Multi-Agent Quadrotor Testbed Control Design: Integral Sliding Mode vs. Reinforcement Learning*

Steven L. Waslander†, Gabriel M. Hoffmann†,‡
Ph.D. Candidate
Aeronautics and Astronautics
Stanford University
{stevenw, gabeh}@stanford.edu

Jung Soon Jang
Research Associate
Aeronautics and Astronautics
Stanford University
jsjang@stanford.edu

Claire J. Tomlin
Associate Professor
Aeronautics and Astronautics
Stanford University
tomlin@stanford.edu

Abstract— **The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) is a multi-vehicle testbed currently comprised of two quadrotors, also called X4-flyers, with capacity for eight. This paper presents a comparison of control design techniques, specifically for outdoor altitude control, in and above ground effect, that accommodate the unique dynamics of the aircraft. Due to the complex airflow induced by the four interacting rotors, classical linear techniques failed to provide sufficient stability. Integral Sliding Mode and Reinforcement Learning control are presented as two design techniques for accommodating the nonlinear disturbances. The methods both result in greatly improved performance over classical control techniques.**

I. INTRODUCTION

As first introduced by the authors in [1], the Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) is an aerial platform intended to validate novel multi-vehicle control techniques and present real-world problems for further investigation. The base vehicle for STARMAC is a four rotor aircraft with fixed pitch blades, referred to as a quadrotor, or an X4-flyer. They are capable of 15 minute outdoor flights in a 100m square area [1].

Fig. 1. One of the STARMAC quadrotors in action.

There have been numerous projects involving quadrotors to date, with the first known hover occurring in October, 1922 [2]. Recent interest in the quadrotor concept has been sparked by commercial remote control versions, such as the

DraganFlyer IV [3]. Many groups [4]–[7] have seen significant success in developing autonomous quadrotor vehicles. To date, however, STARMAC is the only operational multi-vehicle quadrotor platform capable of autonomous outdoor flight, without tethers or motion guides.

The first major milestone for STARMAC was autonomous hover control, with closed loop control of attitude, altitude and position. Using inertial sensing, the attitude of the aircraft is simple to control, by applying small variations in the relative speeds of the blades. In fact, standard integral LQR techniques were applied to provide reliable attitude stability and tracking for the vehicle. Position control was also achieved with an integral LQR, with careful design in order to ensure spectral separation of the successive loops.

Unfortunately, altitude control proves less straightforward. There are many factors that affect the altitude loop specifically that do not amend themselves to classical control techniques. Foremost is the highly nonlinear and destabilizing effect of four rotor downwashes interacting. In our experience, this effect becomes critical when motion is not damped by motion guides or tethers. Empirical observation during manual flight revealed a noticeable loss in thrust upon descent through the highly turbulent flow field. Similar aerodynamic phenomenon for helicopters have been studied extensively [8], but not for the quadrotor, due to its relative obscurity and complexity. Other factors that introduce disturbances into the altitude control loop include blade flex, ground effect and battery discharge dynamics. Although these effects are also present in generating attitude controlling moments, the differential nature of the control input eliminates much of the absolute thrust disturbances that complicate altitude control.

Additional complications arise from the limited choice in low cost, high resolution altitude sensors. An ultrasonic ranging device [9] was used, which suffers from non-Gaussian noise—false echoes and dropouts. The resulting raw data stream includes spikes and echoes that are difficult to mitigate, and most successfully handled by rejection of infeasible measurements prior to Kalman filtering.

In order to accommodate this combination of noise and disturbances, two distinct approaches are adopted. Integral Sliding Mode (ISM) control [10]–[12] takes the approach that the disturbances cannot be modeled, and instead designs

a control law that is guaranteed to be robust to disturbances as long as they do not exceed a certain magnitude. Model-based reinforcement learning [13] creates a dynamic model based on recorded inputs and responses, without *any* knowledge of the underlying dynamics, and then seeks an optimal control law using an optimization technique based on the learned model. This paper presents an exposition of both methods and contrasts the techniques from both a design and implementation point of view.

II. System Description

STARMAC consists of a fleet of quadrotors and a ground station. The system communicates over a Bluetooth Class 1 network. The core of the aircraft are microcontroller circuit boards designed and assembled at Stanford, for this project. The microcontrollers run real-time control code, interface with sensors and the ground station, and supervise the system.

