Principles of Guidance, Navigation and Control of UAVs

Gabriel Hugh Elkaim
University of California Santa Cruz
Santa Cruz, CA, USA

Fidelis Adhika Pradipta Lie
University of Minnesota
Minneapolis, MN, USA

Demoz Gebre-Egziabher
University of Minnesota
Minneapolis, MN, USA

August 6, 2012

1 Abstract

Two complete system architectures for a guidance, navigation and control solution of small UAVs is presented. These systems (developed at the University of California Santa Cruz and the University of Minnesota) are easily reconfigurable and are intended to support testbeds used in navigation, guidance and control research. The systems described both integrate a low cost inertial measurement unit, a GPS receiver, a triad of magnetometers to generate a navigation solution (position, velocity and attitude estimation) which, in turn, is used in the guidance and control algorithms. The navigation solution described is a 15 state Extended Kalman Filter which integrates the inertial sensor and GPS measurement to generate a high-bandwidth estimate of a UAV's state. Guidance algorithms for generating a flight trajectory based on waypoint definitions are also described. A PID controller which uses the navigation filter estimate and guidance algorithm to track a flight trajectory is detailed. The architecture: the hardware, software and algorithms is included for completeness. Hardware in the loop simulation and flight test results documenting the performance of these two systems is given.

2 Introduction

In current usage, the term Uninhabited Aerial Vehicles or UAVs refers to aircraft which fly without a human operator onboard. They span a wide range in size and complexity. The largest UAVs such as Predator or Globalhawk can weigh several thousand pounds and have wing spans on the order of 10 to 100 feet. At the other end of the size spectrum are UAVs whose maximum dimensions and mass are on the order of centimeters and grams, respectively. In general, these vehicles are being used or envisioned for use in operations where they serve primarily as a platform for a sensor payload. The sensor payload can be as simple as an Electro-Optical camera or as complex as synthetic aperture radar used for remote sensing.

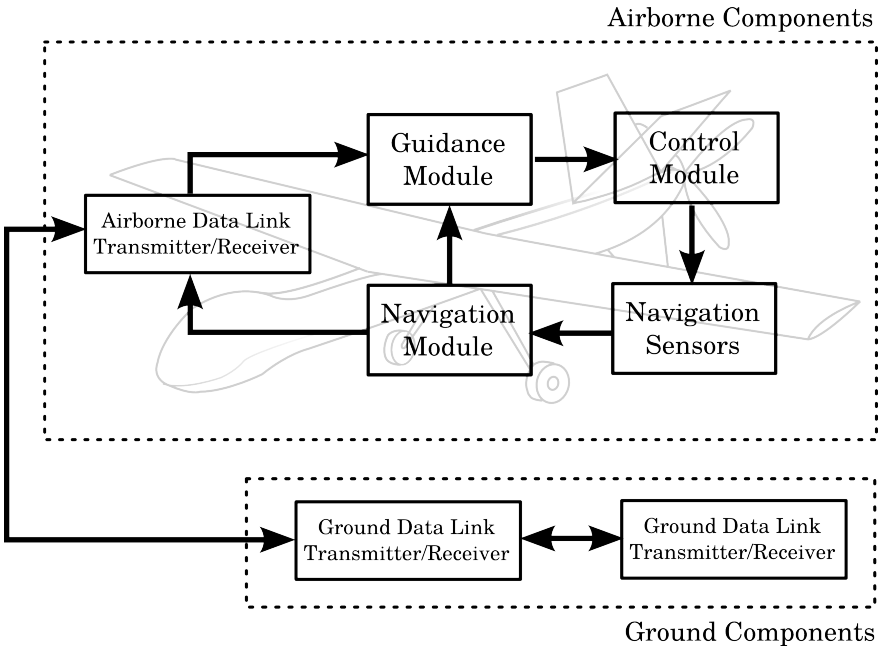


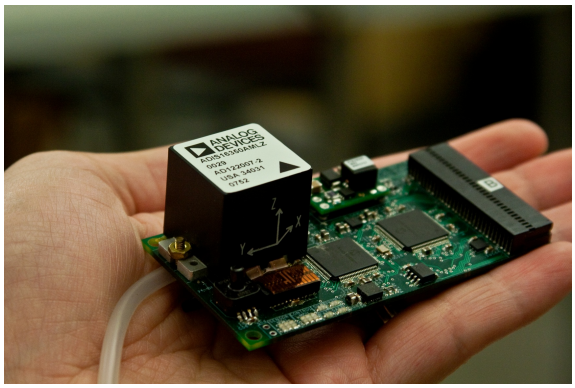
Figure 1: Unmanned Aerial System (UAS) Concept of Operation

Regardless of their size and mission, all UAVs share the need for sensor or sensor systems which provide an estimate of the vehicle's full state vector. The state vector normally consists of three position coordinates, three components of the velocity vector and anywhere between three and nine parameters which describe the vehicle's attitude. In addition to state sensing and estimation, UAVs (especially one that operate autonomously) need control and guidance systems which allow them to maneuver in a way consistent with their mission. In simple terms, the guidance system generate instruction on what state trajectory the UAV should follow in accomplishing its mission. The control system in turn operates the aircraft controls (aileron, elevators, thrust, etc) to follow the trajectory generated by the guidance systems. A high level depiction of a GNC system is shown in Figure 1 and consists of both airborne and ground components.

Off-the-shelf guidance, navigation and control (GNC) solutions for small UAVs (i.e., UAVs which fall in the so-called "Class I" category of vehicles as defined in [UAS CoE, 2010]) exist. Some of these off-the-shelf solutions consist of the vehicle itself, avionics and associated ground support systems. While most of these "turn key" solutions allow some level of customization by the user, they can be very restrictive for use in research environments. For example, in research on the robust control problem, the effect of uncertainties and potential fault modes in the guidance and navigation algorithms must be known. Very few off-the-shelf solutions allow access to the internals of the GNC solution. Thus, a GNC solution that is inexpensive and easily reconfigurable is desirable for the research community.

This chapter describes two such GNC solutions for small UAVs. This chapter discusses the implementation aspect of the GNC modules on University of California Santa Cruz's (UCSC) and University of Minnesota's (UMN) UAVs. The UAV laboratory at UMN is equipped with two fixed-wing aircraft models: Ultrastick 25e and Ultrastick 120. Shown in Fig. 2 is the Ultrastick 120, also known as FASER (Free-flying for Subscale Experimen-

Figure 2: Free-flying for Subscale Experimental Research (FASER) UAV (A Modified Ultrastick 120)



(a)

(b)

Figure 3: The UCSC SLUGS autopilot (a) and Multex Mentor UAV (b)

tal Research). FASER is equipped with conventional flight control surfaces and a pair of wingtip vanes on each sides of the aircraft to measure aerodynamic angles (angle-of-attack and sideslip angle). Prior to joining the UAV Laboratory, FASER's airframe was formerly used by NASA for their research program [Owens et al., 2006]. Extensive wind tunnel testing during its time at NASAs has been used to develop a comprehensive aerodynamic model of the aircraft. At University of Minnesota, FASER is the workhorse for the multi-sensor navigation research such as GPS attitude and heading determination system, synthetic airdata estimator [Lie and Gebre-Egziabher, 2012], and vision-aided navigation [Chu et al., 2011]. Thor, an Ultrastick 25e, is an approximately 65% scaled model of FASER with the same basic configuration. As an addition to the system identification work [Dorobantu et al., 2011], this UAV is used for fault tolerant control research at the University. The UAV Laboratory website [Murch, 2012b] provides a detailed information on the airframe, avionics, and software architecture. The flight software is available as an open source software package available upon request.

The UCSC autopilot, SLUGS, has been developed over the past five years in order to implement a rapidly reconfigurable autopilot for UAV guidance, navigation, and control research. The SLUGS, pictured in Fig. 3a, consists of two fast dsPIC33 microcontrollers (DSC's), and a suite of sensors; the SLUGS-based electric UAV is pictured in Fig. 3b. Details

of the design, development, and deployment of the SLUGS can be found in [Lizarraga, 2009], [Lizarraga et al., 2011a], [Lizarraga et al., 2011b], [Lizarraga et al., 2009a], and [Lizarraga et al., 2009b].

Section 3 describes the UMN's navigation module for attitude and position estimation. Section 4 and 5 describe the inner loop control and guidance algorithm implemented on the UCSC SLUGS platform respectively, including simulation and experimental results.

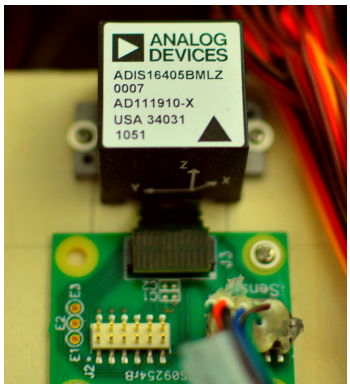
3 Attitude and Position Estimation

In order to be able to guide and control the UAV, the state of the UAV must be available to the controller at high fidelity and high bandwidth. Accurate position is required to perform automatic control for precision applications (e.g., landing). The navigation module in the GNC system implements aircraft state estimation. The advent of powerful computers made available at relatively low cost has allowed sensor fusion for navigation; sensor fusion is a technique of optimally blending information from multiple different (flawed) sensors. Multi-sensor navigation framework aims to obtain the most information about the aircraft states by using minimum combination of sensors. This is key to UAV operations where size, weight, and power are all critical. This section will describe the navigation module implemented on University of Minnesota's UAV.

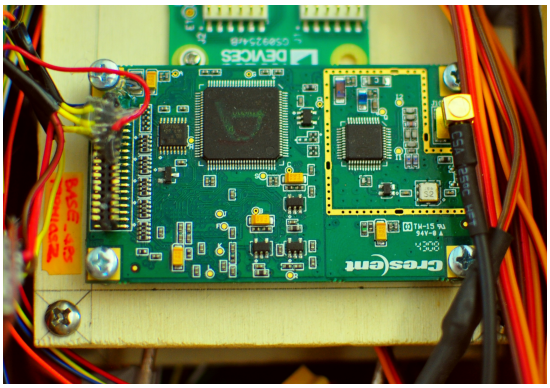
The complete state of the UAV comprises its position, velocity, attitude, airspeed, angle-of-attack, sideslip angle, and rotation (pitch, roll, and yaw) rates. Position, velocity, and attitude are also known as the navigation state [Gleason and Gebre-Egziabher, 2009]. The sensors used to measure these quantities are called navigation sensors: an inertial measurement unit (IMU) and global positioning system (GPS) receiver. Fig. 4a shows the Analog Devices IMU ADIS16405 installed on the UAVs at University of Minnesota. It is a 6 degree-of-freedom temperature calibrated inertial measurement unit with 3-axis accelerometers, 3-axis gyros, and 3-axis magnetometer. With a flight computer that computes the navigation state only from IMU measurements, it is known as an inertial navigation system (INS). Fig. 4b shows Crescent, an OEM GPS receiver from Hemisphere GPS. In addition to the differentially-corrected position and velocity estimates, Crescent also outputs raw pseudorange and carrier phase measurement. These features enable a great deal of navigation research at the UMN UAV Laboratory. Crescent's small form-factor and low power consumption also make it a very suitable choice for UAV applications.

Airspeed, angle-of-attack (α), and sideslip angle (β) are known as the airdata quantities, which are traditionally measured using airdata sensors such as pitot tube and wind vane. Another alternative for measuring these angles is to use a multiple hole pitot tube such as the 5-hole pitot tube as shown in Fig. 5. Recent work [Lie and Gebre-Egziabher, 2012] has shown that this airdata can also be estimated by using sensor fusion from the IMU and GPS.

The navigation state is of particular interest. INS/GPS integration has been studied extensively in the past decades to estimate aircraft's navigation state. This section will focus on the implementation aspects of an integrated navigation systems using inertial sensors and GPS (INS/GPS system). A brief description of INS/GPS is provided; a more in-depth discussion can be found in [Gleason and Gebre-Egziabher, 2009, Groves, 2008, Farrell and Barth, 1999, Titterton and Weston, 2004]. System architecture and implementation challenges are discussed next. Following these descriptions, FASER's flight test result is presented.



