Autonomous Robot Herding Through Physical Interaction

Jordan Liske
Jared McMaster
Anthony Morast
Aubrey Olson
Jacob Oursland
Samuel Vinella
Michael vander Wal
South Dakota School of Mines and Technology

ABSTRACT

The South Dakota School of Mines and Technology Unmanned Aerial Vehicle Team's entry in the 2017 Association for Unmanned Vehicle Systems International International Aerial Robotics Competition is a ducted fan quadrotor helicopter leveraging commercial off-the-shelf hardware and open source software. Localization and mapping is performed through a single machine vision camera using the Direct Sparse Odometry algorithm. Path planning is achieved through a custom goal algorithm and the MoveIt! planning framework. Risk management and fail-safe mechanisms have been implemented at all levels within the control system to prevent undesirable collisions and damage.

INTRODUCTION

This paper describes the South Dakota School of Mines and Technology (SDSMT) Unmanned Aerial Vehicle (UAV) Team's competition vehicle system for the 2017 Association for Unmanned Vehicle Systems International (AUVSI) International Aerial Robotics Competition 7 (IARC 7).

Phase 1 of the IARC is to design and construct an aerial robot that will autonomously enter a 20m x 20m arena, identify and track 10 iRobot Create 2 ground vehicles, and through physical interaction with switches on the top or front of the iRobot Create 2, alter their course to cross one predetermined edge of the arena. A successful completion of Phase 1 of the competition occurs when an aerial vehicle successfully alters the trajectory of at least 7 of the 10 robots across the goal line within 10 minutes, without being disqualified by leaving the arena or three collisions with designated obstacle robots. Effectively, this competition can be thought of as a ground robot herding operation by an autonomous aerial vehicle.

Phase 2 of the IARC will take place once at least two teams successfully complete Phase 1. This phase of the competition will pit two teams against each other to herd their 5 iRobot Create 2 ground vehicles, identified by color, in less time than the competing team. The challenge may be considerably more difficult than in Phase 1 of the competition as there will be another flying vehicle, and the trajectories of the ground robots may be altered unexpectedly by the competing robot during the competition.