The aircraft are capable of sensing position, attitude, and proximity to the ground. The differential GPS receiver is the Trimble Lassen LP, operating on the L1 band, providing 1 Hz updates. The IMU is the MicroStrain 3DM-G, a low cost, light weight IMU that delivers 76 Hz attitude, attitude rate, and acceleration readings. The distance from the ground is found using ultrasonic ranging at 12 Hz.

The ground station consists of a laptop computer, to interface with the aircraft, and a GPS receiver, to provide differential corrections. It also has a battery charger, and joysticks for control-augmented manual flight, when desired.

III. Quadrotor Dynamics

The derivation of the nonlinear dynamics is performed in North-East-Down (NED) inertial and body fixed coordinates. Let $\{\mathbf{e_N}, \mathbf{e_E}, \mathbf{e_D}\}$ denote the inertial axes, and $\{\mathbf{x_B}, \mathbf{y_B}, \mathbf{z_B}\}$ denote the body axes, as defined in Figure 2. Euler angles of the body axes are $\{\phi, \theta, \psi\}$ with respect to the $\mathbf{e_N}$, $\mathbf{e_E}$ and $\mathbf{e_D}$ axes, respectively, and are referred to as roll, pitch and yaw. Let $\mathbf{r}$ be defined as the position vector from the inertial origin to the vehicle center of gravity (CG), and let $\omega_\mathbf{B}$ be defined as the angular velocity in the body frame. The current velocity direction is referred to as $\mathbf{e_v}$ in inertial coordinates.
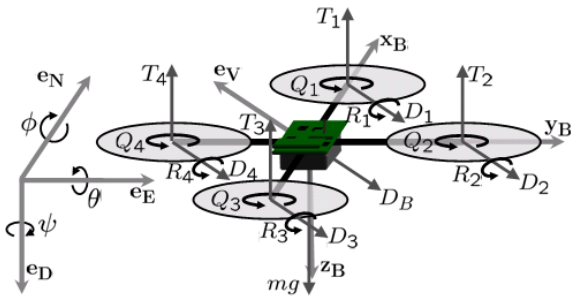

Fig. 2. Free body diagram of a quadrotor aircraft.

The rotors, numbered $1-4$, are mounted outboard on the $\mathbf{x_B}$, $\mathbf{y_B}$, $-\mathbf{x_B}$ and $-\mathbf{y_B}$ axes, respectively, with position vectors $\mathbf{r_i}$ with respect to the CG. Each rotor produces an aerodynamic torque, Q_i, and thrust, T_i, both parallel to the rotor's axis of rotation, and both used for vehicle control. Here, $T_i \approx u_i \frac{k_t}{1+0.1s}$, where u_i is the voltage applied to the motors, as determined from a load cell test. In flight, T_i can vary greatly from this approximation. The torques, Q_i, are proportional to the rotor thrust, and are given by $Q_i = k_r T_i$. Rotors 1 and 3 rotate in the opposite direction as rotors 2 and 4, so that counteracting aerodynamic torques can be used independently for yaw control. Horizontal velocity results in a moment on the rotors, R_i, about $-\mathbf{e_v}$, and a drag force, D_i, in the direction, $-\mathbf{e_v}$.

The body drag force is defined as D_B, vehicle mass is m, acceleration due to gravity is g, and the inertia matrix is $I \in \mathbb{R}^{3\times3}$. A free body diagram is depicted in Figure 2. The total force, $\mathbf{F}$, and moment, $\mathbf{M}$, can be summed as,

$$\mathbf{F} = -D_B\mathbf{e_v} + mg\mathbf{e_D} + \sum_{i=1}^{4}(-T_i\mathbf{z_B} - D_i\mathbf{e_v}) \quad (1)$$

$$\mathbf{M} = \sum_{i=1}^{4}(Q_i\mathbf{z_B} - R_i\mathbf{e_v} - D_i(\mathbf{r_i} \times \mathbf{e_v}) + T_i(\mathbf{r_i} \times \mathbf{z_B})) \quad (2)$$