(a)



(b)

Figure 4: Navigation Sensor of UMN UAV: ADIS16405 (a) and Hemisphere Crescent OEM GPS Receiver (b)

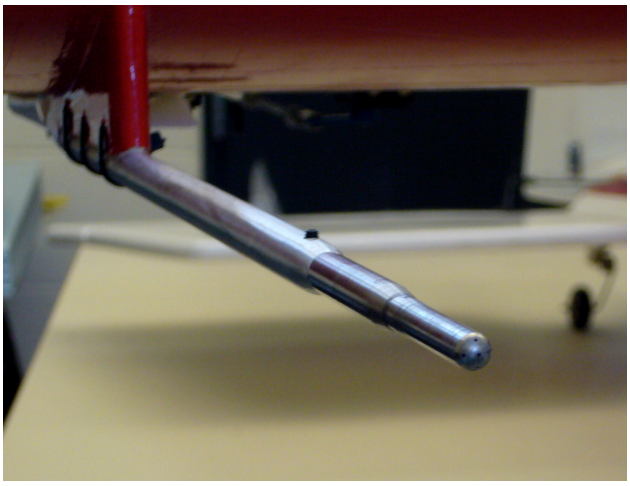


Figure 5: UMN's Thor instrumented with five-hole pitot tube from Goodrich to measure airspeed, α, and β

3.1 INS/GPS Integrated Navigation System

An inertial navigation system (INS) uses of the output of inertial sensors to estimate the vehicle's position, velocity, and attitude. A complete six degree-of-freedom inertial sensor consists of 3-axis accelerometers and 3-axis gyros. The accelerometer measures the specific force acting on the platform and the gyros measures its rotation. Inertial sensors can be categorized according to its resulting navigational accuracy [Gleason and Gebre-Egziabher, 2009, Gebre-Egziabher, 2001]. Automotive-grade micro electro-mechanical system (MEMS) inertial sensors are most suitable for low-cost UAV applications; however, when operating as a standalone navigator, these sensors produce positioning errors on the order of several hundreds meter per minute. These large errors in the position, velocity, and attitude estimates are mainly due to sensor bias and noise that corrupt the measurements. They are also a function of the initial error on the state estimates. The position error grows linearly with the initial velocity error estimate; quadratically with uncorrected bias; and at a cubic rate with attitude error [Titterton and Weston, 2004]. Despite the unbounded growth of the error with

time, one of the important advantages of inertial navigators is that they are self-contained; that is, they require no external signal to provide a navigation solution. In terms of the data rate, the navigation states can be computed at a rate limited only by the processing power of the flight computer.

Contrast GPS to the advantages and disadvantages of an inertial navigator: GPS requires an external signal to operate, and although the navigation solution accuracy depends on the signal quality and the geometry of the satellite in-view, this error is not a function of time [Misra and Enge, 2001]. Although there are specialized GPS receivers that can generate data up to 100 Hz, most low-cost GPS receivers provide output data at 1-10 Hz. In other words, although GPS navigation solution has long-term stability, the bandwidth of the solution is much lower than that of an inertial navigator.

Integration of INS with GPS allows a navigation solution that has the high bandwidth of the inertial sensors and the drift-free long term stability of the GPS solution. Attitude determination is an integral part of strapdown INS. This is because the specific force measured by the accelerometer needs to be transformed into the navigation frame in which the position and velocity are calculated. This transformation calls for the knowledge of the orientation of the platform (defined as the attitude).

Attitude can be equivalently described by a set of three angles known as the Euler angle sequence or the quaternion. An Euler angle sequence consists of the yaw (ψ), pitch (θ), and roll (ϕ) angles that describes three successive rotations about the body z, y, and x axes respectively. Although this representation carries physical interpretation, it is singular at $\theta = \pm 90^o$. Quaternion is a set of four numbers that can be related to the roll, pitch and yaw angles using the following relationship.

$$\begin{aligned} \phi &= \arctan\left(\frac{2q_2q_3 + 2q_0q_1}{2q_0^2 + 2q_3^2 - 1}\right) \\ \theta &= \arcsin\left(-2q_1q_3 + 2q_0q_2\right) \\ \psi &= \arctan\left(\frac{2q_2q_3 + 2q_0q_1}{2q_0^2 + 2q_3^2 - 1}\right) \end{aligned} \tag{1}$$

The quaternion $\mathbf{q} = [q_0 \quad q_1 \quad q_2 \quad q_3]$ represents the transformation from the local North-East-Down coordinate to the body-fixed frame. Although it has no singularity, directly visualizing the aircraft's orientation can be challenging. Hence for remote pilot display purposes and most control algorithms, the quaternion attitude needs to be transformed into its corresponding roll, pitch, and yaw angles. For applications where only benign maneuvers are expected, the Euler angle representation can instead be used.

Attitude determination using gyros output calls for integrating the angular velocity to propagate the attitude forward in time. Since gyros measures inertial rotation, they must be compensated to account for both the earth's rotation rate and the transport rate [Groves, 2008] that due to the earth's curvature. For most low-cost UAV applications, these terms are small ($\sim 10^{-5}$ rad/sec) compared to the noise level in the sensors. Hence, they can be neglected for all practical purposes. When calculating attitude at the IMU

sample rate, the following equations can be used: for an Euler angle attitude representation:

$$\begin{bmatrix} \phi[k+1] \\ \theta[k+1] \\ \psi[k+1] \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi[k])\tan(\theta[k]) & \cos(\phi[k])\tan(\theta[k]) \\ 0 & \cos(\phi[k]) & -\sin(\phi[k]) \\ 0 & \sin(\phi[k])\sec(\theta[k]) & \cos(\phi[k])\sec(\theta[k]) \end{bmatrix} \begin{bmatrix} \omega_x^B[k]\Delta t \\ \omega_y^B[k]\Delta t \\ \omega_z^B[k]\Delta t \end{bmatrix} \tag{2}$$

and for a quaternion attitude representation:

$$\mathbf{q}[k+1] = \mathbf{q}[k] \otimes \begin{bmatrix} 1 & \frac{1}{2}\omega_x[k]^B\Delta t & \frac{1}{2}\omega_y[k]^B\Delta t & \frac{1}{2}\omega_z[k]^B\Delta t \end{bmatrix} \tag{3}$$

where $\otimes$ operator is the quaternion multiplication operator. Details on quaternion algebra can be found on [Lefferts et al., 1982].

In order to improve the navigation solution between subsequent GPS solutions, the filter makes frequent corrections to compensate for the inertial sensor errors. Although more sophisticated sensor error models exist, a simplified model presented in [Gebre-Egziabher, 2001] is used in this implementation. In this model, the sensor output (e.g. gyro) as a function of time can be written as:

$$\omega(t) = \tilde{\omega}(t) + b_{gs} + b_{gd}(t) + w_g \tag{4}$$

where $\tilde{\omega}(t)$ represents the true angular velocity, b_{gs} is the turn-on bias, $b_{gd}(t)$ is a time-varying bias, and w_g is measurement noise that can be regarded as white noise. $b_{gd}(t)$ is modeled as first-order Gauss-Markov process, written as:

$$\dot{b}_{gd}(t) = -\frac{1}{\tau} b_{gd}(t) + w_b \tag{5}$$

This model is robust to parameters that are unobservable when the UAV is not accelerating [Gleason and Gebre-Egziabher, 2009]. Using this model, the estimated sensor bias is not simply the true bias corrupting the measurement; it also accounts for all unmodeled errors that corrupt the sensor measurement.

The architecture of the INS/GPS filter using an Extended Kalman Filter (EKF) [Simon, 2006] is shown on Fig. 6. Included in the state vector (Euler angle representation) are:

$$\mathbf{x} = [\ \underbrace{L \quad \Lambda \quad h}_{\text{Position}} \quad \underbrace{V_{\text{North}} \quad V_{\text{East}} \quad V_{\text{Down}}}_{\text{Groundspeed}} \quad \underbrace{\phi \quad \theta \quad \psi}_{\text{Attitude}} \quad \underbrace{b_{ax} \quad b_{ay} \quad b_{az}}_{\text{AccelBias}} \quad \underbrace{b_{gx} \quad b_{gy} \quad b_{gz}}_{\text{GyroBias}}\ ] \tag{6}$$

Executed at the IMU sample rate is the time-update process.

$$\begin{bmatrix} \phi[k+1] & \theta[k+1] & \psi[k+1] \end{bmatrix}^T = f(\phi[k], \theta[k], \psi[k]) \cdot \omega^B[k-1] \cdot \Delta t \tag{7}$$

$$\mathbf{V}^N[k+1] = \mathbf{V}^N[k] + C_B^N[k]\mathbf{f}^B[k] \cdot \Delta t \tag{8}$$

$$\begin{aligned} h[k+1] &= h[k] - V_{\text{Down}[k]} \cdot \Delta t \\ \Lambda[k+1] &= \Lambda[k] + \frac{V_{\text{East}}[k]}{R_E + h[k]} \cdot \Delta t \\ L[k+1] &= L[k] + \frac{V_{\text{North}}[k]}{R_E + h[k]} \cdot \Delta t \end{aligned} \tag{9}$$

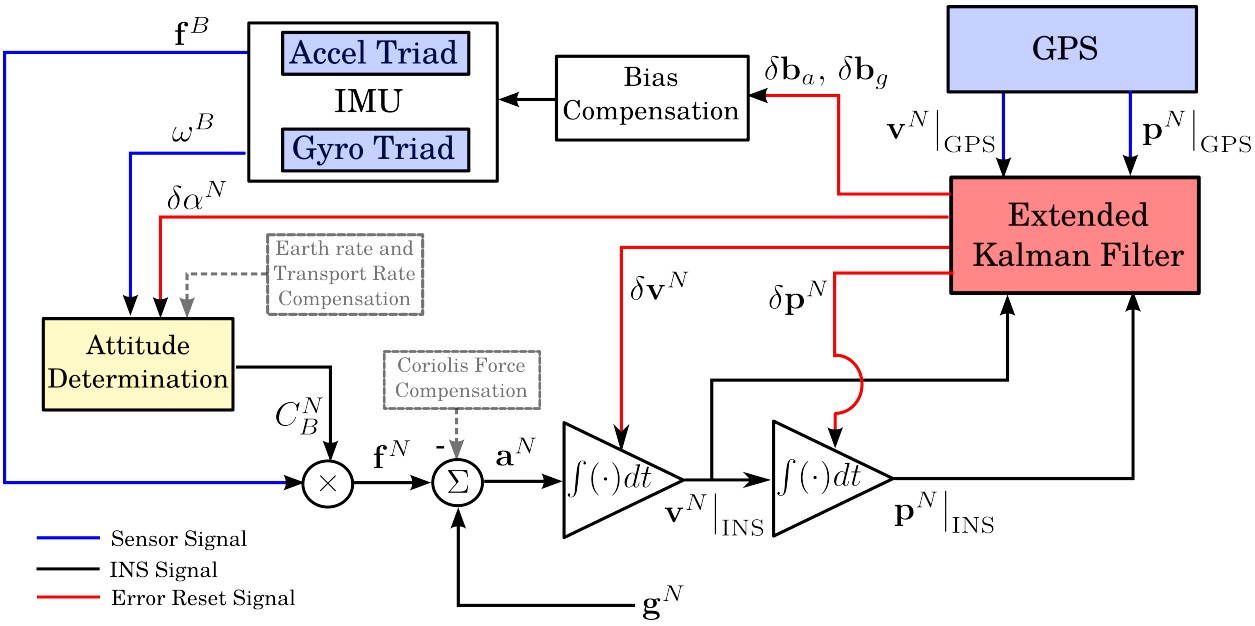


Figure 6: Block Diagram of INS/GPS Integration

The covariance is updated according to

$$\mathbf{P}^{(-)}[k+1] = \Phi[k]\mathbf{P}^{(+)}[k]\Phi[k]^T + \mathbf{Q}[k] \tag{10}$$

The state transition matrix can be approximated as $\Phi[k] = \mathbf{I} + \mathbf{F}[k] \cdot \Delta t$. $\mathbf{F}$ is the Jacobian of the time-update equations and $\mathbf{Q}$ is the process noise covariance matrix. Details on the elements of $\mathbf{F}$ and $\mathbf{Q}$ can be found in [Gleason and Gebre-Egziabher, 2009].

When GPS position and velocity becomes available at time-step k, the measurement update process is outlined as follows. The measurement, $\mathbf{y}$, is defined as follows.

$$\mathbf{y}|_{\text{GPS}} = \left[\begin{array}{cccccc} p_N & p_E & p_D & V_{\text{North}} & V_{\text{East}} & V_{\text{Down}} \end{array} \right]\Big|_{\text{GPS}} \tag{11}$$

where p_N, p_E and p_D are the reported North-East-Down position with respect to a specific location, e.g. the initial position.

