Cerebellar Model Articulation Controller with Introspective Voting Weight Updates for Quadrotor Application

Troy Clark
Department of Electrical and Computer Engineering
Schulich School of Engineering, University of Calgary
Calgary, Alberta, Canada, T2N 1N4
Email: clarkta@ucalgary.ca

C.J.B. Macnab
Department of Electrical and Computer Engineering
Schulich School of Engineering, University of Calgary
Calgary, Alberta, Canada, T2N 1N4
Email: cmacnab@ucalgary.ca

***Abstract*—This paper explores control of a four rotor aerial vehicle, or quadrotor. The quadrotor exhibits under-damped and under-actuated dynamics, so designing a feedback control that can provide robustness to disturbances such as wind and adapt to payloads provides a research challenge. The Cerebellar Model Articulation Controller (CMAC) is a type of neural network that can provide the basis for a stable adaptive neurocontrol. However, traditional robust weight update schemes for the CMAC must often sacrifice performance to eliminate bursting caused by weight drift. This work applies an introspective voting weight update scheme to a CMAC to avoid busting, without reducing performance. The proposed scheme achieves stable, adaptive, and robust control in both simulation and experiment.**

***Index Terms*—Cerebellar Model Articulation Controller (CMAC), Direct Adaptive Control, Neural Network Control, Quadrotor, Aerial Drone, Bursting, Weight Drift, Overlearning**

I. Introduction

The quadrotor can perform a multitude of tasks due to its vertical take-off and land ability, and multi-directional manoeuvrability. A quadrotor uses just four motor outputs to actuate control over a total of six degrees of freedom in a three dimensional operating environment, making it an under-actuated system. In addition, the quadrotor has little opposition or friction to retard its movement, especially in the x and y planes, meaning it is also an under-damped system. Furthermore, nonlinear thrust dynamics such as temperature, altitude, atmospheric pressure, angle of attack, and airspeed have historically made the quadrotor difficult to control via human intuition alone. Developing a feedback control that is robust to wind and adaptive to payloads will foster vehicle autonomy, allowing future research effort to be focused on more advanced applications for this platform.

Linear controls, ranging from simple proportional derivative (PD) control [1], to more advanced variations have been proposed with some success [2], but often in a purely theoretical forum. One study found that stability of a simple PD control could only be guaranteed if the actual system parameters differed by no more than 5% from the modelled values, and defined a maximum system delay of only 23ms [3].

Investigation into quadrotor dynamics has uncovered non-linearities present due to aerodynamic drag and rotor vortices, especially during speed changes [4]. Additionally, extensive work has been done to predict a previously unmodeled disturbance known as blade-flapping [5]. A number of projects, especially [6] and [7], have documented difficulty in predicting and modelling all dynamics of a quadrotor vehicle. As a result, many successful quadrotor control research projects have embraced an adaptive approach, including fuzzy regulators [8], backsteppers [9], and feedback linearizers [10].

However, adaptive controls are not without their own unique challenges. The multi-layer perceptron (back-propagation network) may not converge fast enough to adapt to wind and payloads. The radial-basis-function network (RBFN) can adapt quickly but suffers from the "curse of dimensionality" which limits the practical implementation of an RBFN in a system with as many inputs as a quadrotor [11]. The Cerebellar Model Articulation Controller (CMAC), on the other hand, adapts quickly and avoids the curse of dimensionality.

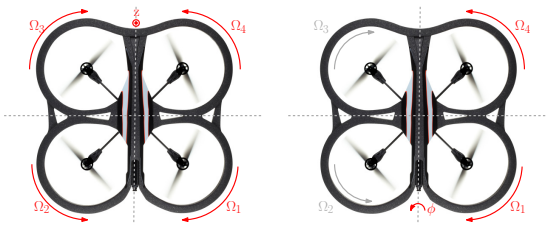
When affected by sinusoidal disturbances, like wind, a neural network will often suffer weight drift. Weight drift of a large magnitude results in bursting [12], which is characterized by sudden unexpected increases in error. Traditional robust modifications may keep weights bounded, but will restrict performance. This trade-off between performance and stability is especially pronounced in the CMAC, where an oscillation between two CMAC memory cells and across the zero-error point immediately leads to opposing weight drift in adjacent cells.

