THE STANFORD TESTBED OF AUTONOMOUS ROTORCRAFT FOR MULTI AGENT CONTROL (STARMAC)

Gabe Hoffmann, Dev Gorur Rajnarayan, Steven L. Waslander, PhD. Candidates

David Dostal, Masters Candidate, Jung Soon Jang, Research Associate,

Claire J. Tomlin, Assistant Professor, Stanford University, Stanford, CA.

Abstract

As an alternative to cumbersome aerial vehicles with considerable maintenance requirements and flight envelope restrictions, the X4 flyer is chosen as the basis for the Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC). This paper outlines the design and development of a miniature autonomous waypoint tracker flight control system, and the creation of a multi-vehicle platform for experimentation and validation of multi-agent control algorithms. This testbed development paves the way for real-world implementation of recent work in the fields of autonomous collision and obstacle avoidance, task assignment formation flight, using both centralized and decentralized techniques.

Introduction

Aerial multi-vehicle testbeds are typically expensive and complex platforms, often requiring large open spaces to operate. Fixed-wing aircraft also have constraints on maneuverability whereas conventional helicopters are dynamically and structurally complex, expensive, and hard to control. In order to circumvent these issues in the development of a multi-vehicle testbed at Stanford University, a quadrotor helicopter was proposed as the base vehicle.

The quadrotor helicopter, or X4 flyer, has four fixed-pitch rotors mounted at the four ends of a simple cross frame. Owing to symmetry, this vehicle is dynamically elegant, inexpensive, and simple to design and construct. It is an omni-directional vehicle, and has no almost constraints on its motion. It can be flown in tight spaces and does not require large safety distances to operate. These facts make it an ideal candidate for a versatile and user-friendly multi-agent testbed.

STARMAC (see Figure 1) is an outdoor testbed for testing and validating multi-agent algorithms and control schemes. It comprises a set of autonomous quadrotor helicopters that can follow prescribed waypoint trajectories using GPS and IMU sensing. The testbed could be extended to operate indoors by incorporating vision based position and velocity estimation or other indoor positioning systems.
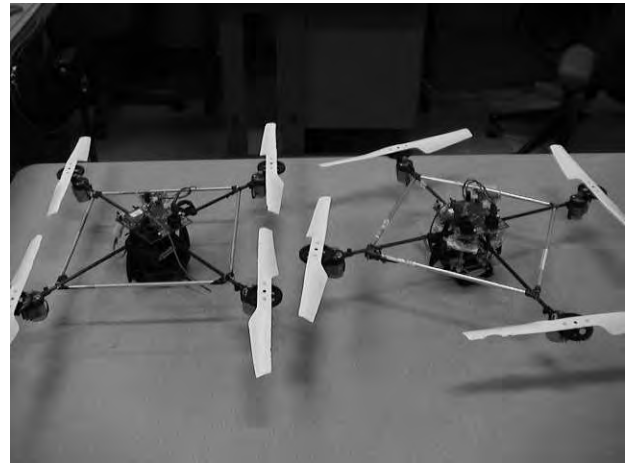
(a) A photograph of two flyers.
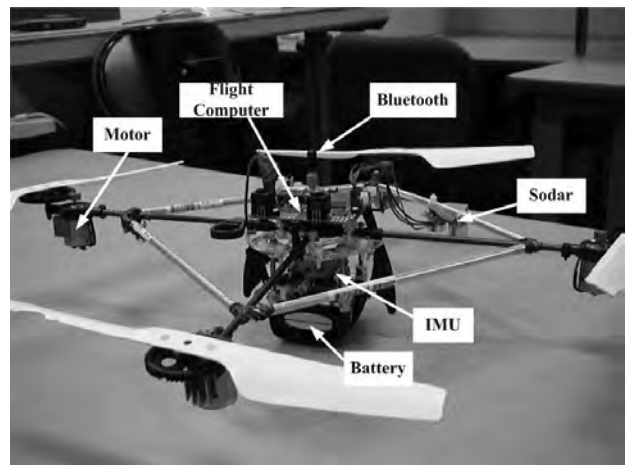
(b) Configuration of the system.

Figure 1: A photograph of the STARMAC.

The goals are to use STARMAC as a testbed for recent algorithms in decentralized optimization [1,2], in which each vehicle in the group will compute its optimal trajectory, which avoids all other vehicles, online using a Nash bargaining formulation which assumes only partial knowledge of the entire system state. In addition, the testbed will be used for validation of multi-player game algorithms. As a future goal, there are plans to equip each flyer with varied sensor suites, including cameras, and use STARMAC as a validation platform for distributed sensing algorithms.