Define the state error as

$$\delta\mathbf{y} = \mathbf{y}|_{\text{GPS}} - \left[\begin{array}{cccccc} p_N & p_E & p_D & V_{\text{North}} & V_{\text{East}} & V_{\text{Down}} \end{array} \right]\Big|_{\text{INS}[k]} \tag{12}$$

The covariance is updated using:

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times9} \end{bmatrix} \\ \mathbf{K}[k] &= \mathbf{P}^{(-)}[k]\mathbf{H}^T\left(\mathbf{R} + \mathbf{H}\mathbf{P}^{(-)}[k]\mathbf{H}^T\right)^{-1} \\ \mathbf{P}^{(+)}[k] &= \left(\mathbf{I} - \mathbf{K}[k]\mathbf{H}\right)\mathbf{P}^{(+)}[k]\left(\mathbf{I} - \mathbf{K}[k]\mathbf{H}\right)^T + \mathbf{K}[k]\mathbf{R}[k]\mathbf{K}[k]^T \end{aligned} \tag{13}$$

The state is updated according to:

$$\delta\mathbf{x} = \mathbf{K}[k]\delta\mathbf{y} \tag{14}$$

Position:

$$\begin{aligned} h^{(+)}[k] &= h^{(-)}[k] - \delta\mathbf{x}(3) \\ \Lambda^{(+)}[k] &= \Lambda^{(-)}[k] + \frac{\delta\mathbf{x}(2)}{R_E + h^{(+)}[k]} \\ L^{(+)}[k] &= L^{(-)}[k] + \frac{\delta\mathbf{x}(1)}{R_E + h^{(+)}[k]} \end{aligned} \tag{15}$$

Velocity:

$$\begin{aligned} V_{\text{North}}^{(+)}[k] &= V_{\text{North}}^{(-)}[k] + \delta\mathbf{x}(4) \\ V_{\text{East}}^{(+)}[k] &= V_{\text{East}}^{(-)}[k] + \delta\mathbf{x}(5) \\ V_{\text{Down}}^{(+)}[k] &= V_{\text{Down}}^{(-)}[k] + \delta\mathbf{x}(6) \end{aligned} \tag{16}$$

When using Euler angle representation, the attitude is updated using the transformation matrix C_B^N.

$$C_B^N(+) = (\mathbf{I} - [\delta\mathbf{x}(7:9)\times]) \cdot C_B^N(-) \tag{17}$$

where $[\cdot\times]$ indicates the skew-symmetric matrix operator. Euler angles can be derived from the updated transformation matrix, $C_B^N(+)$. Accelerometer and gyros bias are updated as follows:

$$\mathbf{b}_a^{(+)}[k] = \mathbf{b}_a^{(-)}[k] + \delta\mathbf{x}(10:12) \tag{18}$$

$$\mathbf{b}_g^{(+)}[k] = \mathbf{b}_g^{(-)}[k] + \delta\mathbf{x}(13:15) \tag{19}$$

3.2 Implementation and Practical Challenges

The avionics suite in the UMN's UAV is implemented on a 32-bit PowerPC Phytec MPC5200B-tiny SoM. It utilizes a real-time operating system (RTOS) and flight software written in C. This computer handles data acquisition; performs guidance, navigation, and control tasks; stores relevant data; and sends information to the ground control station via the telemetry modem. The flight software uses a multi-threaded architecture in which all of the flight critical tasks execute in the highest priority thread at 50Hz, while additional tasks (i.e., those not required to control the aircraft, such as a fault detection filter) are executed in separate, lower priority threads [Murch, 2012a]. As shown in Fig. 7, there are ten modules executed in the highest priority thread (thread 0). Navigation tasks are executed immediately after data acquisition (DAQ). It starts as valid GPS measurements are available. The position and velocity are initialized at the reported position and velocity. Since this initialization occurs on the ground, attitude is initialized to a known attitude of the aircraft on the ground.

One of the key aspects to the correct fusion of inertial sensors and GPS information is aligning data acquisition from non-timing sensors, such as an IMU, with sensors that have inherent timing capability such as GPS. Data acquisition to an IMU is achieved by polling the IMU at the desired sample rate. GPS receivers, however, stream out their navigation solutions (position and velocity) at a preset rate (e.g., 1 Hz). This rate is usually driven by a the receiver clock that has been steered to the GPS time. Despite being precise, the flight computer clock might be running at a different rate from that of the GPS clock. When the IMU data stream is not aligned with the GPS output, incorrect measurements are fed to the EKF with resulting errors in the vehicle state estimates. The magnitude of the error depends on the acceleration experienced by the aircraft and the clock rate difference.

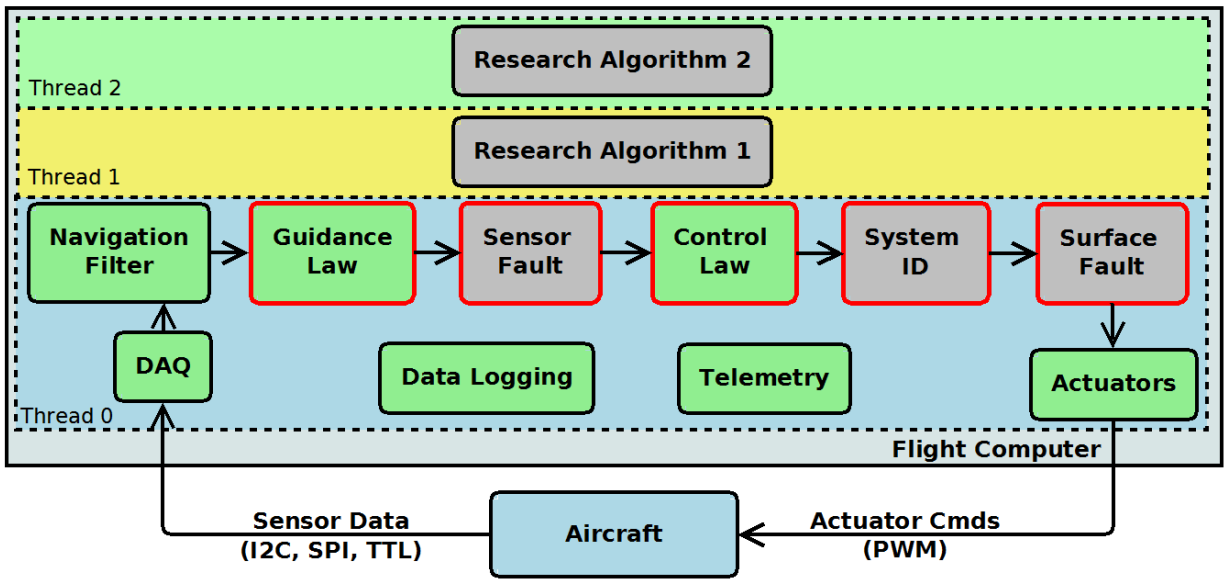


Figure 7: Flight Software Architecture

There are many ways to synchronize IMU and GPS outputs. One way to do this is by performing data acquisition at a deterministic sample rate. By checking the availability of GPS solution at the highest rate available in the flight computer, IMU and GPS data are always synchronized within the resolution of one sampling interval. In this architecture, this is achieved by using the alarm functions provided by the RTOS kernel which are used to trigger module execution. The timing diagram that shows the execution time of each module is shown in Fig. 8.

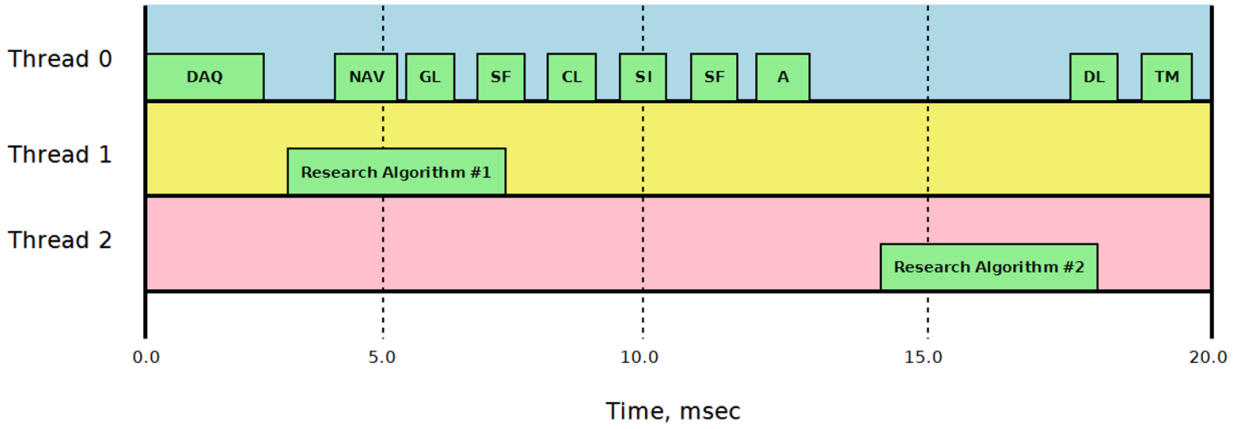


Figure 8: Flight Software Timing Diagram

Tuning the extended Kalman filter is known to improve filter convergence time and short term accuracy [Gleason and Gebre-Egziabher, 2009]. However, in order to tune the filter, a truth reference is needed so that a comparison can be made to the resulting filter estimate. Although tuning the filter is an art, experience can help smooth the process. Having a truth

reference system might be prohibitive for many, it is recommended that the data set used for tuning is from a known trajectory. This helps to develop a feel for how the errors grow and how effective the bias compensations are. Since the states' observability depends highly on the acceleration experienced by the vehicle, it is important that a trajectory with sufficient acceleration is used: acceleration and deceleration on a straight line path and frequent turns should be included. It might also be worth testing the filter performance both during powered and gliding operation since structural vibration might adversely affect the filter performance. Adding damping to the system or simply moving the IMU placement on the UAV can often alleviate vibration problems.

Lastly, when the IMU is not collocated with the GPS antenna, the distance between the IMU and the GPS antenna results in different velocity and position measured by both sensors. This is known as the lever arm problem, and the Kalman filter must account for it to make correct state estimates. There are generally two ways to account for this error. The first way is to transfer the GPS measurement to the location of the IMU by using the information of the aircraft attitude and the known lever arm vector in the body frame. The dependency on estimated attitude to transfer this measurement might lead to inaccurate GPS position and velocity estimates. The second way is by inflating the GPS measurement covariance. This method fails when the inflation factor is so large that it renders the GPS position and velocity useless. For most UAV applications, however, the distance between the GPS antenna and the IMU is not large; thus only a very small inflation required to compensate for this error. In this work, the latter method is employed.

3.3 Flight Test Result

Fig. 9 shows the trajectory flown during one of FASER's flight test. The navigation data are extracted from the the flight computer's data recorder are plotted on Fig. 10. During the flight, there were several GPS outages and the navigation state estimates during these period of free-inertials are plotted in green. When bias corrections are continuously fed to the IMU, this inflight calibration improves the performance of the inertial navigation and slows down error growth during these GPS outages.

