2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

A Robust Environment for Simulation and Testing of Adaptive Control for Mini-UAVs

Manohar B. Srikanth, Zachary T. Dydek, Anuradha M. Annaswamy, and Eugene Lavretsky

Abstract— **A simulation environment is developed to assist in the design, development, and validation of complex controllers with applications to mini-UAVs, such as the four-rotor DraganFly RC helicopter (quadrotor). The simulation system is modular and includes interfaces which allow for substitution of software subsystems with hardware components. This approach enables a smooth transition from the design and simulation phases to the implementation phase. The benefits of the proposed simulation environment are examined through the application of model reference adaptive control to a quadrotor UAV in the presence of actuator uncertainties and nonlinearities.**

I. INTRODUCTION

Quadrotor helicopters have been an increasingly popular research platform in recent years. Their simple design and relatively low cost make them attractive candidates for swarm operations, a field of ongoing research in the UAV community. Quadrotor helicopters typically consist of two pairs of counter-rotating blades mounted on a carbon fiber frame as shown in Figure 1. In designing a controller for these aircraft, there are several important considerations specific to this problem. There are numerous sources of uncertainties in the system–actuator degradation, external disturbances, and potentially uncertain time delays in processing or communication. These problems are only amplified in the case of actuator failures, where the aircraft has lost some of its control effectiveness. Additionally, the dynamics of quadrotors are nonlinear and multivariate. There are several effects to which a potential controller must be robust: the aerodynamics of rotor blade (propeller and blade flapping), inertial antitorques (asymmetric angular speed of propellers), as well as gyroscopic effects (change in orientation of the quadrotor and the plane of the propeller). References [10],[14],[15],[5] further detail the challenges in designing a controller for UAVs with nonlinear dynamics and parameter uncertainties.

The redundancy in the actuators of a quadrotor makes them robust towards a set of partial failures. Though the performance and maneuverability will most likely be reduced in the case of such a failure, it is desirable for a controller to stabilize the system and allow for reduced mode operations such as a safe return, stable hover, etc. Adaptive control is an attractive candidate for this aircraft because of its ability to generate high performance tracking in the presence of parametric uncertainties.

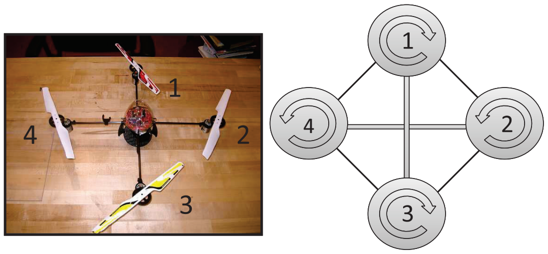


Fig. 1. The two pairs of counter-rotating blades allow the aircraft to hover without rotating about the central point.

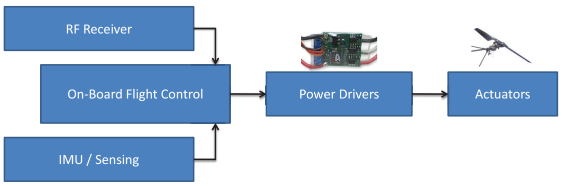


Fig. 2. Quadrotor onboard system. Commands from the transmitter are received by the RF module. An inertial measurement unit (IMU) provides the angular rates to the CPU which generates stabilizing commands. Power drivers amplify the signal from CPU to drive the motors (actuators).

A lightweight, low-power four rotor helicopter such as the commercially available Draganflyer V Ti, from Draganfly Innovations, Inc. provides relatively low onboard computational capability. The onboard CPU handles the flight control system and any communication with a ground based command system. Figure 2 shows the main components of the Draganfly quadrotor (power source not shown). Reference [9] describes a theoretical analysis of the dynamics of the Draganflyer and develops a simple controller which is used to fly the quadrotor autonomously.