Previous Work

X4 flyers have been developed as early as 1922, with the first known successful hover occurring in October of that year [3]. The recent advent of miniature electronics enabled the creation of the Roswell flyer, the first RC X4 flyer from Area Fifty-One Technologies in 1996, which later became the DraganFlyer from RCToys [4]. Keyence [5] also developed two RC sized X4 flyers known as the Engager and Gyrosaucer. The increased availability has spurred interest in their automation. Numerous groups including [6], [7] have attempted single vehicle X4 projects, and most recently, [8] maintained a successful hover of 25 cm in height using a tethered X4 flyer earlier this year.

Multi agent testbeds have been developed by many researchers at leading institutions, including MIT [9], Berkeley [10], Caltech [11], and Stanford [12] to name just a few. The key distinctions (both advantageous and disadvantageous) that separate our testbed from previous work are:

- The X4flyer as the type of vehicle chosen, resulting in a maintenance free aerial testbed,

- The ease of testing multiple vehicle scenarios using such small vehicles,

- The restrictive payload size (< 1 lb.) and high vibration environment (specific to rotorcraft), resulting in significant design constraints.

Outline

This paper begins with a testbed overview which describes the overall system architecture as well as each of the relevant hardware components. A description of the software architecture on STARMAC follows. The paper then describes the flyer dynamics used as a basis for estimator and controller design, and continues with the resulting estimation and control techniques implemented on the flyers. The paper then presents flight test results for a single flyer achieving autonomous hover control. Finally, the future directions of the project are outlined, including a discussion of current theoretical developments to be implemented on STARMAC in the near term.
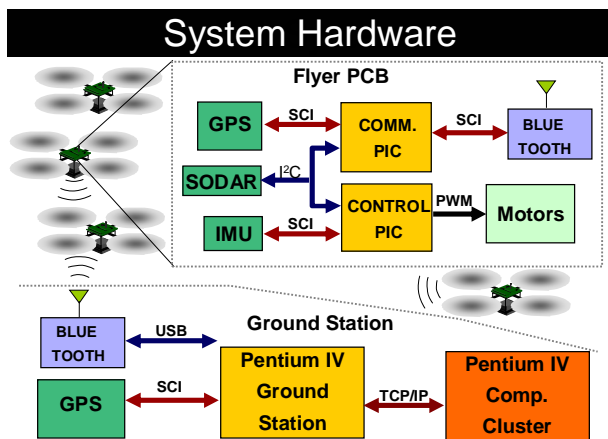
System Overview

STARMAC currently consists of two X4 flyers and a ground station, and is designed with the ability to add up to 2 additional flyers and a ground based computation cluster to the final system. The flyers are controlled by the central ground station through a wireless link, which carries GPS differential correction data and new waypoint commands to each flyer, and flight data from each flyer to the ground station (Figure 2). The trajectory generation may be performed by the ground station computer or by the computation cluster, depending on the complexity of the algorithm implemented.

The main goals for STARMAC include the ability to perform multi-vehicle coordinated tests in order to validate cooperative and non-cooperative multi-agent control algorithms in real time and in a real, variable, outdoor environment. Specifically, collision and obstacle avoidance, trajectory determination, task assignment and coordinated search algorithms are to be tested. Other possibilities include extending the testbed capabilities to perform vision-based navigation, in both indoor and outdoor environments.

X4 flyer

The base vehicle is the Draganflyer III, an off-the-shelf radio-controlled X4 flyer that arrives with onboard gyro-based stability augmentation. The flyer can lift approximately four ounces of payload and fly for about ten minutes at full throttle, due to a recent upgrade to Lithium-polymer batteries which have increased both payload and flight duration and hence have greatly enhanced the abilities of the system.

Figure 2: System Hardware Overview, with expanded view of the Flyer printed circuit board components. SCI refers to serial communication.

The proprietary onboard electronics unit has been completely replaced by a PCB designed and assembled at Stanford, which incorporates all sensing and communication functions needed for autonous flight. Two Microchip PIC18F6520 microcontrollers (referred to as PICs) coordinate the activities of communication, sensing, estimation and control on board the flyer.

Communications Link

Flight data is relayed to the ground and commands are sent to the flyers via a Bluetooth Class II device that has a declared range of over 300 ft, although in practice, data loss occurs as early as 150 ft. This is a different class of device from that found in household appliances, and consequently has greater range. The device operates in the 2.4 GHz frequency range, and incorporates band-hopping, error correction and automatic retransmission. It is designed as a serial cable replacement and operates at a maximum rate of 115.2 kbps.