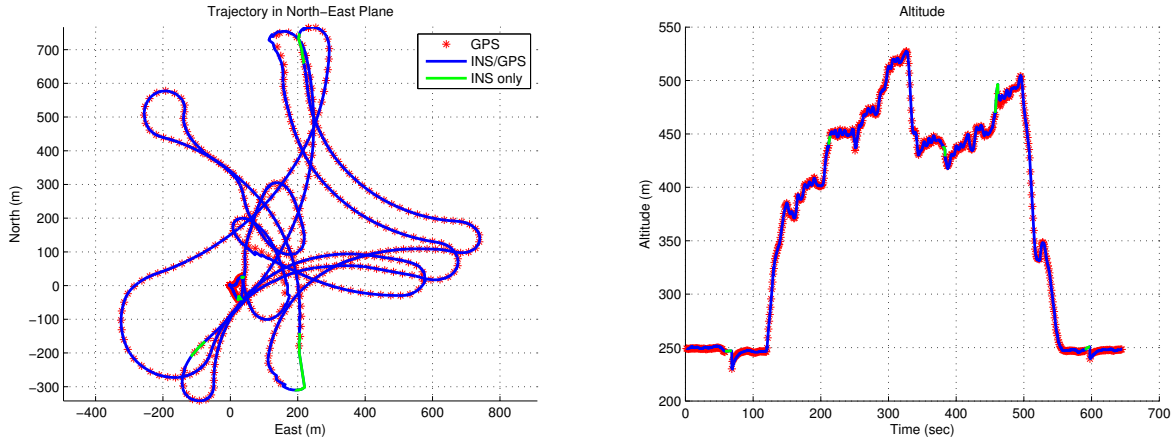


Figure 9: Flight test trajectory

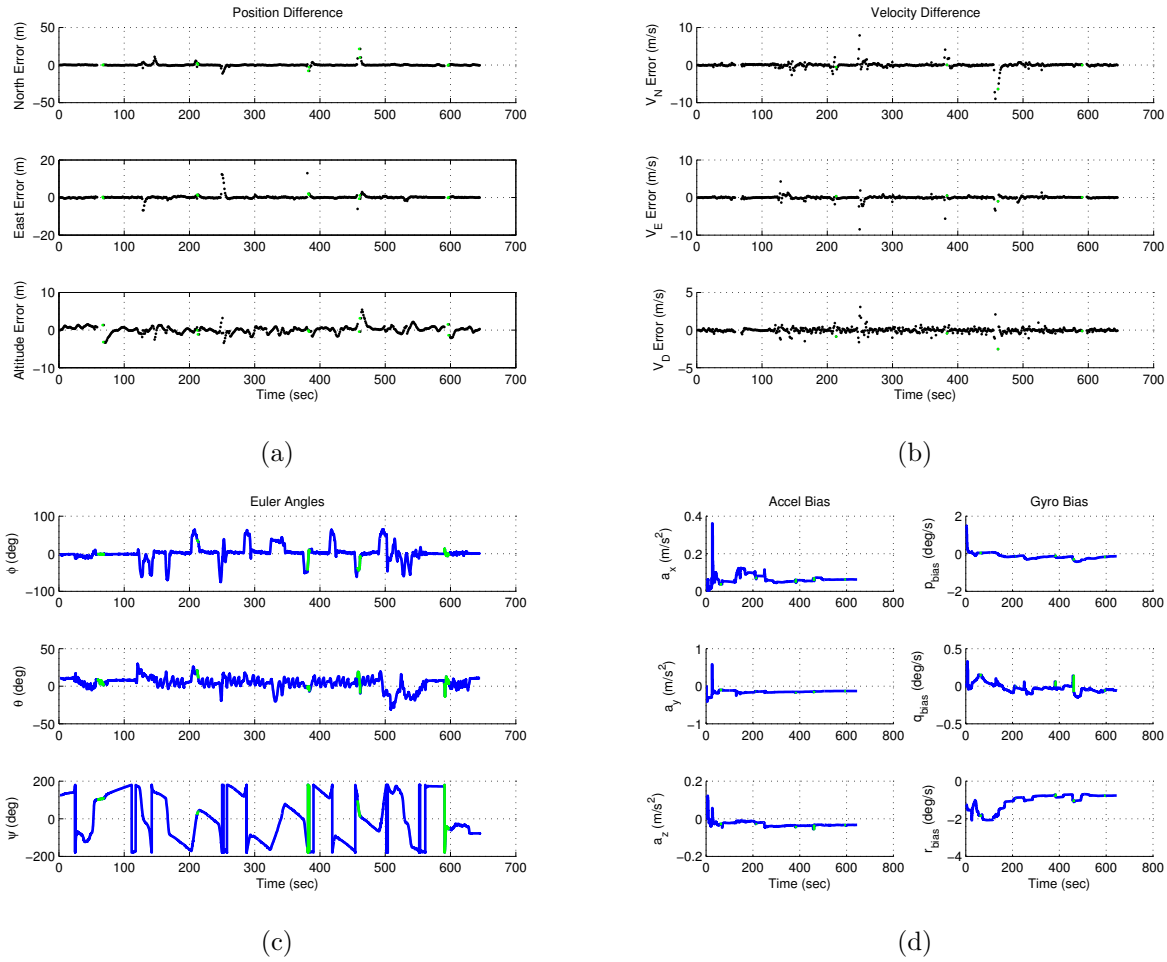


Figure 10: INS/GPS solution: position error (a), velocity error (b), attitude estimate (c), and sensor bias estimate (d)

In Fig. 10a and 10b, the position and velocity estimates from INS/GPS integration system are differenced with the velocity from the GPS measurement. Due to lack of a reference (truth) system, it is not possible to compare the attitude and bias estimates. Hence, the attitude and bias estimates from the INS/GPS filter are plotted on Fig. 10c and 10d. Qualitative evaluation of the filter performance indicates good performance. This can be seen from the smoothness of the INS/GPS solution and good convergence of the inertial sensor bias estimates. As such, the INS/GPS integration provides a high-bandwidth solution with long-term stability suitable for UAV control.

4 Inner Loop Control

In order to fly an aircraft, a low level control system must stabilize the airframe using available sensor inputs and actuators. A higher level outer loop control will implement path following (see Sec. 5) while the inner loop keeps the aircraft flying.

There are myriad ways to implement an inner loop control on a UAV. In this chapter the inner loop controller is based on the SLUGS autopilot [Lizarraga, 2009], [ASL,], and

[SLU,], which uses relatively simple PID controllers in the inner loop. The PID-based inner loop control that has flown on the SLUGS platform is presented, and actual flight data of the implementation is included.

The UCSC SLUGS autopilot is divided into two hardware sections: a control processor and a sensor processor. The sensor processor is tasked with taking the raw sensor measurements and fusing them into a high quality position and attitude estimate (similar to Sec. 3). The control processor is tasked with both the inner loop (stabilization) and the outer loop (guidance). The SLUGS implementation is illustrated in Fig. 11.

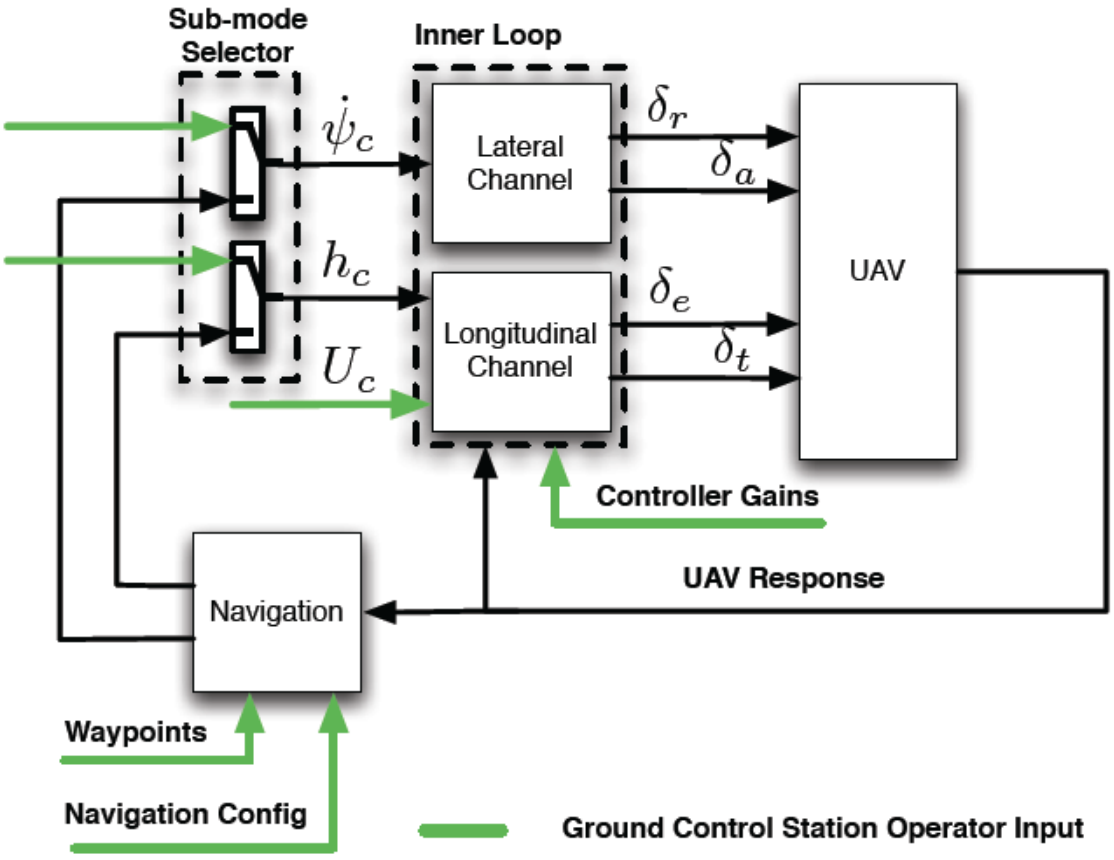


Figure 11: SLUGS block diagram of inner loop and guidance

While there are many different ways to implement a low level control system, the one presented is based on decoupling the longitudinal and lateral flight dynamics into two separate control systems [Stewart, 2001]. The lateral control channel controls rudder and ailerons, while the longitudinal control handles throttle and elevator. While this is reasonable for traditional aircraft-like UAV's flying gentle maneuvers, it is not appropriate for aircraft with high degrees of lateral-longitudinal cross coupling, nor for aircraft performing aggressive aerobatic maneuvers.

Furthermore, each of the two main (lateral and longitudinal) controllers are divided into successive loop closure using Proportional-Integral-Derivative (PID) controllers. PID

controllers are simple to tune experimentally, and while they can theoretically perfectly stabilize a second order plant, in practice they work quite well with higher order plants. Some attention must be paid to integrator wind-up and the use of a numerical derivatives in the PID loop, or the resulting control will have poor performance. Lastly, these controllers are implemented digitally, in order to accommodate the actual autopilot hardware.

4.1 PID Control

As stated above, the Proportional-Integral-Derivative (PID) control can perfectly stabilize a second order plant, given the right gains. However, in practice, it can usually perform well in a variety of settings without the need for a precise model of the underlying plant (a large advantage when system identification is either difficult or imprecise). The classical PID controller consists of an input derived from the measurement of the system and the desired reference for it to track, and an actuator signal on the output. The difference between the measurement and the reference, referred to as the error, is fed into the PID block and an output is generated from the three gains: K_p, the proportional gain is multiplied directly by the error, K_d, the derivative gain is multiplied by the time rate of change of the error, and lastly, K_i, which multiplies the integral of the error.

At its most basic form, the PID control simply consists of three gains, with a first order difference for the derivative and a running sum for the integral. Several embellishments are used to make the PID controller behave better at the corner cases. Firstly, the derivative is changed from the time rate of change of the error to the time rate of change of the measurement; this is done so that the PID controller does not throw the output due to a reference change. If a direct derivative signal is available (for instance pitch rate gyro), then that is used in preference to the first order difference. Also, the first order difference can be changed to longer time steps if noise sensitivity is an issue.

Secondly, saturation limits are included to clip the output if it would saturate the actuator. The integral is changed from a running sum to trapezoidal integration, and anti-windup logic is included. Integral state windup is caused by the integrator continuing to integrate even after the control is saturated. This creates a memory effect within the controller that causes overshoot and degrades the controller performance when coming out of saturation. In order to prevent this, an anti-windup scheme is implemented which checks if the actuator would saturate on the current time step, and does not perform the integration if this is the case.

The integrator state can be reset under software control, such that when the control is engaged, a bumpless transfer is achieved. Furthermore, the integral state can be manipulated when changing the control gains, such that the output remains stable as the new gains are latched into the controller. Again, all of this is done to maintain smooth control for flight.

4.2 Lateral Control

The lateral controller (Fig. 12) uses the rudder and ailerons to keep the aircraft flying in a coordinated turn, and following a commanded turn rate (including a zero turn rate for straight flight). The lateral dynamics of an aircraft include the roll rate damping mode, a spiral mode, and the dutch roll mode (yaw-roll coupling). In this formulation of the inner loop control, yaw rate, $\dot{\psi}_c$ is commanded, and is converted to a commanded roll angle, ϕ_c

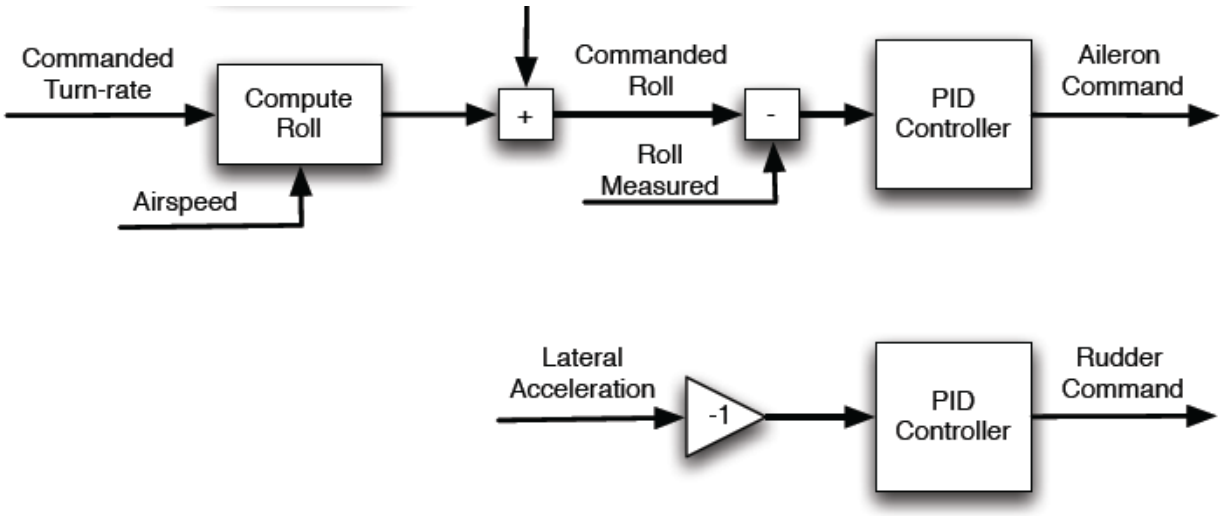


Figure 12: SLUGS lateral inner loop control

using the formulation:

$$\phi_c = \arctan(\frac{\dot{\psi}_c U_m}{g}) \tag{20}$$

where U_m is the measured airspeed, and g is gravity. Eq. 20 assumes that the aircraft is flying in a coordinated turn, that is where the turn rate is constant and the body fixed lateral acceleration, a_y, is zero.