The full nonlinear dynamics can be described as,

$$\begin{aligned} m\ddot{\mathbf{r}} &= \mathbf{F} \\ I\dot{\omega}_\mathbf{B} + \omega_\mathbf{B} \times I\omega_\mathbf{B} &= \mathbf{M} \end{aligned} \quad (3)$$

where the total angular momentum of the rotors is assumed to be near zero, because they are counter-rotating. Near hover conditions, the contributions by rolling moment and drag can be neglected in Equations (1) and (2). Define the total thrust as $T = \sum_{i=1}^{4} T_i$. The translational motion is defined by,

$$m\ddot{\mathbf{r}} = \mathbf{F} = -R_\psi \cdot R_\theta \cdot R_\phi T\mathbf{z_B} + mg\mathbf{e_D} \quad (4)$$

where R_ϕ, R_θ, and R_ψ are the rotation matrices for roll, pitch, and yaw, respectively. Applying the small angle approximation to the rotation matrices,

$$m\begin{bmatrix} \ddot{r}_x \\ \ddot{r}_y \\ \ddot{r}_z \end{bmatrix} = \begin{bmatrix} 1 & \psi & \theta \\ \psi & 1 & \phi \\ \theta & -\phi & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \quad (5)$$

Finally, assuming total thrust approximately counteracts gravity, $T \approx \bar{T} = mg$, except in the $\mathbf{e_D}$ axis,

$$m\begin{bmatrix} \ddot{r}_x \\ \ddot{r}_y \\ \ddot{r}_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{bmatrix} 0 & -\bar{T} & 0 \\ \bar{T} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \phi \\ \theta \\ T \end{bmatrix} \quad (6)$$

For small angular velocities, the Euler angle accelerations are determined from Equation (3) by dropping the second order term, $\omega \times I\omega$, and expanding the thrust into its four constituents. The angular equations become,

$$\begin{bmatrix} I_x\ddot{\phi} \\ I_y\ddot{\theta} \\ I_z\ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & l & 0 & -l \\ l & 0 & -l & 0 \\ K_r & -K_r & K_r & -K_r \end{bmatrix}\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \quad (7)$$

where the moment arm length $l = ||\mathbf{r_i} \times \mathbf{z_B}||$ is identical for all rotors due to symmetry. The resulting linear models can now be used for control design.

IV. ESTIMATION AND CONTROL DESIGN

Applying the concept of spectral separation, inner loop control of attitude and altitude is performed by commanding motor voltages, and outer loop position control is performed by commanding attitude requests for the inner loop. Accurate attitude control of the plant in Equation (7) is achieved with an integral LQR controller design to account for thrust biases. Position estimation is performed using a navigation filter that combines horizontal position and velocity information from GPS, vertical position and estimated velocity information from the ultrasonic ranger, and acceleration and angular rates from the IMU in a Kalman filter that includes bias estimates. Integral LQR techniques are applied to the horizontal components of the linear position plant described in Equation (6). The resulting hover performance is shown in Figure 6.

As described above, altitude control suffers exceedingly from unmodeled dynamics. In fact, manual command of the throttle for altitude control remains a challenge for the authors to this day. Additional complications arise from the ultrasonic ranging sensor, which has frequent erroneous readings, as seen in Figure 3. To alleviate the effect of this noise, rejection of infeasible measurements is used to remove much of the non-Gaussian noise component. This is followed by altitude and altitude rate estimation by Kalman filtering, which adds lag to the estimate. This section proceeds with a derivation of two control techniques that can be used to overcome the unmodeled dynamics and the remaining noise.
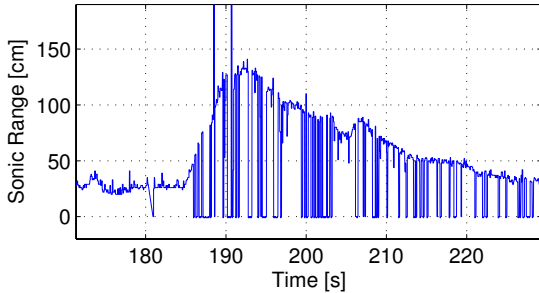

Fig. 3. Characteristic unprocessed ultrasonic ranging data, displaying spikes, false echoes and dropouts. Powered flight commences at 185 seconds.

