10.2514/6.2015-3329

AIAA AVIATION Forum
22-26 June 2015, Dallas, TX
15th AIAA Aviation Technology, Integration, and Operations Conference

Platform-Independent Geofencing for Low Altitude UAS Operations

Mia N. Stevens* and Brandon T. Coloe† and Ella M. Atkins‡
University of Michigan, Ann Arbor, Michigan, 48109, U.S.A.

Small unmanned aircraft systems (UAS) are rapidly proliferating for a variety of commercial and civil applications. Regulators cite detect-and-avoid of manned aircraft as the most significant challenge to integrating small UAS in the National Airspace System. However, most emerging small UAS use cases call for flight within immediate reach of the ground, airspace manned aircraft rarely occupy except on approach or departure. This paper presents a low-altitude geofencing capability for small UAS as a safe alternative to detect-and-avoid. The low-altitude small UAS can be flown manually or through any commercially-available autopilot with the independent geofence unit as a low-cost add-on. The geofencing unit passively monitors the flight, activating as a guidance system override on approach to an altitude or lateral fence boundary. Once active, the geofence turns the UAS back from the border then returns to its monitoring mode. This paper implements geofencing on a small quadrotor flown indoors to eliminate regulatory concerns. While GPS would be a primary means of navigating outdoors, a dual camera, inertial measurement unit, and height sensor package is used to geofence indoors. A Kalman filter integrates sensor data, and a geofence boundary controller overrides the nominal pilot/autopilot guidance inputs to drive the system back within the fenced region. Results show that the fence can be maintained, and that the control authority switch can be done in a stable manner so long as the operator understands when the fence activates and deactivates. Work is ongoing to flight test geofencing outdoors with both multicopter and fixed-wing platforms.

I. Introduction

Unmanned Aircraft Systems (UAS) have the potential to revolutionize our ability to carry payloads ranging from consumer packages and cameras to agricultural chemicals at low energy and cost due to platform size, technology, and versatility. No human pilot will be carried onboard so a small UAS only poses risk of harm to people and property on the ground and in any nearby manned aircraft. Versatile designs such as the multicopter and convertible fixed-wing/VTOL will offer a level of maneuverability and energy efficiency not before realized in aviation.

High-altitude large UAS have been managed with substantial overhead; once approved for flight they typically require a ground operations crew size comparable to if not greater than that required for a manned aircraft. These operations are costly, thus are not likely to grow more rapidly than commercial transport operations. The small, low-altitude UAS sector on the other hand is generally viewed as having tremendous growth potential for public agencies such as law enforcement and private companies supporting package delivery, entertainment, agriculture, newsgathering, etc. Small UAS are also valuable for education and research. Appropriate rules must still be devised, with hobby status, exemption, and self-certification currently required for outdoor flight approval. Policymakers cite detect-and-avoid as the most major concern facing UAS integration in the NAS (National Airspace System). The authors agree: system-wide detect-and-avoid is clearly needed for operation in the airspace typically populated by manned aircraft, especially enroute and terminal area airspace occupied by commercial transport. It is, however, much less clear that manned aircraft detect-and-avoid is a significant concern for flight in the airspace within immediate reaches of the

*Graduate Student, Robotics Program, University of Michigan, Ann Arbor, MI.
†Graduate Student, Aerospace Engineering Department, University of Michigan, Ann Arbor, MI.
‡Associate Professor, Aerospace Engineering Department, University of Michigan, Ann Arbor, MI, Associate Fellow.

Earth's surface, e.g., within a range of zero to 300–500 feet AGL (above ground level). In this immediate reaches airspace, the greatest risk of harm to people and property is posed by a small UAS crashing into the ground or otherwise harmfully interacting with people/property on the ground through harassment, annoyance, or loss of privacy. Risk of collision with manned aircraft can be adequately mitigated if each small UAS remains in immediate reaches airspace away from airport arrival and departure paths. Further, risk of harm and annoyance to people on the ground can be mitigated if small UAS remains immediately over the operator's property or other area approved for flight by the municipality, etc.

This paper proposes electronic geofencing as an alternative to detect and avoid for small UAS flying within immediate reaches of the Earth's surface. The goal of this low-altitude geofencing capability is to guarantee that an aircraft will remain within its designated lateral and vertical "flight box", which in turn means this UAS poses little to no risk to people or property. With a geofence, an aircraft arrests a climb or descent to ensure altitude constraints are respected; the geofence automation then turns back to a central loiter point or slows to a hover (when possible) to ensure lateral fence boundaries are respected. So long as other aircraft remain clear of the low-altitude geofenced airspace, the geofenced UAS can then operate in accordance with its designated mission, e.g., agriculture, pipeline inspection, etc., and the geofence would ensure that any errant behaviors/inputs that might lead to range departure do not actually cause the craft to leave this range. Low-altitude geofencing will not introduce a regulatory nightmare for either public or private operations, in fact quite the opposite. Fixed-wing manned aircraft are not free (or safe) to fly below 500' AGL except on final approach or initial climbout from a runway or with explicit landowner permission (e.g., crop dusting). Helicopters may present a challenge to immediate reaches flight in urban areas, but in rural or suburban areas even helicopters need not routinely dip below 500' AGL unless they have landowner approval or are used in rare circumstances such as medical evacuation.

The geofenced UAS operating exclusively in immediate reaches airspace over a single landowner's property therefore can safely fly now over rural areas without any regulatory oversight or day-to-day interaction with air traffic control. Geofencing will also prove useful in urban areas, e.g., to prevent a drone from flying into the White House lawn, ensure a hobbyist drone remains over an approved flight area such as a neighborhood "drone park", or assure commercial UAS only approach the surface at "drone package delivery boxes." Urban areas may require more coordination to ensure the suite of approved operations are compatible with other municipality and airspace activities. The majority of small UAS are currently flown without specific flight plans. Whether used for hobby or commercial purposes, small UAS behaviors are highly dependent on the experience level of the pilot, accuracy of entered waypoints, steady wind and gusts, and equipment functionality, A geofence would ensure operations are confined to the approved operating volume. The goal of the geofence is to serve as a low-cost safety-preserving add-on or applique to existing small UAS pilot and autopilot control systems. By separating the geofence from the nominal controller, a layer of redundancy not typically available for small UAS is provided.

Below, background in geofencing is provided to connect our concept with Aerospace and other fields that have already exploited this concept. Next, our approach to geofencing is presented. The paper then focuses on our initial "indoor quadrotor flight" application. Sensor and computing systems are summarized. A Kalman filter used to integrate noisy and sporadically-available sensor data is presented and evaluated. The indoor geofence guidance and control system is presented and evaluated over a series of geofence activation time response trends. Performance of the geofence is then analyzed, followed by a discussion of findings and recommendations for future work.

II. Geofencing Background

The Global Positioning System (GPS) provides a low-cost, reliable method for rapidly updating a device's position outdoors. Early handheld GPS units provided *geolocation* capability for people traveling on foot and in vehicles. Miniature GPS receivers are now carried in most every cell phone and most every vehicle-based navigation system. Once users gained trust in GPS for geolocation, a number of *geotracking* applications emerged. Geotracking has enabled fleet managers to better schedule pickups and deliveries and adapt to transit delays.[1] Geotracking also provides a suite of location-based services to cell phone users. Geotracking supports data collection using "people as sensors"[2] and geolocation-based crowd sourcing,[3] and offers researchers a wealth of opportunities for transportation performance analysis.[4] In open environments such

as those to be occupied by small UAS operating outdoors [a], geotracking systems have proven consistently reliable. While jamming and spoofing potential must be considered, other GPS applications such as manned aircraft navigation present greater risk from loss of GPS than will a low-altitude small UAS geofence.

Geotracking has led to applications in a wide variety of domains, e.g., tracking of impaired patients near health care facilities[5] and geo-tagging data from mobile devices to facilitate cloud-based aggregation and processing.[6] In aviation, GPS has become a critical source of data for navigation,[7,8] and is central to Automatic Dependence Surveillance - Broadcast (ADS-B) systems that offer real-time flight tracking information to crews and air traffic control and that are considered a foundation for emerging airborne detect-and-avoid systems.[9,10,11] ADS-B data is currently advisory, providing tracking data but not yet linked to specific detect-and-avoid automation.