978-1-5090-0996-1/16/$31.00

This work will apply an introspective learning algorithm to the CMAC that can avoid weight drift without sacrificing performance previously described in [13], first in simulation and then experimentally.

II. The Quadrotor Vehicle

A quadrotor incorporates four rotors laid out in a box formation. Any two blades on opposing corners of the formation will spin in the same direction. The other two blades will also spin in the same direction, but opposite that of the first pair.



(a) Z Translation (c) Roll Rotation (ϕ)

(b) Pitch Rotation (θ) (d) Yaw Rotation (ψ)

Fig. 1: Quadrotor Dynamics

As defined by [14], the motion of a quadrotor vehicle is straight forward. The most basic movement is simply a change in height, or Z position. This motion is accomplished by varying the speeds of all four rotors in unison, producing a change in Z position that is proportional to the change in speed (1), as shown in Figure 1a.

$$U_1 = \frac{\ddot{z}}{b} = \Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2 \tag{1}$$

Next, increasing the speed of two blades closest to one another, while decreasing that of the pair across the quadrotor, will produce a change in the roll ϕ (2) or pitch θ (3) of the vehicle, as shown in Figures 1c and 1b.

$$U_2 = \frac{\ddot{\phi}}{lb} = \Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2 \tag{2}$$

$$U_3 = \frac{\ddot{\theta}}{lb} = \Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2 \tag{3}$$

As a by-product of this change in angle, a portion of the downward thrust will be directed along one or both of the horizontal planes of the quadrotor, thereby affecting its position and velocity. Finally, increasing the speed of the two blades rotating in the same direction, while decreasing the speed of the counter rotating pair, will introduce an imbalance in the net torque of the vehicle, resulting in a change in yaw ψ (4), as shown in Figure 1d.

$$U_4 = \frac{\ddot{\psi}}{d} = -\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2 \tag{4}$$

These controls consider basic vehicle parameters such as the thrust coefficient of the rotors, b, rotor to vehicle centre distance, l, and aerodynamic drag, d. When formulating a mathematical model of the complete quadrotor, gravitational force, mg, angular velocity, $\dot{\phi}$, $\dot{\theta}$, or $\dot{\phi}$, and rotor inertia, I_r, must all be taken into account.

$$\mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} mg + b \begin{bmatrix} C\phi S\theta C\psi + S\phi S\psi \\ C\phi S\theta S\psi + S\phi C\psi \\ C\theta C\phi \end{bmatrix} U_1 \tag{5}$$

$$\tau = \begin{bmatrix} I_{yy} - I_{xx} \\ I_{zz} - I_{xx} \\ I_{xx} - I_{yy} \end{bmatrix} \begin{bmatrix} \dot{\theta}\dot{\psi} \\ \dot{\phi}\dot{\psi} \\ \dot{\phi}\dot{\theta} \end{bmatrix} + \begin{bmatrix} -I_r \\ I_r \\ 0 \end{bmatrix} \Omega_r + \begin{bmatrix} lb \\ lb \\ d \end{bmatrix} \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{6}$$

Ignoring any nonlinearities present in the thrust coefficient due to environmental considerations, overlooking trivial unmodeled dynamics such as blade-flapping, and neglecting rotor acceleration by slowly varying speeds, the basic quadrotor model is represented by a simple force (5) and torque (6) vector.

III. Control Design

The basic control design of the system employed consists of an inner loop responsible for stabilizing the orientation and vertical position of the vehicle and an outer loop responsible for horizontal position. At one end of the loop is user input, either from a direct control signal or a translated trajectory path. At the other end is the quadrotor vehicle, specifically the motor control signals that actuate the platform. Finally, at the heart of the control scheme is a series of four CMACs.