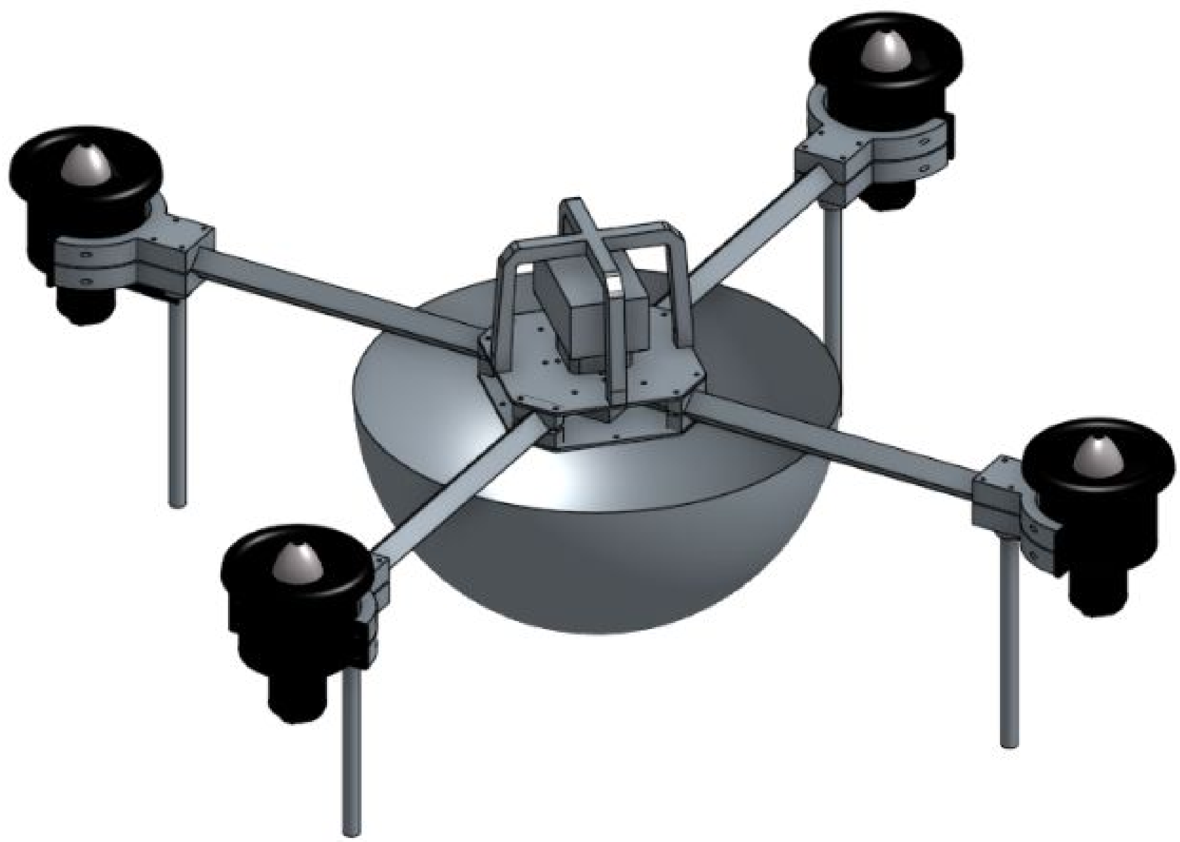

Figure 1. Rendering of SDSM&T UAV Team Vehicle

Conceptual Solution to the Problem

The overall solution is intended to be simple, robust, and self-contained, using commercial off-the-shelf (COTS) hardware and open source software where possible. The controlling paradigm is to "herd" the ground robots across the finish line, using more of a model of constrained diffusion of movement as opposed to direct control of a particular member. Each level of control has fail-safe mechanisms to minimize the risks posed by a failure of any layer of the system. The flight control system has been designed to provide immediate system state and feedback to both the safety pilot via signals to the RC (Remote Control) transmitter and to the rest of the team via the basestation laptop. The software system is based on available open source packages for every subsystem on the vehicle.