The particular focus of this paper is a Simulation, TEst and Validation Environment (STEVE) that assists in the development and facilitates the implementation of onboard and ground-based controllers. STEVE is used not only for the modeling and simulation of the plant and controller, but also for testing and validation of hardware platform functions, such as actuator performance and degradation. STEVE's design allows for hardware-in-the-loop testing, which enables a smooth transition from the design process to on-platform implementation. References [4],[8] cover some of the general issues that arise in the development of software systems for control applications.

This work was supported by the Boeing Strategic University Initiative.

M. Srikanth (srimano@mit.edu), Z. Dydek (zac@mit.edu) and A. M. Annaswamy (aanna@mit.edu) are members of the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02138, USA.

E. Lavretsky (eugene.lavretsky@boeing.com) is a Technical Fellow of the Boeing Company - Phantom Works, Huntington Beach, CA 92647, USA.

978-1-4244-4524-0/09/$25.00 ©2009 AACC

The design of complex controllers for UAV systems presents many challenges for the control designer. Performance differences between simulation and flight tests often necessitate an iterative design process with numerous controller redesigns in order to meet high performance requirements. There is a need for seamless back and forth transition from paper/pencil design to flight testing. The goal of STEVE is to facilitate this iterative design cycle by allowing for intermediate steps and by giving the control designer feedback at every step of the process. This enables fast and efficient design of complex controllers for UAVs.

References [21],[20] propose simulation platforms for control applications of UAVs in particular. The main focus of [21] is to develop an open control platform (OCP) with features such as reconfigurability, extensibility, interoperability, and openness. While OCP is well suited for large scale/complex systems, for a simple system such as a quadrotor it can become overly complex and require significant additional resources, hindering real-time performance. Another discrete-event multi-agent simulation environment [20] is developed in Java. This approach makes hardware-in-the-loop testing, which demands strict real-time rates, considerably more difficult. Furthermore, neither of these systems address the goal of an iterative design process which is at the core of STEVE.

To demonstrate the advantages of STEVE, we examine the application of model reference adaptive control (MRAC) to a quadrotor UAV in the presence of actuator uncertainties and nonlinearities.

II. QUADROTOR MODELING AND CONTROL DESIGN

Since the goal of STEVE is to allow for seamless transition from simulation to flight testing, an accurate, high-fidelity quadrotor model is required. This will ensure that controllers designed and tested using the simulated quadrotor dynamics will perform well on the actual hardware with minimal additional tuning. The model designed and implemented on STEVE is described in Section II-A below. A baseline, fixed-gain controller as well as a model reference adaptive controller (MRAC) are designed using this model. An overview of the controller design procedure is described in Section II-B.

A. Quadrotor Model

A quadrotor model which includes nonlinear aerodynamics as well as actuator dynamics and saturation has been developed for use with STEVE. The dynamics of the quadrotor have been studied in detail by several groups [1], [2]. A simple, rigid-body model of the quadrotor which assumes low speeds is given by:

$$\begin{aligned}
\ddot{x} &= (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)\,\frac{U_1}{m} \\
\ddot{y} &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)\,\frac{U_1}{m} \\
\ddot{z} &= -g + (\cos\phi\cos\theta)\,\frac{U_1}{m}
\end{aligned}$$

$$\begin{aligned}
\ddot{\phi} &= \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_R}{I_x}\dot{\theta}\Omega_R + \frac{L}{I_x}U_2 \\
\ddot{\theta} &= \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) + \frac{J_R}{I_y}\dot{\phi}\Omega_R + \frac{L}{I_y}U_3 \\
\ddot{\psi} &= \dot{\phi}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{1}{I_z}U_4
\end{aligned}$$

where $x, y,$and $z$are the position of the center of mass in the inertial frame; $\phi, \theta,$and $\psi$are the Euler angles which describe the orientation of the body-fixed frame with respect to the inertial frame; $m, I_x, I_y,$and $I_z$are the mass and moments of inertia of the quadrotor respectively; and $J_R$and $\Omega_R$are the moment of inertia and angular velocity of the propeller blades. $U_1, U_2, U_3,$and $U_4$are the collective, roll, pitch, and yaw forces generated by the four propellers.