Direct inputs to the control system include desired roll, pitch, yaw, and Z values. The difference between the measured and desired values for these variables, as well as the rate of error change, are combined to form the auxiliary error of the system (7).

$$\zeta = \Lambda \begin{bmatrix} z - z_d \\ \phi - \phi_d \\ \theta - \theta_d \\ \psi - \psi_d \end{bmatrix} + \begin{bmatrix} \dot{z} - \dot{z}_d \\ \dot{\phi} - \dot{\phi}_d \\ \dot{\theta} - \dot{\theta}_d \\ \dot{\psi} - \dot{\psi}_d \end{bmatrix} \tag{7}$$

A. Cerebellar Model Articulation Controller

The CMAC was originally proposed by James Albus as an analogue to the simple motor control function performed by the human brain [15]. The CMAC is an associative memory device often used as a nonlinear approximator. It receives an input vector, $\mathbf{q} \in \Re^N$, that is applied to a series of offset memory layers, each divided into quantized intervals. On each layer the inputs activate a single N-dimensional hypercube cell and within each cell an activation function, Γ, produces a single value, as shown in Figure 2.

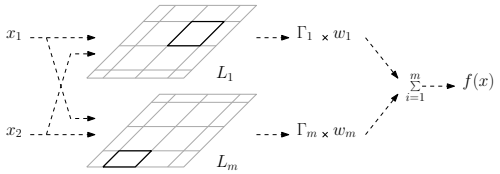


Fig. 2: CMAC Cell Matrix Structure

On the ith layer the unique value produced by the activation function $\Gamma_i(x)$ is multiplied by a weight w_i. The products of each individual layer are then summed to build the CMAC output (8).

$$f(x) = \sum_{i=1}^{m} \Gamma_i(\mathbf{q}) w_i \tag{8}$$

Weights begin with an initial value of zero and are updated with an adaptation rule. When a weight is updated, its value is updated in physical memory and it is accessed by address using a random hash coding scheme. In this way, only (previously) activated weight values are saved, dramatically reducing memory requirements.

B. Lyapunov-Stable Adaptive Control

A CMAC, or indeed any neural network, can be trained to approximate a nonlinearity in a system. In this way, the nonlinearity can be cancelled, greatly simplifying control. In general terms, a nonlinear function, h, subject to an input vector, $\mathbf{q}$, can be approximated by a CMAC with an approximation error, d, in region $\mathcal{D} \subset \Re^N$ as defined by (9).

$$h(\mathbf{q}) = \mathbf{\Gamma}(\mathbf{q})\mathbf{w} + d(\mathbf{q}) \quad \forall \mathbf{q} \in \mathcal{D} \tag{9}$$

such that $|d(\mathbf{q})| < d_{\max}$ where $d_{\max}$ is a positive constant. The maximum modelling error of the system, $d_{\max}$, may be unknown initially, but can be determined experimentally. The ideal weights for the CMAC, w, are unknown, so a weight error (10), becomes the focus of control considerations.

$$\tilde{w} = w - \hat{w} \tag{10}$$

To calculate the stability of the CMAC, Lyapunov's direct method can be used. First, a Lyapunov function representing the error present in the system (11).

$$V = \frac{1}{2\mathbf{c}}\zeta^2 + \frac{1}{2\beta}\tilde{\mathbf{w}}^T\tilde{\mathbf{w}} \tag{11}$$

Here, $\mathbf{c}$ is a positive vector representing system constants, and β is a positive adaptation gain for CMAC weight learning. The derivative of this Lyapunov candidate, defined by (12), will model how the system error will change, and ideally dissipate, over time.

$$\dot{V} = \zeta\left(\frac{f(\zeta) - \dot{\zeta}}{\mathbf{c}} + \mathbf{U}\right) - \frac{1}{\beta}\tilde{\mathbf{w}}^T\dot{\hat{\mathbf{w}}}$$

$$\dot{V} = \zeta(\mathbf{\Gamma}(\mathbf{q})\mathbf{w} + d(\mathbf{q}) + \mathbf{U}) - \frac{1}{\beta}\tilde{\mathbf{w}}^T\dot{\hat{\mathbf{w}}} \tag{12}$$

A CMAC has been used to approximate the function, $\frac{f(\zeta)-\dot{\zeta}}{\mathbf{c}}$. The control (13) and robust e-modification (e-mod) weight update rule (14) are chosen to simplify the derivation.

