PCC's Autonomous Air Vehicle System for IARC 2015

1 June 2015

Frank Manning
Pima Community College

Yanitzin Todd
Embry-Riddle Aeronautical University, Prescott

Tim Worden
The Boeing Company

[1] Abstract

The Pima Community College UAV Club has designed an air vehicle system to compete in the International Aerial Robotics Competition (IARC). The rules require an autonomous air vehicle to herd a group of 10 ground robots while avoiding collisions with a second group of 4 obstacle robots. All 14 ground-based robots are themselves autonomous and move according to their own internal algorithms, including responses to external collisions and magnetic fields. The air vehicle is designed to use machine vision as well as lidar and sonar scanning to sense the positions of ground robots, and to navigate relative to a 20 m x 20 m arena. The arena is marked with a known grid pattern.

[2] Introduction

[2.a] Statement of the Problem

The mission requires an autonomous aerial robot to herd a group of 10 ground robots ("mission robots") within a square 20 m arena. An additional 4 obstacle robots are also present and must be avoided by the air vehicle. All 14 ground-based robots are autonomous and move according to known, internal algorithms.

The arena is indoors on a floor marked by a pattern of 1 m white grid lines internally. The 4 outer boundaries of the arena are marked by a green line on one end and red line on the opposite end. The other 2 boundaries are white lines.

The overall objective is to herd all mission robots to the green boundary. Each mission robot can be steered to a limited extent by applying a magnetic field to certain areas on top of the robot. Steering can also be accomplished by applying a small mechanical force to the front of the robot. One way of doing this is by landing in front of the robot in order to cause an intentional collision.

In parallel with these activities the aerial robot must also avoid colliding with the obstacle robots, each of which has a cylinder extending from the top. The cylinder height is 2 m maximum, but may be shorter. Regarding collisions, note in particular that all ground-based robots will generally be colliding with each other, thus complicating their movements.

The mission terminates at a 10 minute deadline. Other factors may also terminate the mission prior to the time limit, such as a collision with an obstacle robot, or if all mission robots either go out of bounds or reach the green boundary.

[2.b] Conceptual Solution to Solve the Problem

Navigation and mission robot tracking – the air vehicle uses on-board machine vision to detect the grid lines and boundary lines of the arena in order to keep track of the position of the vehicle with respect to the arena. In addition, the vision system determines the positions of the ground robots. The vision system is able to discriminate between mission and obstacle robots because of color. All ground robots are mostly white, except that mission robots have known areas of green or red on top.

Obstacle robot avoidance – a lidar scanner generates a 240 degree horizontal scan pattern in order to sense obstacle robots within a range of 4 m. Four additional sonar sensors are used to cover the 120 degree blind spot of the lidar. Lidar, sonar and camera data are combined in order to avoid obstacle robots.

Disclaimer – this paper describes a conceptual solution that is intended to perform the full IARC mission at a future date. Only a small part of the solution has actually been implemented in hardware or software as of this writing.

[2.b.1] Figure of Overall System Architecture

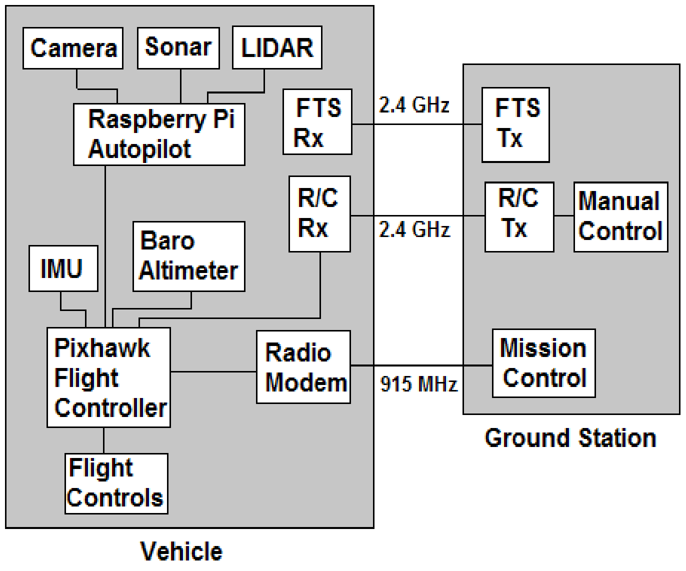

Figure 1. Overall system architecture.

[2.c] Yearly Milestones

Vehicle and machine vision development will be emphasized in 2013/14/15. Enhanced maneuverability and controllability will be done in 2015/16.

