AIAA SciTech Forum
9 - 13 January 2017, Grapevine, Texas
AIAA Information Systems-AIAA Infotech @ Aerospace

10.2514/6.2017-1377

Small Unmanned Aircraft Systems for Project-Based Engineering Education

Pedro F. A. Di Donato[*]
National Civil Aviation Agency - ANAC, São José dos Campos, SP, 12242, Brazil
University of Michigan, Ann Arbor, MI, 48109, USA

Peter E. Gaskell [†]
University of Michigan, Ann Arbor, MI, 48109, USA

Ella M. Atkins [‡]
University of Michigan, Ann Arbor, MI, 48109, USA

Small unmanned aircraft systems (sUAS) have gained significant popularity with applications ranging from aerial imagery acquisition to package delivery and military surveillance. This popularity is reflected in a variety of startups, the hobbyist community, and in academia. sUAS are employed in academia for both research and teaching purposes. Despite widespread use of sUAS at the university level, most literature focuses on research and commercial applications with few publications dedicated to sUAS for engineering education. This paper investigates the use of sUAS and student-designed manipulator payloads to support both Aerospace and Robotics senior and graduate-level courses developed at the University of Michigan. Four project case studies relying on small unmanned systems will be investigated that collectively provide an integrated learning environment over software, sensor, actuator, and decision-making (control) algorithms. Two of the projects also offer the opportunity for students to design, build, and test a manipulator on a fixed base then integrate it within a quadrotor platform tasked with accomplishing an autonomous manipulation mission. The paper discusses specifics of the projects, project and student learning outcomes, and safety measures applied, including institutional safety policy. Specific outcomes and challenges of course projects are discussed.

I. Introduction

Engineers must use their skills to design, build, and adequately test each Aerospace system to meets a client's needs at a manageable cost. Today's Aerospace platforms, small to large, are comprised of integrated hardware and software elements. *Engineering system design* is a complex cognitive process[1] that has received increasing attention over the past two decades.[2] One method to teach engineering design is *project-based* learning (PBL).[1] Projects can be conceived as a competition between collaborative groups, a strategy that motivates students while fostering collaboration to produce a highly effective learning environment.[2] Despite the fact that PBL has usually been incorporated in engineering curricula as capstone and cornerstone design courses, educators have also investigated extending PBL into traditional fundamental engineering courses.[2]

Project-based learning can take advantage of the fact that Small Unmanned Aircraft Systems (sUAS) are rapidly proliferating for a variety of civil, government, and hobby applications.[3] Aerospace education efforts have primarily involved sUAS for "design-build-fly" (DBF) design courses and student team activities. While these exposures are critical to understanding the structure, aerodynamics, and propulsion systems of aircraft, the simple radio-control flight scheme typical for senior capstone courses and competi-

[*] Regulation Specialist (on leave), PhD Student, Aerospace Engineering Department, Student Member.

[†] Research and Development Engineer

[‡] Professor, Aerospace Engineering Department, Associate Fellow.

tions such as the Society of Automotive Engineers (SAE) Aerodesign competition[a] do not include several areas important in an Aerospace project such as the design and integration of sensors, data processing, decision algorithms, software, and mission payload design and management. Especially in the areas related to decision-making and control, experts have long recognized that existing graduate programs do not give sufficient attention to experimental systems education.[4] This status has not changed significantly in 21st century as Aerospace undergraduate and graduate students studying control theory typically graduate without experiencing real-world control and modeling issues such as saturation, sensor noise, and real-world environmental uncertainties, particularly apparent when flying sUAS outdoors away from motion capture.

This paper describes graduate course experimental projects that cover these areas critical to Aerospace and more generally to robotic or vehicle systems. The University of Michigan has flown a fleet of quadrotors in a tethered indoor atrium to support substantial student projects in a graduate Robotics course (ROB 550: Robotic Systems Laboratory) and in two existing Aerospace courses: AEROSP 450 (Flight Software Systems) and AEROSP 552 (Aerospace Information Systems). While AEROSP 450/552 projects focus on guidance, navigation, and control, ROB 550 course projects focus on design, build, control, and automation of student-designed manipulator payloads that allow the quadrotor to interact with its environment, grasping and releasing objects or perching on a bar. All projects have a significant emphasis on algorithms and software design and testing to enable autonomous flight supplemented by manipulation in the case of the robotics projects. The software and payload challenges students face represent a significant portion of future challenges the industry must tackle to deploy small UAS safely and in novel applications.

More than presenting the course projects, this paper discusses the results of their deployment in class, including the challenges faced by students and instructors. These include the broad range of student educational backgrounds, especially in the Robotics course. The paper also presents the required safety measures necessary to operate moderate-size quadrotors capable of carrying manipulator payloads with student-written software.

This paper begins by reviewing literature in which sUAS are applied to engineering education projects (Section II). Section III presents the primary quadrotor platform and its sensors. Course projects are presented in Section IV followed by a description of safety measures in Section V. Section VI offers a discussion of project outcomes, lessons learned, and policy, while conclusions are presented in Section VII.

II. Related Work

Despite the proliferation of sUAS surprisingly little attention has been devoted to publications describing their integration into the engineering curricula.[5] While “design-build-fly” courses are relatively common in Aerospace programs, sUAS also have broader application in engineering education areas such as computer science and electrical engineering. The best sUAS are co-designed over mechanical, electrical and software/computer systems. Incorporation of sUAS into the curriculum faces several challenges. First, many of the most popular multicopters are delivered with proprietary (closed-source) software and interfaces. While open-source options are available, e.g., Pixhawk[b], they typically offer more overhead particularly in digesting software and integrating the appropriate low-cost hardware for safe and effective operation for even the most experienced student. Inner-loop sUAS feedback control would provide a nice supplement to undergraduate Aerospace lab coursework, but even simple experiments can result in sUAS crashes that render hardware inoperative thus limit sustainability even when safety is managed by operating with small platforms or within a netted environment. One way to avoid these difficulties with a simple single degree of freedom system is to employ the AERO-beam,[6] a system composed of one propeller and motor at the end of a beam equipped with inertial sensors. In their paper, the designers discuss how their proposed system could be used with students of different levels to expose them to sensor data and controller design, implementation, and test.

An affordable and popular sUAS platform is the AR-Drone, available on the market since fall 2010[7] with a recent release of open software to facilitate educational use. This platform was utilized in a one-day contest of automatic visual navigation for groups composed of undergraduate and graduate students of different backgrounds.[8] The teams were required to understand and modify five given software modules to guide the sUAS through a course and land at a goal location. In another application, the AR-Drone was incorporated in a graduate level course about software application design.[9] This course includes both active and passive learning components, and students must give predetermined commands to the drone using different devices.

[a] http://students.sae.org/cds/aerodesign/

[b] https://pixhawk.org/

Another popular platform that ships with proprietary code, the DJI Phantom 2, was integrated within an existing computer engineering design module for first-year students.[5] Students who successfully demonstrated control of ground vehicles traditionally used in the module were offered the opportunity to extend their activities to provide outer-loop control the sUAS. The authors discuss the development and use of a geofence and obstacle avoidance system called "hypervisor" to guarantee safety while operating the drones outdoors.

The projects designed for the Aerospace courses described in this paper are similar to the above-discussed projects in that its objective is for students to design, implement, and test guidance, navigation, and control software. Because our focus is on graduate education, the projects described in this paper are more ambitious in scope and can incorporate more advanced concepts in multi-layer control with optimal waypoint sequencing and state machine mode switching layered on lower-level guidance, navigation, and control. The robotics projects described in this paper are also unique in that they include the design-build-test of mechanical devices that are mounted on the sUAS platform to meet application challenge goals. A sUAS with more payload than the AR-Drone or DJI Phantom offer is needed to fly those devices, which increases the challenges of sUAS safety as discussed above.