$$\mathbf{U} = -\mathbf{\Gamma}(\mathbf{q})\hat{\mathbf{w}} - k\zeta \tag{13}$$

$$\dot{\hat{\mathbf{w}}} = \beta(\mathbf{\Gamma}^T(\mathbf{q})\zeta - \nu\|\zeta\|\hat{\mathbf{w}}) \tag{14}$$

where k is a vector of positive definite control gains, and ν is a small positive constant. Setting the Lyapunov Derivative (12) equal to zero results in the definition of an elliptical bound on the error (15) and weight values (16) in the system encapsulating the origin.

$$\dot{v} < 0 \text{ when } |\zeta| > \frac{d_{\max}}{4} + \frac{\nu\|\mathbf{w}\|^2}{4k} \tag{15}$$

or

$$\dot{v} < 0 \text{ when } |\tilde{\mathbf{w}}| > \frac{\|\mathbf{w}\|}{2} + \sqrt{\frac{d_{\max}}{\nu} + \frac{\|\mathbf{w}\|^2}{4}} \tag{16}$$

This result implies that the error and weight values of the system are uniformly ultimately bounded (UUB). The reader is referred to [16] for the modifications required for quadrotor control.

C. Bursting

While the weight and system errors have been shown to be bounded within a Lyapunov surface, the system is not immune to the adaptive disturbance known as bursting. Bursting occurs when the magnitudes of adaptive weights drift towards large values, eventually resulting in an increase in state error [17].

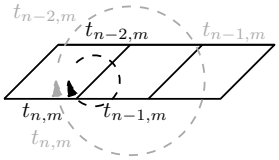


Fig. 3: Oscillating CMAC Cell Cycle Leading to Bursting

Bursting typically occurs in a CMAC when the input trajectory oscillates between two neighbouring or nearby active cells, and the error oscillates over the origin, as shown in Figure 3. This oscillation drives one weight to large positive values, and the other to large negative values. In the case of the quadrotor, this oscillation can be introduced by external factors, especially wind. The physical mechanism for this is often an increase in control signal chatter due to the weights going in opposite directions on each side of the origin, with large enough chatter eventually leading to increased error.

To ensure the system converges to a stable limit cycle, close to the origin, it is important to monitor and limit weight drift. In e-modification, performance may be sacrificed by increasing ν enough to stop weight drift. This can be detrimental to systems where high performance is required, such as the quadrotor. The introspective voting weight update scheme, however, directly monitors weight values to eliminate bursting.

D. Introspective Voting Weight Update

The introspective voting weight update system [18] requires no *a priori* knowledge of model parameters such as maximum disturbance, minimum weights, or maximum weights. Furthermore, the introspective voting system analyses the effect of a weight update before permanently storing it: it ensures that the weight update is indeed reducing the error significantly. In this way, weight growth is restricted to that needed to reduce the error, and unnecessary weight drift is eliminated.

Active weights are updated according to a standard Lyapunov-stable, e-mod style weight update (14). However, when a cell becomes deactivated, the introspective algorithm makes a decision on whether to keep this weight update in permanent memory for the next activation, or move the weight closer to the best value found thus far. This decision is based on the total error over two subsequent cell activations. The number of times the cell has been activated, *i.e.* the number of periodic cycles of the trajectory, is denoted by m, and the number of cell activations on the current cycle is specified by n. The permanent weight update for cell $n-2$ at the moment of a cell activation n is given by (17).

$$\Delta\hat{p}_{n-2} = \begin{cases} \hat{w}_{n-2,m-1} & \text{if vote}_{n-2} > 0 \\ \mu(\hat{b}_{n-2} - \hat{p}_{n-2}) & \text{Otherwise} \end{cases} \tag{17}$$

Here, $\hat{b}_{n-2}$ is the best weight stored in memory; the one that has resulted in the least error over two subsequent cells in the training to this point. The term $vote_{n-2}$ is the result of the introspective voting from each layer, based on whether it has seen its weight update reduce the error since the last cycle (18).