A. Integral Sliding Mode Control

A linear approximation to the altitude error dynamics of a quadrotor aircraft in hover is given by,

$$
\begin{aligned}
\dot{x_1} &= x_2 \\
\dot{x_2} &= u + \xi(g,x)
\end{aligned}
\tag{8}
$$

where $\{x_1, x_2\} = \{(r_{z,des} - r_z), (\dot{r}_{z,des} - \dot{r_z})\}$ are the altitude error states, $u = \sum_{i=1}^{4} u_i$ is the control input, and $\xi(\cdot)$ is a bounded model of disturbances and dynamic uncertainty. It

is assumed that $\xi(\cdot)$ satisfies $\|\xi\| \leq \gamma$, where γ is the upper bounded norm of $\xi(\cdot)$.

In early attempts to stabilize this system, it was observed that LQR control was not able to address the instability and performance degradation due to $\xi(g,x)$. Sliding Mode Control (SMC) was adapted to provide a systematic approach to the problem of maintaining stability and consistent performance in the face of modeling imprecision and disturbances. However, until the system dynamics reach the sliding manifold, such nice properties of SMC are not assured. In order to provide robust control throughout the flight envelope, the Integral Sliding Mode (ISM) technique is applied.

The ISM control is designed in two parts. First, a standard successive loop closure is applied to the linear plant. Second, integral sliding mode techniques are applied to guarantee disturbance rejection. Let

$$
\begin{aligned}
u &= u_p + u_d \\
u_p &= -K_p x_1 - K_d x_2
\end{aligned}
\tag{9}
$$

where K_p and K_d are proportional and derivative loop gains that stabilize the linear dynamics without disturbances. For disturbance rejection, a sliding surface, s, is designed,

$$
\begin{aligned}
s &= s_0(x_1, x_2) + z \\
s_0 &= \alpha(x_2 + k x_1)
\end{aligned}
\tag{10}
$$

such that state trajectories are forced towards the manifold $s = 0$. Here, s_0 is a conventional sliding mode design, z is an additional term that enables integral control to be included, and $\alpha, k \in \mathbb{R}$ are positive constants. Based on the following Lyapunov function candidate, $V = \frac{1}{2}s^2$, the control component, u_d, can be determined such that $\dot{V} < 0$, guaranteeing convergence to the sliding manifold.

$$
\begin{aligned}
\dot{V} &= s\dot{s} = s\big[\alpha(\dot{x}_2 + k\dot{x}_1) + \dot{z}\big] \\
&= s\big[\alpha(u_p + u_d + \xi(g,x) + kx_2) + \dot{z}\big] < 0
\end{aligned}
\tag{11}
$$

The above condition holds if $\dot{z} = -\alpha(u_p + kx_2)$ and u_d can be guaranteed to satisfy,

$$
s\big[u_d + \xi(g,x)\big] < 0, \quad \alpha > 0
\tag{12}
$$

Since the disturbances, $\xi(g,x)$, are bounded by γ, define u_d to be $u_d = -\lambda s$ with $\lambda \in \mathbb{R}$. Equation (11) becomes,

$$
\begin{aligned}
\dot{V} &= s\big[\alpha(-\lambda s + \xi(g,x)\big] \\
&\leq \alpha\big[-\lambda|s|^2 + \gamma|s|\big] < 0
\end{aligned}
\tag{13}
$$

and it can be seen that $\lambda|s| - \gamma > 0$. As a result, for u_p and u_d as above, the sliding mode condition holds when,

$$
|s| > \frac{\gamma}{\lambda}
\tag{14}
$$

With the input derived above, the dynamics are guaranteed to evolve such that s decays to within the boundary layer,

$\frac{\gamma}{\lambda}$, of the sliding manifold. Additionally, the system does not suffer from input chatter as conventional sliding mode controllers do, as the control law does not include a switching function along the sliding mode.

V. Reinforcement Learning Control

An alternate approach is to implement a reinforcement learning controller. Much work has been done on continuous state-action space reinforcement learning methods [13], [14]. For this work, a nonlinear, nonparametric model of the system is first constructed using flight data, approximating the system as a stochastic Markov process [15], [16]. Then a model-based reinforcement learning algorithm uses the model in policy-iteration to search for an optimal control policy that can be implemented on the embedded microprocessors.