III. sUAS Platform and Sensors

III.A. sUAS

The projects described in this paper used a SkySpecs[c] carbon fiber quadrotor platform presented in Figure 1. This quadrotor is also the platform for ongoing research activities.[10] The propulsion system consists of four T-Motor MN-4010 370 kV motors with fully ducted 12 in propellers. The system can carry payloads weighing up to 1 kg with a flight time of approximately 15 minutes.

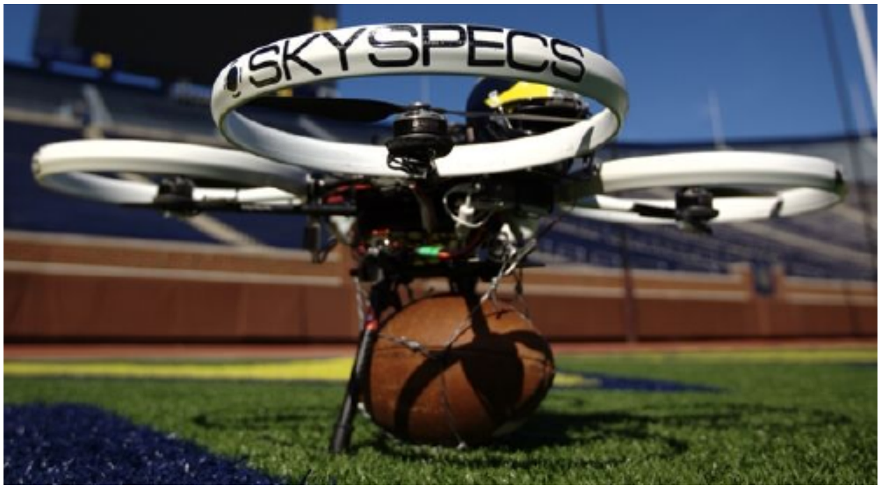

Figure 1. UAS Platform - SkySpecs Quadrotor

The quadrotor is equipped with a commercial DJI Naza-M flight controller[d] that provides attitude stabilization. Using such a "black-box" inner loop flight controller maintains quadrotor stability despite limited student experience in programming a low-level flight controller. Note that for projects involving manipulator integration a separate control lab with a safer fixed-base system offers students exposure to the challenges of embedded feedback control. We intentionally avoid student implementation and test of inner-loop control on the larger quadrotor for two reasons. First, quadrotors are dynamically unstable,[11] which makes manual flight not feasible without an automatic stabilization inner loop. Second, testing low-level controllers would significantly increase the risk of the operation since a poor performing controller would result in unstable flight, and manual flight override will not help as a human also cannot manually issue quadrotor motor commands that stabilize the platform. We, therefore, begin with outer-loop control emulating the pilot's "stick inputs" as the first layer of embedded controller implementation and testing.

[c] `http://www.skyspecs.com`
[d] `http://www.dji.com/product/naza-m`

III.B. Embedded Development Platform

The different course projects presented in this paper make use of a general architecture shown in Figure 2. The platform for the development of student projects is the BeagleBone Black single-board computer[e]. This platform provides a familiar and accessible Linux-based environment for all software development. A custom breakout board gives easy access to digital, analog, and serial interfaces as well as providing additional circuitry to provide protection during power faults and high voltage discharge events. The use of the breakout board allows different sensors to be connected to the development platform depending on the project.

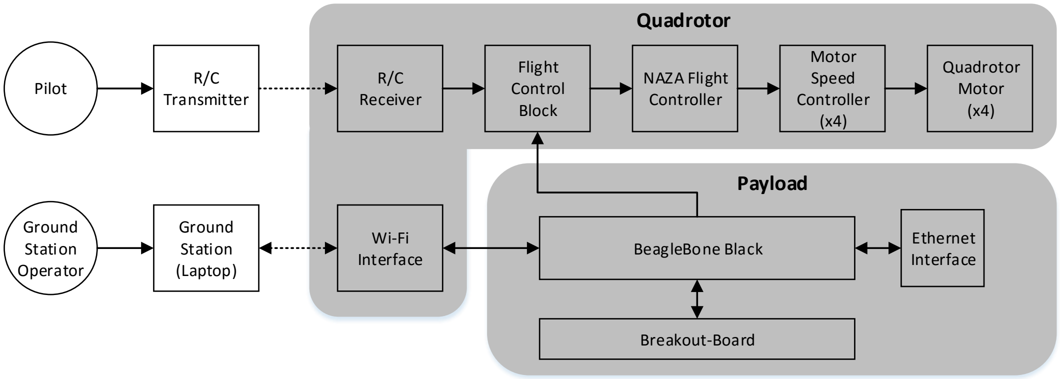

Figure 2. SkySpecs Quadrotor General System Diagram

The inner loop Naza-M flight controller can receive control signals from two sources: from the pilot using a remote controller and from the BeagleBone Black computer. The switch between both input sources is accomplished using an electronic board named “Flight Control Block” originally provided by SkySpecs along with the platforms, and software running on the onboard computer which listens to a particular channel of the remote controller. With this design, the pilot can activate autonomous operation and retake control using the remote controller at any time, similar to the “buddy box” used for decades by hobbyists for flight training. A ground station operator is connected to the on-board computer using a Wi-Fi interface to start and stop the different software to be tested. The onboard BeagleBone Black has an additional Ethernet interface which is used to provide an interface with the motion capture system that minimizes packet loss and delays.

III.C. Sensors and Actuators

A variety of sensors were utilized for projects presented in this paper. Off-the-shelf motors, electronic speed controllers, and digital servos were used to actuate the sUAS and its payload. Students were provided with a basic code written in C and/or Python to handle the communication with each of these peripherals, with C code used whenever computational complexity or real-time execution speed might be an issue. Provided code modules leave “hooks” for students to insert their own code to provide a realistic path for student success. Lecture content covers algorithm and software design, both for provided modules and student-written modules to ensure complete concept exposure. Students learn to “pay attention” to lecture content for provided modules because understanding all code modules’ functionality can, in turn, facilitate debugging of student-written code.

III.C.1. Inertial Measurement Unit (IMU)

The IMU provided to the students is the model LSM9DS0 on a small daughter board.[12] It provides a three-axis accelerometer, gyro, and magnetometer and communicates with the BeagleBone using the Inter-Integrated Circuit (I2C) protocol. While not needed for flight stabilization, the additional IMU allows

[e] `https://beagleboard.org/`

students to gain experience programming an attitude and heading reference system and provides reference data for the outer-loop flight controller.

III.C.2. Computer Vision

The primary sensor used for computer vision is CMUcam5 Pixy [f]. It combines a camera and microcontroller running an optimized color blob detection algorithm. The sensor can detect multicolor fiducial tags at 30 Hz. By using colored fiducial tags placed around the flight area, students can implement a simple indoor localization system even without the motion capture system. The sensor communicates with the BeagleBone via Serial Peripheral Interface (SPI). The Pixy provides a particularly valuable learning experience on sensor sensitivity to environmental conditions. The Pixy is sensitive to ambient lighting and reflections, which necessitates recalibration as cloud cover and day/night conditions evolve. Further, the Pixy has a limited field of view requiring software to handle switches into and out of the area populated with localization targets.