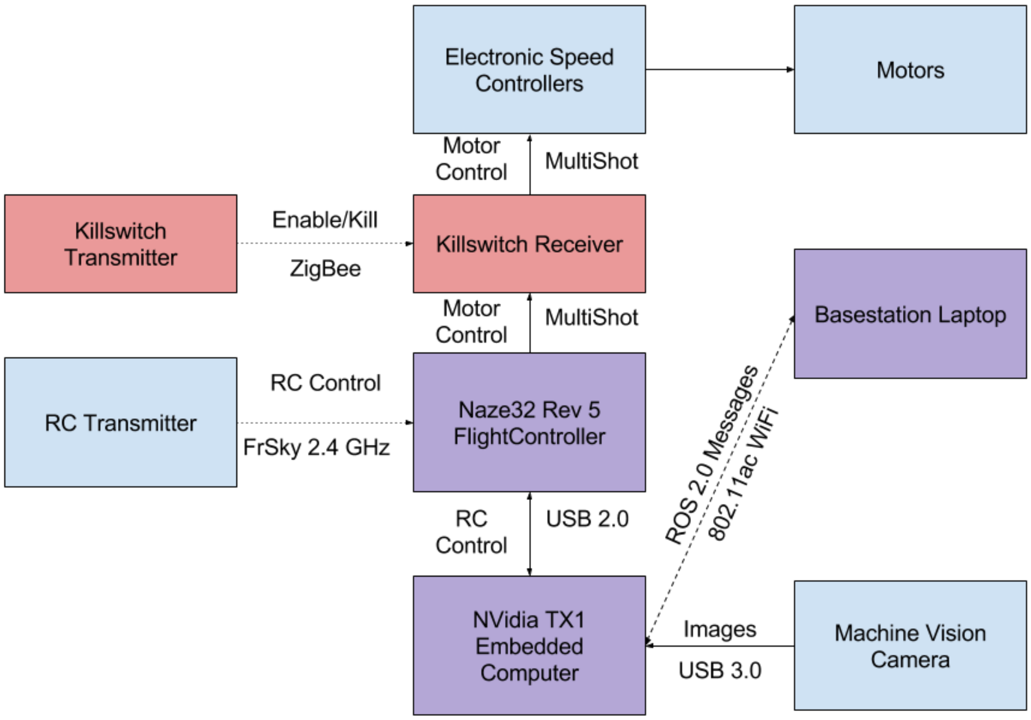

Figure 2. Overall System Architecture

Milestones

Jan 2017 – UAV team re-activated at SDSMT for the express purpose of competing in IARC 7. Initial planning and controlling charter established.

Feb 2017 – Construction begins on Type 01 (not shown) of quadcopter built completely from COTS parts.

Mar 2017 – ROS software installed and custom camera drivers developed.

Apr 2017 – Mock competition held for data collection.

Apr 2017 – Modeling begins of virtual area for overall mission planning, Type 02 (*Figure 1*) printing and construction begins.

May 2017 – Publication of paper, website, and other required IARC 7 materials. Further modeling work and completion of V2 airframe and initial test flights.

Jun 2017 – (proposed) Continuation of test flights for V2. Further development of mission theory and refinement of visual, path planning, and command and control.

Jul 2017 – (proposed) Completion of testing and readying system for competition at end of month

AIR VEHICLE

The SDSMT UAV Team's vehicle system is based on common quadrotor helicopter designs made with COTS hardware. It differs in that the usual propellers have been replaced with ducted fans. This is intended to reduce risk of damage caused by collisions with the environment and other vehicles, while allowing interaction with the front bumper of the iRobot Create 2s. The landing gear is as distal as possible to maximize the ability of slightly radial impacts with the strike plate.

Propulsion and Lift System

There are four ducted fans fixed to the airframe, two rotating in the clockwise direction and two in the counterclockwise direction. Thrust differentials are created by changing the rotational speed of individual fans to affect the roll and pitch. Yaw control can be performed by adjusting the rotational speed of the rotating fans to apply an overall torque to the vehicle about the Y-axis. The ducted fans localize thrust, which reduces the likelihood of inadvertently triggering the dorsal strike plate, and create inherent lateral collision durability as they are physically contained in the propeller shroud.