In order to model the aircraft dynamics as a stochastic Markov process, a Locally Weighted Linear Regression (LWLR) approach is used to map the current state, $\mathbf{S}(t) \in \mathbb{R}^{n_s}$, and input, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$, onto the subsequent state estimate, $\hat{\mathbf{S}}(t+1)$. In this application, $\mathbf{S} = \begin{bmatrix} r_z & \dot{r}_z & \ddot{r}_z & V \end{bmatrix}$, where V is the battery level. In the altitude loop, the input, $\mathbf{u} \in \mathbb{R}$, is the total motor power, u. The subsequent state mapping is the summation of the traditional LWLR estimate, using the current state and input, with the random vector, $\mathbf{v} \in \mathbb{R}^{n_s}$, representing unmodeled noise. The value for $\mathbf{v}$ is drawn from the distribution of output error as determined by using a maximum likelihood estimate [16] of the Gaussian noise in the LWLR estimate. Although the true distribution is not perfectly Gaussian, this model is found to be adequate.

The LWLR method [17] is well suited to this problem, as it fits a non-parametric curve to the local structure of the data. The scheme extends least squares by assigning weights to each training data point according to its proximity to the input value, for which the output is to be computed. The technique requires a sizable set of training data in order to reflect the full dynamics of the system, which is captured from flights flown under both automatic and manually controlled thrust, with the attitude states under automatic control.

For m training data points, the input training samples are stored in $X \in \mathbb{R}^{(m) \times (n_s + n_u + 1)}$, and the outputs corresponding to those inputs are stored in $Y \in \mathbb{R}^{m \times n_s}$. These matrices are defined as

$$X = \begin{bmatrix} 1 & \mathbf{S}(t_1)^\mathsf{T} & \mathbf{u}(t_1)^\mathsf{T} \\ \vdots & \vdots & \vdots \\ 1 & \mathbf{S}(t_m)^\mathsf{T} & \mathbf{u}(t_m)^\mathsf{T} \end{bmatrix}, \quad Y = \begin{bmatrix} \mathbf{S}(t_1+1)^\mathsf{T} \\ \vdots \\ \mathbf{S}(t_m+1)^\mathsf{T} \end{bmatrix}$$
(15)

The column of ones in X enables the inclusion of a constant offset in the solution, as used in linear regression.

The diagonal weighting matrix $W \in \mathbb{R}^{m \times m}$, which acts on X, has one diagonal entry for each training data point. That entry gives more weight to training data points that are close to the $\mathbf{S}(t)$ and $\mathbf{u}(t)$ for which $\hat{\mathbf{S}}(t+1)$ is to be computed.

The distance measure used in this work is

$$W_{i,i} = \exp\left(\frac{-||\mathbf{x}^{(i)} - \mathbf{x}||}{2\tau^2}\right)$$
(16)

where $\mathbf{x}^{(i)}$ is the ith row of X, $\mathbf{x}$ is the vector $\begin{bmatrix} 1 & \mathbf{S}(t)^\mathsf{T} & \mathbf{u}(t)^\mathsf{T} \end{bmatrix}$, and fit parameter τ is used to adjust the range of influence of training points. The value for τ can be tuned by cross validation to prevent over- or under-fitting the data. Note that it may be necessary to scale the columns before taking the Euclidean norm to prevent undue influence of one state on the W matrix.

The subsequent state estimate is computed by summing the LWLR estimate with $\mathbf{v}$,

$$\hat{\mathbf{S}}(t+1) = \left(X^\mathsf{T} W X\right)^{-1} X^\mathsf{T} W^\mathsf{T} \mathbf{x} + \mathbf{v}$$
(17)

Because W is a continuous function of $\mathbf{x}$ and X, as $\mathbf{x}$ is varied, the resulting estimate is a continuous non-parametric curve capturing the local structure of the data. The matrix computations, in code, exploit the large diagonal matrix W; as each $W_{i,i}$ is computed, it is multiplied by row $\mathbf{x}^{(i)}$, and stored in WX.