[3] AIR VEHICLE

The air vehicle is adapted from a vehicle being developed for long range communications. The vehicle has a balloon radome intended for enclosing a large directional antenna. For the IARC, the radome and directional antenna are removed.

The vehicle consists of a highly modified 3DRobotics Y6, which has a tricopter configuration. Most of the Y6 internal electronics and sensors are retained, as are 2 of the 3 motor pylons, plus 4 of the 6 motors.

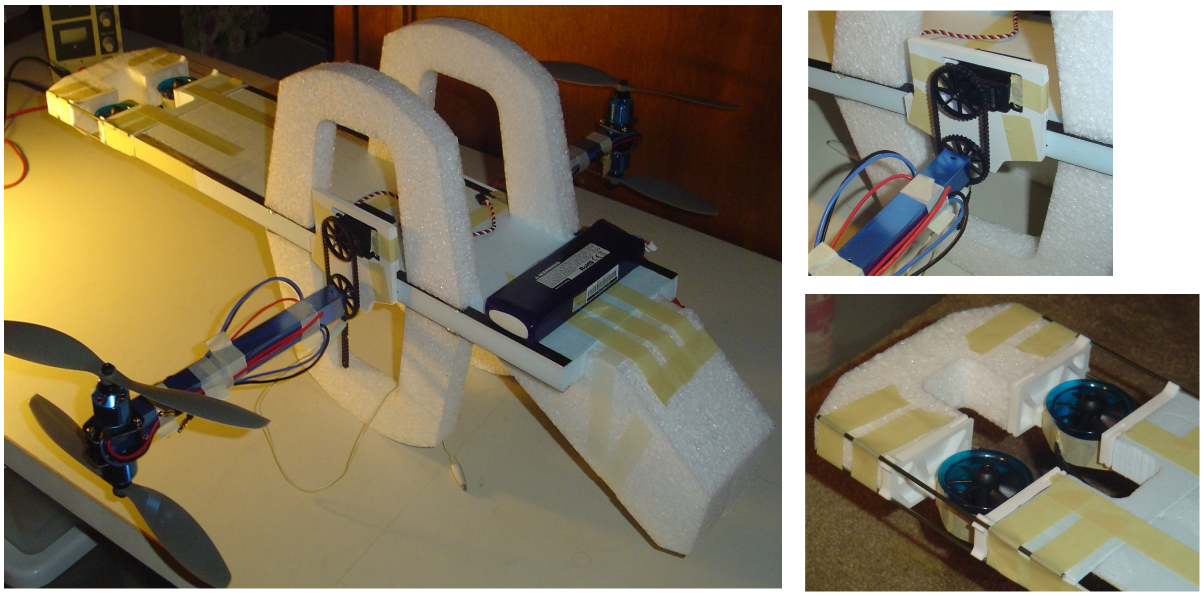

Figure 2. Vehicle, tilt mechanism, thruster.

The Y6 main body and tail pylon are replaced with a 20 mm thick styrofoam plank reinforced with carbon strips. The original landing gear is replaced with structures of expanded polypropylene (EPP). Blocks of EPP are also located in the nose and tail to minimize crash damage.

The vehicle has a total of 6 rotors for propulsion and attitude control. The Y6 is modified such that all 6 rotors have some form of Thrust Vector Control (TVC). All rotors can tilt through a 180º sweep angle.

Of the original 6 main rotors in the Y6, 4 are retained as-is, each with a diameter of 254 mm. The main rotors are grouped in 2 countrarotating pairs. The 4 rotors are also modified to tilt as a single unit about the pitch axis for increased maneuverability.

The remaining 2 rotors in the Y6 tail are replaced with a thruster that consists of 2 small electric ducted fans (EDF) with 40 mm diameter rotors. Each EDF implements TVC with independently-controllable tilt angles of 180º about the roll axis.

[3.a] Propulsion and Lift System

Altitude controlled by total thrust produced by the 4 main rotors.

Horizontal translation in the X-direction is controlled by tilting the 4 main rotors about the pitch axis. This allows the lift vector to be tilted without rotating the fuselage, which by comparison allows rapid changes to longitudinal acceleration.

Horizontal translation in the Y-direction depends on the mode:

Mode 1 -- Low airspeed

The thruster TVC generates a thrust component in the Y-direction for vernier control of lateral acceleration. Again this can be done rapidly and without rotating the fuselage.