While Merriam-Webster (online) does not yet offer a definition for the term *geofence*, Wikipedia (http://en.wikipedia.org/wiki/Geo-fence) defines a geofence as "a virtual perimeter for a real-world geographic area" and cites applications where notifications are generated when children, wildlife, firearms, and vehicles "travel unexpectedly". To-date most geofencing applications have centered around designating areas to "fence" and generating notifications/warnings when area boundaries are not respected. Long before computers and automation, real "fences" were built not just to notify but to physically turn domestic animals and potential trespassers away from property boundaries. Analogously, the next step for the virtual geofence is to "close the loop" with automation that can guarantee the geofenced entity will remain in its designated operating range or box despite the absence of real physical fencing barriers. Then, just as we trust a physical fence to keep us safe from wandering or dangerous animals and private from people wandering through our yards, the geofence can ensure our ground or flight vehicles will also be "turned back" from range boundaries.

The concept of geofencing to keep a small UAS in a designated test range has been introduced previously[12,13,14,15] but to-date has not been rigorously developed or flight-tested as an automation aid. Basic geofencing or electronic leashing functionality already exists in some off-the-shelf autopilot systems, such as the open-source Ardupilot (http://ardupilot.com/) with upgraded hardware such as Pixhawk[16] (http://plane.ardupilot.com/wiki/common-autopilots/common-pixhawk-overview/) and APM (http://plane.ardupilot.com/wiki/common-autopilots/common-apm25-and-26-overview/). Open-source geofencing solutions offer an excellent means to explore geofencing but raise a number of questions related to redundancy, operator input, and software pedigree. First, current autopilot units offer geofencing as an option rather than a requirement: what if the operator chooses to never activate a geofence? Second, the geofence is completely defined by the user. There is no hard altitude limit imposed, nor are coordinates double-checked with a database for validity. Finally, products like Ardupilot are open-source, which is great for education, research, and community-based development, but the community may unknowingly integrate a modification that intentionally or unintentionally defeats the geofencing capability. For these reasons, we take the approach of using a separate geofencing unit with independent sensing and software.

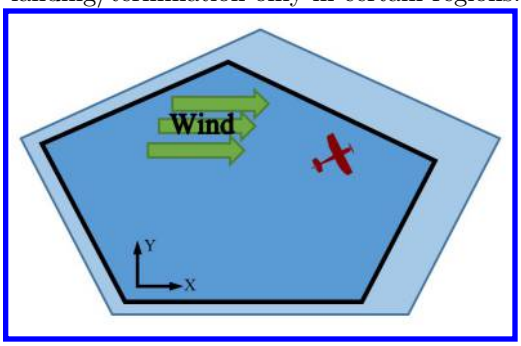
Finally, public acceptance and regulatory issues must be considered independent of technology readiness. Will a geofencing capability gain traction in the community? How do we ensure policy makers consider geofencing carefully rather than continuing to focus strictly on detect-and-avoid? Experts are beginning to recognize the technology and policy needs of *immediate reaches airspace* small UAS operations distinct from those for manned aircraft transiting at higher altitudes .[17,18] The geofencing capability introduced in this paper offers a means to safely segregate this new small UAS airspace layer from the NAS governed under existing the existing Federal Aviation Regulations.

III. Geofencing Concept

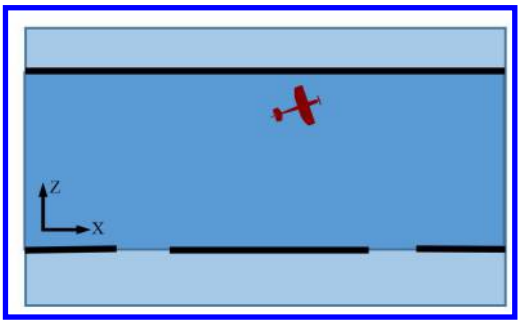
As outlined above, geofence automation must confine a UAS to a specified flight area. A geofence guidance system maintains an ordered list of GPS coordinates defining the outer edge of the flight area, shown as the outer-most light-blue corners of Figure 1(a). A maximum altitude (flight ceiling) constraint is shown as the top-most light-blue corners of Figure 1(b), while an altitude floor is depicted as the bottom-most light-blue corners of Figure 1(b). The floor may extend to the ground, indicating the UAS is not constrained by a geofence floor, instead the nominal guidance system plus pilot can descend to land anywhere within the geofenced operating area. A nominal outdoor geofence system might rely on a GPS unit, an IMU, and an off-the-shelf flight control system. The geofence guidance system anticipates breaches of the fenced area by

[a] Urban canyons are not discussed here as buildings provide actual walls to avoid, and geofence boundaries would lie above and beyond urban canyon environments.

considering the turning flight dynamics of the aircraft and the current (steady or average) wind conditions, as shown by the dark blue areas in Figure 1(a) and Figure 1(b) that shift the geofence flight boundaries accordingly. In Figure 1(b), the black lines show the altitude limits at which the aircraft would be instructed to climb/descend or level off to avoid an altitude ceiling or floor breach. The breaks in the floor fence designate safe descent and landing zones for the aircraft in regions where only certain landing sites are safe. In rural applications over a single property with no structures, a geofence would typically have no floor. In urban areas or approved flights over multiple properties, the geofence may be approved to descend to landing/termination only in certain regions.



(a) Planar view of a geofence shifted by a measured average wind.



(b) Side view of geofence altitude boundaries with takeoff/landing openings in floor.

Figure 1. Planar and side views of example geofence.

A low-cost geofence guidance unit might serve as a black box attached to the aircraft's nominal flight controller and power source. The geofence would pass pilot/autopiot guidance our "outer loop" control commands through to the aircraft except in cases where the aircraft is in danger of breaching the fence. When an imminent fence breach is detected, the nominal pilot/autopilot guidance commands will be overridden to turn the UAS back from the fence boundary. Once the geofence recovery criteria are met, e.g., vehicle is at a central recovery waypoint and/or sufficiently far from the fence boundary, the geofence returns control to the aircraft's default controller. If a return to the center of the flight zone is not possible or if there is an error detected with the aircrafts default controller, the geofence system connected to an appropriately-equipped autopilot would automatically land the aircraft in as controlled a manner as possible or flight terminate in a safe zone designated by breaks in the geofence floor.

Due to policy and travel time/cost considerations, we were restricted to conduct all flight testing for this paper indoors. While the geofence protocol and sensor package were highly modified, the concept of a dual-mode monitor/override geofence capability was still demonstrated. Details of the quadrotor and its geofencing prototype are thus the focus of our case study and flight tests as presented below.

IV. Indoor Quadrotor Case Study

This paper focuses on the implementation and evaluation of an indoor geofence system for a multicopter. Figure 2 shows the multicopter flight test platform utilized for geofence development and experimental validation. The quadrotor hardware, sensors, and computing architecture is summarized below.

A. The SkySpecs Quadrotor

The SkySpecs (www.skyspecs.com) quadrotor platform was used for flight testing. It is equipped with a commercial off-the-shelf DJI Naza-M autopilot (http://www.dji.com/product/naza-m). Our geofence prototype was implemented on a BeagleBone Black processor (http://beagleboard.org/) and connected through custom electronics developed by SkySpecs that supported pilot command pass-through or override by the BeagleBone geofence. The quadrotor flew tethered in the University of Michigan's FXB building atrium to ensure flight safety. The geofencing goal was to enforce maximum altitude and lateral boundary constraints for the multicopter platform flown indoors. Geofence maximum altitude was defined with respect to the building floor, and lateral boundaries were defined relative to the initial launch state and marked by visually-distinct targets.

Figure 2. SkySpecs quad rotor test vehicle. (http://imgick.mlive.com/home/mlive-media/pgmain/img/ann-arbor_photos/photo/-cee9773b56d9e324.JPG)

B. Sensors and Computing System

The Figure 3 system diagram shows our geofence augmentation to the SkySpecs multicopter. The off-the-shelf DJI Naza-M is augmented with geofencing prototype sensors (blue outline) and a Beaglebone Black processor. The PX4Flow (https://pixhawk.org/modules/px4flow) provides data in both indoor and outdoor environments, while 3D Position provided by GPS outdoors is not available indoors. To facilitate indoor positioning for our geofence tests we utilize the Pixy camera (http://www.cmucam.org/) to track targets marking geofence boundaries. These sensors are lightweight and relatively low-cost. While the Beaglebone itself has a relatively large footprint, its online tool chain and Linux-based development environment facilitate prototype development and flexible high-rate data acquisition.