III.C.3. Sonar Distance Sensor

The MB1010 LV-MaxSonar®-EZ1 sonar sensor[g] was used to estimate ground distance. A sensor of same type is also used attached to the optical flow. The interface with Beaglebone is achieved using Universal Asynchronous Receiver / Transmitter (UART). This sensor provided a reliable height estimate for small pitch and roll angles and for heights within the range of the sensor. As with Pixy, students gained experience implementing software to manage out-of-range values as well as the application of filters (e.g., median filter) to eliminate spurious signals that can significantly degrade the quality of standard Kalman filter estimates.

III.C.4. Optic Flow

The PX4Flow optic flow smart camera[h] combines a processor, an image sensor, and a gyroscope. These measurements are fused with the sonar distance sensor that can be attached to the system. The final set provides $x - y$ speed estimates as well as optical flow values, gyro rates, altitude, and optical flow quality. The processor can also output integrated values of some of these measurement. As with Pixy, the optic flow data is sensitive to environmental conditions, specifically overall lighting and level of texture in the overflown surface. A smooth surface renders the PX4Flow ineffective, requiring software to evaluate data quality and consistency before passing results into an outer-loop flight controller.

III.C.5. Motion Capture

The OptiTrack commercial motion capture system from NaturalPoint Inc.[13] was used as an alternative to onboard navigation systems. This system provides accurate indoor 3D localization and 3D attitude estimates, effectively replacing an INS (inertial navigation system) with a better quality alternative. Motion capture can provide a valuable source of truth information to validate other sensor data and estimation/filtering/fusion algorithms. On the other hand, motion capture can also provide students with a false sense that the sensing challenge is "easy and solved", so one must take care to expose students to additional sensors more realistic for use outside motion-capture-equipped environments. The amount of data provided by a 60Hz (tunable) update rate was considered suitable for the student-designed decision and control algorithms associated with our projects, which in turn required the use of Ethernet communication instead of Wi-Fi. This limitation did not significantly impact the flight since the larger quadrotors were flown with tethers to ensure flyaway did not occur (see Section V for more details).

III.C.6. Servos

For the robotics projects, two types of digital rotary actuator servos were used: standard radio-control (R/C) servos with pulse width modulation (PWM) input and Dynamixel[i] servos. Dynamixel servos offer a higher level of accuracy and accept a broader set of input directives but require instructions in a particular protocol

[f] `http://www.cmucam.org/projects/cmucam5/wiki`

[g] `http://www.maxbotix.com/Ultrasonic_Sensors/MB1010.htm`

[h] `https://pixhawk.org/modules/px4flow`

[i] `http://www.robotis.us/dynamixel/`

which require a dedicated interface board. Students learned about PWM, serial protocols, and motor control as part of the lectures. One robotic manipulation lab modules asked students to manually generate PWM output from the Beaglebone which provided exposure to embedded code relying on counter/timer and serial communication interfaces. Modern speed controllers and servos are typically quite reliable, so students did not face appreciable challenges in direct servo control once interface code was in place. Multi-thread, socket-driven software data management and process timing provided the primary learning opportunities related to servo control.

IV. Course Projects

IV.A. Flight Software systems (AEROSP 450) and Aerospace Information System (AEROSP 552) Projects

IV.A.1. Course Overview and Project General Educational Goals

The third author has developed two courses at the University of Michigan related to software and computing: Flight Software Systems (AEROSP 450) and Aerospace Information Systems (AEROSP 552). Ideally, these courses would be offered as a two-semester sequence, but the faculty voted to not require AEROSP 450 as a prerequisite for AEROSP 552, so in practice, the distinction is that undergraduates select AEROSP 450 as a technical elective and graduates select AEROSP 552. AEROSP 450 covers a number of topics ranging from data structures and complexity, real-time software and scheduling, and automata / state machines to embedded control. AEROSP 552 covers topics ranging from state machines, timed and hybrid automata to information theory, search, and planning (deterministic and uncertain). Obviously, the lecture-based content in this courses is significant, so lab exposures are in the form of end-of-term (final course) projects.

The final course project for each class is developed over the last half of the course to introduce the students to a real application of Aerospace Guidance, Navigation, and Control along with high-level task management (via a state machine) and search-based flight planning. For most Aerospace students who do not elect student team experiences, AEROSP 450 or 552 will be the first exposure to implementing concepts learned in AEROSP 450/552 and in prerequisite flight controls courses. Working with real sensors and actuators presents major challenges to students. Sensors have a communication protocol and can present latency, saturation, drop-outs and imprecision. Data structures must be managed such that they are modified consistently and shared among all dependent functions with sufficient frequency to ensure stable controller operation. Actuators commanded through serial interface also present uncertainty and latency. For many who have worked almost exclusively in a modeling and simulation environment such as Matlab, the project offers the first experience in translating an algorithm or equations to software. We discourage automatic code generation because part of the learning experience is in manually implementing the code at least once, much like students still learn to perform basic math functions such as long division before being offered calculators. While most students hear about implementation challenges in different courses, theoretical control and dynamics courses do not devote time to discussing them in detail, instead ending the discussion once stability, controllability, and performance metrics (e.g., margins) are satisfied. Real-world exposures reveal that real-world controller implementations rarely exhibit the excellent behavior seen in simulation providing an excellent learning opportunity.

This paper discusses two different sUAS projects used for AEROSP 450 and AEROSP 552 during Winter term offerings in 2015 and 2016. While the first project focuses on state estimation, the second focuses attention on flight planning. The payload configurations used for each project are presented in Figure 3.

IV.A.2. Project #1 Description and Specific Educational Goals

Project #1 required students to develop software that allows the quadrotor to autonomously maintain its position and also translate from one waypoint to another. While these waypoint coordinates were known, the only sensor data input sources were onboard sensors: IMU, Pixy, and PX4Flow (see left diagram of Figure 3). These data streams were converted by outer-loop control software into PWM control signals then sent to the Naza-M inner loop flight Controller. These signals substitute the ones commanded by a pilot using a remote control. Autonomous takeoff and landing were not necessary. Therefore, students were asked to accomplish three main tasks:

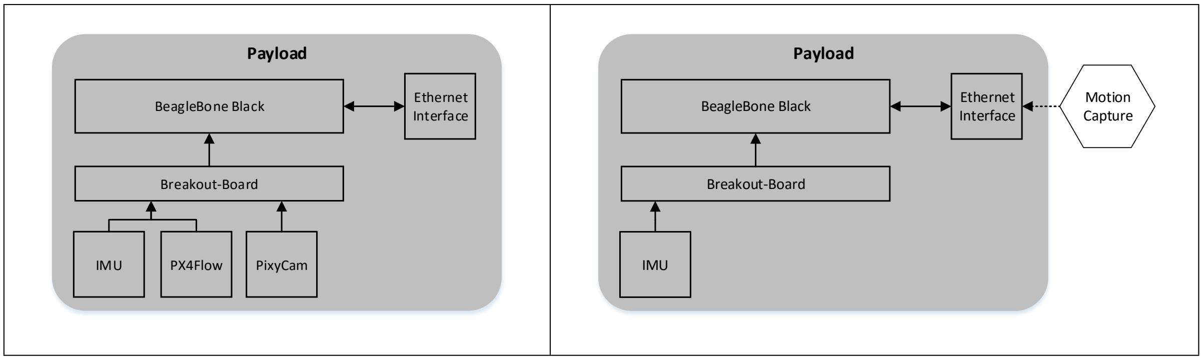

Figure 3. Quadrotor Payload Configuration for AEROSP 450/552 Projects. Left, Project #1. Right, Project #2

1. Estimate the current position/velocity and attitude/angular velocity using the available onboard sensors;

2. Implement a controller that would convert the estimated and desired position to appropriate commands for the Naza-M;

3. Implement logic that would make the sUAS to hover over each waypoint and to move to the next desired waypoint once hover had been accomplished over a preset amount of time.