An actuator model has been developed using simple DC motor equations and momentum balance to determine the thrust generated by the propellers. Since the onboard motor drivers use pulse width modulation (PWM) to vary the speed of the motors, there are hard actuator saturation limits which correspond to 0% and 100 % duty cycle. Actuator failures have also been included as a multiplication of the motor thrusts by unknown constants $0 \leq \lambda_i \leq 1$where a value of $\lambda = 1$corresponds to the no-failure case and $\lambda = 0$corresponds to a complete failure.

B. Controller design

The overall control architecture consists of an simple onboard controller, a fixed-gain baseline controller, as well as an augmented adaptive controller. This approach primarily addresses the problem of actuator uncertainty, but is also robust with respect to other types of uncertainties.

The Draganfly quadrotor comes shipped from the manufacturer with an onboard inner-loop controller designed to increase stability and ease of use by human operators. Since the specifics of the onboard control system are not in the public domain, it is assumed that the control designers used an LQR PD controller in order to regulate the aircraft about the level hover position. This assumption was validated when the results from the simulation and the actual flight tests were compared. Thus, the onboard command signal is given by $\delta_{OB} = K_x x$where $K_x$minimizes a quadratic cost function

The baseline controller, designed using classical control techniques, consists of several decoupled sub-loops which allow the quadrotor to follow $x, y, z,$and ψcommands. The altitude $z$and the yaw angle $\psi$are controlled individually through a set of PID controllers. The pitch loop feeds the $x$position and pitch angle θback through a pair of lead controllers which generate pitch commands that will allow

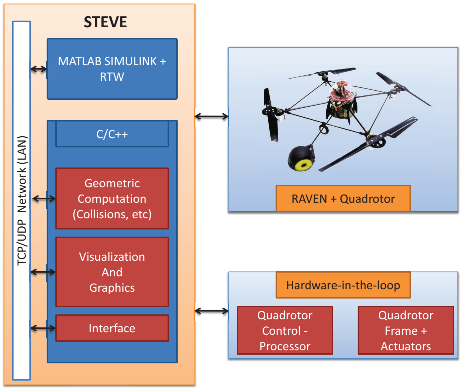


Fig. 3. STEVE: Simulation, TEst and Validation Environment.

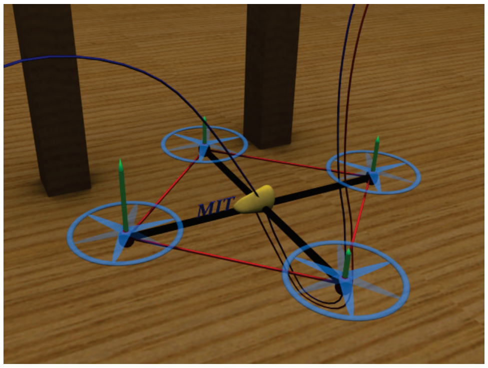


Fig. 4. Quadrotor Visualization. Blue curve is the commanded trajectory, and the red curve is the trajectory followed by the simulated quadrotor. The green bars/arrows at the center for the propeller show the force generated at the actuator.

the aircraft track the desired trajectory. The roll loop, which is essentially equivalent to the pitch loop due to the symmetry of the aircraft, generates appropriate roll commands. This approach is quite robust and has been implemented in previous work on this aircraft [3], [17].

The main problem that needs to be addressed by the augmented model reference adaptive controller (MRAC) is the accommodation of uncertainties that occur due to actuator anomalies. These uncertainties are represented by a combination of two features: the parametric uncertainty that represents loss of control effectiveness, and a saturation nonlinearity in the actuator. The goal of the adaptive controller is to recover/maintain nominal closed-loop system performance, even in the presence of uncertainties. A detailed description of the adaptive control design process and proof of stability can be found in [11], [13], [12].