Sensors

The current onboard sensor suite includes the Trimble Lassen LP – a differential-capable GPS unit, the Devantech SRF08, an ultrasonic altimeter (referred to as Sodar), and the MicroStrain 3DM-G, a complete three-axis IMU (Inertial Measurement Unit) for attitude detection. GPS updates occur at 1

Hz, and exhibit standard differential error ranges of 1-2m in the horizontal plane. The Sodar has a sampling rate of 12 Hz, a range of 2 meters, and an accuracy of 5-10 centimeters, depending largely on the noise environment and the reflecting surface. Finally, the IMU provides gyro-stabilized attitude, attitude rate, and acceleration information at 76 Hz. It has been shown to work extremely well in low vibration environments but loses accuracy as vibration amplitude increases.

Ground Station

The ground station is a standard laptop computer running Microsoft Windows XP and National Instruments Labview 7, which handles all serial communication to the flyers and the differential GPS unit, and will manage the TCP/IP ethernet connection to the computation cluster via built in Labview modules as well. Labview also facilitated the design of a GUI for debugging, monitoring and remote control of the testbed vehicles. The Bluetooth software module creates virtual serial port connections to each flyer which are visible in Labview, and hence communication with all flyers is abstracted to a standard serial interface, which can be implemented using built Labview features. Manual flight is performed via standard joystick input to the ground station laptop.

Computation Cluster

The computation cluster will involve 1-4 PC computers running Matlab 6.5, which can communicate with each other using Matlab Java libraries. The PCs and base station are connected via TCP/IP, and algorithm calculations can be performed in both centralized and decentralized manners.

Software Architecture

The real time control of aerial vehicles requires precise timing and robust code design. To perform all control functions onboard the flyer and implement a controller with as high a bandwidth as possible, the computations required for the estimation and control laws must be pared down to reasonable efficiency. As such, most of the resulting software architecture was selected to ensure both the continuous operation of the fixed

real-time control process and a robust communications channel with the ground station from which user commands originate.

On Board

Onboard the X4 flyer, core functionality has been implemented using two PIC processors. The Control PIC is connected to each of the motors via PWM outputs, to the IMU via a serial port, and to both the SODAR and the Communication PIC via a shared I^{2}C bus. The Communications PIC is connected to both the Bluetooth device and the GPS sensor via separate serial ports as well as to the shared I^{2}C bus.

Control PIC

The Control PIC software is designed to be a deterministic single loop code which polls for necessary information from its attached sensors and from the Communications PICs and performs all calculations once each loop. This architecture was chosen to ensure accurate timing of the control outputs to the motors. With such a design, a timing budget was required in order to ensure all calculations could be performed each cycle despite worst-case communication delays at all interfaces. The loop is timed off the 76Hz continuous IMU data stream, and cannot wait to reattempt any communication if unsuccessful. As a result, none of the communication methods can be blocking on the Control PIC.

Communications PIC

The Communications PIC code services two serial connections by polling for incoming data, and uses an interrupt routine to service the Control PIC communication channel. This avoids blockage of the time-critical control loop. The I^{2}C bus is driven by the interrupt routine and receives flight data or provides the most recent ground station command to the Control PIC immediately upon request. At fixed intervals, the main Communications PIC code reads GPS data and transmits/receives data to/from the ground station. This architecture was chosen to ensure that the Communication PIC does not interfere with the timing of the Control PIC, and to avoid the possibility of interrupt collisions, which might have resulted if more interrupts had been enabled for the various serial communications necessary.

Ground Station

The Labview code running on a Windows platform is inherently multi threaded and non-deterministic. However, Labview timing seems to be quite reliable, and we have found communications with the ground station to be limited by the serial data rate much more than by the ability of the ground station to handle the incoming data. There are four main types of threads in the Labview code, and all scheduling and timing is handled via built in functions and structures. The thread types include: flyer data receive and display thread (one for each flyer), flyer command generate and send thread (one for each flyer), data logging thread (collects all data), and GPS differential unit management thread (only one instance). The GUI displays a tab for each flyer which display full state information received from the flyer, and a tab for monitoring the differential GPS unit.

Aircraft Dynamics

The X4 flyer can be modeled as a primarily linear system for a moderate range of the vehicle flight envelope. Since the ultimate goal of STARMAC was to investigate interaction amongst vehicles, this simplifying assumption greatly reduced the complexity of the control design problem. A complete nonlinear model is presented and was used for simulation.