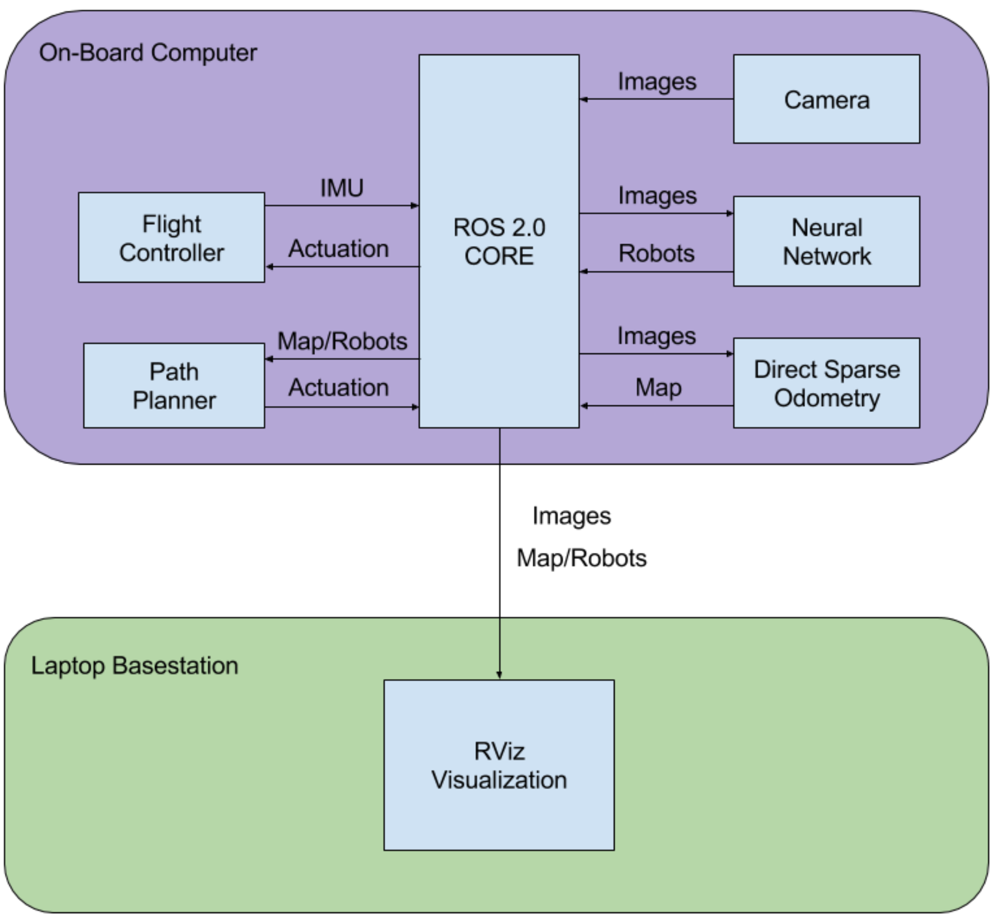

Figure 3. Control System Architecture

Guidance, Navigation, and Control

The onboard processing is performed by an NVidia TX1, which is a cost-effective single-board computer with high performance computation capabilities on a quad-core ARM CPU and integrated GPU. The flight controller is a customized version of the open source BetaFlight flight control software. The RC Transmitter runs a customized version of the OpenTX 2.0 RC transmitter software. The on-board computer runs the open source Ubuntu 16.04 operating system using the open source Robot Operating System (ROS) 2.0 [1] for software module integration.

At the heart of the system is the Naze32 Rev flight controller operating a custom version of the BetaFlight flight control software modified to permit controller inputs from both the RC receiver and commands from the TX1 on the USB 2.0 port. Selection of input signals is dependent upon the state of the mode switch on the RC Transmitter. Motor control signals are then sent to the Electronic Speed Controls (ESCs) to drive the motors.

Stability Augmentation System

The BetaFlight flight control software provides stability augmentation system built into the controller. Without commanded inputs the flight controller will attempt to maintain a level attitude.

Navigation

The state of the aerial vehicle system and environment is measured by an on-board machine vision camera and inertial measurement unit (IMU) integrated in the flight controller. A map of the world is synthesized from these measurements using the Direct Sparse Odometry (DSO) [2] algorithm. The images are then fed into a neural network trained to recognize the iRobot Create 2s and return the location of the robots.

Control

A trajectory that optimizes for the goal is computed using the MoveIt! motion planning framework [3] and resulting actuation commands are sent to the flight controller. The main trade off, as in any planning algorithm, is the distance/time in the future that the algorithm is capable of planning out. The control software uses a heuristic load-balancing approach that takes distance to target, number of players available, and transition to state to minimize the amount of processing power necessary at certain times and allow rapid updating when approaching physical interaction with a single member of the herd.

Flight Termination System

The killswitch is an Radio Frequency (RF) transmitter-receiver pair using the ZigBee protocol that gates the control signal to the ESCs. The interrupter on the airframe receives a constant signal to affirm that the handheld safety switch is still active and transmitting. Switching the killswitch to the “KILL” position on the handheld transmitter ceases the signal, resulting in the airborne killswitch interrupting the control signal to the ESCs.

PAYLOAD

The payload is a downward facing Leopard Imaging LI-USB30-M021 HD camera mounted to 2-axis gimbal allowing for pitch and yaw rotation. The camera and gimbal system is enclosed in a plastic dome that allows full 360-degree view below the quadcopter (save the minimal landing gear). This protective dome acts as a point of physical interaction for the front actuator and dorsal strike plate on the iRobot Create 2s.

Sensor Suite

The aerial vehicle system includes the onboard machine vision camera and the IMU integrated in the Naze32 Rev5 for sensing the state of the aerial vehicle and surrounding environment.