III. SIMULATION, TEST AND VALIDATION ENVIRONMENT

While controller design and simulation has its own challenges, implementing the controller on a physical platform poses yet another set of challenges. Existing simulation packages alone do not provide the flexibility, scalability, and computational efficiency necessary to deal with the challenges posed by our requirements. To meet these requirements, the Simulation, TEst and Validation Environment (STEVE) was developed using a combination of existing simulation packages and our own in-house developed modules. Figure 3 describes the STEVE architecture. The modularity of STEVE allows integration of available software modules without modifying the existing system. It also permits independent code development, and allows dynamic addition and deletion of components such as hardware-in-loop interfaces, sensors, controllers, collision detection and response, and visualization modules. This approach has created a robust and full-featured environment in which one can take complex controllers from theory to simulation to hardware testing to free flight.

The overarching simulation system included in STEVE consists of Matlab Simulink with the Real-time Windows Target. In order to make the controller design and implementation process fast, easy and efficient, the following core functionalities have also been included in STEVE:

- Real-time system
- Collision Detection and Response System
- Visualization system (Custom OpenGL Graphics)
- Hardware-in-the-loop interface
- Flight test (motion capture system) interface

The various modules in the system are connected via LAN (local area network) and communicate using the UDP protocol. In this way, modules can reside on the same computer or they can be distributed across multiple computers on the same LAN. A description of the modules included in STEVE is given below.

A. Real-time System

Since the system intends to allow for hardware-in-the-loop testing, the entire simulation process must be run in real time. There are two ways in which this can be achieved. One is using the Matlab Simulink real-time feature, and the other is using C++ multimedia timers. Although Simulink real-time feature is sufficient for many applications, compatibility issues with some data acquisition boards and the presence of large number of modules in the Simulink model make it difficult to implement. Multimedia timers are set of functions available in Windows to perform timed tasks. Using multimedia timers, we are able to run the simulation up at speeds of up to 800 Hz.

B. Collision Detection and Response

A collision imparts an impulsive force on the quadrotor, and can occur in any direction. External disturbance rejection is one of the features of the augmented adaptive control structure. The proposed simulation environment integrates a collision detection algorithm and hence allows the user to evaluate the performance of the controller to external disturbances.

The collision detection algorithm is a computationally intensive process. In applications where a large number of obstacles are present, such as a swarm of UAVs, this module could be run on its own dedicated machine. Open source libraries (ICOLLIDE [19]) were used to implement

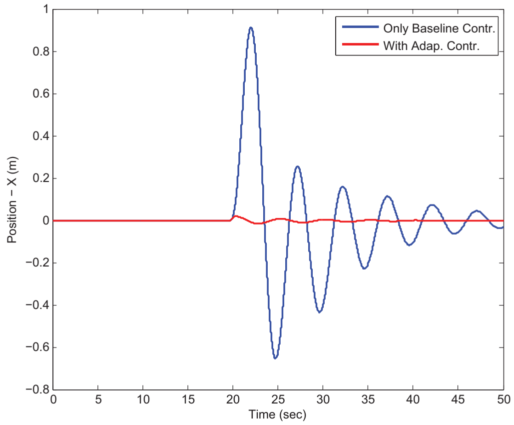


Fig. 5. Performance of the adaptive controller to collisions. Collision force (impusive) occurs at the x-axis motor, resulting in a force in z-axis (up). This leads to perturbations in x-axis, which is plotted. Adaptive controller responds to this external force more aggresively than the baseline controller in minimizing the perturbations.

the collision detection module. The main simulation system sends out the pose parameters of the quadrotor to the collision detection system, and in return gets the collision response. The collision response consists of the following information: points of intersection of quadrotor geometry with other objects, penetration depth at those points, the and reaction force vectors. This information is then used to update the dynamics of the simulated 6DoF quadrotor. Figure 5 shows the disturbance rejection of baseline and adaptive controller to a vertical (up) impact.