The primary educational goal of this project was to offer students experience with different sensors and fusion of their data through low-level and integrated estimation algorithms. Teams were free to use the sensors and combine their information in any way they wanted, though techniques such as moving average, median filtering, and least squares / Kalman Filtering (for those with appropriate background) was encouraged. During the project, students experienced a number of challenges in working with real sensors. Inertial sensors, especially cheap ones, are noisy, affected by vehicle vibration and by significant bias that changes as a function of temperature and other unobserved factors. Sonar altitude measurements have dropouts. Optical flow and vision systems can be significantly affected by the lighting condition and overflown floor optical texture as discussed above.

Another primary educational goal was to offer the students the opportunity to implement and tune a simple controller such as Proportional-Derivative-Integral (PID) to command the sUAS. For many students this is the first time they have implemented a controller on a real hardware platform even though they have designed and tested controllers in simulation previously. These simulations usually do not include issues such as numerical differentiation imprecision, signal delays, and controller saturation. All of these need to be addressed when implementing the feedback controller in a real-world system.

Finally, the design of logic to transition between different control modes allow students to put into practice concepts learned in the class such as finite state machines and integrated planning and control. Other concepts discussed in class such as data structures and algorithms are applied in a general form throughout project development. Students do not necessarily leave the course as experts in software development, but they leave with improved understanding of translating math methods and algorithms to embedded code, and actually making it work in an embedded flight system.

IV.A.3. Project #2 Description and Specific Educational Goals

Project #2 focused on flight planning and graph search rather than sensor data fusion. The primary project goal was on high-level planning and search, so a motion capture system provided accurate position and attitude information to de-emphasize work required for sensor data filtering and logic. An IMU data processing and validation subproject was still included to offer students exposure to IMU sensor data challenges, particularly when affixed to a vibrating frame. The right diagram of Figure 3 shows the payload configuration used for this project.

The primary project requirement was to develop software that enables the quadrotor to successfully perform the following in a final demonstration:

- Preflight Sequence:
  1. Record all motion capture “visit targets” in the flight area. These targets indicate waypoints the quadrotor must autonomous visit at an altitude of 1.5 meters.
  2. Assuming a travel distance cost metric, compute an optimal waypoint sequence to visit all waypoints using an A* search algorithm.
  3. Build a smooth reference trajectory to guide the quadrotor through the planned waypoint sequence stopping to hover at each waypoint for 5 s.
- Flight Sequence:
  1. The quadrotor is placed approximately at the (0,0) location in the flight area. The pilot manually commands the quadrotor to lift off and hover at an initial altitude of roughly 1.5 meters directly above the takeoff location.
  2. The pilot activates autonomous control but remains ready to take over manual control as needed. The quadrotor autonomously follows the planned waypoint sequence then returns to and hovers over the starting location.
  3. The pilot then de-activates the autopilot and lands the quadrotor manually.

The main educational objective of this project was to give students the experience of designing an integrated planning and control system for the quadrotor. While this was also a goal of project #1, the availability of high-quality data input allowed the students to dedicate significantly more time to this part of the project. This extra time resulting from the use of motion capture also allowed the inclusion of a more sophisticated flight planning task. Such problem allows students not only to put in practice the graph-search theory learned in class but also to apply it in a real-world autonomous flight system.

IV.B. Robotics Systems Laboratory (ROB 550) Projects

IV.B.1. Course Overview and Project General Educational Goals

Robotic Systems Laboratory (ROB 550) is a multidisciplinary laboratory course for graduate students with exposures to sensing, reasoning, and acting for physically embodied systems. It is an eight-hour-per-week lab course, with two hours dedicated to lectures and six to the laboratory. Unlike the Aerospace courses, the lab is emphasized which allows a sequence of long-term projects to be the focus of the course. Lecture topics are aimed at providing technical background for methods used in projects, e.g., manipulator and rigid-body mobile robot kinematics, sensor-based localization and mapping, task sequencing and motion planning, feedback control, communication protocols, and mechanical design. Laboratory-based group projects expose students to design, test, and integration of complex integrated mechanical, electrical, and software systems.

ROB 550 is one of the two mandatory courses for the Robotics Master’s and Doctorate degree program at the University of Michigan. Students often enter with different backgrounds and are more competent in one but not all of the following areas: software, electrical or mechanical hardware. While students with strong computer programming backgrounds tend to find the projects more tractable, students without strong programming backgrounds have repeatedly found success with course projects given hard work and regular interaction with teammates and course instructors. Teams switch between projects so students typically have the opportunity to work with a combination of strong and not-too-strong team members during the course. Regardless of student area of expertise, instructors encourage team members to engage on all part of the project regardless of their backgrounds to foster learning, and with close supervision (and with mature graduate students) this process tends to work reasonably well.

The below text describes two ROB 550 projects that used sUAS during the Fall of 2014 and 2015 respectively. These projects represent a subset of the three or four main projects assigned during each term of ROB 550. An important component of these projects is to require the (graduate) student teams to submit a final report in a professional conference paper format that offers experience in documenting experiments beyond simply demonstrating functionality. The two sUAS ROB 550 projects were unique in that they were composed of two main parts. The first involved the design-build-test (DBT) of a manipulator initially affixed to a hand-carried base for testing prior to transitioning to the sUAS as a flown payload. The second project phase was to integrate the payload with the Beaglebone embedded computer and subsequently with the quadrotor for full flight system integration and testing.

The manipulator DBT module fulfills a mechanical design objective of the ROB 550 course. While the instructors present to students some possible design solutions to the problem, students are free and encouraged to use their creativity to address the project's manipulation goals. This design process also provides an opportunity for students to work with Computer-Aided Design (CAD) software, laser cutters, and 3-dimensional printers. Such experience is extremely valuable since for some students with computer and electrical background this is the first time they work on a mechanical system design.

The second part of the project was dedicated to sensor data fusion as well as planning and control of the manipulator. Project #2 also included control of the sUAS. This part of the ROB 550 project shares similarities with the AEROSP 450/552 projects, but most of the focus in ROB 550 is directed to the payload and not sUAS control. Such difference in focus is a direct consequence of the different objectives of the courses. Figure 4 presents the payload configuration for both project realizations. The main difference for the ROB 550 projects when compared to Figure 3 is the presence of the manipulator servos controlled by the Beaglebone Board.