Forces and Moments

Each rotor produces a thrust T_i, a drag moment Q_i about its axis of rotation, a rolling moment R_i perpendicular to its axis of rotation, and a drag force D_i in the direction $\mathbf{e_{wi}}$, which is opposite to direction of its relative velocity with the oncoming wind. The drag moments Q_i are proportional to the rotor thrust, and are given by $Q_i = K_r T_i$. Let the direction of flight be $\mathbf{e_W}$. Let $\mathbf{x_B}$, $\mathbf{y_B}$ and $\mathbf{z_B}$ denote the body axes. Denote the position vector from the center of gravity (cg) of the vehicle to the rotor disc center by $\mathbf{r_i}$. Let the free stream dynamic pressure be q_∞, the drag coefficient of the aircraft in forward flight C_D, the reference area S, the mass of the

aircraft m, and gravity g. A free body diagram, axes and rotor numbering are defined in Figure 3. The total force $\mathbf{F}$ and moment $\mathbf{M}$ acting on the vehicle in the body frame are as follows.

$$\mathbf{F} = -q_{\infty} S C_D \mathbf{e_W} + m\mathbf{g} + \sum_{i=1}^{4} [-T_i \mathbf{z_B} - D_i \mathbf{e_W}]$$

(1.1)

$$\mathbf{M} = \sum_{i=1}^{4} [Q_i \mathbf{z_B} + R_i \mathbf{e_{W_i}} + D_i (\mathbf{r_i} \times \mathbf{e_{W_i}}) + T_i (\mathbf{r_i} \times \mathbf{z_B})]$$

(1.2)

The reason for the four-rotor design now becomes apparent. This configuration takes advantage of the symmetry in the system and cancels out several moments. The rotors can be grouped into the front-back pair, and the left-right pair. Suppose the pairs rotate in opposite directions. There are then two rotors spinning clockwise and two rotors spinning anti-clockwise. The yawing moments generated by the rotors cancel out to zero. A rolling moment can be generated by speeding up one of the motors in the left-right pair and slowing the other by the same amount. Similarly, a pitching moment can be generated by speeding up one of the motors in the front-back pair and slowing down the other by the same amount. A yawing moment can be generated by speeding up one pair and slowing down the other by the same amount. Note that in each the above cases, the only effect is the generation of a single moment. The sum of rotor thrusts is held constant.

In fact, the symmetry effects extend even further. A single rotor in forward flight also generates a moment perpendicular to its axis of rotation, which we denoted by R_i. However, in this configuration, the rolling moments R_i also cancel out due to the opposite directions of rotation. This cancellation takes place as long as the net angular momentum of all four rotors is zero (this holds at all times except during transients). Even in this case, owing to small inertias and large moment arms, the changes in thrust are small, so the differences in rotor speeds will be small. The resulting rotor rolling moments will be much smaller. For this reason, we will neglect the rotor rolling moments R_i in subsequent discussions.
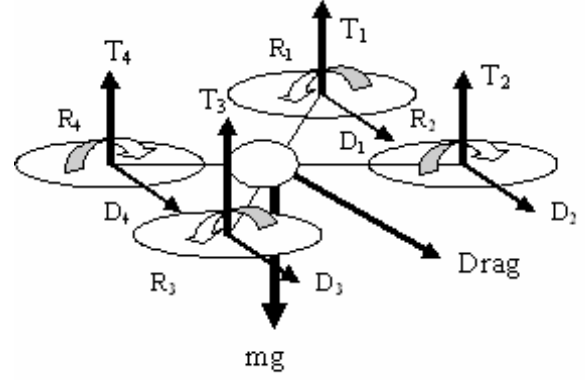
Figure 3: X4 Flyer Free Body Diagram.

If we make the rather reasonable assumption that the STARMAC helicopter is a planar vehicle, then we can further simplify the terms $r_i \cdot z_B$ to a simple multiplication by a moment arm l. Further, if we consider operating points close to hover, then we can assume the drag terms D_i and the total drag D to be small. We can also assume that the angular rates p, q, and r are small. Assume that the aircraft is symmetric about the x, y, and z axes. Then, by the perpendicular axis theorem

$$I = I_{xx} = I_{yy} = I_{zz} / 2$$

Rotational Dynamics

The moment of inertia tensor $\mathbf{I}$ is assumed diagonal, as described previously, the exact governing moment equation is given by

$$\mathbf{I}\dot{\omega}_B + \omega_B \times \mathbf{I}\omega_B = \mathbf{M}$$
$$= \sum_{i=1}^{4} [Q_i \mathbf{z_B} + R_i \mathbf{e_{W_i}} + D_i (\mathbf{r_i} \times \mathbf{e_{W_i}})$$
$$+ T_i (\mathbf{r_i} \times \mathbf{z_B})]$$