Guidance Navigation and Control (GNC) Sensor

The internal flight controller, with added information about the environment from the camera, sends information to the NVidia TX1 where it is integrated into the ROS environment and fed to the MoveIt! flight path planning software. After the software does its planning and path creation, that data is converted back into attitude and airspeed information for the flight controller. The Naze32 Rev5 then determines necessary rotation speed to create a certain attitude, vertical and lateral movement velocities, and converts that data into an electric potential which is sent to the individual motors. All of this data is logged at both the flight controller and central processor level for later debugging.

Mission Sensors

The only mission sensor is the machine vision camera. The images are fed to the DSO algorithm which allows visual triangulation of position based on historic data and reference key frame difference analysis. The output of the DSO algorithm is a point cloud that is provided to the path planning algorithm.

Target Identification

Target identification is performed by feeding the images from the machine vision camera to the neural network trained to identify iRobot Create 2s. The pixels are then combined with the point cloud from the DSO algorithm to localize the iRobot Create 2s on the map.

Threat Avoidance

Threat avoidance is primarily done through a combination of the above described methods. Points representing obstacles are identified by the DSO algorithm in the point cloud and the path planning algorithm generates a trajectory to avoid the obstacles.

Communications

The initial way of "waking" the robot is completely physical. First, plug in the battery sources and wake the primary and secondary processing units. Once online, there are two communication paths that are used to control the airframe. The first is an RF interface that allows manual control of the airframe through a handheld RF transmitting controller. The RC Transmitter additionally can send basic commands to the airframe, resulting in actions like "enter autonomous mode", "automatic takeoff", "enter mission mode", etc. The second is over TCP/IP into the integrated WiFi chip on the NVidia Jetson TX1. The basestation laptop and wireless router are set up to receive telemetry. These two communication methods allow the robot to enter various modes of flight and competition and transmit telemetry and performance data for monitoring.

Power Management System

Power is supplied by a pair of lithium polymer battery packs, which can supply the necessary current to the ducted fan motors, controllers, and computers. Power distribution to the ESC is performed by a wiring harness. Power regulation and distribution to the controllers and computer is through a COTS switching regulator for RC aerial vehicles.

OPERATIONS

The general operation of the airframe has multiple steps, phases of flight, and operational functionality to verify. The overall walk-around is detailed below. After the airworthiness is determined and the telemetry base station is set up, the aircraft is ready to engage in the competition round. Essentially, run up the aircraft, command it to takeoff and center over the arena, tell it to enter mission mode, allow the 10 minute round to expire, tell it to exit mission mode and re-center over the arena, command it to land back at its takeoff point, and then power the whole system down so batteries can be changed and the airframe prepared for the next run. Additionally, at all phases of flight, the pilot be ready to take manual control if the aircraft enters a run-away or potentially dangerous situation to avoid loss of aircraft or potential harm to any person. If that is not sufficient the judge will have a “kill switch” emergency procedure to cut all power to the motors sacrificing the airframe instead of causing injury.

Flight Preparations

The initial flight preparation involves setting up the base station laptop and private network on associated router and getting all the command software and shells up and running. Additionally, ensure to check all the connections on the airframe to ensure structural stability and electrical connectivity during the flight. Spin the rotors and look for cracks, make sure the batteries are sufficiently charged and intact, ensure the landing gear is stable, and verify the overall airworthiness of the aircraft.

Checklists

Normal Start-up

- Ensure kill switch is set to interrupt mode to prevent power to the motors
- Ensure all switches (SA-SD) are in the down position
- Plug in both battery connections
- Switch power to the on position
- Wait for Naze32 Flight Controller and NVidia Jetson TX1 to complete startup
- Verify link between RC Transmitter and RC Receiver
- Verify link between laptop base station, NVidia Jetson TX1 and Naze32 Flight Controller
- Set kill switch to “ENABLE” position, permitting the ESCs to operate the motors
- Command low upward thrust from RC Transmitter (insufficient for takeoff)
- Switch the kill switch to the “KILL” position
- Verify that the motors no longer function
- Return to idle state in preparation for takeoff
- Switch the kill switch to the “ENABLE” position