The commanded roll angle, ϕ_c, is used as the reference to the PID control, with the plant output being the actual roll angle, ϕ that comes from the attitude estimation algorithm. The commanded bank angle is limited with a saturation block to keep the roll angle from becomming too extreme. In the case of the SLGUS small UAV, this is limited to $\pm 40°$.

The output generated by the PID block is to the ailerons, which are used to drive the roll error (commanded $-$ actual) to zero. In this case, the derivative of the roll error is taken directly from the body fixed gyros (with the bias again taken care of by the attitude estimation algorithm). That is, while the full derivative of roll rate:

$$\dot{\phi} = p + (q \sin\phi + r \cos\phi) \tan\theta \tag{21}$$

where $[p, q, r]$ are the body fixed roll, pitch, and yaw rates, and ϕ, θ are the roll and pitch euler angles respectively. Note that with a small pitch angle and yaw rate, this can be approximated as:

$$\dot{\phi} \simeq p \tag{22}$$

Lastly, when the roll angle command, ϕ_c is the opposite polarity from the previous command, the integral state is reset to improve roll tracking performance.

The above describes the turn rate command to aileron control system, which will drive the roll error to zero. However, Eq. 20 depends on coordinated flight to be effective. In order to ensure coordinated flight, a second PID loop is closed around to rudder. The commanded input to the turn coordinator PID block is the negative of the lateral acceleration (this

accounts for the fact that the rudder is behind the center of mass of the aircraft, and has a negative sign in the transfer function), and the output is the rudder actuator command.

The PID loop will actuate the rudder in order to drive the body fixed lateral acceleration, a_y to zero. Here, again, the fact that the output of the aircraft affected by the rudder has a direct measurement of its derivative from the body fixed gyros can be used for the D term. That is:

$$\dot{\psi} = \frac{1}{\cos\theta}(q\sin\phi + r\cos\phi) \tag{23}$$

which for small angles of pitch and roll, can be reliably approximated as:

$$\dot{\psi} \simeq r \tag{24}$$

Note that the turn coordinator, with the derivative feedback from the body fixed yaw rate gyro, r, has the effect of also acting as a yaw damper for the aircraft. More traditional autopilots often use the rudder solely as a yaw damper, without the turn coordination function, and rely on the directional stability of the aircraft to keep the turns coordinated. In that case, the rudder feedback command would be based on a high passed version of the yaw rate gyro. Experience with small UAVs indicates that the former approach works better for model scale aircraft, giving both coordinated turns and decent yaw damping.

4.3 Longitudinal Control

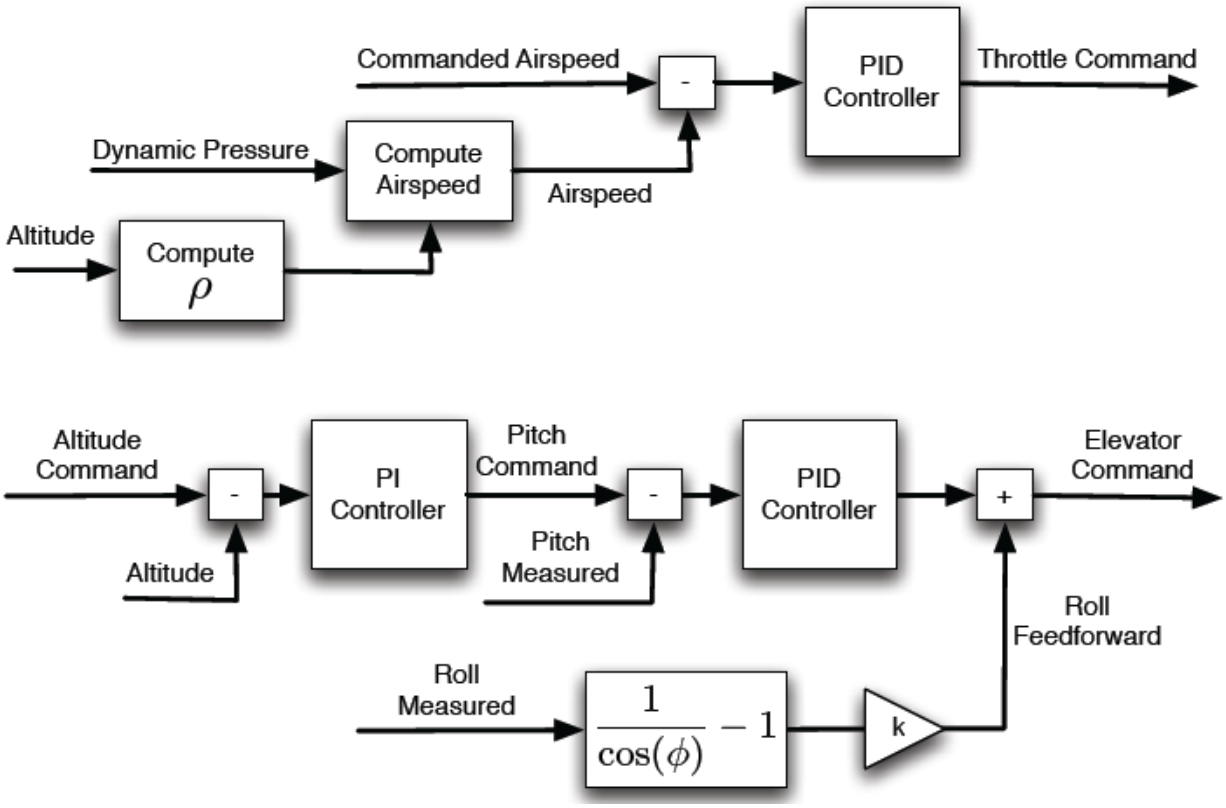


Figure 13: SLUGS longitudinal inner loop control

The longitudinal low level control loops (Fig. 13) operate similarly to the lateral ones, with the other two flight controls being throttle and elevator. There are two modes of operation for a longitudinal low level controller: *climb/descent* and *level flight*. In *climb/descent* mode, the throttle is set to a fixed value (high for climb, low for descent) and the airspeed and climb rate are controlled through the aircraft pitch via the elevator.

In *level flight* mode, airspeed is controlled directly by the throttle, and altitude by aircraft pitch via the elevator. Since airspeed is being kept constant, the altitude responds rapidly to even small changes in aircraft pitch. As a design choice, only the *level flight* mode is implemented, given that the UAV will spend most of its mission in level flight at constant altitude following waypoints.

The airspeed hold control loop takes in a commanded airspeed, U_c, and compares this to the measured airspeed. Note that the airspeed control uses airspeed, and not ground speed (as available from GPS). Compensation for wind is handled within the higher level guidance controller, rather than in the low level stabilizing controllers.

The aircraft does not, however, have an onboard measurement of airspeed. Rather, it has an onboard measurement of dynamic pressure, q, and altitude, from which the airspeed can be extracted. That is:

$$U_m = \sqrt{\frac{2q}{\rho}} \tag{25}$$

where q is the dynamic pressure, and ρ is the atmospheric density. However, ρ is a function of altitude, such that:

$$\rho = \rho_0(1 - \frac{h_m}{44331})^{4.255876} \tag{26}$$

where ρ_0 is the sea-level air density ($1.025 Kg/m^3$) and h_m is the measured altitude in meters. This approximation is good for most altitudes reachable by UAVs, and is sufficient for control. The output of the PID block is the throttle actuator. In the case of the airspeed, there is no direct derivative to be used in the throttle loop (though the body fixed longitudinal acceleration could be used). As such, some care must be used in both the derivatives, and in tuning the airspeed hold loop in general.

Too aggressive gains on the airspeed hold loop causes the throttle to surge and cut in flight, with unpleasant results. This is due both to the noise on measured airspeed and also due to the lag in response to the throttle input. Here tuning the gains for an acceptable error and relying on the integral term to close the error works well.

The altitude control loop consists of two chained PID controllers. The first is a straight PI controller (the derivative gain is set to 0), that takes in commanded altitude, h_c and measured altitude, h_m as its error, and outputs a pitch command. The pitch command is saturated at $\pm 15^\circ$ with the integral windup stopped if the saturation is reached. The limited pitch command is used as a reference input for the second PID loop, and is compared to the aircraft pitch, θ, from the attitude estimation to generate the error. The output of the second PID loop drives the elevator, which works to match the aircraft pitch attitude to the commanded one. Here, the derivative term comes straight from the body fixed gyros, with the full equation:

$$\dot{\theta} = q\cos\phi - r\sin\phi \tag{27}$$

which for small bank angles can be reduced to:

$$\dot{\theta} \simeq q \tag{28}$$

The direct derivative term allows the pitch to elevator PID control to be aggressive, yet still retain good damping characteristics. This PID controller will drive the aircraft pitch to the commanded pitch until the desired altitude is reached.

While the aircraft is flying straight and level, this control works very well. However, while the aircraft is in a turn, the aircraft will descend. This is because the lift from the wings must match the aircraft weight, but in a turn part of the lift is directed inwards to cause the turn itself. More precisely, in order not to lose altitude in a turn, the lift is related to the weight and the bank angle as:

$$L = \frac{W}{\cos\phi} \tag{29}$$

where L is lift, W is aircraft weight, and ϕ is roll angle. This is a simplified model assuming lift only from the wings, and not accounting for sideslip angles of the fuselage for knife-edge type flight. Using this model, the change in lift is:

$$\Delta L = W(\frac{1}{\cos\phi} - 1) \tag{30}$$

which indicates how much the lift must be increased to maintain altitude in a coordinated circular turn. Pilots are trained to increase elevator (back stick) when entering a turn to maintain altitude. In order to ensure that the UAV holds altitude during turns, a feedforward gain proportional to the increase in lift is included:

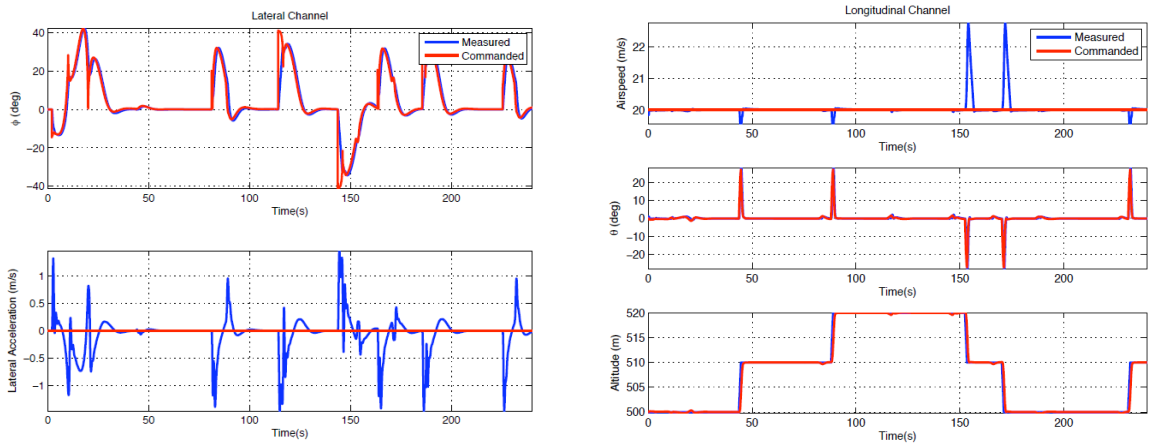
$$\delta e_{ff} = K_{ff}(\frac{1}{\cos\phi} - 1) \tag{31}$$

where δe_{ff} is the additional elevator command due to the feedforward term. Note that in the implementation, the roll angle, ϕ, is low pass filtered to reduce unwanted pitch oscillations resulting from attitude estimation noise, and is also limited to $\pm 60^{\circ}$. At angles beyond this amount, the UAV will simply enter into an accelerated stall trying to hold altitude, and the low level control system will no longer be able to stabilize the aircraft.

4.4 Trim Conditions

Before leaving the subject of the low level inner loop controllers, it is necessary to discuss aircraft trim conditions. In the case of the SLUGS small scale UAV, trim is established by the safety pilot before the autopilot is engaged. At the moment the autopilot is switched on, the inner control loops assume the aircraft is currently trimmed for straight and level flight.

The inner PID control loops above each output a *change* in actuator output summed with the original trim condition. Note that for the SLUGS autopilot, this trim is established by a human safety pilot. The trim could just as easily be established using a conventional 6DOF model of the airframe and computing trim for various flight conditions. That is, for every combination of airspeed, climb rate, and altitude, there is a throttle and elevator setting that will keep the UAV flying in steady state at those conditions. The ailerons and rudder surfaces are assumed to be set at zero for straight flight.