(1.3)

If we assume small perturbations about hover, we make the following approximations: The rotor rolling moments R_i are negligible. Since the vehicle is planar and symmetric, the drag terms $D_i (\mathbf{r_i} \times \mathbf{e_{Wi}})$ only apply on the left-right rotor pair, and they cancel out. The rotor drag moments are given by $Q_i = K_r T_i$. Let the perturbation angular rates be p, q and r. We now have

$$I\overset{\square}{p} = -Iqr + l(T_4 - T_2)$$

$$I\overset{\square}{q} = -Ipr + l(T_1 - T_3) \qquad (1.4)$$

$$2I\overset{\square}{r} = K_r(T_1 + T_3 - T_2 - T_4)$$

If we also assume that the angular rates are small and thus neglect their products, the equations become linear, and we can cast them in matrix form as follows.

$$\begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & 2I \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & -l & 0 & -l \\ l & 0 & -l & 0 \\ K_r & -K_r & K_r & -K_r \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}$$
$$(1.5)$$

The body angular rates are easily related to the inertial Euler angle rates, $\dot{\varphi}, \dot{\theta}, \dot{\psi}$, when the Euler angles themselves are small. This is indeed the case if the perturbations from hover are small. Then, the Euler angle rates are approximately equal to the body angular rates. Thus, we get

$$\begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & 2I \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & -l & 0 & -l \\ l & 0 & -l & 0 \\ K_r & -K_r & K_r & -K_r \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}$$
$$(1.6)$$

We observe that the equations are decoupled. This is the form of equations for rotational dynamics used for the inner-loop controller design.

Translational dynamics

Define the total thrust in the body frame as

$$T = T_1 + T_2 + T_3 + T_4$$

If the position vector of the cg in the inertial reference frame is **r**, the translational dynamics can be written as follows

$$m\ddot{\mathbf{r}} = -q_\infty SC_D \mathbf{e_W} + m\mathbf{g} + \sum [-T_i \mathbf{z_B} - D_i \mathbf{e_{W_i}}]$$
$$(1.7)$$

Since this is near hover, we ignore the drag terms and use rotation matrices corresponding to the Euler angles to get

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = R_\psi \cdot R_\theta \cdot R_\phi \cdot \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \qquad (1.8)$$

where R_ψ, R_θ, R_ϕ are the rotation matrices corresponding to the Euler angle rotations ψ, θ, ϕ. If these angles are small, then

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 1 & \psi & -\theta \\ -\psi & 1 & \phi \\ \theta & -\phi & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \quad (1.9)$$

State Estimation

Attitude and attitude rate estimation is not required, as it is already performed in the IMU as received from MicroStrain. We found these sets of signals to be both accurate and reliably noise free at moderate vibration amplitudes, although they do decay at maximum flyer thrust levels. Acceleration data was quite noisy throughout the flyer operating regime, and so attitude information is used instead of direct acceleration measurements in the estimation of translational acceleration.

Altitude estimation using a Kalman filter was necessary in order to determine altitude rate, as it was not sensed directly, and the characteristics of the Sodar sensor proved quite difficult to manage. The problem lay in receiving false echoes due to motor noise, and as such, we added acceleration limited reasonability checks to the data, and selected the first available measurement that satisfied the limits from a list of up to 17 measurements stored by the unit during each measurement. Ultimately, an IR sensor for a limited portion of the flight envelop was added to augment the Sodar signal where possible.

Position estimation uses both GPS position and velocity measurements, as well as attitude information in a Kalman filter to update the position and velocity estimates at 10 Hz. The GPS data is used to correct for integration bias of the small angle approximated accelerations derived from the attitude information.

Controller Synthesis

This section outlines the approach used for control of the X4 flyer. Wherever possible, the simplest linear controllers were used, and only where the situation required were more complex nonlinear effects considered, as in the altitude control loop, for example.

Altitude loop control

The altitude dynamics of the STARMAC can be modeled as a double integrator:

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = u + \Delta(x, g) \tag{1.10}$$

where $x = [h\ \dot{h}]^T$ is the state, u is the control input, g is the gravity constant, and Δ is the matched uncertainty. To design a robust state feedback control to stabilize the system to the origin ($x = 0$) in the presence of the uncertainty Δ, a sliding mode control is applied.