Mode 2 -- High airspeed

The roll angle of the entire vehicle can be varied, which tilts the lift vector for lateral acceleration. This type of control is typical of helicopters and multirotor vehicles.

[3.b] Guidance, Navigation and Control

Attitude control

Pitch -- Controlled by thrusters in the tail.

Yaw – Controlled by differential torque on the 4 main rotors.

Roll – Controlled by the thrust difference between left and right main rotors.

Note that the tail thrusters generate an unwanted yawing moment whenever they generate a force component in the Y direction. This yawing moment is automatically countered by differential torque on the main lift propellers.

Alternatively, thruster control is flexible enough that the thrusters can theoretically be used for yaw control if needed.

Also note that, depending on the strategy used for maneuvering, the fuselage can be held at a level attitude (roll angle = pitch angle = 0), independent of maneuvering. Alternatively, the fuselage can similarly be held at a zero roll angle and constant pitch angle within a pitch range of 0º to 90º. As an example, the vehicle can hover with the fuselage in a 90º nose-down attitude.

[3.b.1] Stability Augmentation System

The flight controller, based on an off-the-shelf Pixhawk unit, reads IMU sensors, including accelerometers, gyros and magnetometers. The processor uses an Extended Kalman Filter to calculate Euler angles. Various PID controls are used to actively control rotation rates, Euler angles, rotation rates and altitude.

[3.b.2] Navigation

Navigation is performed primarily by a machine vision system that tracks grid lines in the arena.

A Hokuyo URG-04LX scanning laser rangefinder is also used to detect obstacle robots. The lidar sensor is augmented by sonar sensors that cover the lidar's blind spot.

The primary altimeter is a barometric pressure sensor, augmented by a Sharp IR rangefinder that is used to compensate for variations in barometric pressure.

[3.b.3] Figure of Control System Architecture

The autopilot is currently a Raspberry Pi system, which also processes machine vision. Flight control functions are planned to be written in Ada 2012 on the RPi. We may also offload some autopilot functions to a bare-metal ARM Cortex M4 (STM32F407) programmed in Ada 2012. The Ravenscar subset of the language is used. In the future we may also investigate the use of the Crazyflie open source drone project. At this writing, the AdaCore company is reportedly rewriting Crazyflie software from C to SPARK, which is a high-reliability subset of Ada.

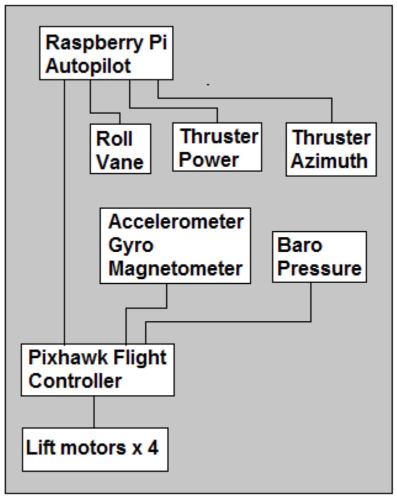

Figure 3. Control system architecture.

[3.c] Flight Termination System

The flight termination system (FTS) allows an operator to remotely cut power to the lift motors and thruster in an emergency. This is a crucial safety feature intended to prevent injury and property damage. The FTS consists of a conventional R/C system on 2.4 GHz spread spectrum.

The R/C system is connected through an electronic interface to the IARC Common Safety Switch, as described in the Mission 7 rules.

[4] PAYLOAD

[4.a] Sensor Suite

[4.a.1] GNC Sensors

The vehicle is cannibalized from a 3DRobotics Y6 tricopter with the following equipment:

Flight Controller (FC): Pixhawk
32-bit STM32F427 Cortex M4 core with floating point unit.
168 MHz/256 KB RAM/2 MB Flash
32 bit STM32F103 failsafe co-processor
Firmware: APM:Copter 3.1 (open source)
OS: NuttX RTOS

Sensors:
ST Micro L3GD20H 3-axis 16-bit gyroscope
ST Micro LSM303D 3-axis 14-bit accelerometer / magnetometer
Invensense MPU 6000 3-axis accelerometer/gyroscope
MEAS MS5611 barometer

Power System:
Ideal diode controller with automatic failover
Servo rail high-power (7 V) and high-current ready
All peripheral outputs over-current protected, all inputs ESD protected

Ground station uses APMPlanner2 program, also open source

[4.a.2] Mission Sensors

[4.a.2.1] Target Identification

The vehicle will have two control systems on board, a Pixhawk flight controller (FC) to keep the UAV stable and an autopilot (AP) to map and fly the UAV as well as accomplish target identification.