$$vote_{n-2} = \begin{cases} |\hat{w}_{n-2,m-1}| & \text{if } \Delta E_{n-2}/\hat{w}_{n-2,m-1} < 0 \\ & \text{and } E_{n-2,m}E_{n-2,m-1} > 0 \\ & \text{and } |\Delta E_{n-2}| > \delta_{E,\min} \\ -|\hat{w}_{n-2,m-1}| & \text{otherwise} \end{cases} \tag{18}$$

Here E_{n-2} is the total state error over cell $n-2$ and $n-1$. A vote will go in favour of a weight update if three conditions are met. First, the error and weight update must be of opposing signs to signify that the update is aiding the system. Second, the sign of the error must be the same as it was in the last calculation to show that the weight update is still moving in the correct direction. And most importantly, the weight update must significantly decrease the output error of the system, the variable of significance being represented by $\delta_{E,\min}$.

IV. Results

Through simulation and experimentation the performance of a CMAC with introspective voting weight updates is compared to a traditional proportional-integral-derivative (PID) scheme, and a CMAC using a leakage style e-mod robust weight update. The goal is to demonstrate that the introspective voting CMAC scheme is stable, adaptive, robust, and better suited to autonomy than existing controls.

A. Simulation

In the first simulation, the stability and robustness of the introspective weight voting scheme is compared to a traditional e-mod style update. In a MATLAB environment, a quadrotor model is asked to take-off and hover at a height of one meter. The learning and adaptation gains of the CMAC are tuned for performance, over stability, and a simulated sinusoidal wind disturbance is added. This represents the worst type of disturbance for bursting behaviour.

The e-mod CMAC update scheme is observed to follow a bursting cycle (red solid line in Figure 4, top).

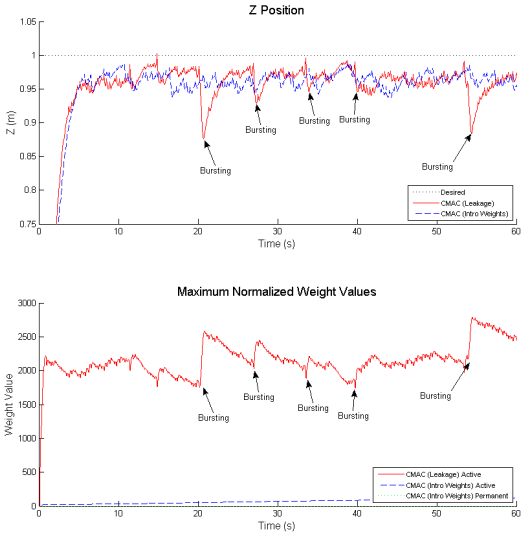


Fig. 4: Z Position and Maximum Weight Values Showing Bursting Phenomenon in Simulation

Long periods of stable progress towards the error origin are interrupted by sudden, unpredictable spikes in error due to weight drift (red solid line in Figure 4, bottom). The introspective voting scheme shows a stable error value (blue dashed line in Figure 4, top) and the weights converge (blue dashed line and green dotted line in Figure 4, bottom).

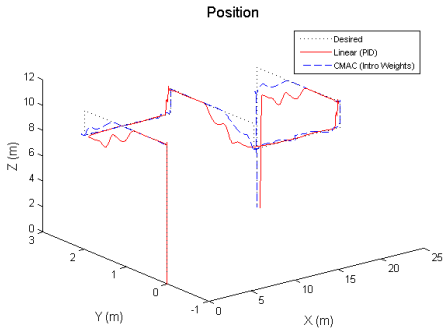


Fig. 5: Inspection Position Comparison

In a more advanced simulation, a position tracking loop is added outside the stabilization loop. This higher level control adds autonomy to the quadrotor by allowing the vehicle to track desired locations in three dimensional space. Position response of a PID (red solid line) and the introspective voting CMAC (blue dashed line) to the tracking control along X, Y, and Z axes are shown in Figure 5. In the Z direction, the PID and CMAC