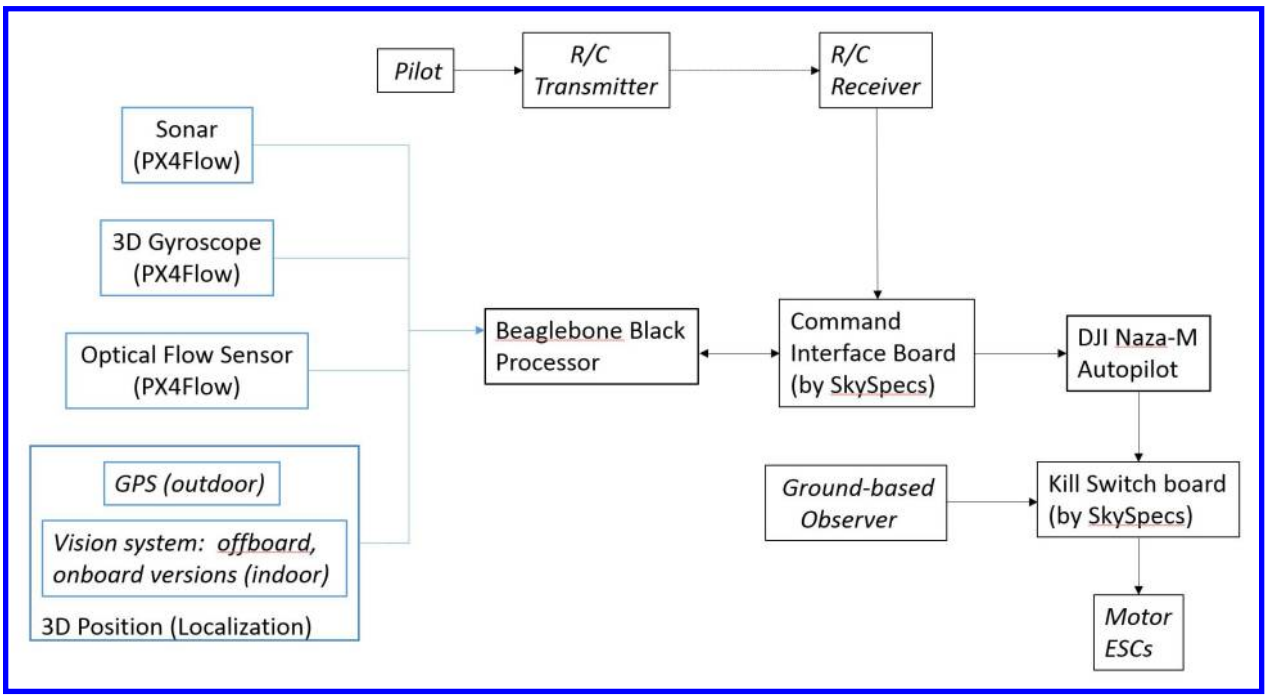


Figure 3. Multicopter Geofence System for Indoor/Outdoor Operation.

The Beaglebone-based geofencing system intercepts the (forward, back, left, right, up, down, autopilot on/off) command stream from the pilot through an interface and switching board (the Command Interface Board) made available by SkySpecs, the local startup company that manufactured and assembled the quadrotor platforms.

The geofencing system monitors 3D position with respect to a user-defined operating volume. If the geofence is activated the Beaglebone replaces pilot commands with values that will stop traversal outside the test range. This is the simplest possible geofencing solution to address the case of exiting the geofenced volume due to pilot input error. The nominal control inputs provide the first level of oversight; the geofence provides a second or backup layer of oversight to ensure the vehicle does not depart the test range; a fully-independent kill switch capability is also included to provide an observer flight terminate ability at any time by turning off all motors.

The set of low-cost geofencing sensors shown in Figure 3 collectively provide 3-D position and attitude information to the BeagleBone processor. This data is sufficient both to activate the geofence and to compute guidance inputs to the low-level Naza-M flight controller.

1. *PX4Flow*

The PX4Flow contains gyroscope, sonar, and optical flow sensors. The gyroscope unit provides attitude and angular rate data. The sonar sensor provides a measurement of altitude above the ground/floor, functioning both indoors and outdoors until the ground goes out of sensor range in which case pressure and GPS (outdoors) would provide altitude. The optical flow sensor provides planar velocity measurements based on pixel differences from one image frame to the next.

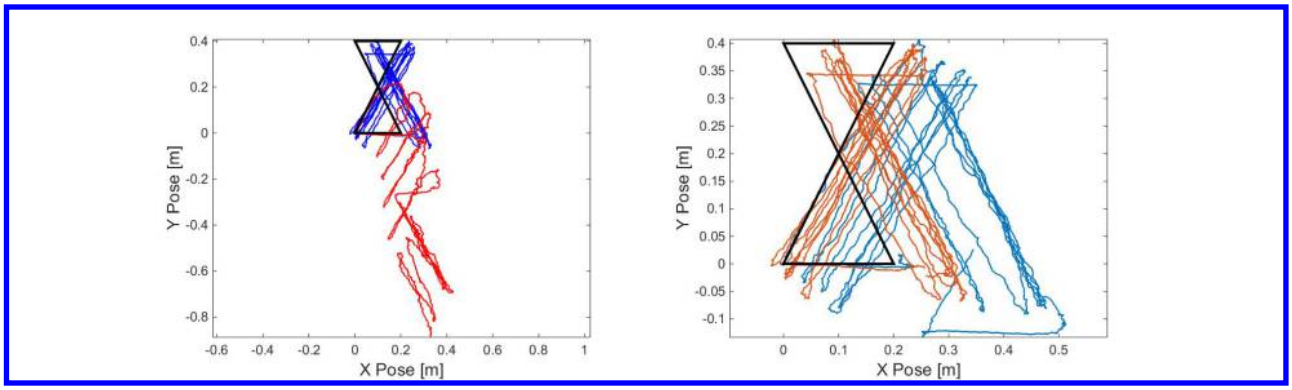
The PX4Flow is connected to the Beaglebone Black through an I2C link. With video downlink disabled, PX4Flow data is received at a frequency of between 110 Hz and 115 Hz. Each I2C message contains the following data of interest:

- x-axis and y-axis optical flow velocities
- quality of optical flow measurement
- x-axis, y-axis, and z-axis gyroscope angular rate measurements
- z-axis sonar-based altitude measurement

The PX4Flow sonar can measure distances from 0.3 meters to 5 meters, which is a sufficient range to measure altitude for indoor flight. To understand the error in optic flow velocity measurements when traversing over an indoor floor, we attached the sensor to a 2D planar manipulator, which allowed the PX4Flow to be commanded to a specific (x, y) location within the manipulator range of motion with an accuracy of a few thousandths of an inch. The change in (x, y) position between data frames was calculated by integrating the velocity measurements based on the time stamps of consecutive measurements.

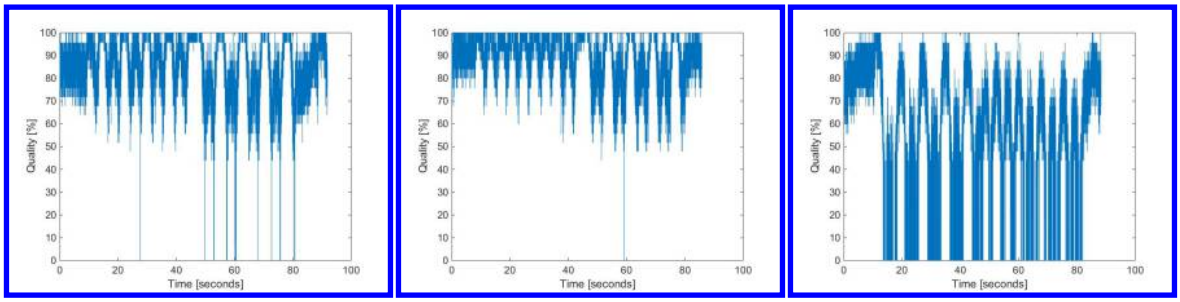
Figure 4(a) shows the actual path of the PX4Flow in black, the integrated pose over well-lit concrete marked with black tape in blue, and the integrated pose over poorly-lit concrete marked with black tape in red. The deviation from the actual path (black) is greater for the poorly-lit path (red) than for the brightly-lit path (blue), showing that the system is not able to perform well in low-lighting conditions. Figure 4(b) shows the actual path of the PX4Flow in black, the integrated position over unmarked concrete in blue, and the integrated position over concrete marked with black tape in red. The black tape was applied to the floor to add features to the ground, thereby improving the optical flow data.