The matrix being inverted is poorly conditioned, because weakly related data points have little influence, so their contribution cannot be accurately numerically inverted. To more accurately compute the numerical inversion, one can perform a singular value decomposition, $(X^\mathsf{T} W X) = U \Sigma V^\mathsf{T}$. Then, numerical error during inversion can be avoided by using the n singular values σ_i with values of $\frac{\sigma_{max}}{\sigma_i} < C_{max}$, where the value of C_{max} is chosen by cross validation. In this work, $C_{max} \approx 10$ was found to minimize numerical error, and was typically satisfied by $n = 1$. The inverse can be directly computed using the n upper singular values in the diagonal matrix $\Sigma_n \in \mathbb{R}^{n \times n}$, and the corresponding singular vectors, in $U_n \in \mathbb{R}^{m \times n}$ and $V_n \in \mathbb{R}^{m \times n}$. Thus, the stochastic Markov model becomes

$$\hat{\mathbf{S}}(t+1) = V_n \Sigma_n^{-1} U_n^\mathsf{T} X^\mathsf{T} W^\mathsf{T} \mathbf{x} + \mathbf{v}$$
(18)

Next, model-based reinforcement learning is implemented, incorporating the stochastic Markov model, to design a controller. A quadratic reward function is used,

$$R(\mathbf{S}, \mathbf{S_{ref}}) = -c_1(r_z - r_{z,ref})^2 - c_2 \dot{r}_z^2$$
(19)

where $R : \mathbb{R}^{2n_s} \rightarrow \mathbb{R}$, $c_1 > 0$ and $c_2 > 0$ are constants giving reward for accurate tracking and good damping respectively, and $\mathbf{S_{ref}} = \begin{bmatrix} r_{z,ref} & \dot{r}_{z,ref} & \ddot{r}_{z,ref} & V_{ref} \end{bmatrix}$ is the reference state desired for the system.

The control policy maps the observed state $\mathbf{S}$ onto the input command $\mathbf{u}$. In this work, the state space has the constraint of $r_z \geq 0$, and the input command has the constraint of $0 \leq u \leq u_{max}$. The control policy is chosen to be

$$\pi(\mathbf{S}, \mathbf{w}) = w_1 + w_2(r_z - r_{z,ref}) + w_3 \dot{r}_z + w_4 \ddot{r}_z$$
(20)

where $\mathbf{w} \in \mathbb{R}^{n_c}$ is the vector of policy coefficients $w_1, \ldots, w_{n_c}$. Linear functions were sufficient to achieve good stability and performance. Additional terms, such as battery level and integral of altitude error, could be included to make the policy more resilient to differing flight conditions.

Policy iteration is performed as explained in Algorithm 1. The algorithm aims to find the value of $\mathbf{w}$ that yields the greatest total reward R_{total}, as determined by simulating the system over a finite horizon from a set of random initial conditions, and summing the values of $R(\mathbf{S}, \mathbf{S}_{ref})$ at each state encountered.

Algorithm 1 Model-Based Reinforcement Learning

1: Generate set $\mathbf{S_0}$ of random initial states
2: Generate set T of random reference trajectories
3: Initialize $\mathbf{w}$ to reasonable values
4: $R_{best} \leftarrow -\infty$, $\mathbf{w}_{best} \leftarrow \mathbf{w}$
5: **repeat**
6:     $R_{total} \leftarrow 0$
7:     **for** $\mathbf{s_0} \in \mathbf{S_0}, t \in T$ **do**
8:         $\mathbf{S}(0) \leftarrow \mathbf{s_0}$
9:         **for** $t = 0$ to $t_{max} - 1$ **do**
10:             $\mathbf{u}(t) \leftarrow \pi(\mathbf{S}(t), \mathbf{w})$
11:             $\mathbf{S}(t+1) \leftarrow LWLR(\mathbf{S}(t), \mathbf{u}(t)) + \mathbf{v}$
12:             $R_{total} \leftarrow R_{total} + R(\mathbf{S}(t+1))$
13:         **end for**
14:     **end for**
15:     **if** $R_{total} > R_{best}$ **then**
16:         $R_{best} \leftarrow R_{total}$, $\mathbf{w}_{best} \leftarrow \mathbf{w}$
17:     **end if**
18:     Add Gaussian random vector to $\mathbf{w}_{best}$, store as $\mathbf{w}$
19: **until** $\mathbf{w}_{best}$ converges

In policy iteration, a fixed set of random initial conditions and reference trajectories are used to simulate flights at each iteration, with a given policy parameterized by $\mathbf{w}$. It is necessary to use the same random set at each iteration in order for convergence to be possible [15]. After each iteration, the new value of $\mathbf{w}$ is stored as $\mathbf{w}_{best}$ if it outperforms the previous best policy, as determined by comparing R_{total} to R_{best}, the previous best reward encountered. Then, a Gaussian random vector is added to $\mathbf{w}_{best}$. The result is stored as $\mathbf{w}$, and the simulation is performed again. This is iterated until the value of $\mathbf{w}_{best}$ remains fixed for an appropriate number of iterations, as determined by the particular application. The simulation results must be examined to predict the likely performance of the resulting control policy.