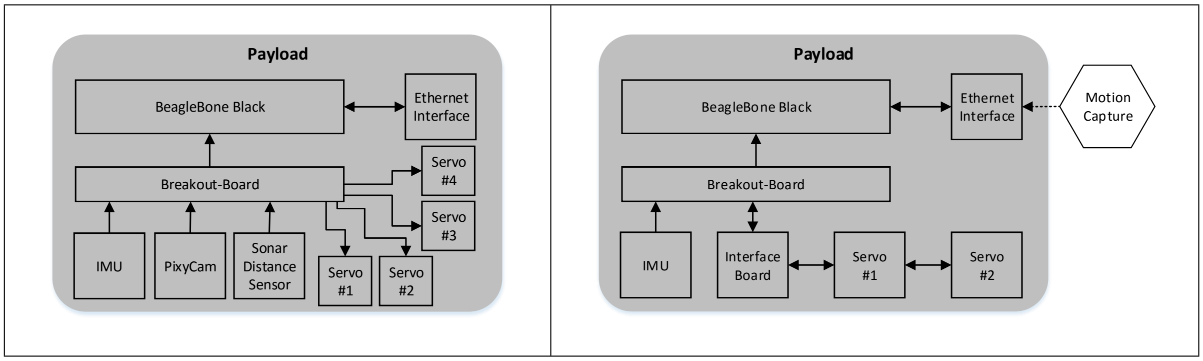

Figure 4. Quadrotor Payload Configuration for ROB 550 Projects. Left, Project #1. Right, Project #2

IV.B.2. Project #1 Description and Specific Educational Goals

This ROB 550 project's ultimate goal is to enable the quadrotor to semi-autonomously collect a ping-pong ball affixed to a small structure and deliver it inside a goal made of an empty cylindrical container. Throughout this flight, the sUAS is directly piloted by one member of the group. The BeagleBone computer is used to provide the required semi-autonomous behavior described below:

1. Receive from the ground station an authorization to try to retrieve the ball;
2. Once authorized, estimate the relative position of the ball using the Pixy and IMU feedback;
3. Once inside the reachable range of the manipulator (e.g., delta arm) affixed to the quadrotor, it should automatically command ball retrieval;
4. Receive an authorization from the ground station to either re-attempt retrieval (back to step 2) or to deliver the ball;
5. Once authorized, the relative position of the goal is estimated using the Pixy and IMU feedback;
6. When the dropoff canister is within the reachable range of the grasping manipulator (delta arm), it should automatically command ball release into the container.

The payload configuration available for all groups is presented in the left side of Figure 4.

To provide a way for students to test their manipulator and algorithms, stands were built to simulate the bottom plate and landing gear of the quadrotor. This mounting system assures the manipulator will be solidly mounted and will be commanded to remain clear of quadrotor structure during operation. Figure 5 presents a design of the stands and a schematic of the configuration for ball catching used during the competition. The first part of the project, mechanical DBT, involved designing the manipulator including space for the BeagleBone and Sensors as well as servos. This manipulator was attached to the sUAS or stand by four bolts in a specific pattern. Although not mandatory, students were encouraged to use a Delta parallel manipulator to achieve a precision pick-and-place capability with minimal CG shift during operation. All groups adopted a Delta parallel manipulator with designs differing on the actual mechanism to retrieve and release the ball located at the end effector. Additional design requirements included the ability to safely stow the manipulator to allow a safe landing. Students were limited to designs with at most four servos. Most of the manipulators were manufactured with wood using a laser cutter, while a few parts were 3D printed. Figure 6 presents an example of a student design for the upper plate of the Delta parallel platform. The links shown are connected to an end effector shown in Figure 7 designed to capture the ball with the use of an additional servo.

Once the manipulator was built, the next part of the project involved implementing and testing the software to read the sensor data and command the manipulator. Regarding sensor data, teams were free to use the Pixy, IMU, and Sonar Distance Sensor in any way desired. Groups were also free to design at most five Pixy color targets and allocate them in any pattern around the ping-pong ball support. Most of the teams concentrated their efforts on the Pixy considering the restricted time frame and general difficulties with IMU and sonar data. Noise is one of the examples of challenges that were already discussed in Section IV.A but also observed in this ROB 550 project. This experience with real sensors and design choices offered substantial learning experiences.

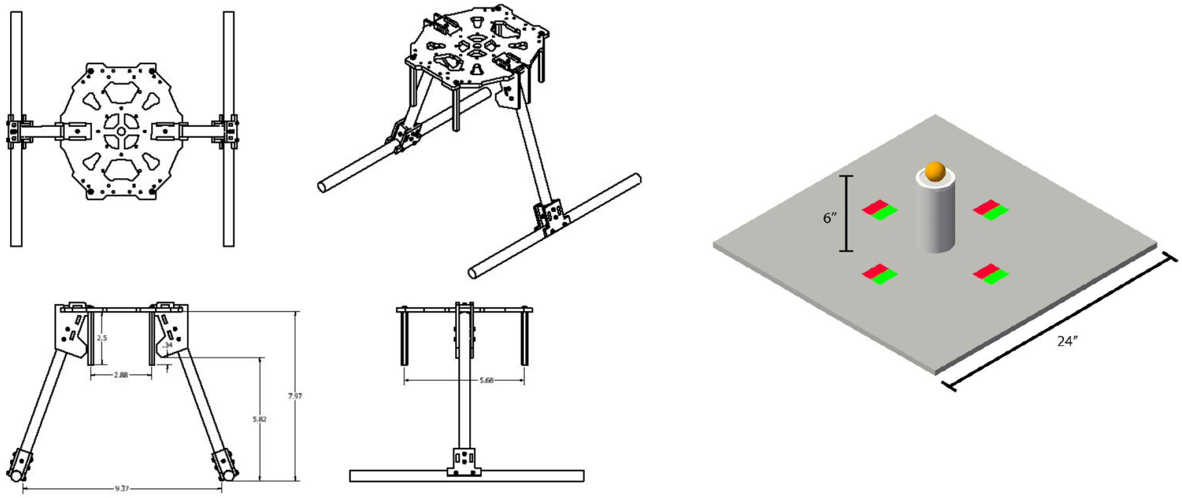

Figure 5. ROB 550 Project #1 Platform for Manipulator Tests and Arrangement for Ball Retrieval

Students were provided with generic software to position a Delta arm end effector, but this code had to be integrated with the quadrotor code and also modified to account for manipulator dimensions, and mounting location. Baseline code availability, like with the AEROSP projects, allowed the groups to concentrate on the logic to transform the position error to end effector position guidance commands as well as implementing an algorithm to control the custom mechanism (for grasping) each student designed for their delta arm end-effector. Note that it was not mandatory to have an actuated end effector. In fact, some groups only used the three servos required for the Delta platform. The educational objective was to ask the students to implement finite-state machines or similar techniques to deal with the required level of the autonomous behavior of the project.

After manipulator, grasping, and manipulator control software were validated with the mounting stands, the last step was integration with the quadrotor for flight testing. sUAS deployment presented all the challenges discussed in the context of the AEROSP projects. Even with manual flight, students were also exposed to challenges in quadrotor downwash flow disturbances and inaccuracy in position maintenance over time limiting the reliability of target grasping success. As an additional benefit, students gained experience flying quadrotors in a "test flight" mode given that control software didn't always work as planned.