The quality value represents the level of confidence the sensor has in the current optical flow measurement. The higher a quality value, the better the flow estimate is expected to be. Lower quality values indicate less confidence, and values of zero quality indicate that the sensor considered the data to be of too poor quality to pass to the user. In each plot in Figure 5, the quality measurement is at its highest when the PX4Flow is at rest and the downward spikes are indicative of accelerations and constant velocities. Figure 5(a) shows the quality measurements for a well-lit area with limited ground features (unmarked concrete). Figure 5(b) shows the quality of measurements for a well-lit area with ground features (concrete marked randomly with black tape). Figure 5(c) shows quality measurements for a dimly-lit area with ground features (concrete marked randomly with black tape). These three graphs show that the quality of the optical flow data is more sensitive to the lighting of an area than to the presence of distinct features within the field of view.



(a) Blue - bright lighting, red - dim lighting. (b) Blue - featureless, red - featured.

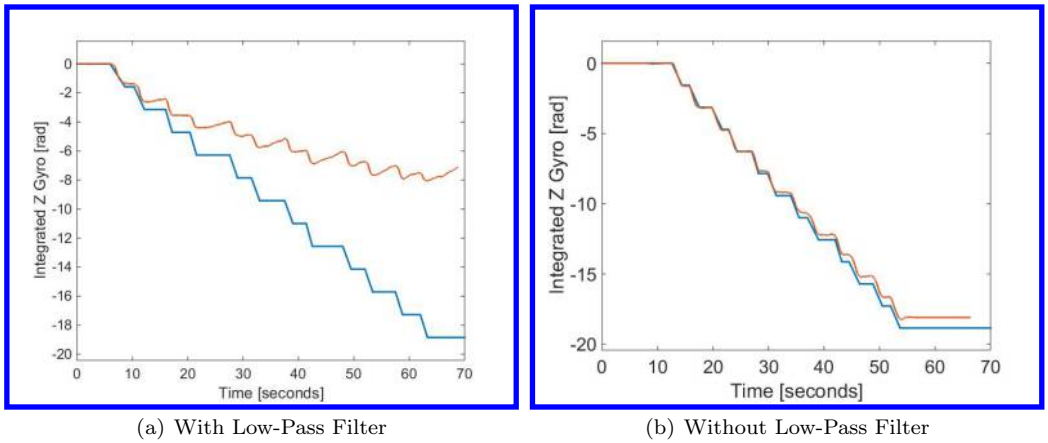
Figure 4. $X - Y$ positions based on integrated optical flow velocities.



(a) Featureless, bright lighting. (b) Featured, bright lighting (c) Featured, dim lighting

Figure 5. Optical flow quality measurements for different test environments.

The current PX4Flow firmware contains a low-pass filter for each gyroscope to continuously remove any gyroscope (gyro) bias. We elected to remove this low-pass filter from the z-axis gyroscope because this continuous bias update causes the rejection of accumulated angular displacement. Figure 6(a) shows the integrated z-axis gyroscope data from the PX4Flow with the low-pass filter in place through twelve consecutively-executed 90 degree turns. As can be seen from the graph, the low-pass filter rejects the motion as accumulated bias, unlike in Figure 6(b) which shows the same motion without the low-pass filter. The removal of the low-pass filter allows for the accurate accumulation of z-axis rotations, which is particularly important for the eventual implementation of this system outdoors and on fixed wing aircraft.



(a) With Low-Pass Filter (b) Without Low-Pass Filter

Figure 6. Integrated PX4Flow z-gyroscope data, twelve 90 degree turns. Blue - actual, red - measured angle.

2. CMUcam5 Pixy

Two different 3D positioning systems are shown in Figure 3. GPS will be used as the primary positioning system in outdoor flight given the need for a low-cost COTS design but will include logic to land automatically or return control to the pilot should GPS signal be lost or otherwise deemed invalid. Because GPS is not available in most indoor locations, two different camera configurations were considered for geofence testing indoors. The first method used offboard cameras to find the vehicle by mounting visually-distinct tags to the multicopter and utilizing the associated software library developed and debugged by University of Michigan colleague Ed Olson (http://april.eecs.umich.edu/wiki/index.php/AprilTags). The APRIL tag system has been demonstrated to provide robust calibration and localization capabilities over a wide range of distances and lighting conditions. The second method required mounting a CMUcam5 Pixy camera on the multicopter and flying it over color tags distributed across the floor at known locations. The APRIL tag method could consistently provide 3D position data, similar to GPS, but could not easily be moved from one test site to another due to its reliance on offboard cameras located at known positions. The Pixy camera solution only provides 3D position data when the color tags are within the onboard camera's field of view, but Pixy only relies on color tags placed at known locations on the floor. The Pixy camera method was selected due to its ease of relocation.

The CMUcam5 Pixy camera detects color codes (i.e., patterns of one or more color blocks) and reports signatures, pixel locations, and angles of the detected color codes. By placing color code tags at known locations around the perimeter of the geofence, the detection and measurements of the color code tags can be used to reorient the multicopter and correct for any positional drift that may have occurred. Figure 7 shows the multicopter with two color code tags, placed at (0m, 0.75m) and (-0.75m, 0m) relative to the multicopter.

Figure 7. Multicopter with two Pixy color code tags and flight safety tethers.

The Pixy tag (x, y) position is expressed by the sensor in pixel coordinates. The transformation from pixels coordinates to the local multicopter frame is show in Equation 1, where $pixy_{center}$ is the pixel coordinates of the center of the Pixy frame, tag_{pixel} is the (x, y) position of the color code tag expressed in pixel coordinates, $altitude$ is the current altitude of the multicopter, $pixy\ focal\ length$ is the focal length of the Pixy camera lens, and tag_{local} is the position of the Pixy color code tag relative to the multicopter's current position.

$$tag_{local} = (pixy_{center} - tag_{pixel}) * \frac{altitude}{pixy\ focal\ length} \tag{1}$$

The transformation of the Pixy tag from the local multicopter frame to the world frame is shown in Equation 2, where $(x_{world}, y_{world}, \theta_{world})^T$ is the position of the multicopter in the world coordinate frame.

$$tag_{world} = \begin{bmatrix} x_{world} \\ y_{world} \\ \theta_{world} \end{bmatrix} + \begin{bmatrix} \cos\theta_{world} & -\sin\theta_{world} & 0 \\ \sin\theta_{world} & \cos\theta_{world} & 0 \\ 0 & 0 & -1 \end{bmatrix} * tag_{local} \tag{2}$$

Figure 8 shows positional data from an indoor flight test. The flight path of the multicopter is represented in blue, where the purple overlay indicates positions at which Pixy color code tags were detected and used to correct the position estimate. The purple overlay only occurs near the Pixy tags due to the field of view of the Pixy camera (75 degrees along the x-axis, 47 degrees along the y-axis). The red data set is the Pixy tag data in world coordinates. The Pixy tags are located at the (0m, 0.75m) and (-0.75m, 0m), which is approximately where they appear in the figure; the positional error of the Pixy tags is due to the estimate's dependency on the multicopter position estimate. If the multicopter position was known exactly, then the primary errors in Pixy tag location estimate would be due to camera distortion and sensor noise.

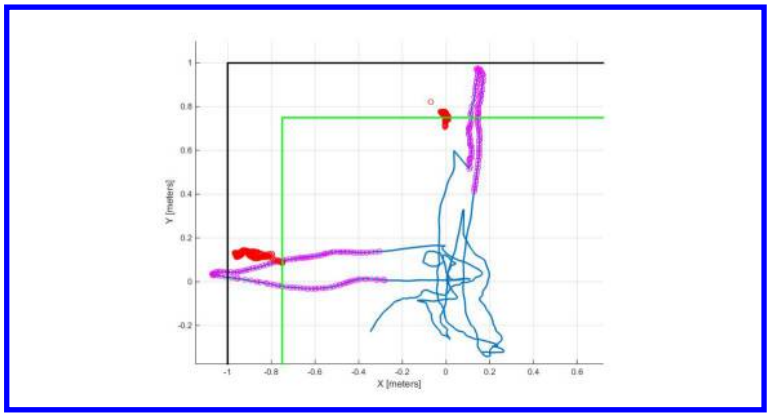


Figure 8. Positions at which Pixy tag data was used during flight test and the detected locations of the Pixy color code tags. Blue - flight path, purple - positions in flight path that utilized Pixy tag data, red - positions of detected Pixy color code tags, green - geofence, black - average geofence overshoot.