Suppose the first component of (1.10) can be stabilized by a smooth state feedback control $x_2 = \phi(x_1)$, where $\phi(0) = 0$, i.e., the origin of $\dot{x}_1 = \phi(x_1)$ is asymptotically stable. To proceed with the design of the sliding mode control, set the sliding surface s as

$$s = x_2 - \phi(x_1) \tag{1.11}$$

If $s = 0$, then $x_2 = \phi(x_1)$ and the state x_1 approaches the origin asymptotically. Therefore, u can be designed to bring s to zero in finite time and maintain the condition $s = 0$ for all future time. Using $V = \frac{1}{2}s^2$ as a Lyapunov function candidate, we obtain

$$\dot{V} = s\left(\dot{x}_2 - \frac{\partial \phi}{\partial x_1} \dot{x}_1 \right)$$
$$= s\left(u + \Delta(x, g) - \frac{\partial \phi}{\partial x_1} x_2 \right) \tag{1.12}$$

Take the control u as

$$u = \frac{\partial \phi}{\partial x_1} x_2 + v \tag{1.13}$$

where v is an arbitrary part of the control input which will be defined subsequently. Then (1.12) becomes

$$\dot{V} = sv + s\Delta(x, g) \tag{1.14}$$

Assumption 1: Δ *satisfies the following inequality*

$$\| \Delta(x, g) \|_\infty \leq \delta(x) + \gamma |g| \tag{1.15}$$

where the continuous function $\delta(x) \geq 0$ *and* $\gamma \in [0, 1)$ *are known.*

Using the above assumption, (1.14) becomes

$$\dot{V} = sv + s\Delta(x, g)$$
$$\leq sv + |s|\big(\delta(x) + \gamma |g|\big) \tag{1.16}$$

Take the arbitrary control input v as

$$v = -K(x) \cdot sign(s) \tag{1.17}$$

where

$$K(x) \geq \delta(x) + b, \quad \forall \, \| x \| \leq r_0 \tag{1.18}$$

for some $b > \gamma |g|$, and $sign(\cdot)$ is the signum nonlinearity. Then,

$$\dot{V} \leq -K(x)|s| + \delta(x)|s| + \gamma |g| \|s\|$$
$$\leq -\big(b - \gamma |g|\big)|s| \tag{1.19}$$

This inequality ensures that any trajectory starting away from the manifold, $s = 0$, will reach it in finite time and will remain confined to the manifold $s = 0$ [13] once achieved.

The sliding mode controller (1.17) contains the discontinuous nonlinearity $sign(\cdot)$, which is known to cause *chattering* when applied to real systems. One simple approach to eliminate chattering in the altitude control output is to approximate the signum nonlinearity by a saturation nonlinearity [14]. This is achieved by smoothing out the control discontinuity in a thin boundary layer. As a result, the control input v is taken as

$$v = -K(x) \cdot sat\left(\frac{s}{\varepsilon}\right) \qquad (1.20)$$

where

$$sat(y) = \begin{cases} y & \text{if } |y| \leq 1 \\ sign(y) & \text{if } |y| > 1 \end{cases}$$

and ε is a positive constant. The resulting the control input u for the altitude loop control is given by

$$u = \frac{\partial \phi}{\partial x_1} x_2 - K(x) \cdot sat\left(\frac{s}{\varepsilon}\right) \qquad (1.21)$$

Attitude loop control

An inner loop attitude controller was designed using for the simplified second order rotational dynamics model described by equation (1.6), using standard LQR techniques. The cost matrix Q was chosen such that angular deviations were penalized more than rate deviations. Also, the pitch and roll loops were penalized identically, with a reduced penalty on the yaw deviations. This controller was implemented in floating point onboard the X4 flyer. The control inputs for each rotor corresponding to pitch, roll and yaw are computed separately, and then added to the altitude input and the nominal thrust to yield the net thrust value for each rotor. This controller worked well at low thrust levels, but performance at higher thrust levels was degraded owing to the increased vibration levels. This was solved by implementing a softer attitude controller with lower costs on attitude deviations. This improved noise rejection at the cost of tracking performance.

Flight Test Results

Figure 4 shows a STARMAC flyer in the air. Figure 5 (overleaf) shows altitude, roll, and pitch data for a STARMAC flyer in hover, tracking commanded altitude and attitude. The test was run outdoors, and attitude deviations reflect manual attitude command inputs to counteract wind disturbances. The altitude command was held constant. However, net vertical thrust changes with attitude and this effect is not currently accounted for in the control scheme, leading to some altitude deviation during maneuvers.