C. Visualization System

The visualization system is built on OpenGL graphics libraries. It allows real-time rendering of the quadrotor with trajectory and force parameters overlayed on top, as shown in Figure 4. Real-time visualization is an invaluable tool during hardware-in-the-loop testing. While the physical quadrotor is bolted down to the test stand, the visualization system displays the path of the virtual quadrotor as the simulation runs. This visualization gives the control designer the tools necessary to closely examine the behavior of the controller, allowing them to make adjustments and perform tuning quickly and efficiently. The states of the quadrotor are sent to the visualization module via the TCP/IP layer, which is used to update the graphics at a rate of 30 frames per second. The use of the TCP/IP protocol allows for running the visualization system on another computer, further distributing the computational load.

D. Hardware-in-the-loop Interface

Hardware-in-the-loop testing involves replacing the simulated elements of quadrotor with actual physical elements. This is an intermediate step before doing a full flight test. STEVE's interface to the hardware-in-the-loop components enables efficient testing and tuning of adaptive controllers. Figure 6 describes this setup. The quadrotor frame is mounted to a sensitive force-torque sensor which is then mounted to a rigid test stand. The force-torque data acquisition is done with an NI DAQ card. Noise filters are implemented to remove high frequency noise and also to suppress the structural modes that arise as a result of the aircraft being bolted to the test stand. The force and torque measurements are then used to evolve the 6DoF rigid body dynamics. It is here that external forces, such as aerodynamic forces and collision-response forces, are included as well. The controller uses the state of the rigid body dynamics and generates motor command signals which are then transmitted to the quadrotor via the NI DAQ board and PWM Amplifiers. Actuator protection logic is implemented to limit the current of the command signals. A separate module also monitors the approximate current flow, which is used to estimates the total energy consumption.

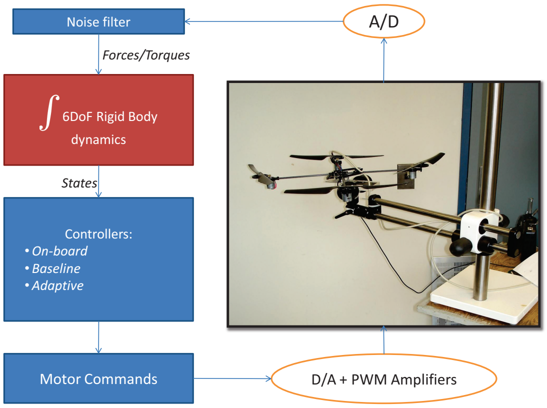


Fig. 6. Hardware-in-loop setup.

E. Motion capture system Interface

This interface enables STEVE to run free flight tests of the adaptive controller on the quadrotor hardware. A camera based motion capture system the states of the quadrotor. During these tests, the controller components of STEVE are used to generate control signals which are then sent to the quadrotor through RF transmitter.

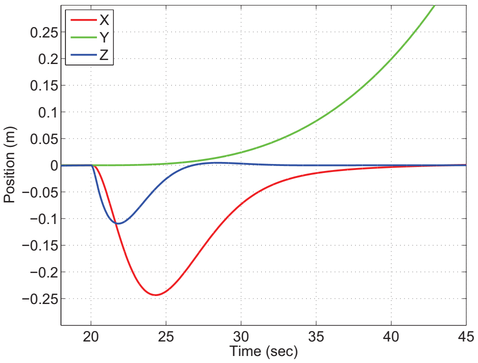
IV. SIMULATION RESULTS

In order to test the advantages of STEVE, several simulation studies of advanced flight controllers were performed in the presence of various uncertainties. In the first set of simulations, the quadrotor is commanded to maintain a fixed hover position. One of the actuators is then subjected to either a loss of control effectiveness or a sudden change in the actuator saturation limits. Both the baseline controller and adaptive controller are implemented using STEVE, and their performances for satisfactory regulation are compared. In the second set of simulations, the quadrotor is commanded to follow a three dimensional trajectory, and at a point in between, the aircraft is again subjected to an actuator anomaly. The tracking performances of the adaptive and baseline controllers are then compared.