The AP will be a Raspberry Pi with the Pi camera attached to it (see Figure 6). The RPi is run Raspbian GNU/Linux 7 (wheezy) with OpenCV 2.3.1 to control the camera. The camera is set to 640 x 480 pixels and 30 fps to process the frames in real time. The camera is capable of 2592×1944 pixels stills and 1080p 30 fps video. OpenCV did not work with this setup of the RPi and camera so the library file from Emil Valkov [1] was added to get it to work. This library allowed us to address the camera directly.

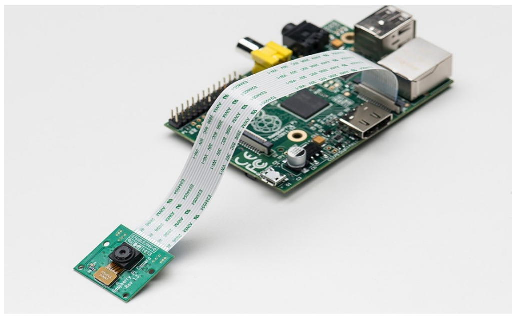

Figure 4. Raspberry Pi with camera.

OpenCV

There are three main goals that needed to be solved with OpenCV and they are line detection, objection detection with tracking, and optical flow. Doing one of these could be done but the problem is that all three needed to done in real time.

There are two methods for do Line Detection, Hough-based method and LSWMS (**Line Segment** detection using **Weighted Mean-Shift).** LSWMS is a faster method because the image does not need to be converted to gray scale and the number of lines is set by weight. The grid is mapped by getting the compass reading from the FC. This is then added to the on-board map to track the objects and goals.

Object Detection is done by taking the image and filtering out the colors that are not needed to get a black and white image for each of the three colors of the ground robots (see Figure 5). Each new image is then processed to get an X ,Y location for each robot on the map. The map is evaluated to determine what direction the ground robots are going and if the needed to be turned or avoided. The red and green robots will be directed towards the goal and the white robots will be avoided.

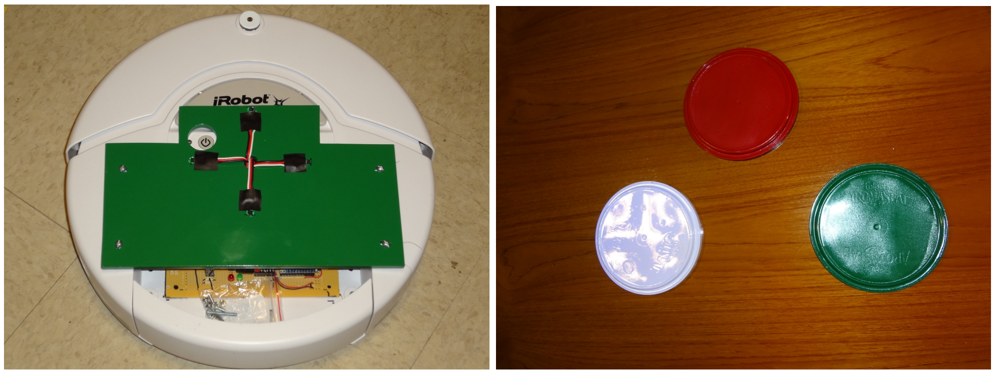

Figure 5. Ground robot and three test targets.

Optical flow is done by comparing the last image to the new image to see what has changed. The changes are the calculated to tell what direction we are moving in. This information is then used to update the UAV's location on the map and build a flight plan that will help to get as many of the colored robots across the goal line and to fly around the white robots.

Raspberry Pi

The RPi was chosen because it is small and light, with a camera, and runs Linux. The problems that we ran into were that the camera was not fully supported in OpenCV. There is a lot of work being done on the RPi so the amount of information out there is great if you can find what you need. The RPI has a USB WiFi adaptor that links up to the ground station for telemetry and remote access. Along with getting the flight data from the UAV we can also reprogram it's setting or upload new version of the application as we need to.

[4.a.2.2] Threat Avoidance

Protective Cage

A main component that contributes to the structural integrity of the vehicle is the cage. It is not there to give the robot a robust look. Instead, it provides protection to the rotors in case of any collision with moving obstacles. Moreover, the lower portion of the cage houses rows of thin magnet strips that will be used to redirect the iRobot Create® ground robots.