**Figure 4: STARMAC in flight**

Future Direction

STARMAC stands to become a unique testbed that will enable the implementation of many cutting-edge multi-agent control techniques in an easy-to-use real-world environment. The system has been designed to accommodate 4-6 flyers. As the individual flyer design nears completion, additional flier are being constructed Implementation of the waypoint tracking algorithm and of the computation cluster is to be completed.

Once completed, the aim is to implement decentralized optimization techniques that allow the flyers to act as individual agents in a system, and coordinate, or compete, to achieve system level goals such as collision avoidance and formation flight. Such work will see individual decision makers communicating in the computation cluster to arrive at team decisions.

An exciting field of research that could be investigated using the STARMAC platform is that of vision based navigation and control. There remains sufficient payload available on the current flyer design to incorporate a small wireless camera, and such a development project would further expand the opportunities available for discovery using the STARMAC testbed.
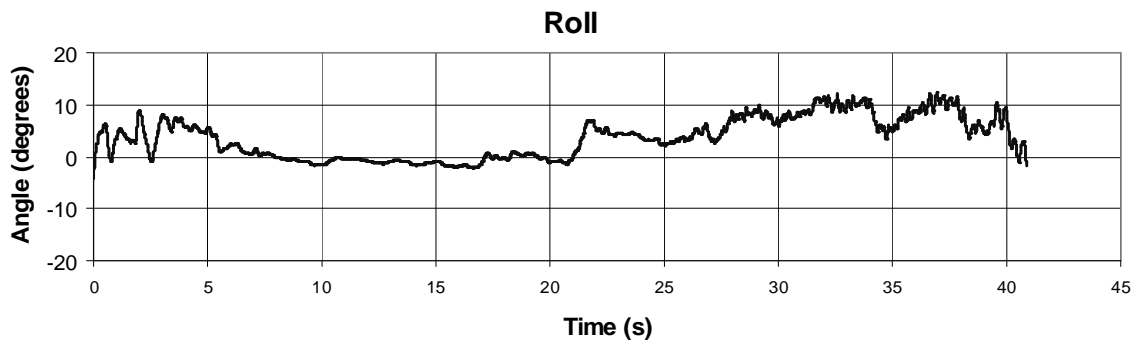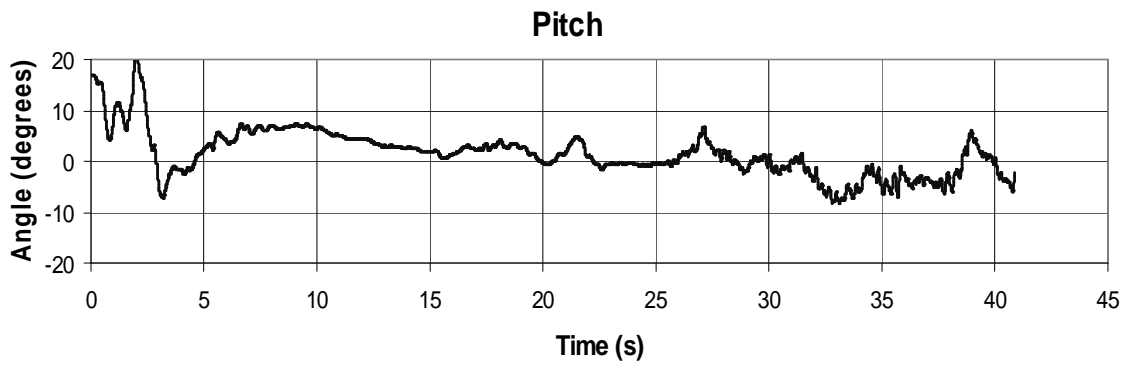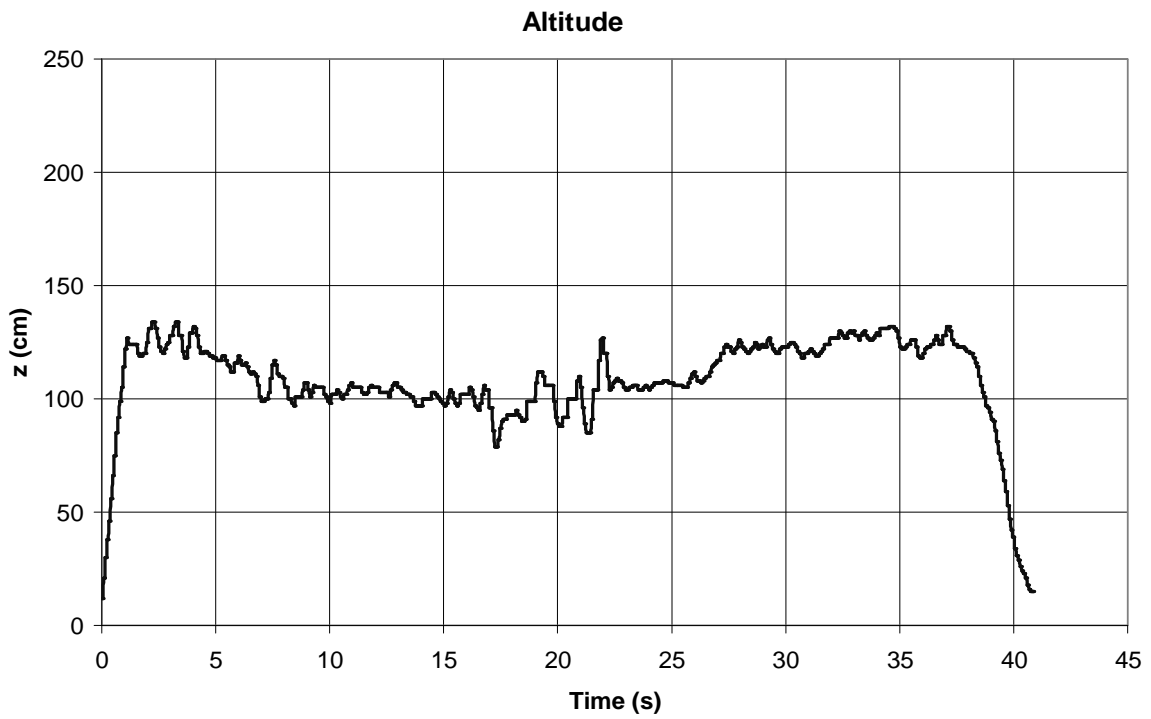
Figure 5: Onboard Controller Test Data