(a) (b)

Figure 14: Simulation of the inner loop control using the HIL simulator: (a) the lateral channel, (b) the longitudinal channel

4.5 Simulation and Flight Test Results

In order to validate the inner loop control, data from the HIL simulation is presented showing the results of the inner loop PID controllers. Fig. 14(a) shows the lateral controller. Note that at every change in commanded roll angle, the lateral acceleration receives a large disturbance, which must be washed out. In panel (b) is the longitudinal control, with both pitch and airspeed control. Again, note that the spikes out of speed are due to large climb or descent inputs to the controller.

Flight test data from the inner loop control is presented in Fig. 15, which shows excellent performance on pitch and roll commands, a reasonable attenuation of lateral acceleration (with a limit to less than 1g for most of the flight time), and a fairly good airspeed hold. This is in the presence of wind, gusts, and other disturbances. Note that the bias on the lateral acceleration is most likely from the accelerometer being mounted with a slight tilt on the airframe.

5 Guidance

UAV guidance, navigation, and control (GNC) ultimately signifies the ability to follow a desired trajectory through the sky. With attitude estimation established (Sec. 3), and inner loop stabilizing the aircraft (Sec. 4), what is left is to guide the UAV along the desired trajectory rejecting disturbances such as wind.

Again, while there are several ways to implement this guidance control, this section discusses the guidance algorithms implemented on the SLUGS autopilot [Lizarraga, 2009]. In order to deliver a mission to the UAV from the ground station, a simple set of GPS-based waypoints, along with an altitude and speed for each leg of the mission, is transmitted to the UAV via telemetry link. Furthermore, the interest is in flying the aircraft on the trajectory, and thus the legs are blended into each other using circular arcs rather than forcing the UAV to overfly each waypoint. Note that this is a design choice, and both variants have a utility depending on the specific mission.

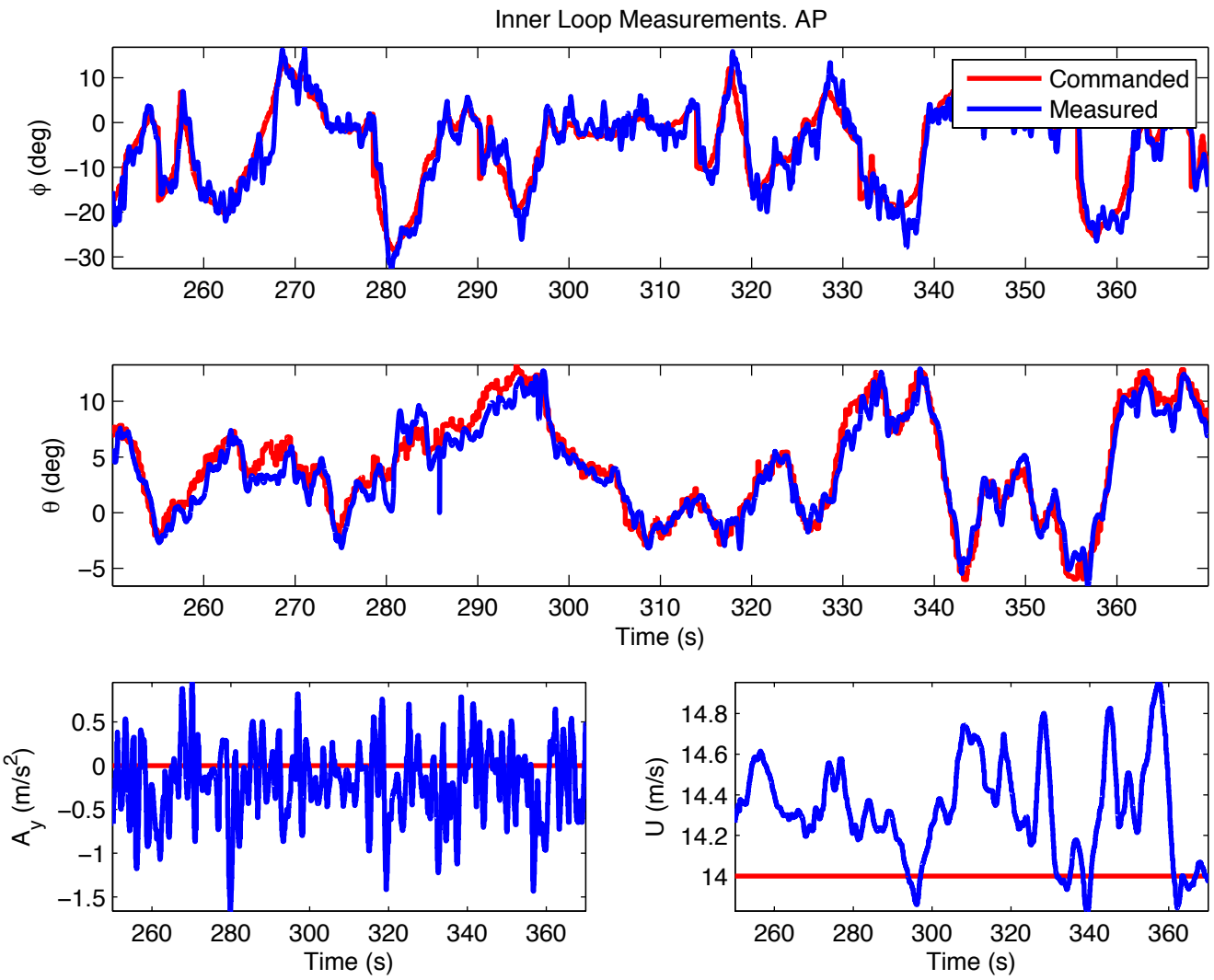


Figure 15: SLUGS inner loop flight test data.

The SLUGS guidance algorithm is based on an extension of a simple line-of-sight guidance law originally developed for ground robotics. Much of the art of waypoint guidance consists of determining which leg of the trajectory the UAV is on, and when to switch to the next leg of the trajectory.

The basic approach is to define a Serret-Frenet frame which points from the current waypoint to the next ($\vec{T}$, along the current leg), with one axis down and the third defined by the cross product between the along track, $\vec{T}$ and down, $\vec{B}$, unit vectors. The projection of the UAV position onto the $\vec{T}$ direction is used to determine when to switch to the next leg. In practice this is better than using a distance from the next waypoint to switch (especially in very windy conditions).

5.1 General Tracking

The outer loop guidance law follows closely the work developed in [Amidi and Thorpe, 1991], [Park et al., 2004], and [Park et al., 2007]. The original derivation is presented here for completeness. The guidance algorithm steers the velocity vector toward the line of sight; this is one form of pursuit guidance. Making the commanded acceleration proportional to $\sin\eta$ is only one of number of possible guidance laws. The basic guidance algorithm is to determine an aim point and steer the velocity vector towards it.

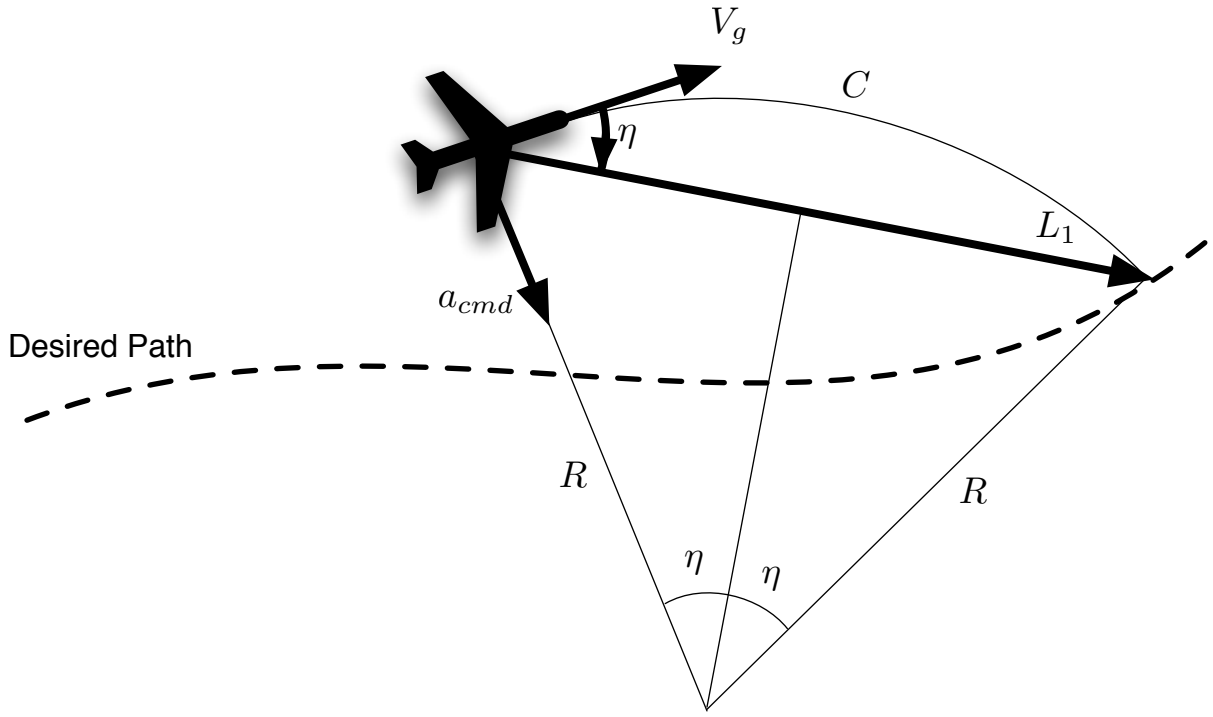


Figure 16: Navigation control law geometry. Reproduction of Figure 1 in [Park et al., 2004]

Referring to Fig. 16, V_g is the UAV's groundspeed and C is a circular arc of radius R that originates at the UAV and intercepts the desired path. L_1 is a constant lookahead distance from the UAV to the path in the desired direction of travel. From elementary trigonometry:

$$\frac{|L_1|}{2} = R\sin\eta. \tag{32}$$

Additionally, from elementary kinematics it is known that the centripetal acceleration, a_c, required for a point mass to follow the circular arc C is given by:

$$a_c = \frac{|V_g|^2}{R}, \tag{33}$$

Thus the UAV must command a lateral acceleration of a_c. Solving Eq. 32 for R and substituting it into Eq. 33 produces the following control law for commanded acceleration:

$$a_{cmd} = 2\frac{|V_g|^2}{|L_1|}\sin\eta. \tag{34}$$

The only requirements for the implementation of this control law are to select the lookahead distance $|L_1|$ and determine $sin\,\eta$, the sine of the angle from the velocity vector to L_1. η is sometimes called the line of sight angle. Choice of $|L_1|$ is analogous to feedback gain, with a larger L_1 corresponding to smaller gains; $sin\,\eta$ is found from the vector cross product of V_g and L_1.

$$\sin\eta = \frac{V_g \times L_1}{|V_g||L_1|} \tag{35}$$

For the UAV to actually track the desired trajectory, the lateral acceleration command must be converted to an appropriate bank angle command using the steady-state turn equation:

$$\phi_{cmd} = \tan^{-1}\frac{a_{cmd}}{g} \tag{36}$$

5.2 Straight Line Tracking

The guidance algorithm is most easily described using a straight line (though it is certainly not limited to such). In Fig. 17, the UAV is following a straight line segment from waypoint P_0 to P_1. A Serret-Frenet coordinate frame [Etkin, 2005] is attached to the initial waypoint, P_0, such thatthe calculations are always in local path coordinates.

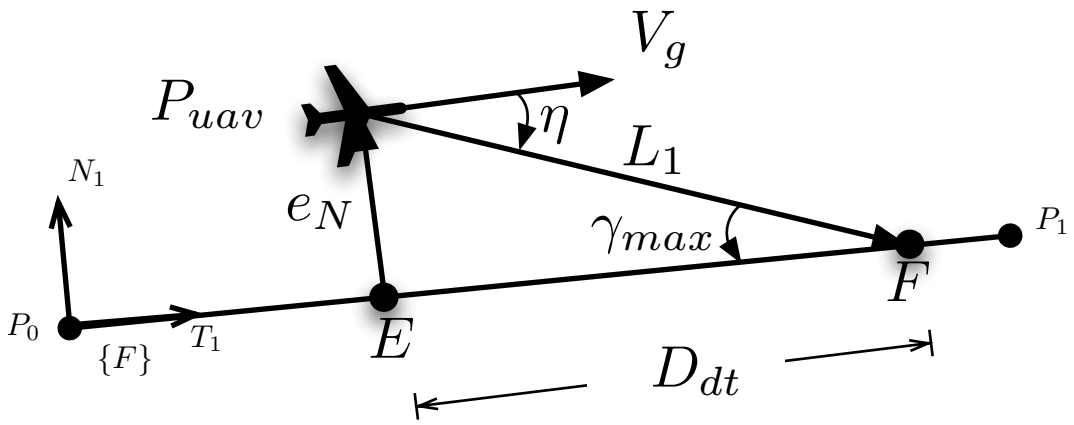


Figure 17: Geometry Description of the angle η computation.

$$\begin{aligned} T &= [x_T, y_T, z_T] = \frac{(P_1 - P_0)}{|P_1 - P_0|}, \\ N &= \frac{1}{|[-y_T \;\; x_T \;\; 0]|}[-y_T \;\; x_T \;\; 0]^\top, \\ B &= T \times N, \end{aligned} \tag{37}$$

Given that e_N is the the error in the N direction. Then

$$|e_N| = N^\top(P_{uav} - P_0), \tag{38}$$

E is the closest point on the path to the UAV. The intersection of L_1 with the desired path is shown by the point F. The down track distance or EF separation, denoted D_{dt} can be computed as:

$$D_{dt} = \sqrt{|L_1|^2 - |e_N|^2}, \tag{39}$$

with the vectors E and F given by:

$$E = P_{uav} - |e_N|N, \tag{40}$$

$$F = E + D_{dt}T. \tag{41}$$

From these relations, the vector L_1 can then be computed as:

$$L_1 = F - P_{uav} = D_{dt}T - |e_N|N. \tag{42}$$

which is then used in Eq. 35 to determine η and hence the commanded acceleration. For sufficiently small tracking errors, the L_1 guidance looks like a PD controller on lateral error [Park et al., 2007].