V. Localization through a Kalman Filter

The lateral position and orientation of the multicopter is determined using a modified form of an Extended Kalman Filter. Altitude z is directly estimated from the sonar with no further filtering. A modified form of the Kalman Filter is used because the DJI Naza-M autopilot is being used as the black-box inner-loop controller to map the signals from the guidance or outer-loop controller to multicopter motor commands. As such, there is no known mapping from the commands to the accelerations of the multicopter, thus no direct way to propagate system dynamics to estimate velocities at the next time step. Therefore, the lateral 2-D state carried through the filter for this work is limited to the (x, y) position and yaw or heading of the multicopter.

The prediction step of the modified Extended Kalman Filter propagates the current state, μ, forward by the velocities measured by the PX4Flow. For the Q matrix, the x and y covariances are inversely proportional to the quality of the optical flow measurement. The yaw covariance is constant, determined by the quality of the PX4Flow gyroscope measurement.

$$\theta = \mu_\theta + u_\theta * \delta t \tag{3}$$

$$\hat{\mu} = \begin{bmatrix} \cos\theta & -\sin\theta & \mu_x \\ \sin\theta & \cos\theta & \mu_y \\ 0 & 0 & \theta \end{bmatrix} * \begin{bmatrix} u_x * \delta t \\ u_y * \delta t \\ 1 \end{bmatrix} \tag{4}$$

$$F = \begin{bmatrix} 1 & 0 & -\delta t * (\sin\theta + \cos\theta) \\ 0 & 1 & \delta t * (\cos\theta - \sin\theta) \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$\hat{\Sigma} = F * \Sigma * F^T + Q \tag{6}$$

The update step of the modified Extended Kalman Filter corrects the state estimate based on the observed Pixy tags. The R matrix is computed based on the average distance from the observed tag to the multicopter squared.

$$K = \Sigma * H^T * (H * \Sigma * H^T + R)^{-1} \tag{7}$$

$$\hat{\mu} = \mu + K * (z - \hat{z}) \tag{8}$$

$$\hat{\Sigma} = (I - K * H) * \Sigma \tag{9}$$

VI. Guidance or "Outer-loop Controller" Formulation and Results

The SkySpecs quadrotor test platform is equipped with a commercially-available DJI Naza-M autopilot as discussed above. This autopilot nominally receives pulse width modulation (PWM) signals from the pilot through a standard radio-controlled receiver. These signals are mapped by the Naza-M to "outer-loop" thrust, yaw, pitch, and roll commands. The SkySpecs Command Interface Board either passes pilot commands directly through to the Naza-M or feeds commands from the BeagleBone to the Naza-M. For a normal flight, the BeagleBone command path is active throughout; the BeagleBone passes through pilot commands except when the geofence is active. Since the Naza-M autopilot is proprietary, the exact nature of its embedded inner-loop controller is unknown; we therefore treat this inner-loop controller as a black box.

For all flight tests, the quadrotor pilot nominally provides outer-loop control inputs through an off-the-shelf 2.4 GHz transmitter; the geofence monitors estimated quadrotor lateral position and altitude throughout the flight. When the geofence perimeter has been violated, geofence automation overrides pilot inputs. The geofencing software for our quadrotor implementation is effectively an outer-loop controller prescribing motions to guide the quadrotor back to a reference point within the geofenced area. Manual control is returned to the pilot for our tests after a user-defined period of time has elapsed and the quadrotor is contained within its geofenced operating volume. Either an altitude or lateral geofence violation will activate the entire 3-D geofence controller to reduce the potential for pilot mode confusion.

A. Input Characterization

Once active, the geofence controller publishes PWM commands to the Naza-M in the same manner as the manual transmitter. When both control sticks of the transmitter are in their neutral, centered positions, the Naza-M commands the quadrotor to maintain a trimmed hover condition apart from external disturbances. We refer to the corresponding PWM signals as *baseline* commands, each of which are approximately 1500 μs in PWM signal width. Input deviations below this baseline are referred to as "low" commands and deviations above this baseline are referred to as "high" commands. Depression of the control sticks through their full range of motion reveal that for each input the minimum PWM command is approximately 1075 μs and the maximum PWM command is approximately 1920 μs. Such minimum and maximum commands are not realized, however, in manual flight with default Naza-M sensitivity settings as such large commands result in destabilizing motions. Instead, flight test data from well-controlled manual flights suggests that user inputs rarely exceed a range of 1400-1600 μs to produce desired motions. We therefore manually impose saturation limits on computed control commands.

Table 1 indicates the motions that correspond to low and high commands for each of the inputs, as well as the saturation limits that are imposed on each within the controller. Note that positive and negative pitch and roll correspond with the right-handed convention of the body-fixed frame definition used throughout. That is, the positive y-axis extends through the aft-end of the quadrotor and the positive x-axis extends through the starborad side of the quadrotor. By this convention, roll is said to be the rotation about the y-axis and pitch is said to be the rotation about the x-axis. Yaw is said to be about the positive z-axis, which extends through the undercarriage of the quadrotor.

B. Control Law

Once input-output characteristics were defined, an outer-loop controller was developed to turn the quadrotor back into the geofenced area following a perimeter breach. The general framework of this controller is based on a simple proportional-derivative (PD) feedback mechanism, which leverages information about the quadrotor's current state in comparison to some desired state to update control commands. The states of interest in this controller are the global Cartesian position (x, y), the altitude h, the yaw angle ψ, and the time rate of change of each. This state is therefore defined as $\mathbb{X} = [\mathbf{x} \quad \dot{\mathbf{x}}]^\intercal$, where $\mathbf{x} = [h \quad x \quad y \quad \psi]$.

Input	Low Command	High Command	Saturation Limits [μs]
Thrust	Decrease Altitude	Increase Altitude	[1425, 1575]
Pitch	Positive Pitch	Negative Pitch	[1380, 1620]
Roll	Negative Roll	Positive Roll	[1380, 1620]
Yaw	Negative Yaw	Positive Yaw	[1425, 1575]

Table 1. Mapping of input PWM signals to quadrotor response and manually-imposed saturation limits.

Upon each iteration of the controller, the difference between the current estimated state of the quadrotor, $\hat{\mathbb{X}}$, and the desired state set-point value, $\mathbb{X}_{\text{set}}$, is computed to yield the proportional and derivative state errors, $\delta\mathbf{x}$ and $\delta\dot{\mathbf{x}}$:

$$\delta\mathbb{X} = \begin{bmatrix} \delta\mathbf{x} \\ \delta\dot{\mathbf{x}} \end{bmatrix} = \mathbb{X}_{\text{set}} - \hat{\mathbb{X}} \tag{10}$$

This error term, $\delta\mathbb{X}$, is then used to compute the control inputs for thrust, pitch, roll, and yaw by mapping these state errors to control deviations from the baseline input as follows:

$$\mathbf{u} = \mathbf{u}_{\text{base}} + \mathbf{K}\delta\mathbb{X} \tag{11}$$

where the control vector $\mathbf{u} = [u_{\text{thrust}} \quad u_{\text{roll}} \quad u_{\text{pitch}} \quad u_{\text{yaw}}]^\intercal$, $\mathbf{u}_{\text{base}}$ is a column vector of constant baseline commands associated with each of the inputs (in this case 1500 μs for each), and $\mathbf{K} \in \mathbb{R}^8 \times \mathbb{R}^4$ is the constant-valued gain matrix. In this format, the gain matrix can be decomposed into $\mathbf{K} = [\mathbf{K}_P \quad \mathbf{K}_D]$, where $\mathbf{K}_P$ denotes the proportional gains and $\mathbf{K}_D$ denotes the derivative gains.

In the present work, it is assumed that the quadrotor operates with a small yaw angle throughout its manual flight and upon activation of the controller a yaw set-point of 0 radians is commanded. This simplification further enables the assumption that state errors are decoupled when computing control input commands. That is, the x-position can be completely controlled through roll and the y-position can be completely controlled through pitch. Thus, the gain matrices can be expressed as follows: $\mathbf{K}_P = \text{diag}\{K_{P,\text{thrust}}, K_{P,\text{roll}}, K_{P,\text{pitch}}, K_{P,\text{yaw}}\}$ and $\mathbf{K}_D = \text{diag}\{K_{D,\text{thrust}}, K_{D,\text{roll}}, K_{D,\text{pitch}}, K_{D,\text{yaw}}\}$.

