control a rover, or easily modified to do so.

6.1 QGroundControl

QGroundControl (QGC) is a full flight control and mission planning GUI software package that is compatible with any MAVLink enabled drone. [1] It is open source and is configured for use with ArduPilot and PX4 Pro. QGC runs on Windows, OS X, Linux, and iOS and Android tablets. QGC has video streaming with instrument overlays, allows mission planning including map point selection, rally points, and even a virtual fence to keep the drone from going beyond a specified area. QGC is a mature software package that has excellent libraries and support with very good documentation. [2] Additionally, QGC works with ArduPilot which is known to work with the PXFMini , the autopilot we intend to use and can be configured for rovers. [3] QGC appears be GUI only with no CLI functionality.

Pros: easy to use, great documentation, compatible with MAVLink, tested on the PXFMini. Can be used on all major pc and mobile platforms. Supports rovers.

Cons: does not appear to have CLI support.

6.2 Tower/DroneKit-Android

(http://android.dronekit.io/) (https://play.google.com/store/apps/details?id=org.droidplanner.android&hl=en) Tower is a Android mobile app that works with most drones that use the MAVLink protocol. Tower allows basic map point selection and allows drawing a flight path on the tablet. [4] It is based on the open source DroneKit-Android framework. DroneKit-Android has good documentation providing code snippets with working example code. [5] Because of the modularity of Android development and access to Tower source code, adding feature and interfaces to the existing app should be relatively easy. Tower is not configured for rovers, so we would have to write the functionality into it. DroneKit-Android and Tower are only available on Android.

Pros: is easy to use, has basic map point selection, adding features should be relatively straight-forward.

Cons: is only available on Android, configuring for rovers requires writing code for support.

6.3 LibrePilot

(https://www.librepilot.org/site/index.html)

LibrePilot (LP) is a full flight control and mission planning software package. It is open source and operates via GUI and allows map point selection. LP is compatible with OpenPilot control system exclusively. It does not work with any other hardware but OpenPilot hardware and does not have rover support. It has some helpful documentation, such as Windows build instructions, but the information is very basic. The source code does seem to have decent commenting which could help since we would need to heavily modify the code base for rover support. LP runs on Linux, Mac, Windows, and Android. [6]

Pros: Has a nice GUI for map point selection, mission planning, and vehicle control. If used in the OpenPilot ecosystem, it should communicate well. Runs on most major pc platforms.

Cons: Is locked in to the OpenPilot ecosystem. Does not have rover support out of the box which will require extensive coding. Does not run in iOS.

6.4 User interface choice

QGroundControl is our user interface choice.

LibrePilot has similar features to QGC but being locked in to the OpenPilot ecosystem is a deal breaker. We need to be able to use the PXFMini and LibrePilot cannot do that. LP is also not configured for rovers, which would require extensive coding.

Tower is the most modest of the user interface options. It is only available on Android, does not have rover support and has limited options for navigation and vehicle control. For these reasons we reject Tower as a viable option.

QGroundControl runs on all major pc and mobile platforms, is configured to run rovers, and uses the MAVLink protocol. It has a nice GUI and allows map point selection and advanced mission planning. It is known to work with the PXFMini flight controller, a component we want to use as part of the ARC build. These features give us a platform that is meets our needs and is flexible, should our needs change. Therefore, QGroundControl is the clear user interface choice for the ARC project.

6.5 References

[1] QGroundControl.com, Unknown. [Online]. Available: http://qgroundcontrol.com/. [Accessed: 15- Nov- 2016].

[2] QGroundControl.com, Unknown. [Online]. Available: https://donlakeflyer.gitbooks.io/qgroundcontrol-user-guide/content/. [Accessed: 15- Nov- 2016].

[3] ArduPilot Dev Team, ’PXFmini Wiring Quick Start’, 2016. [Online]. Available: http://ardupilot.org/rover/docs/common-pxfmini-wiring-quick-start.html. [Accessed: 15- Nov- 2016].

[4] Fredia Huya-Kouadio, ’Tower’, 2016. [Online]. Available: https://play.google.com/store/apps/details?id=org.droidplanner.andro [Accessed: 15- Nov- 2016].

[5] 3D Robotics Inc., ’DroneKit’, 2015. [Online]. Available: http://dronekit.io/. [Accessed: 15- Nov- 2016].

[6] LibrePilot, ’Open-Collaborative-Free’, 2016. [Online]. Available: https://www.librepilot.org/site/index.html. [Accessed: 15- Nov- 2016].