Takeoff

- Verify system is ready for flight mode in base station
- Set SA to up position to enable Autonomous Mode
- Verify that the system is in Autonomous Mode on the laptop base station
- Set SB to up position to perform takeoff
- Set SC to up position to enter arena
- Airframe should locate the arena boundaries and center itself at height sufficient to observe the entire arena
- Apply manual control inputs to move the robot off center to verify manual override and subsequent correction

Enter Mission Mode

- Set SD to up position to enable Mission Mode
- Verify that the system is in Mission Mode on the laptop base station and acting accordingly
- Monitor execution over the 10 minute round and be ready with Emergency Manual Control and Emergency Shutdown checklists

Exit Mission Mode

- Set SD to down position to disable Mission Mode
- Verify that the airframe returns to hover over the center of arena
- If aircraft does not respond appropriately, resort to manual control to RTB and land aircraft at nearest suitable spot.

RTB and Landing

- Set SC to down position to initiate RTB
- Monitor progress towards landing pad
- Set SB to down position to initiate landing procedure

Normal Power-down

- Once airframe is landed, command kill switch to interrupt power to motors.
- Set throttle to idle
- Set SA to down position to enable Manual Mode
- Power off NVidia Jetson TX1
- Disconnect batteries from electrical bus and ensure that primary and secondary processors are powered down

Emergency Shutdown

- Actuate kill switch from transmit to interrupt mode

Emergency Manual Control

- Switch RF controller to manual mode and apply necessary corrections to take control of the aircraft.

Figure 4. RF Control Station

Man/Machine Interface

The main controller is a Taranis X9D Plus programmable RC Transmitter with status display. It contains numerous digital control switches, dials, joysticks, and trim tabs. The pilot will use only the front switches to send modal commands and the joysticks for manual throttle and attitude control when necessary.