Input commands computed by (11) are not directly submitted to the Naza-M; they must first be validated against manually-imposed saturation limits to ensure that they are of an acceptable magnitude. If the computed control is found to exceed saturation bounds in either direction, the closest saturation limit is instead passed to the Naza-M. The reasons to impose this saturation limit are twofold. First, this method ensures that the control inputs commanded by the autopilot are always within the feasible range of commands, [1075, 1920] μs. This prevents the Naza-M from entering unstable operating modes that may arise from unexpected/erroneous input commands. Additionally, the presence of saturation limits restrains the autopilot from commanding *excessive* feasible commands that could contribute to system destabilization. By imposing these saturation bounds, the vehicle is constrained to smooth motions that do not produce large amplitude pitch and roll angles.

It should also be noted that there is additional logic present in the control software specific to the control of altitude and yaw angle. Namely, there is a state-tolerance *deadband* present in each that will default to the baseline input command if the altitude and yaw are within a certain tolerance region of the set-point. For altitude this tolerance band allows for a 5 centimeter discrepancy in either direction of the set-point, whereas the yaw tolerance allows for a 0.009 radian disparity in either direction. Outside of these regions, both yaw and altitude control operate according to (11).

C. Set-Point Design

By virtue of the geofence's goal, the final desired state $\mathbb{X}_{\text{set}}$ supplied to the controller should be some position well-within the geofenced region with all reference velocity terms equal to zero. Commanding this set-point immediately as the geofence is violated, however, will result in an unfavorable transient response characterized by large overshoot and oscillation. The reason for this poor transient response is attributed to the fact that a perfect step-input is impossible for a real system to track.

To mitigate these transient response issues, a pre-filter has been developed that varies the set-point gradually in a manner that is much more reasonable for the system to track. In particular, this pre-filter linearly ramps the commanded set-point from the value of the quadrotor's state when the fence is initially breached to the final desired state in a specified interval of time. This corresponds to the following logic,

$$\mathbb{X}_{\text{set}}(t) = \begin{cases} \hat{\mathbb{X}}_0 + \left(\dfrac{\mathbb{X}_{\text{set},f} - \hat{\mathbb{X}}_0}{t_{\text{des}}} \right) t & \text{if} \quad t - t_0 \leq t_{\text{des}} \\ \mathbb{X}_{\text{set},f} & \text{if} \quad t - t_0 > t_{\text{des}} \end{cases} \tag{12}$$

where t_0 is the time at which the geofence is initially violated and $\hat{\mathbb{X}}_0$ is the corresponding state estimate of the quadrotor at this point; $\mathbb{X}_{\text{set},f}$ is the desired or reference state of the quadrotor, and t_{des} is the user-prescribed ramp period.

For this implementation, the specified ramp interval is three seconds from perimeter breach and the ultimate set-point is 0.8 meters directly above the origin point from which the quadrotor launched. The dashed lines in Figure 9 show the set-point time history corresponding to x- and y-positions. Note that the pre-filter is only applied to altitude, x-position, and y-position set-points currently because they were observed to have the poorest transient responses in the presence of a step-input command. Future developments of the controller should extend this pre-filter to all set-points.

D. Gain Tuning and Controller Performance

In absence of a dynamic model for the quadrotor, gain matrix $\mathbf{K}$ was initially populated with estimates for proportional and derivative gains that ensure full saturation of manually-imposed control limits at specific values of error. These saturating values of error were chosen based on a qualitative understanding of the system dynamics garnered from performing manual flights. In particular, it was evident before implementing the controller that roll and pitch commands required large derivative gains in order to provide damping to an otherwise very lightly-damped motion. Additionally, yaw and upward thrust commands appear to require very little derivative gain, since both motions are heavily damped: yaw by the rotational inertia of the spinning propellers, and upward thrust by gravity. Therefore, the yaw and thrust controllers were developed with purely proportional feedback (i.e. $K_{D,\text{thrust}} = K_{D,\text{yaw}} = 0$).

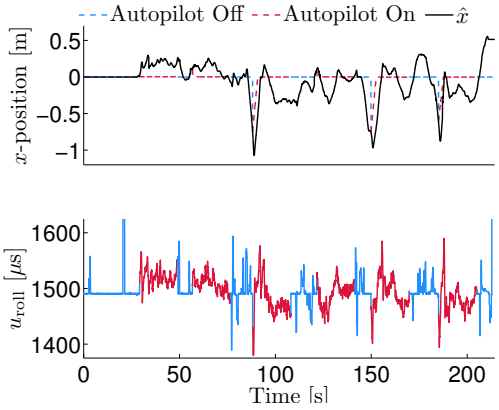
Through a series of flight tests, these gains and saturation limits were tuned manually to an acceptable level of performance. We do not claim the controller as a major contribution to this work; its purpose is to demonstrate stable geofencing. Therefore, this gain tuning process was not extensive. It should additionally be noted that yaw control was omitted from this tuning process since the Naza-M accurately maintained a fixed yaw angle without outer-loop control inputs. Expanding geofence control to support non-zero yaw commands is a subject of future work.

The final, tuned controllers for x- and y-positions were determined to have derivative gains whose magnitudes are double that of their corresponding proportional gains ($K_{D,\text{roll}} = K_{D,\text{pitch}} = 200$ and $K_{P,\text{roll}} = K_{P,\text{pitch}} = 100$). This ensures that the controller limits overshoot by commanding inputs to preemptively damp the quadrotor's motion as it approaches its final set-point. Figure 9 shows the time responses of these controllers in a flight test during which the geofence controller is commanded to remain active for 20 seconds once it is initialized. This long geofence control period enables us to examine the transient response of the geofence controller.

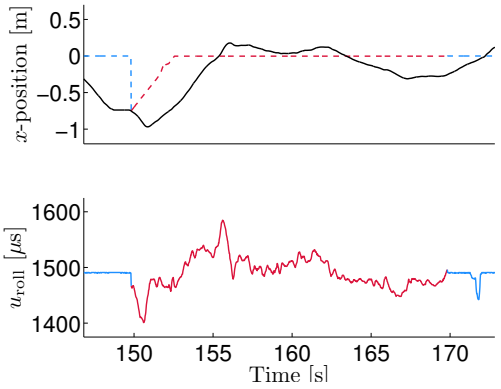
Each of the present controller responses are comprised of two plots. The first shows the time history of a given component of the estimated state $\hat{\mathbb{X}}$ (solid black line) in comparison to the commanded set-point $\mathbb{X}_{\text{set}}$ (dashed-line), while the second shows the time history of the associated control input that is supplied to the Naza-M. In each plot, color is used to distinguish how control commands are transmitted to the Naza-M; light-blue segments indicate manual control, whereas red segments correspond to geofence controller output.

Figures 9(a) and 9(c) indicate that the geofence controller becomes active six times during this flight. The resulting responses in the x- and y-position are favorable, since the quadrotor is always restored to the geofenced region in a stable manner well before the 20 second period has elapsed. It is evident that these controllers are most effective when the quadrotor slowly breaches the geofence perimeter in their respective directions. This is demonstrated in Figure 9(b) and Figure 9(d), where the quadrotor's position is restored quickly to the set-point with little overshoot and small-amplitude transient oscillations.

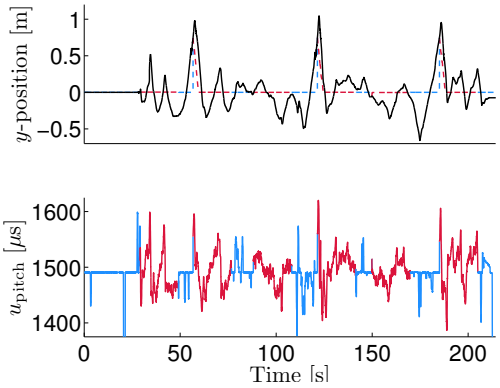
Altitude controller response is shown for the same flight test in Figure 10(a). Once again, we observe that the quadrotor is quickly restored to a position well-within the geofenced altitude ceiling (1.1 meters



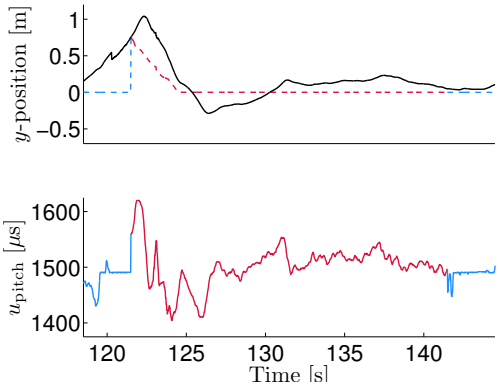
(a) x-position control with roll commands.