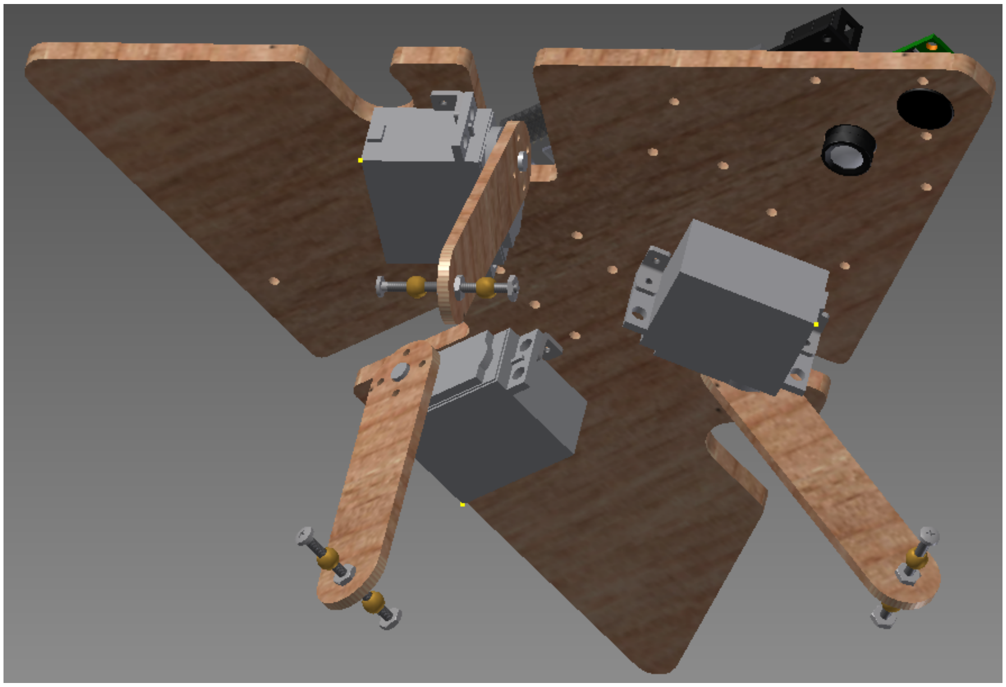

Figure 6. ROB 550 Project #1 Example of Manipulator Top Plate Design

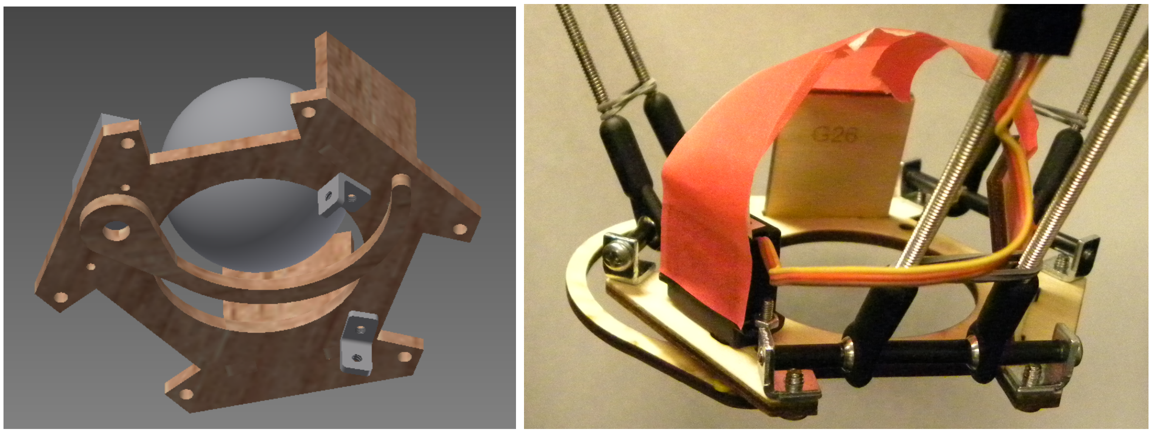

Figure 7. ROB 550 Project #1 Example of End Effector Design and Construction

IV.B.3. Project #2 Description and Specific Educational Goals

This project's ultimate goal was to enable the quadrotor to autonomously perch, passively maintain the perch, and takeoff from the perching bar. Only high-level commands from a ground station operator were allowed, i.e., initiate the perch, interrupt/stop the perch sequence. Additionally, a pilot supervised the autonomous flight to override when necessary as well as providing manual takeoff and landing inputs. This second ROB 550 project required students to write an algorithm to control both the sUAS and the designed mechanism to allow perching. As compensation, the sensing part of the project was facilitated since the motion capture system was available for position and attitude data. This payload architecture is presented in the right illustration of Figure 4.

The first part of the project included the DBT of a gripper responsible for grasping the bar. Groups were required to design a gripper with at most one servo. For actual flights, the teams could duplicate the gripper to provide longitudinal stability and additional grasping torque. The DBT was considered one full project for the course with dedicated competition and report. The competition involved testing the maximum torque the gripper could generate before it slipped on a grasped bar of the same diameter as the sUAS perching bar. Students were encouraged to design grippers that could be produced using 3D printers with the objective to give them experience with this emerging prototyping and manufacturing technology. Figure 8 presents a gripper designed by one of the groups and the actual mounting of two of the grippers in the platform that connects to the sUAS. This design uses the same mechanical design principle as handcuffs to grasp the perching bar.

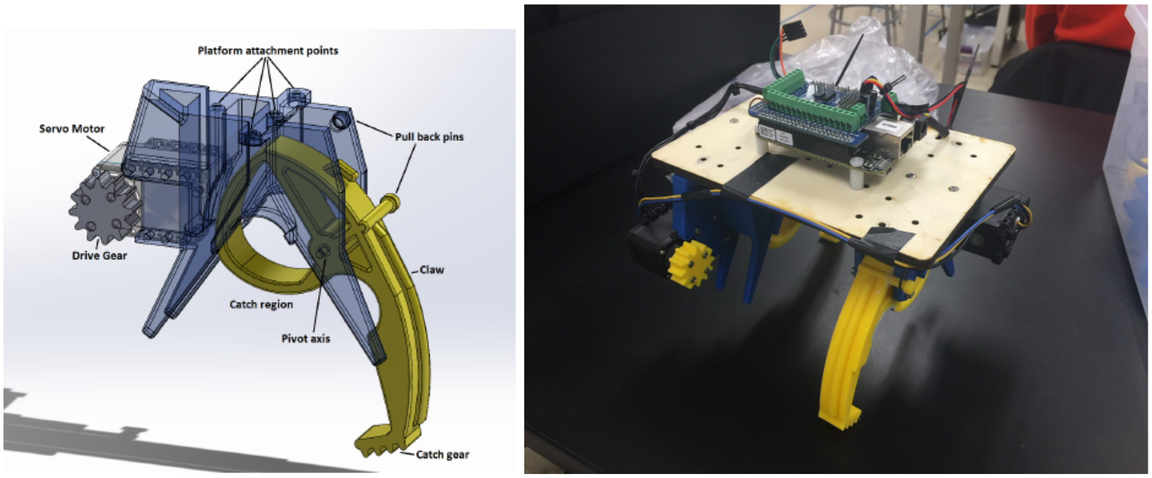

Figure 8. ROB 550 Project #2 Example of Gripping Design and Mounting on the Platform

After the competition and design reports, groups were reorganized for the flight project. Each flight group formed by students from different DBT teams were allowed to use any of the grippers from one team member's gripper design group. This rule was made to avoid over-penalizing any group that ended with a gripper design that performed poorly on the gripping torque competition. The use of the motion capture system allowed students to focus on the development of a flight controller similar to the AEROSP 552 Project #2. The difference was that their algorithm must add the state machine and guidance/control commands to perform the actual perching and releasing sequence. Active control of the sUAS using the propeller thrust after perching was not necessary because the best gripper designs were able to provide sufficient torque to maintain the sUAS perched with an attitude tolerance.

V. Safety Measures

The projects presented in this paper were developed using a sUAS with 12-inch propellers flying indoors in a building atrium. This enabled all-weather flight and avoided issues with the Section 333 policy[j] in place during the time of these projects. While a pilot could override the autonomous student-written code with remote control, fast quadrotor dynamics could restrict the pilot's ability to recover before a crash. We developed a protocol that enabled each team to successfully achieve project goals while minimizing risk primarily to people and secondarily to the quadrotors so we could re-use them over multiple terms/years. While there were events of sUAS loss of control, all operations during four course projects were consistently safe, and only one quadrotor was damaged beyond repair during the first term of use (by not responding to override commands and wrapping itself in the tethers).