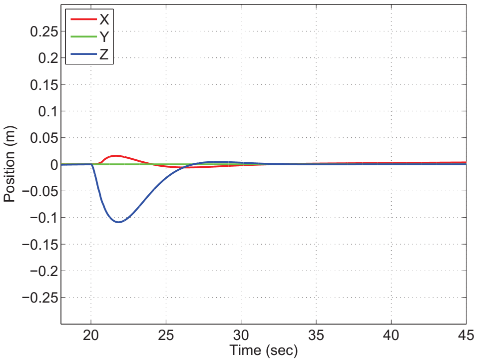
A. Regulation

The simulation environment, STEVE, discussed in the previous sections is used to compare the baseline controller, which consists of the onboard control and the outer loop linear controller, to the adaptive controller

For these simulations, the quadrotor is commanded to hover at the point defined as $(0, 0, 0)$in xyzspace. An 80% loss of control effectiveness is injected into one of the actuators at a time $t = 20$seconds. As can be seen in Figure 7, the baseline controller gradually departs from the hover position. However, the adaptive controller is able to maintain stability in this case.



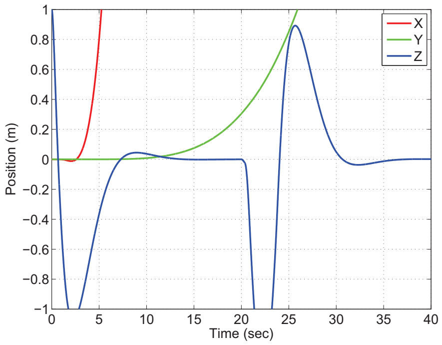
(a) Baseline



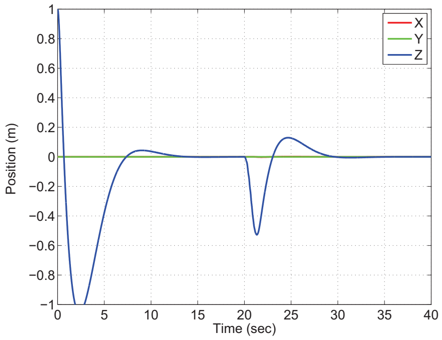
(b) Adaptive

Fig. 7. Regulation performance of the (a) baseline and (b) adaptive controller when subjected to an 80% loss of control effectiveness in one actuator.

Another class of actuator anomalies is an uncertainty in the actuator saturation limits. For the following example, the upper actuator saturation limit was reduced to 74% of the nominal value. Figure 8 shows that the baseline controller is very sensitive to the change in saturation limits, becoming unstable even before the failure is injected at a time of $t = 20$seconds. The adaptive controller, on the other hand, retains stability in spite of both the saturation uncertainty as well as the loss of control effectiveness.



(a) Baseline



(b) Adaptive

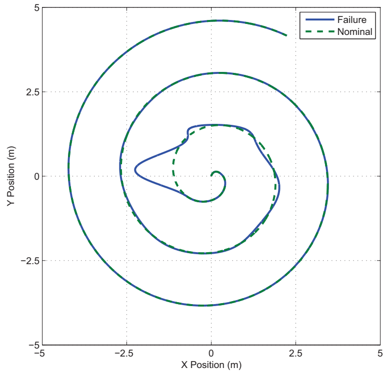
Fig. 8. Regulation performance of the (a) baseline and (b) adaptive controller when subjected to an 74% reduction of the upper saturation limit of one actuator.