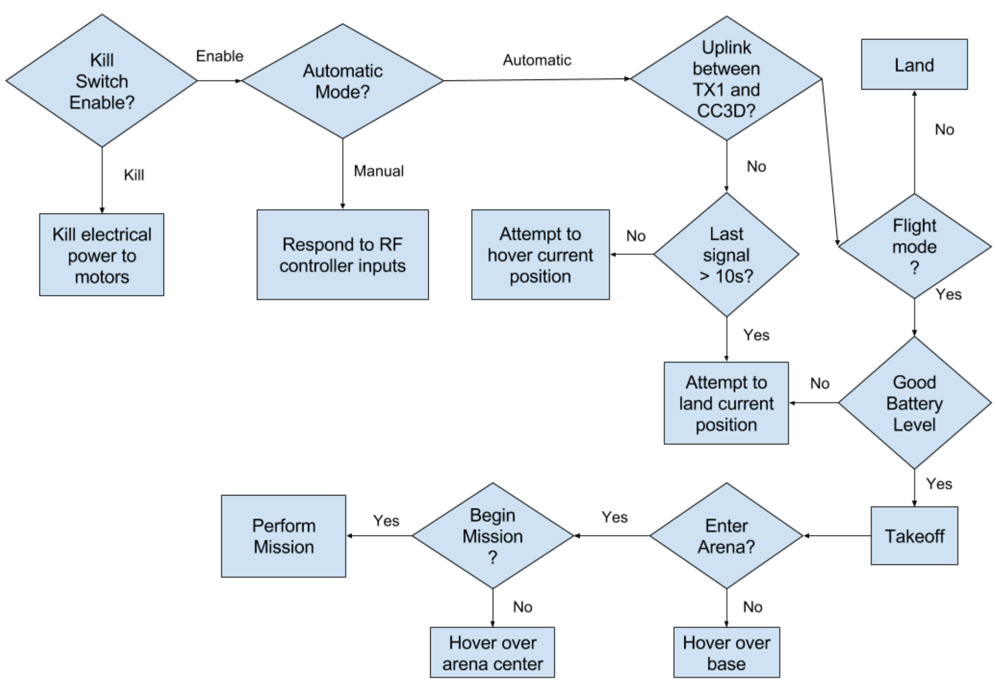

Figure 5. Kill Switch and Overall Basic Decision Tree

RISK REDUCTION

The main overarching paradigm for risk reduction is making the robot as robust and durable as possible. To this end, all the key components are enclosed in some shape or fashion. The fans are ducted to prevent damage from lateral collision. All of the computational components are on the top of the airframe ensuring that it will only be damaged from a vertical collision or complete loss of control of the airframe and inversion. The camera is protected by a plastic dome that also serves as a strike plate. The kill switch is another form of risk reduction as outlined previously. And the final piece, which is also the most likely to fail, is the software controlled obstacle avoidance.

Vehicle Status

The vehicle status is communicated locally on the airframe and remotely to the RF controller and laptop. Locally, a basic health status is downlinked from the NVidia Jetson TX1 to the flight controller based on the time of last commanded input. If $\frac{1}{5}$ of a second passes, the flight controller assumes that something may be wrong with the control algorithm on the NVidia Jetson TX1 and attempts to halt movement and hover in a neutral attitude. If 10 seconds pass with no command from either the NVidia Jetson TX1 or RC Transmitter the flight controller attempts to land in its current position.

Shock/Vibration Isolation

All of the components have foam tape at between the contact points of intersection and attachment. This should isolate vibration and provide limited shock absorption. It will also provide a dampening effect from higher frequency sympathetic vibrations. Additionally, the tolerance of the parts is low enough, in both COTS and 3D printed, to provide sufficient rigidity to prevent bowing, ambulation around joints, and low frequency oscillations across the airframe

EMI/RFI Solution

The only command and control is through RF in the form of the RF controller and the kill switch. The first solution is to use different frequencies for outright deconfliction. The second is to avoid communication as much as possible. Capable onboard processing drastically reduces the importance of OTA communications. Lack of status/command and telemetry to the RF controller and base station respectively does not prevent function of the airframe. Only lack of a signal from the kill switch will produce a deleterious state.

Safety

The main safety features are inherent in the physical design methodology and hardware/software mentioned earlier. Self-contained and enclosed systems prevent damage of fragile components and also reduce abrasive/free-rotating/cutting surfaces at the same time. Hardware redundancy in control through automatic, manual, and emergency kill switch ensures that the airframe will be ultimately under human control. Software redundancy can be seen in Figure 4 earlier.

Modeling and Simulation

Physical modeling was primarily done in SolidWorks 2017 for component creation and printing. The model was re-created in Onshape for display, rendering, and export to the Gazebo simulation environment. Mission simulation was done in Gazebo, an open-source robot modeling program that gives pre-existing models for iRobot Creates, programmable behavior, a physics engine, import capability, and plug-in compatibility for ROS and thus capable of simulating all of the operational algorithms used in the vehicle system.

Testing

Testing of the system is performed primarily in the simulation environment. After implementation of subsystem, the physical vehicle or relevant subsystem is tested in hardware. Using the ROS 2.0 framework, this transition from simulation to physical testing is simply a matter of altering message publishing and subscribing from simulated devices to physical device.

CONCLUSION

The South Dakota School of Mines and Technology Unmanned Aerial Vehicle Team's aerial vehicle system should be capable of satisfying the requirements of the International Aerial Robotics Competition 7 Phase 1 mission requirements by providing a robust, responsive platform with significant compute capabilities and modern sensor fusion and planning algorithms. The platform aims to minimize risk of collision and damage by providing several mechanisms for detecting and recovering from faults.

REFERENCES

1. "ROS 2 Design - ros2/ros2 Wiki." http://design.ros2.org/. Accessed 31 May. 2017.
2. "Computer Vision Group - Visual SLAM - DSO: Direct Sparse Odometry." 13 Nov. 2016, http://vision.in.tum.de/research/vslam/dso. Accessed 31 May. 2017.
3. "MoveIt! Motion Planning Framework." http://moveit.ros.org/. Accessed 31 May. 2017.