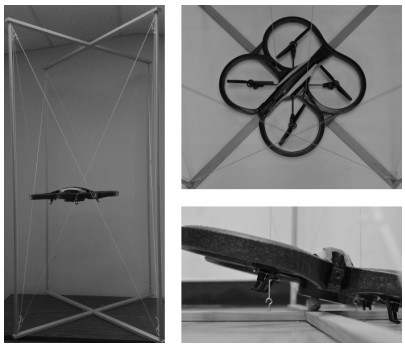


Fig. 6: Quad-Rotor Testing Enclosure

performed similarly, with errors of 0.60m and 0.63m respectively. In the Y plane, the PID provided an error of 0.47m, which was 30% greater than that of the CMAC. Finally, in the X direction, the PID exhibited the largest RMS error, 1.21m, which was 61% greater than that of the introspective scheme. In addition to a significantly lower RMS error, the increased speed with which the CMAC is able to adapt to changes in desired position is clearly evident.

In this simulation, the PID has been tuned for stability, which has decreased the RMS error primarily in steady-state regions. Unfortunately, tuning for performance garnered less predictable results. In this tracking simulation, a CMAC with introspective voting weight updates is able to adapt to challenging input vectors with speed and accuracy superior to that of a PID scheme.

B. Experimentation

Control experiments are performed using an AR.Drone 2.0 quadrotor with direct motor control over UDP via a custom MATLAB user interface. An enclosure was designed to enable tuning to be performed safely, while introducing as little system interference as possible, as shown in Figure 6. Vehicle information is retrieved from the AR.Drone 2.0 fifteen times per second, and is visually presented in red for the PID control, and blue for the introspective voting CMAC.

In the first experimental trial, the stabilizing effect of the PID and CMAC control schemes is analyzed. To understand initial performance while minimizing the negative effects of an untuned control scheme, rotor speeds are limited to restrain the quadrotor. The vehicle is placed upon a spherical surface supported by wedges, resulting in initial roll and pitch angles of approximately ten degrees. The vehicle is then commanded to stabilize itself to roll and pitch angles of zero degrees while height and yaw control inputs are fixed.

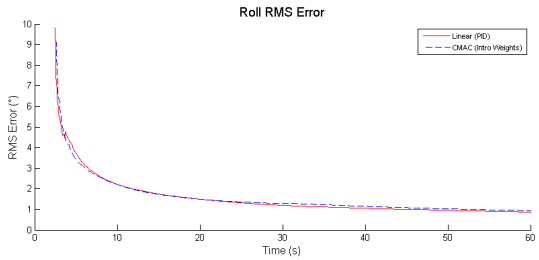


Fig. 7: Stabilization Roll Angle Error Comparison

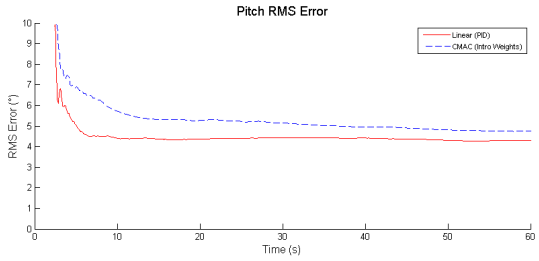


Fig. 8: Stabilization Pitch Angle Error Comparison

In experimentation, both the PID and introspective voting CMAC perform with greater error than in simulation. Part of this decreased performance is the result of drift errors in roll and pitch measurements which were observed to be as high as one degree. Yet, stable performance is achieved by both control schemes, within five seconds for the PID and ten seconds for the CMAC.

For the PID control, the initial roll angle measures 9.8 degrees, for the CMAC it is 9.3 degrees. The PID scheme exhibits an RMS error of 0.9 degrees over the entire simulation, while the CMAC provides a deviation of 0.8 degrees (Figure 7). The results in the pitch plane are similar (Figure 8). The initial pitch angle is measured at 10.0 degrees for the PID control and 10.1 degrees for the CMAC. The CMAC provides an overall error of 4.7 degrees and the PID varies by an RMS value of 4.8 degrees.