The first risk mitigation employed is the use of a quadrotor with ducted propellers. More than providing protection to the propeller itself, such ducts act as a barrier to minimize the chance that a propeller will "slice" a user or observer. This barrier is of particular importance when a student or instructor is manipulating the quadrotor itself. Examples of such scenarios are changing batteries, connecting ethernet cables, and resetting the onboard computer.

A second safety measure was the progressive testing/training and operation approach enforced for all pilots, manipulators, and software. All students first experimented with controlling small, typical toy-like sUAS. Only students that could demonstrate adequate quadrotor control were allowed to pilot the larger sUAS. For the ROB 550 projects, groups demonstrated safe and reliable manipulator control in the lab environment before being allowed to test the manipulator on the sUAS. An example of such "safe" control

[j] FAA Modernization and Reform Act of 2012. Pub. L. 112-95 126 Stat. 11-155

was an ability to provide an override command able to retract the Delta arm to its stowed position in the case of project #1 to allow for a safe landing. In the case of software, an example of this progressive approach was that student groups were required to demonstrate flight control with a simple hover over the present position before transitioning between waypoints autonomously. This allowed an observer to immediately judge whether the controller was functioning, and tether operators could input “disturbances” to assess robustness before the quadrotor manipulator interacted with its grasping target.

All other employed risk mitigations were related to the atrium flight arena arrangement shown in Figure 9. These measures start with a restriction on flight operation hours. Since the building is used for different classes throughout the day, flight operations were restricted to the evenings. Even if such restriction poses a burden to all involved in the activities, it significantly reduces risk exposure to passersby. Also to maintain people not involved in the operations at a safe distance from the sUAS, portable chains were used to delimit the flight arena as shown in Figure 9. Anyone that did not need to stay close to the sUAS was required to be outside the arena, even the ground station operator.

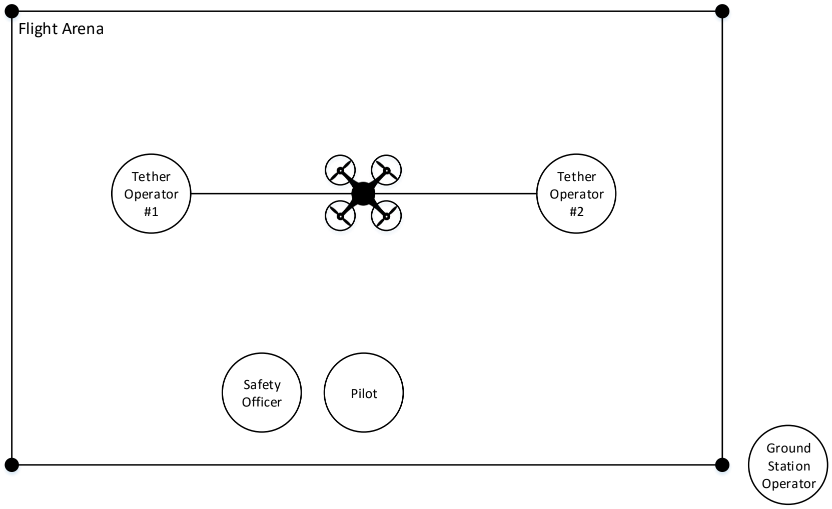

Figure 9. Flight Arena Standard Configuration for Risk Mitigation

Probably the most effective safety measure adopted was the use of two tethers to restrict the sUAS movement in case of undesired behavior. Such tethers were controlled by two operators on opposite sides of the arena who could restrict the quadrotor position and attitude in a manner that offered time to the pilot to intervene or turn off the motors in the worse case. Tethers have the major advantage over any other system of being entirely independent of sUAS sensors, actuators or software. They are effective even if the undesired behavior is due to the pilot improper commands through the remote control. The ability of tether operators to restrain motion gives tether systems a distinct advantage over flight in a netted arena for our manipulation-oriented projects that do not require a large flight volume. While the net will contain the quadrotor, the sUAS and its payload can be severely damaged during loss of control events when untethered. While this may be a valuable lesson, it also would result in costly and time-intensive repairs throughout each project.

While a geofencing system[5, 10] could be effective in avoiding loss of control or range departure conditions, its implementation needs to be completely independent to prevent inappropriate student code interaction that could render geofencing ineffective. Geofencing also needs to be robust to sensor failure. In fact, on more than one occasion the sUAS had to be held with the tethers as a result of the motion capture system data stoppage due to the sUAS move to a position outside its coverage area.

As a final safety measure, all flights were conducted with an instructor (typically one of the last two authors) acting as the Safety Officer. This officer would stay at the side of the pilot overseeing the operation and giving instructions to all involved, especially in the case of undesired sUAS behaviors. In extreme cases, the officer would take the remote control from the pilot to directly command the sUAS.

VI. Discussion

PBL Advantages and Disadvantages

Project-based-learning (PBL) is an effective tool to increase students' interest in a subject, foster creativity and teamwork, and provide real-world realizations of concepts learned otherwise in theoretical and simulation contexts. The advantages of providing sUAS project exposures also come with challenges for both students and instructors. Probably the greatest challenge is the significant time commitment required for both instructors and students. Projects that foster creativity and involve hardware are prone to overhead required for system and software debugging, repairs, and adaptation of operational protocols given the inevitable non-ideal functionality of components and the integrated system. While motivated students can handle and even thrive with extensive hours in the laboratories, it is not beneficial to their overall education to over-commit their time to one course. Time estimation and management will always be an issue but can be tweaked by simplifying modules and capabilities, e.g., introducing ideal motion capture data or a validated flight controller into the system. The goal is then to make sure students realize they are being exposed to certain challenges and that this doesn't mean other capabilities are easy.

Limited Flight Test Time Provides a Valuable Experience

The sUAS used in our course projects could only be flown during reserved evening intervals. Restriction on available flight time is a direct consequence of the amount of work necessary to prepare the flight arena and associated safety features. While such difficulties come from the current course restrictions, limited availability of testing time is common in projects students will face in real life working as engineers. During other ROB 550 projects entirely performed in the laboratory, it was common to see students testing several approaches and parameters of their sensor data processing software and controllers without devoting time to carefully analyzing the outputs of the previous implementation and designing a new approach to be tried. The use of sUAS forced student groups to implement this feedback in their design development. In fact, successful groups developed models based on output data to support their control system development. The test time limitation also encouraged groups to develop software that could be rapidly updated during testing.

Real-World Sensing and Control Experience is Valuable to Student Education

It is interesting to observe how students with excellent theoretical background can struggle with implementing simple controllers in a laboratory setting. These difficulties come from "real-world" sensor behaviors that are usually not expected and can be frustrating including data dropouts, noise and the necessity to deal with data arriving at different rates and formats. Another challenge is in realizing a software implementation of an actual embedded real-time controller. This aspect is challenging especially to students without good programming background because it is not usually covered in control courses. Students are trained designing and testing controllers using advanced simulation tools such as Simulink or a mathematical system representation. Our projects are helpful in expanding this vision to include data acquisition and filtering, controller execution frequency and delays, environment disturbances, and other system behaviors not modeled during design and analysis.

Why not Autocode Control Code?

All projects presented in this paper are developed by students using C, C++ and Python embedded in the sUAS of AEROSP 450, 552, and ROB 550. Certainly the choice of language will evolve over the decades along with processor architectures, operating systems, development environments, and hardware interfaces. However, basic handling of "bits and bytes" is essential to understand along with module execution frequencies and delays, abstraction and precision of different data representations, communication requirements and overhead, etc. Automatic coding is increasingly popular in the industry to support fast prototyping and minimize the potential for low-level coding errors. Considering that academia is also making use of such automated coding tools for both research and teaching, one could question the necessity of students to learn how to program the actual software, e.g., in C. Even though auto-coding tools are available, our goal is education. Students must understand the foundation of how such tools work to understand what might go wrong when controllers are deployed. The University of Michigan provides a specific laboratory course for teaching undergraduate students embedded feedback control.[14] In this course C code and autocode generated by the Matlab® allow students to gain experience in both and learning each other advantage. Our courses instead combine Python and C/C++ to demonstrate how multiple languages can similarly strike an appropriate balance between manual coding and a scripting language that is slower but more compact and easier to manage/debug.