Structure Overview

The cage is built out of hollow carbon fiber rods (Figure 7). The diameter of the rods has not been determined at this moment. However, the roughest estimate is around a 3.3 mm outer diameter and a 2 mm inner diameter. The overall shape of the cage from a top view perspective (Figure 6) looks like a rectangle with semicircle ends. The rectangle has a length of 442 mm and a width of 340 mm. There is a spacing distance of 43 mm between the rotors and carbon fiber skeleton. Lastly, from a right side view perspective, the cage resembles a ladder in that it consists of two parallel rods crossed by perpendicular rods evenly spaced.

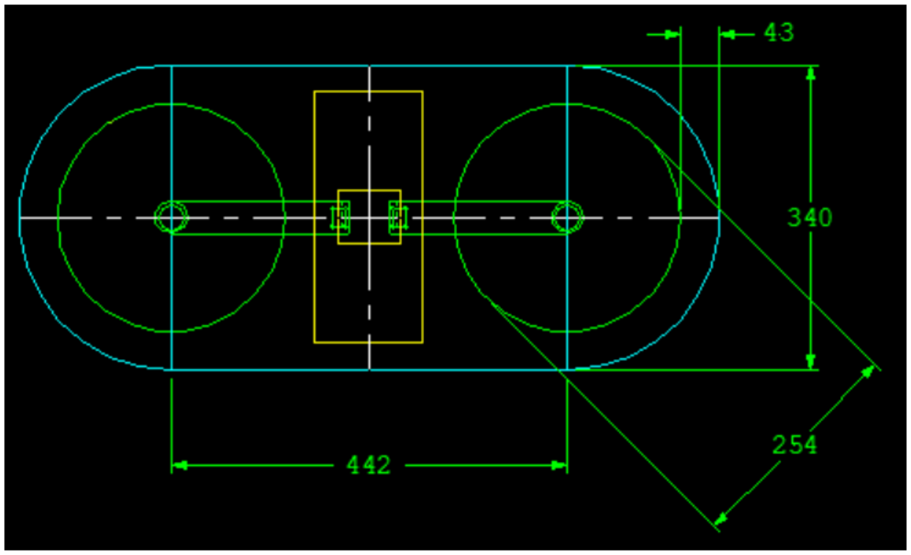

Figure 6. This image depicts a top view of the cage (light blue contour), propellers (green circles), robot computer (yellow), along with some overall dimensions (green).

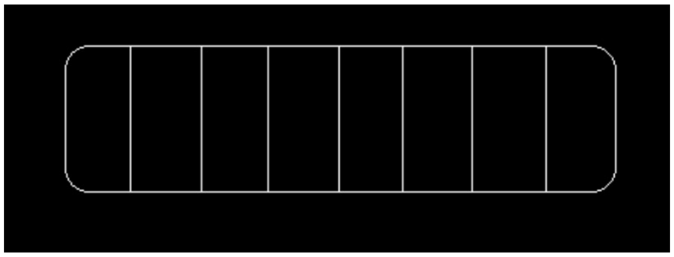

Figure 7. This is a symbolic representation of the cage's side view.

Minimizing Mass of Protective Cage

The cage encloses the 4 main rotors, which handle yaw and roll control. Pitch control is provided by a pair of small EDFs positioned outside the cage. This strategy reduces weight – otherwise the cage would need to be larger and heavier to enclose tail rotors. Doing without the protection of the cage is considered to be a minor problem because small EDFs are easier to protect from crash damage compared to larger open propellers.

The EDFs can be small because they are not required to rapidly rotate the entire vehicle, as explained in sections [3.a] and [3.b]. The EDFs instead counteract the pitching moment required to tilt the main rotor assembly, which has a much lower moment of inertia, thus requiring significantly lower power for a given tilt rate. The other functions of the EDFs are mostly to keep the fuselage level and produce small lateral acceleration, neither of which requires large thrust.

Recovery from Inverted Landing

The vehicle is designed to be able to recover from an inverted landing. This is a side benefit of the ability to tilt the main lift propellers, which can tilt through a sweep angle of 180º. In order to perform an inverted takeoff, the tail rotors need to elevate the tail as much as possible. Combined with tilting the main rotors back as far as possible, the main thrust vector can be oriented vertically, ready for takeoff (Figure 8)

.