By using a Gaussian update rule for the policy weights, $\mathbf{w}$, it is possible to escape local maxima of R_{total}. The highest probability steps are small, and result in refinement of a solution near a local maximum of R_{total}. However, if the algorithm is not at the global maximum, and is allowed to continue, there exists a finite probability that a sufficiently

large Gaussian step will be performed such that the algorithm can keep ascending.

VI. FLIGHT TEST RESULTS

A. Integral Sliding Mode

The results of an outdoor flight test with ISM control can be seen in Figure 4. The response time is on the order of 1-2 seconds, with 5 seconds settling time, and little to no steady state offset. Also, an oscillatory character can be seen in the response, which is most likely being triggered by the nonlinear aerodynamic effects and sensor data spikes described earlier.
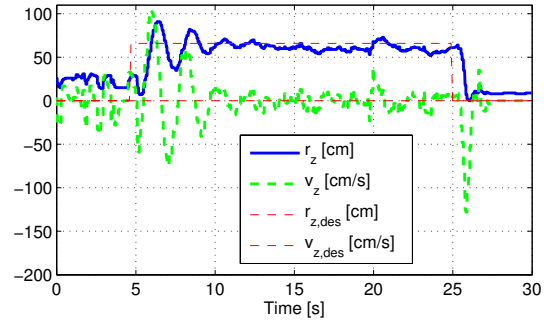

Fig. 4. Integral sliding mode step response in outdoor flight test.

Compared to linear control design techniques implemented on the aircraft, the ISM control proves a significant enhancement. By explicitly incorporating bounds on the unknown disturbance forces in the derivation of the control law, it is possible to maintain stable altitude on a system that has evaded standard approaches.

B. Reinforcement Learning Control

One of the most exciting aspects of RL control design is its ease of implementation. The policy iteration algorithm arrived at the implemented control law after only 3 hours on a Pentium IV computer. Figure 5 presents flight test results for the controller. The high fidelity model of the system, used for RL control design, provides a useful tool for comparison of the RL control law with other controllers. In fact, in simulation with linear controllers that proved unstable on the quadrotor, flight paths with growing oscillations were predicted that closely matched real flight data.

The locally weighted linear regression model showed many relations that were not reflected in the linear model, but that reflect the physics of the system well. For instance, with all other states held fixed, an upward velocity results in more acceleration at the subsequent time step for a throttle level, and a downward velocity yields the opposite effect. This is essentially negative damping. The model also shows a strong ground effect. That is, with all other states held fixed, the closer the vehicle is to the ground, the more acceleration it will have at the subsequent time step for a given throttle level.
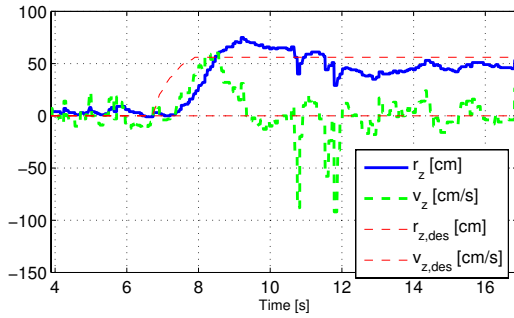
Fig. 5. Reinforcement learning controller response to manually applied step input, in outdoor flight test. Spikes in state estimates are from sensor noise passing through the Kalman filter.

The reinforcement learning control law is susceptible to system disturbances for which it is not trained. In particular, varying battery levels and blade degradation may cause a reduction in stability or steady state offset. Addition of an integral error term to the control policy may prove an effective means of mitigating steady state disturbances, as was seen in the ISM control law.

Comparison of the step response for ISM and RL control reveals both stable performance and similar response times, although the transient dynamics of the ISM control are more pronounced. RL does, however, have the advantage that it incorporates accelerometer measurement into its control, and as such uses a more direct measurement of the disturbances imposed on the aircraft.