The PID scheme requests significant rotor speed fluctuations, which leads to more erratic initial vehicle movement, as well as increased wear on the vehicle over time. The CMAC exhibits a much smaller maximum range of rotor speeds, 1,750 RPM, whereas the PID scheme requests a maximum range of 10,569 RPM; over six times greater (Figure 9). Ultimately, the success of this test for both controls provides a basis upon which more advanced applications can be explored.

In the next experimental application, the quadrotor is

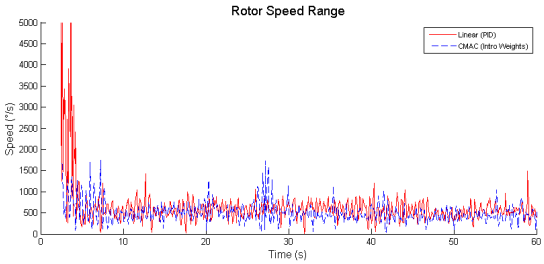


Fig. 9: Stabilization Rotor Speed Range Comparison

manually accelerated to a hovering position before the control system takes over to stabilize the roll and pitch angles of the vehicle. Again, the height control is fixed and the yaw control disabled. Only the ability of the control to stabilize roll and pitch angles is investigated.

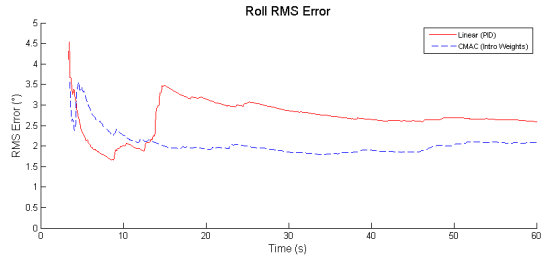


Fig. 10: Hover Roll Angle Error Comparison

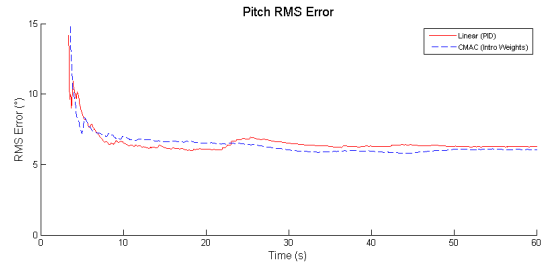


Fig. 11: Hover Pitch Angle Error Comparison

During the hover task the baseline PID control exhibits a combined roll RMS error of 2.6 degrees (Figure 10). The CMAC achieves a 19% better result, with an error of only 2.1 degrees. When stabilizing pitch, the CMAC and PID perform similarly; with RMS errors of 6.1 degrees 6.2 degrees respectively (Figure 11).

The PID scheme immediately exerts a significant control force, while the introspective voting CMAC learns the dynamics of the system and exert control effort more

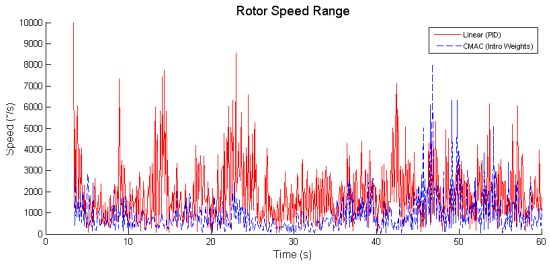


Fig. 12: Hover Rotor Speed Range Comparison

efficiently. The PID scheme requests a wider maximum range of rotor speeds, 10,027 RPM, as compared to the CMAC which required a maximum variance of just 8,034 RPM (Figure 12).

TABLE I: PID Control Gains

	k_p	k_i	k_d
Tracking Simulation	3	4	2
Hover Experiment	0.6	0.7	0.003

Again, the PID stresses the vehicle more during this stabilization experiment, while providing performance similar to, if not slightly worse than, the introspective voting CMAC. Furthermore, while PID gains were retuned specifically for each task, any work done to improve the learning variables of the CMAC enhanced performance in all trials (Table I). The comparison of the experimental results obtained is slightly more encouraging than that noted in the early simulation trials, before the CMAC began to show significant performance improvements over the PID in more advanced tasks.