(b) Close-up of fifth autopilot controlled response in Figure 9(a).



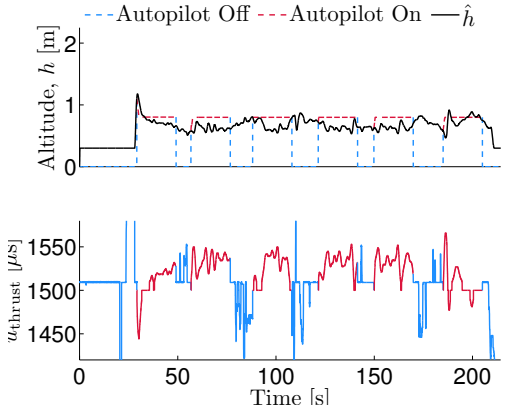
(c) y-position control with pitch commands.



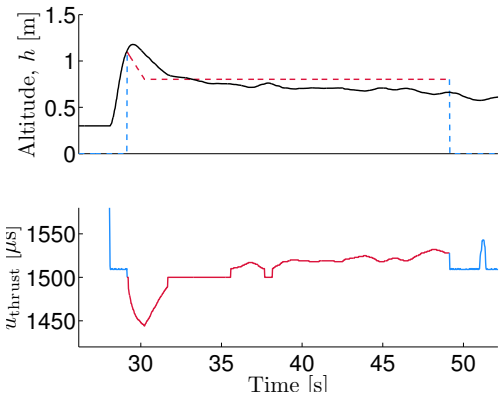
(d) Close-up of fourth autopilot controlled response in Figure 9(c).

Figure 9. Tuned x- and y-position controller response, with $K_{D,\textbf{roll}} = K_{D,\textbf{pitch}} = 200$ and $K_{P,\textbf{roll}} = K_{P,\textbf{pitch}} = 100$.

above the floor) for all controlled regions. In particular, the first controller response, shown in Figure 10(b), demonstrates an example in which the quadrotor attempts to breach the geofence ceiling. Notice that the quadrotor's motion is near-immediately halted from further exceeding this boundary and the quadrotor quickly is restored to the set-point of 0.8 meters.



(a) Altitude control with thrust commands.



(b) Close-up of first autopilot controlled response in Figure 10(a)

Figure 10. Tuned altitude controller, with $K_{D,\textbf{thrust}} = 0$ and $K_{P,\textbf{thrust}} = 200$.

All demonstrated controller responses are shown for relatively low-velocity attempts to breach the geofence. To account for high-velocity breaches, augmentation of the present controller is required. Namely, pre-filters need be applied to derivative set-points and the altitude controller requires additional derivative error feedback that accounts for imbalanced gravitational damping. All geofence outer-loop controllers could additionally benefit from addition of integral feedback to mitigate inherent steady-state error.

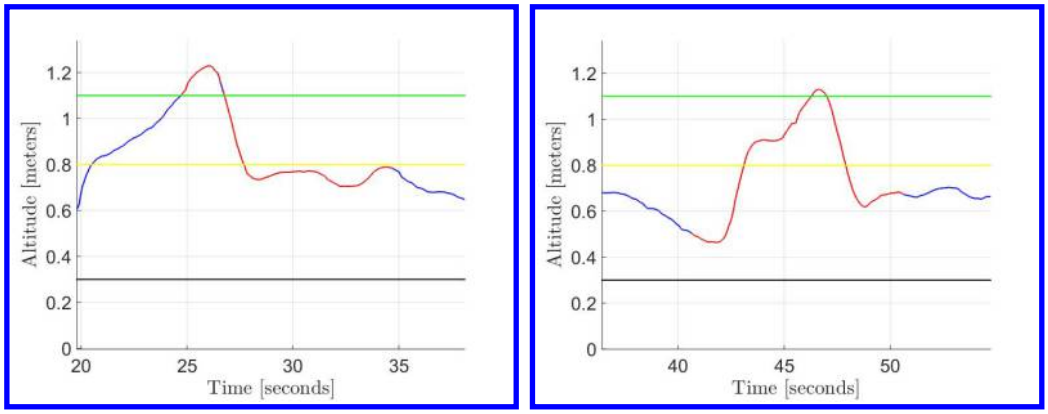
VII. Geofencing Results

Lateral position and altitude time histories were collected over a series of intentional geofence breach attempts by the pilot. Table 2 presents overshoot statistics based the farthest excursions outside each geofence boundary observed in each test. The average of both x and y overshoot is between 0.20 m and 0.25 m, while the standard deviation is between 0.06 m and 0.11 m. Overshoot is consistently correlated with quadrotor velocity when the geofence is activated.

Table 2. Planar peak overshoot data statistics

Overshoot Statistic	Value [m]
X Average	0.2309
Y Average	0.2448
X Standard Deviation	0.0620
Y Standard Deviation	0.1047

Figure 11 demonstrates the response of the geofence to a deviation from set point altitude, 0.8 m as shown by the yellow lines. The altitude geofence is set at 1.1 m, the green line, and is activated in Figure 11(a). Figure 11(b) shows the result of the geofence activating while the altitude of the multicopter is lower than the set point altitude.

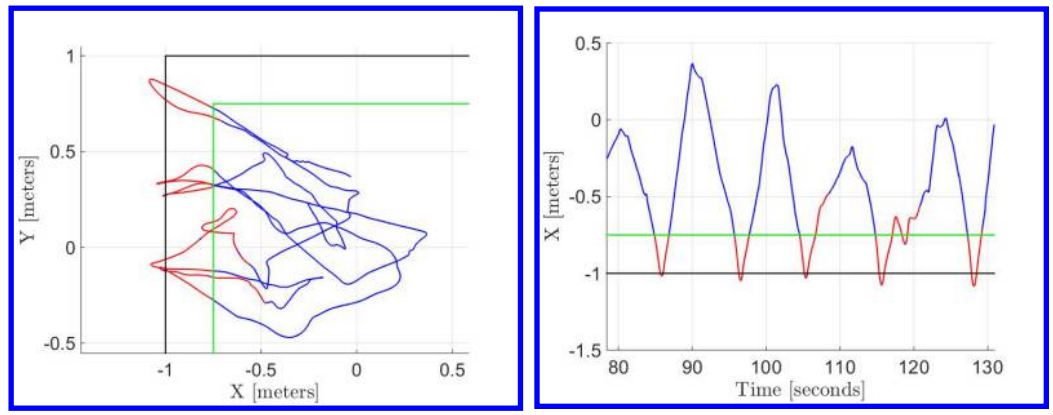


(a) High altitude geofence breach. (b) Low altitude correction.

Figure 11. Altitude geofence corrections. Blue - autopilot off, red - autopilot on, green - geofence, yellow - altitude set point, black - minimum sonar distance.

Figure 12 shows several examples of planar geofence breaches along the x-axis (Figures 12(a) and 12(b)) and the y-axis (Figures 12(c) and 12(d)). Each set of breaches is displayed in the x-y plane and the time domain. The x-y plane shows the ways in which the multicopter moves based on user input and autopilot input. The time domain plots display the same data as the related planar plots. Only the axis in which the geofence breaches occur is displayed in the time domain.

Figure 13 shows the full test flight from which the plots in Figure 12 are taken. In these graphs, the green line is the geofence boundary: $x = -0.75m$ and $y = 0.75m$. The black line is the geofence boundary plus the approximate average overshoot distance: $x = -1.0m$ and $y = 1.0m$. This data was generated with the geofence controller activating for a minimum of a two second duration or period. If the multicopter is still outside of the geofence at the end of the two seconds of autopilot control, the geofence controller re-activates.



(a) X breaches (b) X breaches vs time

(c) Y breaches (d) Y breaches vs time

Figure 12. Blue - autopilot off, red - autopilot on, green - geofence, black - average geofence overshoot

VIII. Conclusions and Future Work

Geofencing automation has the potential to offer small UAS the ability to safely and freely fly in immediate reaches airspace without air traffic control oversight or detect-and-avoid. This paper has presented an initial geofencing automation concept and demonstrated its use over a series of indoor quadrotor geofencing experiments. These geofencing experiments demonstrate that a simple low-cost onboard sensor suite and outer-loop geofencing guidance algorithm can effectively implement the geofence without destabilizing or confusing at least an operator who was well aware of the geofence boundaries.