5.3 L_2^+ Control

The SLUGS autopilot extends the L_1 pursuit guidance to account for some shortcomings of the control law (and for clarity refers to this as L_2^+ control). Firstly, it was noticed during flight test experiments that the L_1 guidance exhibited large overshoots when turning downwind (hence an increasing V_g). Analysis showed that in order to solve this, the new L_2 vector should be a function of groundspeed. Thus, $|L_2| = \mathcal{T}^*|V_g|$, where $\mathcal{T}^*$ is a constant, and the commanded acceleration becomes:

$$a_{cmd} = 2\frac{|V_g|}{\mathcal{T}^*}\sin\eta. \tag{43}$$

Furthermore, when the lateral error, $|e_N|$ is larger than $|L_2|$, the aim point cannot be found. To enforce that an aim point is always found, define a maximum intercept angle, γ_{max}; the down-track distance to the aim point defined by γ_{max} is

$$D_{dt} = \frac{|e_N|}{\tan\gamma_{max}}. \tag{44}$$

as shown in the lower aircraft in Fig. 18.

When $|e_N|$ is greater than $|L_2|$, such as during initial intercept, D_{dt} may become too large. For example the aim point may go beyond the next waypoint. To prevent this a bound is placed on the down path distance of the aim point. This bound is set to be a constant, $\mathcal{M}^*_{\mathcal{DP}}$, times $|L_2|$

$$D_{dt_{min}} = \min(D_{dt}, |L_2| \cdot \mathcal{M}^*_{\mathcal{DP}}). \tag{45}$$

This is shown for the upper aircraft in Fig. 18. The down track distance of the aim point is:

$$D^*_{dt_{min}} = \begin{cases} D_{dt_{min}} & |e_N| > |L_2| \\ \max(D_{dt_{min}}, \sqrt{|L_2|^2 - |e_N|^2}) & |e_N| \leq |L_2| \end{cases} \tag{46}$$

If the aim point gets beyond P_1 the UAV will continue along this line without ever changing direction. First compute D_{wp_1}, the along-track distance from the UAV to P_1.

$$D_{wp_1} = T^\top(P_1 - P_{uav}) \tag{47}$$

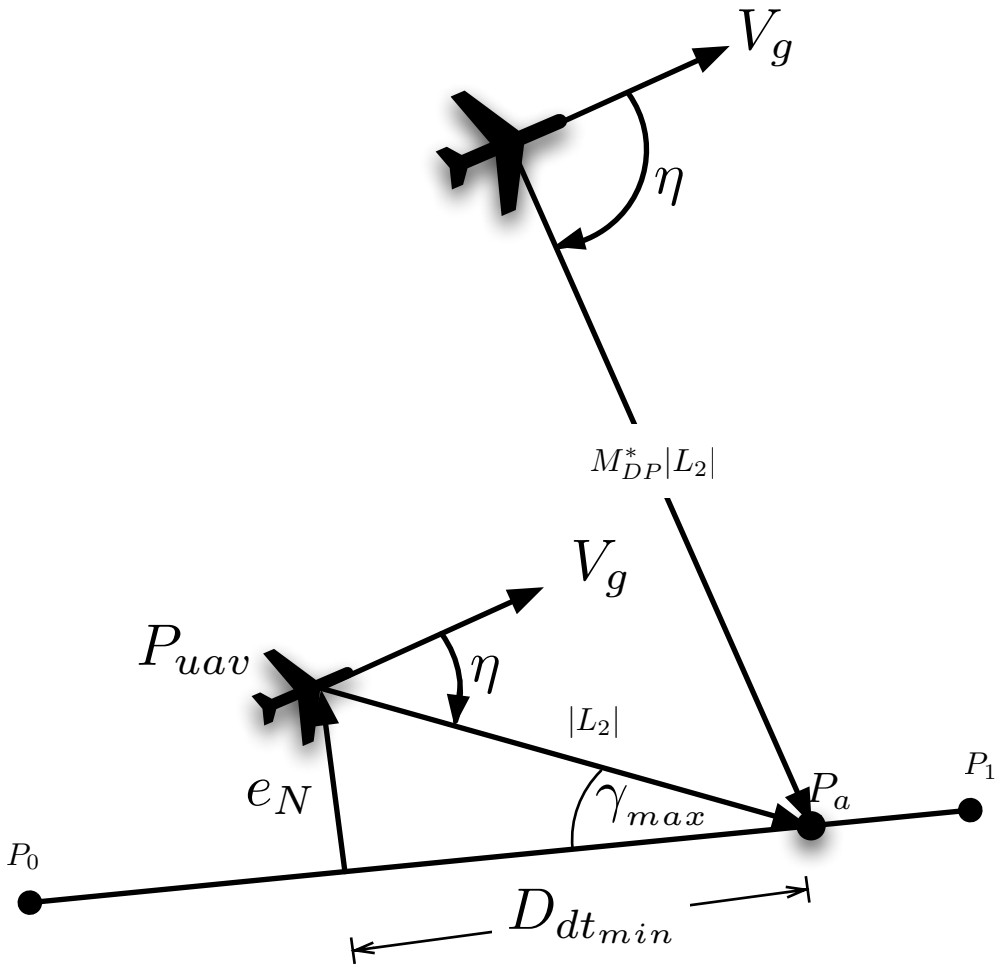


Figure 18: L_2^+ Geometry.

This will be a negative number if the UAV is beyond P_1. The distance from P_1 back along T to the aim point is:

$$D_a = D_{wp_1} - D^*_{dt_{min}}. \tag{48}$$

The final aim point P_a is then computed:

$$P_a = -T_1 \cdot max(0, D_a) - P_{wp_1}. \tag{49}$$

The *max* operation ensures that the aim point does not extend beyond P_1, i.e., if D_a ever becomes negative due to UAV position or failure of the waypoint switching logic.

The acceleration command is then computed with Eq. 43, where the $sin\,\eta$ is obtained from the cross product of V_g and P_a, Eq. 35.

Note that when the aircraft is initially pointed in the opposite direction from T, η is greater than 90°. In this case, a maximum lateral acceleration is imposed to return the vehicle to the correct flight path, where:

$$a_{max} = g \tan \phi_{max} \tag{50}$$

and ϕ_{max} is the maximum bank angle permitted (typically set to 30-60° for a small UAV). In order to use the L_2^+ controller in this condition, a maximum allowable η is required:

$$\eta_{max} = min\left(\frac{\pi}{2}, \mathcal{T}_{LEAD}\frac{a_{max}/2}{U_{comm}}\right) \tag{51}$$

where U_{comm} is the commanded airspeed from the inner loop controller, and $\mathcal{T}_{LEAD}$ is the lead time required for the aircraft to roll out of a steep bank (and determined experimentally for the airframe).

5.4 Waypoint Switching

As previously stated, the guidance strategy is to move from one segment to the next of the flight mission by connecting the two segments with a circular arc. There are myriad ways to switch between segments; this is simply the one implemented on the SLUGS. The L_2^+ implementation uses a policy of early waypoint switching to prioritize path-following instead of waypoint-precision.

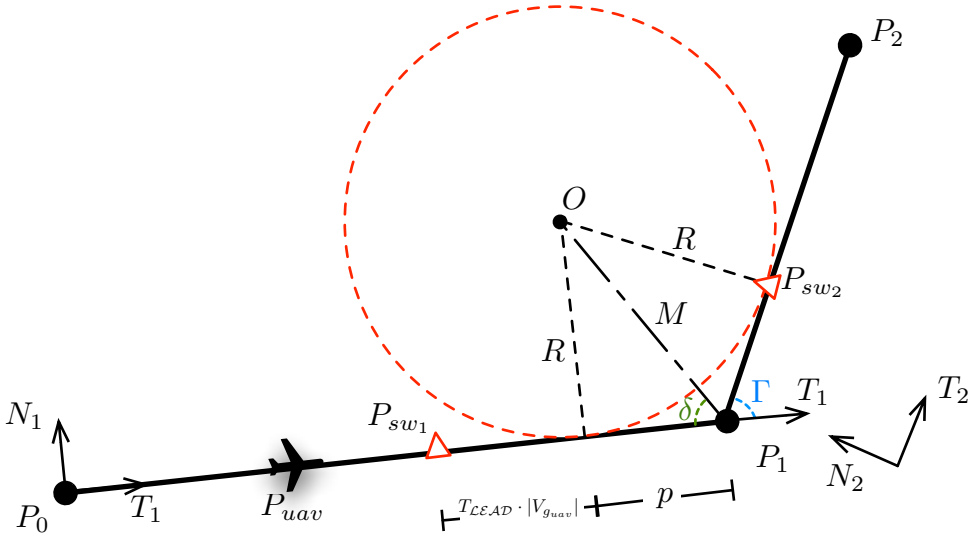


Figure 19: Waypoint Switching Geometry.

Let C be a circle of radius R given by:

$$R = \frac{(U_c + |wind|)^2}{a_{max}}, \tag{52}$$

Here U_c is the commanded airspeed, $|wind|$ is the windspeed, and a_{max} is the maximum acceleration for the UAV. In the presence of wind the actual radius of curvature of the vehicle over the ground changes as the track angle changes. To avoid this problem in defining switching points, use the maximum possible ground speed in the numerator, thus creating a circle whose radius is larger than any curve over the ground.

The circle C is centered at a point O such that it is tangent to each of two waypoint path legs, P_{sw1} and P_{sw2}, as shown in Fig. 19. The first tangent point, P_{sw1}, determines the location where the UAV will switch to start tracking the next waypoint segment.

Referring to the geometry in Fig. 19,

$$\Gamma = \arccos(T_1^\top T_2) \tag{53}$$

$$\delta = \frac{\Gamma - \pi}{2} \tag{54}$$

$$p = M \cos\delta \tag{55}$$

$$M = \frac{R}{\sin\delta} \tag{56}$$

$$p = \frac{R}{\tan\delta} \tag{57}$$

In flight tests, it was discovered that initiating the turn just at the switch point given by Eq. 57 was not adequate because of the lag time of the roll dynamics of the UAV. Therefore the concept of a lead time, $\mathcal{T}_{LEAD}$, was introduced to initiate the turn. When this lead time is multiplied by the groundspeed, it gives the extra distance from the waypoint to the switch point. The new switch point distance is

$$p = \mathcal{T}_{LEAD} \cdot |V_g| + \frac{R}{\tan\delta} \tag{58}$$

Finally, the vector position of the switch point P_{sw1} is given by:

$$P_{sw1} = P_1 - pT_1 \tag{59}$$

Note that during the transition from missions leg, the L_2 vector intercepts the circular arc, not the straight line segments. Also, depending on the exact geometry of the waypoints, and the aircraft speed, it may be that just after switching legs, the aircraft is already beyond the next segment switching point. In this case, the logic immediately switches again to the next leg.

5.5 Point Acquisition and RTB

One advantage of the L_2^+ controller is that it is quite robust. For instance, without any change in the logic, the same controller (with its commanded lateral acceleration/bank angle), can just as easily drive the UAV to a point as well as follow an arbitrary curve.

For the case of an acquisition point, P_a, the angle η is computed from the current UAV position and P_a:

$$\sin\eta = \frac{V_g \times (P_a - P_{uav})}{|V_g||(P_a - P_{uav})|} \tag{60}$$

Again with the limits of η_{max} and downrange distance imposed to limit bank angle and lateral acceleration, implemented in Eq. 33 and Eq. 36. This controller will essentially "point at the point" until it overflies the acquisition point, P_a, at which point the switch logic will cause it to circle the point. Of interest is that nowhere in the formulation does the point P_a have to be fixed; the same L_2^+ controller can track a moving target using its same logic assuming you have a position estimate of the target. Simulations has shown that for target speeds moving at or below the ground speed of the UAV, the UAV always acquires the target.