Student Evaluation Based on Conference Paper Format Reports

All projects presented in this paper are evaluated based on final reports students are asked to write and submit as in teams. Students know that for their final grade it is more important to write a report that explains "why" something did or did not work rather than simply demonstrating that everything works. The competition itself is used only for motivation and a small grade bonus for winners. This approach worked satisfactorily as students know they can get a good grade even if their project was unsuccessful during the competition. Further, many graduate students are exposed to paper-writing for experimental systems for the first time given that their "normal" research often focuses on theoretical results supported by simulation.

Implemented Safety Measures Were Successful

The tether and partitioning safety systems were capable of restraining the quadrotors as well as preventing students from entering the flight area. The movement restriction offered by tethers was acceptable for all our projects to-date, because accuracy and precision were targeted over flight volume traversed. The use of a dedicated space for flight testing may allow flights with fewer safety restrictions which will reduce setup time, but our setup has served as an exemplary model for ongoing campus-wide quadrotor flight safety protocols.

Numerous student groups within and outside engineering are now venturing into multicopter (sUAS) activities both indoors and outdoors. For our indoor tests we dealt with safety from a facility, test setup, and personnel perspective, but outdoor flight projects is beginning to see closer oversight by university safety committees. At the University of Michigan, the executive officers have banned sUAS flight over campus except for groups receiving specific operational approval. Meanwhile the FAA has authorized student groups to fly without formal training or approval as they are viewed as close to "hobbyists", while others can operate under Part 107. All our experiences indicate that students going out with off-the-shelf quadrotors can easily get into trouble, both with respect to flying away or out of control and operating dangerously close to manned aircraft, e.g., our hospital heliport adjacent to an open field popular for hobby flight. The University of Michigan is in the process of developing validation and approval processes that will apply to all university entities whether they are operating as a student team, course project team, or research group. This process will involve training for entities that do not pass an initial validation activity. In this manner we can consistently operate safely and be "good neighbors" to other planes in the sky and entities on the ground.

VII. Conclusions

This paper has presented a series of four different sUAS course projects that expose students to real-world integrated software, sensing, and decision-making required for autonomous operation of an sUAS or other robotic platform. These projects are significantly different than usual design-build-and-fly projects used in capstone Aerospace engineering courses in their emphasis on autonomy and decision-making at multiple levels. Robotics course projects asked students to design-build-test manipulators that are attached to the sUAS for a mission goal such as object pick-and-place or quadrotor perching. Aerospace course projects focus on state estimation using sensor data fusion along with integrated planning and control. The projects described in this paper show the potential for sUAS to be utilized as a real-world Aerospace and Robotics interdisicplinary educational platform. We look forward to the continued deployment of sUAS for new educational projects as well as commercial and civil applications in future years.

Acknowledgement

The authors would like to thank the students that provided the design figures used in this paper, and the University of Michigan for providing resources to purchase sUAS and support equipment for our project-based laboratories. The first author is supported in part by CAPES-Brazil through the Science without Borders program, process number BEX 19130/12-0.

References

[1] Dym, C. L., Agogino, A. M., Eris, O., Frey, D. D., and Leifer, L. J., "Engineering Design Thinking, Teaching, and Learning," *Journal of Engineering Education*, Vol. 94, No. 1, 2005, pp. 103–120, doi: 10.1002/j.2168-9830.2005.tb00832.x.

[2] Froyd, J. E., Wankat, P. C., and Smith, K. A., "Five Major Shifts in 100 Years of Engineering Education," *Proceedings of the IEEE*, Vol. 100, No. Special Centennial Issue, May 2012, pp. 1344–1360, doi: 10.1109/JPROC.2012.2190167.

[3] Morris, L. V., "On or Coming to your Campus Soon: Drones," *Innovative Higher Education*, Vol. 40, No. 3, 2015, pp. 187–188, doi: 10.1007/s10755-015-9323-x.

[4] Antsaklis, P., Basar, T., deCarlo, R., McClamroch, N. H., Spong, M., and Yurkovich, S., "Report on the NSF/CSS Workshop on New Directions in Control Engineering Education," *IEEE Control Systems*, Vol. 19, No. 5, Oct 1999, pp. 53–58, doi: 10.1109/MCS.1999.793442.

[5] Huggard, M. and Mc Goldrick, C., "Droning On: Reflections on Integrating UAV Technology into a Computer Engineering Design Laboratory," *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE '16, ACM, New York, NY, USA, 2016, pp. 504–509, doi: 10.1145/2839509.2844650.

[6] Lawlor, O., Bogosyan, S., Vural, Y., Thompson, I., Moss, M., and Gokasan, M., "AERO-beam: An Open-Architecture Test-Bed for Research and Education in Cyber-Physical Systems," *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, 11 2015, pp. 005080–005086, doi: 10.1109/IECON.2015.7392897.

[7] Krajník, T., Vonásek, V., Fišer, D., and Faigl, J., "AR-Drone as a Platform for Robotic Research and Education," *Research and Education in Robotics - EUROBOT 2011: International Conference, Prague, Czech Republic, June 15-17, 2011. Proceedings*, edited by D. Obdržálek and A. Gottscheber, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 172–186, ISBN: 978-3-642-21975-7, doi: 10.1007/978-3-642-21975-7_16.

[8] Nitschke, C., Minami, Y., Hiromoto, M., Ohshima, H., and Sato, T., "A Quadrocopter Automatic Control Contest as an Example of Interdisciplinary Design Education," *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, 2014, pp. 678–685, doi: 10.1109/ICCAS.2014.6987866.

[9] Winterfeldt, G. and Hahne, C., "Controlling Quad-Copters: a Project-Based Approach in the Teaching of Application Design," *2014 IEEE Global Engineering Education Conference (EDUCON)*, 4 2014, pp. 961–968, doi: 10.1109/EDUCON.2014.6826216.

[10] Stevens, M., Coloe, B., and Atkins, E., "Platform-Independent Geofencing for Low Altitude UAS Operations," *Proceedings of the 15th Aviation Technology Integration, and Operations Conference, Dallas, TX, USA*, 2015, AIAA 2015-3329, doi: 10.2514/6.2015-3329.

[11] Castillo, P., Lozano, R., and Dzul, A., "Stabilization of a mini rotorcraft with four rotors," *IEEE Control Systems*, Vol. 25, No. 6, Dec 2005, pp. 45–55, doi: 10.1109/MCS.2005.1550152.

[12] "LSM9DS0 - Datasheet," Available at: `https://cdn-shop.adafruit.com/datasheets/LSM9DS0.pdf`, accessed June 6, 2016, August 2013.

[13] Natural Point Inc., "NatNet API, User's Guide," Available at: `http://optitrack.com/static/documents/NatNet%202.9.0%20API%20User%20Guide.pdf`, accessed June 6, 2016, 2015.

[14] Freudenberg, J. S. and Krogh, B. H., "Industry needs for embedded control education," *Proceedings of the 2005, American Control Conference, 2005.*, June 2005, pp. 4543–4550 vol. 7, doi: 10.1109/ACC.2005.1470712.