The next steps in the development of the indoor geofence are to implement a more robust yaw outer-loop controller, extend the localization software to function with GPS for outdoor flight, and modify the Kalman Filter scheme to propagate velocities as well as positions over time. Platform independence is a key challenge to geofence development. The key challenges to extending geofence automation to other multicopter and fixed wing platforms are in the development of flexible guidance or outer-loop control laws that can be automatically adapted to the each platform, and in development of a standardized interface between the geofence system and a variety of small UAS platform.

Numerous research questions must be addressed in future work before geofencing automation can be certified and infused into commercially-available products. First, how in general will the operator and suite of small UAS autopilot technologies effectively interact with the geofence automation? We were keenly aware of the geofence in all our indoor flight tests, but the geofence might be a surprise to an operator focusing on the primary mission. While lost link "flight termination" geofencing such as that offered on hobbyist PCM receivers currently functions well over rural ranges, this "hobbyist geofence" of sorts certainly can't just crash the plane (flight terminate) over people or property even if link is lost. The geofence must therefore override even PCM flight termination in cases where a geofence floor is present for safety reasons.

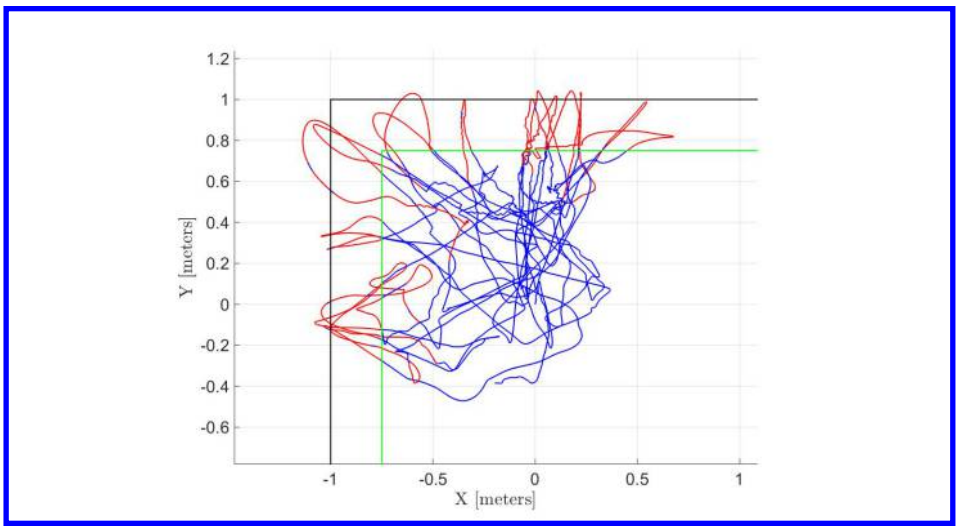


Figure 13. Repeated deliberate breaches of geofence by pilot. Autopilot duration - 2 seconds. Blue - autopilot off, red - autopilot on, green - geofence, black - average geofence overshoot

The geofence must enhance not compromise safety. It therefore must not activate in situations where it doesn't understand how to fly the plane (e.g., damage/failures) even if the aircraft is departing the geofenced test range, except in conditions where the geofence automation is able to safely execute a flight termination sequence as a last-resort backup. Finally, any mass-produced geofencing capability must be easy-to-understand and use by the "average small UAS operator" who may have little flight training beyond perhaps the "aeronautical knowledge test" under development by the FAA. Geofencing products are needed for a large variety of small UAS configurations ranging from multicopters to the spectrum of fixed-wing designs from glider to aerobatic. The underlying feedback control problem for some of these platforms remains challenging, especially given damage or failures, and geofence automation relies on a robust feedback controller to execute its turn-back guidance instructions.

IX. Acknowledgments

The authors would like to thank Peter Gaskell from the University of Michigan Robotics Program and Kelly Hayhurst from NASA Langley for their support throughout the project. We also thank University of Michigan Autonomous Aerospace Systems Lab undergraduates Jerry Lin and David Hershey for their initial work to develop the geofencing automation concept. This work was supported in part by NASA contract NNX11AO78A.

References

[1]Melekhova, O., Abchir, M.-A., Châtel, P., Malenfant, J., Truck, I., and Pappa, A., "Self-Adaptation in Geotracking Applications: Challenges, Opportunities and Models," *ADAPTIVE 2010, The Second International Conference on Adaptive and Self-Adaptive Systems and Applications*, 2010, pp. 68–77.

[2]Resch, B., "People as sensors and collective sensing-contextual observations complementing geo-sensor network measurements," *Progress in Location-Based Services*, Springer, 2013, pp. 391–406.

[3]Souliotis, N., Tsadimas, A., and Nikolaidou, M., "Real-time information about public transport's position using crowdsourcing," *Proceedings of the 18th Panhellenic Conference on Informatics*, ACM, 2014, pp. 1–6.

[4]Morgul, E. F., Yang, H., Kurkcu, A., Ozbay, K., Bartin, B., Kamga, C., and Salloum, R., "Virtual Sensors: A Web-based Real-Time Data Collection Methodology for Transportation Operation Performance Analysis," *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2014, 2014.

[5]Yuce, Y. K., Gulkesen, K. H., and Barcin, E. N., "An Approach to Geotracking Patients with Alzheimers Disease," *Studies in Health Technology and Informatics Series, IOS Press*, Vol. 176, 2012, pp. 121–125.

[6]Bauer, M., Dobre, D., Santos, N., and Schmidt, M., "Scalable processing of geo-tagged data in the cloud," *NEC Technical Journal*, Vol. 7, No. 2, 2012, pp. 97.

[7]Chatterji, G., Menon, P., and Sridhar, B., "GPS/machine vision navigation system for aircraft," *Aerospace and Electronic Systems, IEEE Transactions on*, Vol. 33, No. 3, 1997, pp. 1012–1025.

[8]Cabler, H. and DeCleene, B., "LPV: New, improved WAAS instrument approach," *Proceedings of the 15th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2002)*, 2001, pp. 1013–1021.

[9]Olmos, B. O., Mundra, A., Cieplak, J. J., Domino, D. A., and Stassen, H. P., "Evaluation of near-term applications for ADS-B/CDTI implementation," *98WAC-73, World Aviation Congress Conference*, 1998.

[10]Krozel, J. and Andrisani, D., "Independent ADS-B verification and validation," *AIAA Aviation, Technology, Integration, and Operations Conference Proceedings*, 2005, pp. 1–11.

[11]Sampigethaya, K. and Poovendran, R., "Enhancing ADS-B for Future UAV Operations," *AIAA Infotech@ Aerospace 2012*, 2012.

[12]Atkins, E. M., "Autonomy as an Enabler of Economically-Viable Beyond-Line-of-Sight Low-Altitude UAS Applications with Acceptable Risk," *AUVSI North America*, 2014.

[13]Williams, B. P., Clothier, R., Fulton, N., Lin, X. n., Johnson, S., and Cox, K., "Building the Safety Case for UAS Operations in Support of Natural Disa ster Response," *Proceedings of the 14th AIAA Aviation Technology, Integration, and Operations Conference (AIAA Aviation 2014)*, American Institute of Aeronautics and Astronautics Inc., 2014.

[14]Anderson, R., Moncayo, H., Prazenica, R., Mirmirani, M., and Noriega, A., "Development of a Surrogate Autonomous Aircraft for Entry in the NASA Airspace Operation Challenge," *Proceedings of the AIAA Infotech@Aerospace Conference*, American Institute of Aeronautics and Astronautics Inc., 2015.

[15]Roadman, J., Elston, J., Argrow, B., and Frew, E., "Mission Performance of the Tempest Unmanned Aircraft System in Supercell Storms," *Journal of Aircraft*, Vol. 49, No. 6, 2012, pp. 1821–1830.

[16]Meier, L., Tanskanen, P., Heng, L., Lee, G. H., Fraundorfer, F., and Pollefeys, M., "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Autonomous Robots*, Vol. 33, No. 1-2, 2012, pp. 21–39.

[17]McNeal, G. S., "Drones and Aerial Surveillance: Considerations for Legislators," *Brookings Institution: Project On Civilian Robotics*, 2014.

[18]Kopardekar, P., "Safely Enabling Low-Altitude Airspace Operations: Unmanned Aerial System Traffic Management (UTM)," Available at http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20150006814.pdf.