B. Command Tracking

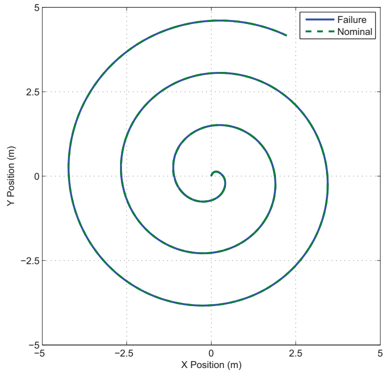
In this second set of simulations, the quadrotor is commanded to follow a complex three dimensional trajectory. The commanded trajectory used for this testing consists of a gradual spiral outward from $(0, 0, 0)$in the $x - y$plane as well as a change in altitude. This corresponds to a helix-like motion.

For this example, we have introduced a 50% loss of control effectiveness in one of the actuators, as well as a 20% reduction in the actuator saturation limits. Looking at a birds-eye view of the trajectory followed by the quadrotors, we can see a dramatic difference between the baseline and adaptive controllers in Figure 9. In this example both controllers retain stability and eventually follow the commanded trajectory. However the large departure from the trajectory as seen in Figure 9 (a) may be undesirable especially if, for example, the quadrotor is operating in a tight environment with many walls, obstacles, or other UAVs.

The architecture of STEVE allows one to design and test complex controllers efficiently. Prior to the development of STEVE, the design of the adaptive controller was carried out in a pure simulation environment (design phase), and then ported as C++ code to a flight test system (flight phase). In our experience, this two phase process involves unnecessary steps that consume time and energy. There will always be performance differences between simulation results and flight test results; in a two phase process it is often difficult to take lessons learned from the flight phase back to the design phase. The STEVE architecture, on the other hand, provides a seamless path from design phase to flight via several phases of hardware-in-the-loop testing. Furthermore, at each step along the way, visualizations increase transparency and give



(a) Baseline



(b) Adaptive

Fig. 9. Trajectory executed by the (a) baseline and (b) adaptive controller.

the user insight as to the behavior of the controller and the dynamics. STEVE is also general enough to be applied to a class of rotorcraft UAVs coupled with a variety of controllers.

V. SUMMARY

In this paper, a modular simulation environment, STEVE, is proposed to assist in the design, development, and validation of complex controllers for UAVs. The advantages of STEVE are examined via the application of model reference adaptive control to a four-rotor DraganFly RC helicopter. Regulation and command tracking are examined in the presence of actuator failures.

The simulations demonstrated that the proposed adaptive controller is robust towards actuator uncertainties, and is effective in stabilizing the quadrotor during hover as well as during command tracking. STEVE enabled a fast and efficient iterative design of this complex controller and provided tools to evaluate the controller's performance. By allowing for internmediate steps such as hardware-in-the-loop testing and by giving the control designer feedback at every step of the process, STEVE simplifies the design of complex controllers for UAVs. Hardware-in-the-loop tests and flight tests using more realistic test cases are currently being tested using STEVE.

VI. ACKNOWLEDGMENTS

This research is funded by the Boeing Company Strategic University Initiative. We would like to thank group members of Aerospace Controls Laboratory of MIT, and in particular Professor How, Brett Bethke and Buddy Michini, for providing us with valuable information and access to the RAVEN facility.