V. Conclusion

The control challenges presented by an under-actuated and under-damped quadrotor vehicle in a practical environment were identified and overcome using a CMAC with an introspective voting weight update. This work confirmed the adaptability and robustness of an existing CMAC control scheme in simulation and experimental application through the creation of simulated and physical testing environments. The ability of the introspective voting scheme to eliminate bursting found in traditional weight update schemes was confirmed in simulation. In experiment, it was found that the CMAC slightly outperforms PID while requiring significantly less control effort.

References

[1] J. Li and Y. Li, "Dynamic analysis and pid control for a quadrotor," in *Mechatronics and Automation (ICMA), 2011 International Conference on*, 2011, pp. 573–578.

[2] L. Jin-song, Y. Lian, and W. Le-tian, "Aaptive control-optimization of a small scale quadrotor helicopter," *Mechanika*, no. 5, pp. 559–566, 2013.

[3] D. Lara, G. Romero, A. Sanchez, R. Lozano, and A. Guerrero, "Robustness margin for attitude control of a four rotor mini-rotorcraft: Case of study," *Mechatronics*, vol. 20, no. 1, pp. 143–152, 2 2010.

[4] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *Proc. of the AIAA Guidance, Navigation, and Control Conference*, 2007, pp. 1–20.

[5] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a quad-rotor robot," in *Proceedings Australasian Conference on Robotics and Automation 2006*. Australian Robotics and Automation Association Inc., 2006.

[6] I. Kroo and P. Kunz, "Development of the mesicopter: A miniature autonomous rotorcraft." in *American Helicopter Society (AHS) Vertical Lift Aircraft Design Conference*, 2000.

[7] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 3277–3282.

[8] B. Castillo-Toledo, S. Di Gennaro, and F. Jurado, "Trajectory tracking for a quadrotor via fuzzy regulation," in *World Automation Congress (WAC), 2012*, 2012, pp. 1–6.

[9] H. Bouadi, M. Bouchoucha, and M. Tadjine, "Sliding mode control based on backstepping approach for an uav type-quadrotor," *World Academy of Science, Engineering and Technology*, vol. 26, pp. 22–27, 2007.

[10] H. Voos, "Nonlinear control of a quadrotor micro-uav using feedback-linearization," in *Mechatronics, 2009. ICM 2009. IEEE International Conference on*, 2009, pp. 1–6.

[11] E. Stingu and F. L. Lewis, "An approximate dynamic programming based controller for an underactuated 6dof quadrotor," in *Adaptive Dynamic Programming And Reinforcement Learning (ADPRL), 2011 IEEE Symposium on*, 2011, pp. 271–278.

[12] C. Coza, C. Nicol, C. Macnab, and A. Ramirez-Serrano, "Adaptive fuzzy control for a quadrotor helicopter robust to wind buffeting," *Journal of Intelligent and Fuzzy Systems*, vol. 22, no. 5, pp. 267–283, 2011.

[13] K. Masaud and C. Macnab, *An Introspective Learning Algorithm that Achieves Robust Adaptive Control of a Quadrotor Helicopter*. American Institute of Aeronautics and Astronautics, 06/19; 2014/05 2012. [Online]. Available: http://dx.doi.org/10.2514/6.2012-2518

[14] L. R. G. Carrillo, A. E. D. Lpez, R. Lozano, and C. Pgard, *Quad rotorcraft control: vision-based hovering and navigation*. Springer Science and Business Media, 2012.

[15] J. S. Albus, "A model for memory in the brain." *National Aeronautics and Space Administration*, vol. 6456, 1971.

[16] C. M. Nicol, C. and A. Ramirez-Serrano, "Robust adaptive control of a quadrotor helicopter," *Mechatronics*, vol. 21, no. 6, pp. 927–938, 2011.

[17] B. D. O. Anderson, "Adaptive systems, lack of persistency of excitation and bursting phenomena," *Automatica*, vol. 21, no. 3, pp. 247–258, 5 1985.

[18] K. Masaud and C. J. B. Macnab, "Preventing bursting in adaptive control using an introspective neural network algorithm," *Neurocomputing*, vol. 136, no. 0, pp. 300–314, 7/20 2014.