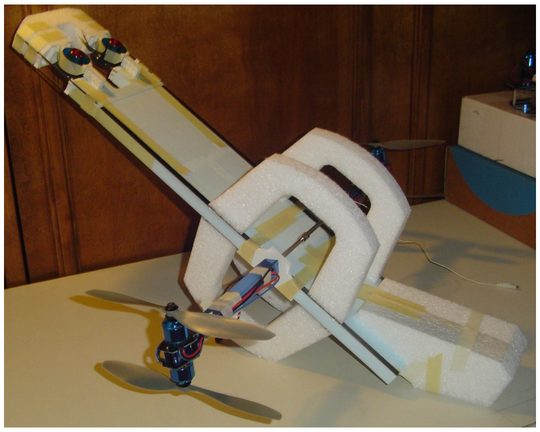

Figure 8. Vehicle in inverted takeoff orientation.

[4.b] Communications

A radio modem allows 2-way communications between the ground station and air vehicle.

Radio: 3DR Radio Telemetry V2 (915 MHz), based on HopeRF HM-TRP module

[4.c] Power Management System

A 11.1 VDC lithium-polymer battery powers the propulsion and lift systems, as well as all electronics on the vehicle.

A power distribution board routes power to individual ESCs that drive all 6 rotors. The board also contains voltage and current sensors for the main battery, as well connections to BEC voltage regulators built into the 6 ESCs. The BECs supply power to the rest of the system.

[4.d] Sub-Vehicles

No sub-vehicle is used.

[5] OPERATIONS

[5.a] Flight Preparations

[5.a.1] Checklists

- **Mechanical**
  - Fuselage
    - Check for damage
  - Protective cage
    - Attachment points
    - Check for cracks
  - Lift propellers
    - Propeller integrity
- **Communications**
  - Check Data link integrity
- **Controls**
  - Exercise controls in sequence
  - Check for proper operation

- Rotation mechanism/pylon
  - Pylon damage
  - Rotater structure
  - Rotater mounting
- Thrusters
  - Servo condition
  - Mechanical integrity of ducts
  - Propeller integrity

[5.b] Man/Machine Interface

Since the vehicle spends most of its time hovering or flying at relatively low airspeeds, it's not required to have a low drag coefficient. Therefore the structure of the vehicle is open, with equipment easily accessible for operation, maintenance and replacement.

[6] RISK REDUCTION

[6.a] Vehicle Status

A large number of parameters are streamed in real time to the ground station from the air vehicle. Parameters include Euler angles, angular rates, voltage, current and altitude.

[6.a.1] Shock/Vibration Isolation

The sensors are mounted on soft foam to isolate the internal gyros and accelerometers. In addition, all propeller blades are balanced in order to reduce vibration. The large lift motors use an outrunner configuration with no mechanical gear reduction, which further reduces vibration.

[6.a.2] EMI/RFI Solutions

In the Pixhawk flight controller, all peripheral outputs over-current protected, all inputs ESD protected.

[6.a.3] Software Risk Reduction

Software reliability is enhanced by the Ada 2012 programming language.

[6.b] Safety

The cage structure reduces somewhat the chances of injury due to spinning propellers.

[6.c] Modeling and Simulation

A software test harness was utilized to accomplish unit tests of software components.

[6.d] Testing

Flight testing lends itself to an academic lab environment, since testing can occur indoors in cluttered environments. Large outdoor flight test areas are not required.

[7] CONCLUSION

The Pima Community College UAV Club has designed an air vehicle system to herd a group of 10 ground-based robots in a predefined arena, while simultaneously avoiding a second group of 4 obstacle robots. The vehicle uses machine vision as well as lidar and sonar scans to sense its environment.

Acknowledgements – the team wishes to thank Chien-Wei Han for his support of the project as faculty advisor, and Helen Fan for her longtime administration of our listserv, since 2005. We also wish to thank Andrew and Robert Wagoner, of SteeleDesign LLC, for generously allowing us to use company facilities for meetings and flight testing. Thanks also to Jerry Petrey for his invaluable help in getting our ARM board up and running with a bare-metal Ada 2012 compiler.

[8] REFERENCES

[1] Official Rules for the International Aerial Robotics Competition MISSION 7, Version 10.2 April 2014, http://www.aerialroboticscompetition.org/downloads/mission7rules_041914.pdf

[2] Manning, Frank. Todd, Yanitzin. Worden, Tim. An Indoor Aerial Robot for Herding Ground Robots, 1 June 2014. Most of this paper was copied verbatim except for incremental changes due to improved system design.

[3] Emil Valkov Raspberry Pi Camera with OpenCV, October 19th, 2013 http://robidouille.wordpress.com/2013/10/19/raspberry-pi-camera-with-opencv/

[4] https://github.com/bitcraze/crazyflie-firmware