The *initial mission point*, P_I, before the first waypoint, is determined using concept borrowed from instrument flying in which all aircraft must first fly to a well defined point

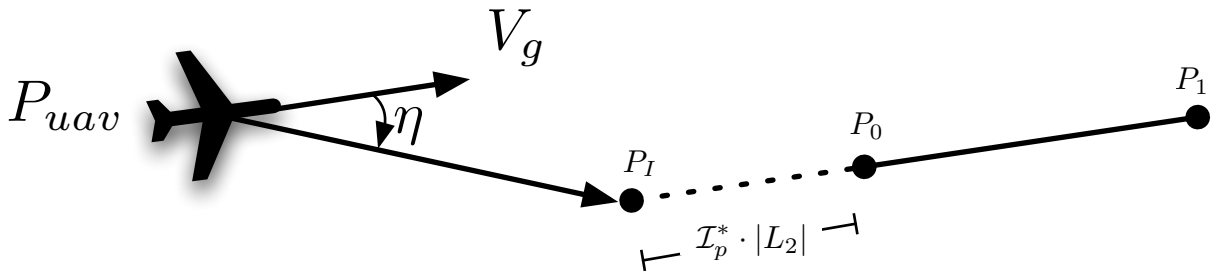


Figure 20: Initial Point Geometry.

before proceeding to land. The initial point is determined by projecting a fixed point in front of the first leg of the mission a constant distance in front of the initial waypoint, Fig. 20:

$$P_I = P_a = P_0 - T_1 \cdot (\mathcal{I}_{\mathcal{P}}^* \cdot |L_2|) \tag{61}$$

where $\mathcal{I}_{\mathcal{P}}^*$ is simply a constant tuned to the specific aircraft.

Lastly, the L_2^+ controller implements a *return to base* (RTB) functionality by simply recording the original base position, P_b, and using this as the acquisition point if either the mission waypoints are completed, or if communication to the UAV fails.

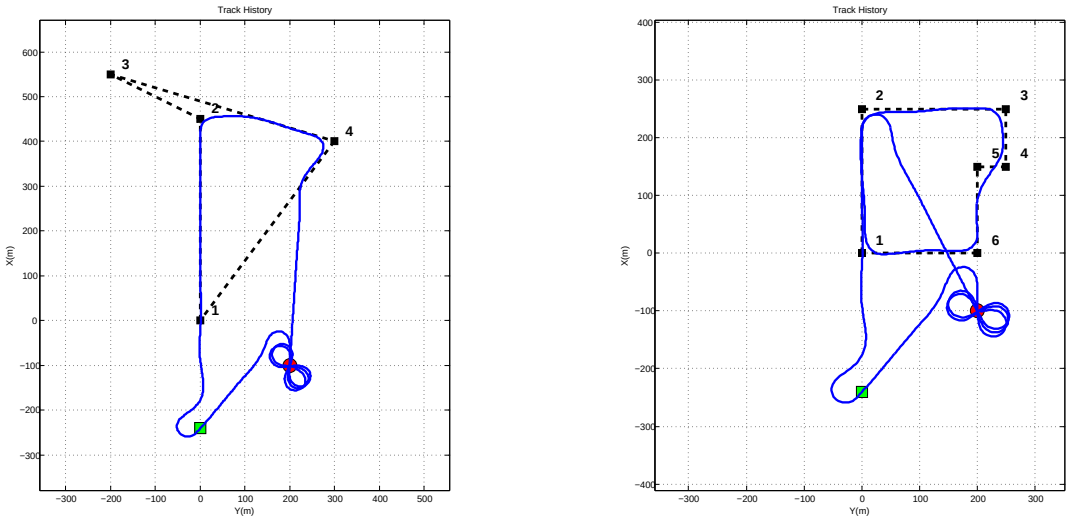
5.6 Simulation and Flight Test Results

The SLUGS autopilot has both a full simulation (based on MATLAB's Simulink), as well as a hardware-in-the-loop (HIL) simulation with the algorithms running on the SLUGS embedded hardware, while the aircraft is simulated through a full 6DOF model running on a separate computer. Fig. 21 demonstrates the full L_2^+ running in simulation, showing the initial point, transitions, and a RTB at the end of the flight.

The SLUGS-based UAV, a small electric RC aircraft (Multex Mentor) was flown at UCSC, using both the inner loop control and outer loop guidance described above. Fig. 22(a) shows the flight test results [Lizarraga, 2009], in the presence of strong wind and real flight disturbances. Panel (b) shows the L_2 vector in real time as the aircraft transitions through the waypoints.

6 Conclusion

In this chapter the complete system architecture for two UAV GNC solutions is described. The systems described a low cost and easily reconfigurable solution which are intended to support research efforts associated with guidance, navigation and control of small UAVs. The purpose of the description and discussion provided was to show how to implement (hardware and software) a low cost GNC solution. It was not intended, however, to be the definitive GNC solution for small UAVs. Such solutions do not exist and this fact was the impetus for writing this chapter. Size, weight, and power constraints associated with small UAVs precludes a "one size fits all" GNC solution. Thus, while the guidance, navigation and control solutions presented in the chapter were suited for the two UAVs described and the mission they were intended for (flight control and guidance research), it does not mean they are ideal for application based on other sensors, vehicles or mission. However, they are flexible enough that they can be adapted to support other missions.



(a) (b)

Figure 21: Return to base implementation in two different waypoint paths. Both have an initial point (green square) and return to the base (red circle). Note the logic skips the waypoint 2 in (a), and waypoint 5 in (b).

References

[SLU,] SLUGS website. `http://slugsuav.soe.ucsc.edu`.

[ASL,] University of California Santa Cruz Autonomous Systems Lab. `http://asl.soe.ucsc.edu`.

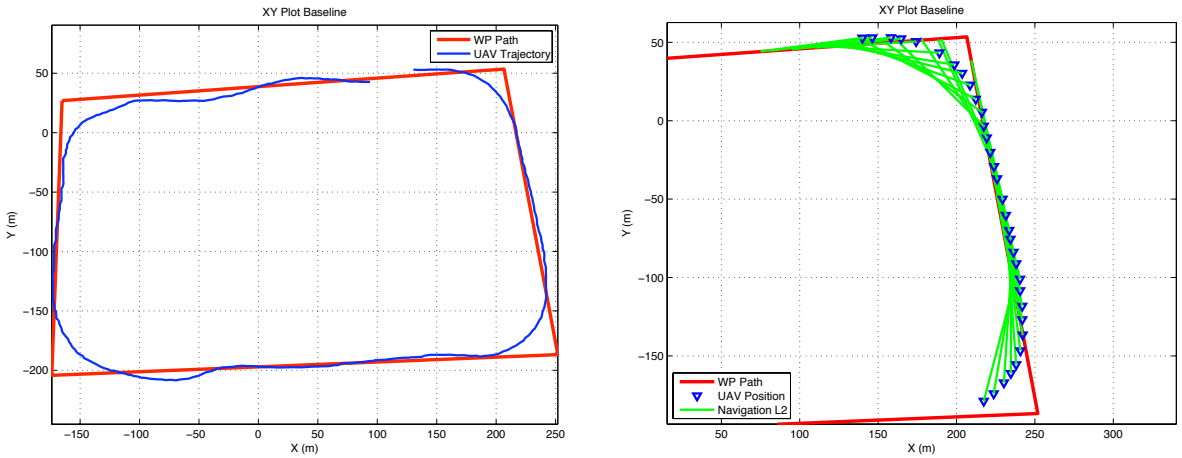
[Amidi and Thorpe, 1991] Amidi, O. and Thorpe, C. (1991). Integrated mobile robot control. *Proc. SPIE*, 1388(504).

[Chu et al., 2011] Chu, C.-C., Lie, F. A. P., Lemay, L., and Gebre-Egziabher, D. (2011). Performance comparison of tight and loose INS-Camera integration. In *Proceedings of the 24th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2011)*, pages 3516–, Portland, OR.

[Dorobantu et al., 2011] Dorobantu, A., Murch, A., Mettler, B., and Balas, G. (2011). Frequency domain system identification for a small, low-cost, fixed-wing UAV. In *AIAA Guidance, Navigation, and Control Conference*, Portland, OR.

[Etkin, 2005] Etkin, B. (2005). *Dynamics of Atmospheric Flight*. Dover.

[Farrell and Barth, 1999] Farrell, J. and Barth, M. (1999). *The Global Positioning System and Inertial Navigation*. McGraw-Hill Professional.



(a) (b)

Figure 22: Flight data for the L_2^+ controller on the Mentor platform, showing path tracking performance in (a) and the evolution of the L_2 vector at the waypoint switches in (b)

[Gebre-Egziabher, 2001] Gebre-Egziabher, D. (December 2001). *Design and Performance Analysis of Low-Cost Aided Dead Reckoning Navigator*. PhD thesis, Department of Aeronautics and Astronautics, Stanford University, Stanford, CA.

[Gleason and Gebre-Egziabher, 2009] Gleason, S. and Gebre-Egziabher, D. (2009). *GNSS Applications and Methods*. Artech House.

[Groves, 2008] Groves, P. (2008). *Principles of GNSS, Inertial, and Integrated Navigation Systems*. Artech House.

[Lefferts et al., 1982] Lefferts, E. J., Markley, F. L., and Shuster, M. D. (1982). Kalman filtering for spacecraft attitude estimation. *Journal of Guidance, Control, and Dynamics*, 5(5):417–429.

[Lie and Gebre-Egziabher, 2012] Lie, F. A. P. and Gebre-Egziabher, D. (2012). A synthetic airdata system. In *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, MN.

[Lizarraga, 2009] Lizarraga, M. (December 2009). *Design, Implementation and Flight Verification of a Versatile and Rapidly Reconfigurable UAV GNC Research Platform*. PhD thesis, Department of Computer Engineering, University of California Santa Cruz, Santa Cruz, CA.

[Lizarraga et al., 2011a] Lizarraga, M., Curry, R., and Elkaim, G. (2011a). Reprogrammable uav autopilot system (part 1) - system hardware and software. *Circuit Cellar*, (249):24–35.

[Lizarraga et al., 2011b] Lizarraga, M., Curry, R., and Elkaim, G. (2011b). Reprogrammable uav autopilot system (part 2) - testing and results. *Circuit Cellar*, (250):36–43.

[Lizarraga et al., 2009a] Lizarraga, M., Dobrokhodov, V., Elkaim, G. H., Curry, R., and Kaminer, I. (2009a). Simulink based hardware-in-the-loop simulator for rapid prototyping of uav control algorithms. In *AIAA Infotech Conference*, Seattle, WA.

[Lizarraga et al., 2009b] Lizarraga, M., Elkaim, G. H., Horn, G., Curry, R., Dobrokhodov, V., and Kaminer, I. (2009b). Low cost rapidly reconfigurable uav autopilot for research and development of guidance, navigation and control algorithms. In *ASME/IEEE MESA09*, San Diego, CA. International Conference on Mechatronic and Embedded Systems and Applications, International Conference on Mechatronic and Embedded Systems and Applications.

[Misra and Enge, 2001] Misra, P. and Enge, P. (2001). *Global Positioning System, Signals, Measurements, and Performance*. Ganga-Jamuna Press.

[Murch, 2012a] Murch, A. (2012a). UMN UAV Flight Code Documentation. `http://http://www.uav.aem.umn.edu/uav/doxygen/html/index.html`.

[Murch, 2012b] Murch, A. (2012b). University of Minnesota UAV Laboratory. `http://www.uav.aem.umn.edu`.

[Owens et al., 2006] Owens, D., Cox, D., and Morelli, E. (2006). Development of a low-cost sub-scale aircraft for flight research: The FASER project. In *25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference*, San Francisco, CA.

[Park et al., 2004] Park, S., Deyst, J., and How, J. (2004). A new nonlinear guidance logic for trajectory tracking. *AIAA Guidance, Navigation and Control Conference and Exhibit*.

[Park et al., 2007] Park, S., Deyst, J., and How, J. P. (2007). Performance and Lyapunov stability of a nonlinear path-following guidance method. *Journal of Guidance, Control and Dynamics*, 30(6):1718.

[Simon, 2006] Simon, D. (2006). *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*. Wiley.

[Stewart, 2001] Stewart, J. (2001). *Calculus: Early Transcendentals*. Brooks/Cole, fourth edition edition.

[Titterton and Weston, 2004] Titterton, D. and Weston, J. (2004). *Strapdown Inertial Navigation Technology*. Institution of Engineering and Technology.

[UAS CoE, 2010] UAS CoE (2010). Eyes of the Army: US Army Roadmap for Unmanned Aircraft Systems 2010-2035. `http://www.rucker.army.mil/usaace/uas/`.