References

[1] G. Inalhan, D. M. Stipanovic, and C. J. Tomlin. Decentralized optimization, with application to multiple aircraft coordination. In *Proceedings of the 41st IEEE Conference on Decision and Control*, Las Vegas, December 2002.

[2] S. L. Waslander, G. Inalhan, and C. J. Tomlin. *Decentralized Optimization via Nash Bargaining*. Kluwer Academic Press, 2004.

[3] P.Lambermont, *Helicopters and Autogyros of the World*, 1958

[4] Website, http://www.rctoys.com

[5] Website, http://www.keyence.co.jp/hobby/english/saucer.html

[6] Kroo, Ilan et al., *The Mesicopter: A Miniature Rotorcraft Concept*, Stanford University, July 2000.

[7] Castillo, Pedro, Dzul, Alejandro, and Lozano, Rogelio, *Real-Time Stabilization and Tracking of a Four-Rotor Mini Rotorcraft,* IEEE Transactions on Control Systems Technology, Vol. 12, No. 4, July 2004.

[8] Pounds, P., Mahony, R., Hynes, P. and Roberts, J., *Design of a Four-Rotor Aerial Robot,* in Proc. of 2002 Australian Conference on Robotics and Automation, Auckland, November 2002.

[9] Richards, A., Kuwata, Y., and How, J. *Experimental demonstrations of real-time MILP control*, In Proceeding of the AIAA Guidance, Navigation, and Control Conference, 2003.

[10] T. J. Koo, J. Liebman, C. Ma, B. Horowitz, A. Sangiovanni-Vincentelli, and S. Sastry *Platform-Based Embedded Software Design for Multi-Vehicle Multi-Modal Embedded Software,* Embedded Software, Alberto Sangiovanni-Vincentelli and Joseph Sifakis (Eds.), Lecture Notes in Computer Science, Springer-Verlag, *2002.*

[11] Lars Cremean], William B. Dunbar, Dave van Gogh, Jason Hickey, Eric Klavins, Jason Meltzer and Richard M. Murray, *The Caltech Multi-Vehicle Wireless Testbed*, in Proceedings of the 41st IEEE Conference on Decision and Control, 2002.

[12] Christopher M. Clark, Eric W. Frew, Henry L. Jones, and Stephen M. Rock. *An Integrated System for Command and Control of Cooperative Robotic Systems*, In Proceedings of the 11th International Conference on Advanced Robotics, Portugal, June 2003.

[13] Khalil, Hassan, 1996, *Nonlinear Systems*, Prentice-Hall.

[14] Slotine, Jean-Jacques and Weiping Li, 1991, *Applied Nonlinear Control*, Prentice-Hall.