REFERENCES

[1] S. Bouabdallah, P. Murrieri, and R. Siegwart, Design and Control of an Indoor Micro Quadrotor. *In Proc. ICRA*, New Orleans, Louisianna, 2004.
[2] Gabriel M. Hoffmann, Haomiao Huang, Steven L. Waslander, and Claire J. Tomlin, Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment, *In Proc. AIAA GNC*, Hilton Head, South Carolina, 2007.
[3] Glenn P. Tournier, Mario Valenti, Jonathan P. How, and Eric Feron, Estimation and Control of a Quadrotor Vehicle Using Monocular Vision and Moire Patterns, *In Proc. AIAA GNC*, Keystone, Colorado, 2006.
[4] Miaobo Dong, Ben M. Chen, Guowei Cai, and Kemao Peng, Development of Real-time Onboard and Ground Station Software System for a UAV Helicopter, *Journal of Aerospace Computing, Information, and Communication,* Vol.4 Aug 2007.
[5] Mac Schwager, Anuradha Annaswamy, and Eugene Lavretsky, Adaptation-Based Reconfiguration in the Presence of Actuator Failures and Saturation. *In Proc. Amerial Controls Conference, pp-2640-2645, 2005. Portland. OR. USA.*
[6] D. Blake Barber, Stephen R. Griffiths, Timothy W. McLain, and Randal W. Beard, Autonomous Landing of Miniature Aerial Vehicles. *Journal of Aerospace Computing, Information, and Communication. Vol 4. May 2007.*
[7] Gabriel M. Hoffmann, Haomiao Huang, Steven L. Waslander, and Claire J. Tomlin, Quadrotor Helicopter Flight Dynamics and Control: Theory and Expriment. AIAA Guidance, Navigation, and Control Conference and Exhibit.20-23 August 2007, Hilton Head, South Carolina
[8] John P. Enright and David W. Miller, Formal Verification of Real-time Middleware, *Journal of Aerospace Computing, Information, and Communication. Vol 4, June 2007.*
[9] Phillip M^{c}Kerrow, Modeling the Draganflyer four-rotor helicopter. *In Proc. IEEE ICRA, New Orleans, LA, Paril 2004. pp-3596-3601*
[10] A. Benallegue, A Mokhtari, and L. Fridman, Higher-Order sliding-mode observer for a quadrotor UAV. *Int. J. Robust Nonlinear Control 2008*; 18:427-440
[11] S. P. Karason and A. M. Annaswamy, Adaptive control in the presence of input constraints, *IEEE Transaction on Automatic Control*, vol. 46, no. 11, pp. 2325–2330, November 1994.
[12] Z. T. Dydek, H. Jain, J. Jang, A. M. Annaswamy, and E. Lavretsky, Theoretically verifiable stability margins for an adaptive controller *In Proc. AIAA Conference on Guidance, Navigation, and Control (GNC'06)*, Keystone, Colorado, August 2006.
[13] K. S. Narendra and A. M. Annaswamy, Stable Adaptive Systems. Englewood Cliffs, N.J.: Prentice-Hall.
[14] Jong-Rae, Model-error control synthesis: a new approach to robust control. *PhD Dissertation,* Texas A&M University, College Station, TX, August 2002
[15] Castillo P, Lozano R, Dzul A. Modeling and Control of Mini-flying Machines. Springer: Berlin, 2005.
[16] Kim B.K. Disturbance observer based approach to the design of sliding mode controller for high performance positioning systems. *Fifteenth Triennial World Congress (IFAC)*, Barcelona, 2002.
[17] M. Valenti, B. Bethke, G. Fiore, J. How, E. Feron "Indoor Multi-Vehicle Flight Testbed for Fault Detection, Isolation, and Recovery", AIAA Conference on Guidance, Navigation and Control, August 2006, Keystone, CO
[18] http://www.rctoys.com/ Draganfly Innovations Inc. 2108 St. George Avenue, Saskatoon, SK S7M0K7, Canada.
[19] http://www.cs.unc.edu/~geom/collide/
[20] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, Gabriel Balan "MASON: A Multiagent Simulation Environment", SIMULATION, Vol. 81, Issue 7, July 2005 517-527
[21] Linda Wills, Suresh Kannan, Sam Sander, Murat Guler, Bonnie Heck, J.V.R Prasad, Daniel Schrage, and George Vachtsevanos " An Open Platform for Reconfigurable Control", IEEE Controls Systems Magazine, June 2001 pp 49-64