C. Autonomous Hover

Applying ISM altitude control and integral LQR position control techniques, flight tests were performed to achieve the goal of autonomous hover. Position response was maintained within a 3m circle for the duration of a two minute flight (see Figure 6), which is well within the expected error bound for the L1 band differential GPS used.
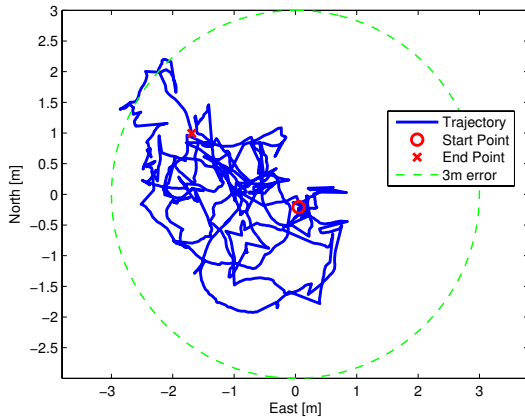

Fig. 6. Autonomous hover flight recorded position, with 3m error circle.

VII. CONCLUSION

This paper summarizes the development of an autonomous quadrotor capable of extended outdoor trajectory tracking control. This is the first demonstration of such capabilities on a quadrotor known to the authors, and represents a critical step in developing a novel, easy to use, multi-vehicle testbed for validation of multi-agent control strategies for autonomous aerial robots. Specifically, two design approaches were presented for the altitude control loop, which proved a challenging hurdle. Both techniques resulted in stable controllers with similar response times, and were a significant improvement over linear controllers that failed to stabilize the system adequately.

REFERENCES

[1] Hoffmann, G., Rajnarayan, D. G., Waslander, S. L., Dostal, D., Jang, J. S., and Tomlin, C. J., "The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC)," *23rd Digital Avionics System Conference*, Salt Lake City, UT, November 2004.
[2] Lambermont, P., *Helicopters and Autogyros of the World*, 1958.
[3] DraganFly-Innovations, "www.rctoys.com," 2003.
[4] Pounds, P., Mahony, R., Hynes, P., and Roberts, J., "Design of a Four-Rotor Aerial Robot," *Australian Conference on Robotics and Automation*, Auckland, November 2002.
[5] Altug, E., Ostrowski, J. P., and Taylor, C. J., "Quadrotor Control Using Dual Camera Visual Feedback," *ICRA*, Taipei, September 2003.
[6] Bouabdallah, S., Murrieri, P., and Siegwart, R., "Design and Control of an Indoor Micro Quadrotor," *ICRA*, New Orleans, April 2004.
[7] Castillo, P., Dzul, A., and Lozano, R., "Real-Time Stabilization and Tracking of a Four-Rotor Mini Rotorcraft," *IEEE Transactions on Control Systems Technology*, Vol. 12, No. 4, 2004.
[8] Bramwell, A., Done, G., and Blamford, D., *Bramwell's Helicopter Dynamics*, Butterworth-Heinemann, 2nd ed., 2001.
[9] Devantech, "http://www.robot-electronics.co.uk/htm/srf08tech.shtml," SRF08 Ultrasonic Ranger.
[10] Ütkin, V., Guldner, J., and Shi, J., *Sliding Mode Control in Electromechanical Systems*, Taylor-Francis Inc., 1999.
[11] Khalil, H. K., *Nonlinear Systems*, Prentice Hall, 1996.
[12] Jang, J. S., *Nonlinear Control Using Discrete-Time Dynamic Inversion Under Input Saturation: Theory and Experiment on the Stanford DragonFly UAVs*, Ph.D. thesis, Stanford University, 2004.
[13] Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
[14] Doya, K., Samejima, K., ichi Katagiri, K., and Kawato, M., "Multiple Model-based Reinforcement Learning," Tech. rep., Kawato Dynamic Brain Project Technical Report, KDB-TR-08, Japan Science and Technology Corporatio, June 2000.
[15] Ng, A. Y. and Jordan, M. I., "PEGASUS: A policy search method for large MDPs and POMDPs," *Uncertainty in Artificial Intelligence*, 2000.
[16] Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E., "Autonomous inverted helicopter flight via reinforcement learning," *International Symposium on Experimental Robotics,*, 2004.
[17] Atkeson, C. G., Moore, A. W., and Schaal, S., "Locally Weighted Learning," *Artificial Intelligence Review*, Vol. 11, No. 1-5, 1997, pp. 11–73.