Aircraft Maneuvering via Genetics-Based Adaptive Agent

H. Brown Cribbs, III*
Department of Aerospace Engineering and Mechanics
The University of Alabama
Box 870280
Tuscaloosa, Alabama 35487-0280
(205) 348-4661

Abstract

Aircraft flight challenges both human pilots and artificial (auto-) pilots. Aircraft maneuvering supplies a rich and complex task for machine learning. A rudimentary *flight test maneuvering agent* (FTMA) possesses the potential to be moved from aircraft to aircraft in a robust but general manner. This study presents an FTMA, whose adaptive qualities are tested in two, NASA developed, hi-fidelity flight simulations—the X-31A Enhanced Fighter Maneuverability Demonstrator and the F-106 Delta Dart. These two aircraft possess dramatically different dynamics and control laws. The agent approach contributes generality of interface and adaptation to this complex problem.

KEYWORDS: adaptive behavior, aircraft maneuvering, artificial neural networks, genetic algorithms, hybrid intelligent models, intelligent agents, reinforcement learning

1 Introduction

Aircraft maneuvers help aerospace engineers rate aircraft performance and identify potential hazards. Maneuver autopilots have been designed to improve the quality and repeatability of dynamic maneuvers, but these autopilots augment the aircraft's control laws. Hence, a general maneuver autopilot, that may be moved from aircraft to aircraft is not possible due to the tight coupling of the autopilot to the aircraft's dynamics.

Machine learning may fill the role of maneuver autopilot in initial simulation studies. This offers flexibility, in that the agent may then be moved to another aircraft type and retrained. The agent presented here learns a maneuver policy for the rudimentary task of starting from one flight condition (a set altitude and speed) and transitioning to another flight condition. While a relatively mundane task, climbing to a preset altitude and accelerating can be counterproductive tasks. This simple test proves quite challenging and shows the complexity of maneuvering in the flight environment.

Previous aircraft studies using genetic algorithms (GAs) employed simplified models of the aircraft's dynamics (Smith & Dike, 1995; Stroud, 1998). These studies proved beneficial in evaluating combat strategies, but to serve flight test, fully flight verified models need consideration. Ryan (1995) evolved stick trajectories off-line to seek out the ways the X-31A might depart from controlled flight. This work used the hi-fidelity X-31A simulation developed by NASA's Hugh L. Dryden Flight Research Center at Edwards Air Force Base, California. Simulations, like the X-31A, model the aircraft's aerodynamics, control laws, engines, equations of motion, etc. (Norlin, 1995). Using Dryden's hi-fidelity flight simulations, an agent-based approach to maneuvering is studied. This environment, composed of six degrees-of-freedom (three translational, three rotational), allows motion in three dimensions with multiple hazards, e.g., crashing or departing from controlled flight.

Building an agent to work in such as hostile and highly-dimensioned environment presents many challenges. The adopted design doctrine chooses to hybridize reinforcement learning (RL), artificial neural networks (ANNs), and genetic algorithms (GAs).

* brown@galab3.mh.ua.edu

2 An Artificial Chuck Yeager: The FTMA

Intelligent behavior may be described as a two-part process of predicting and acting (Fogel et al., 1966). The agent predicts long-term cost with a genetics-based artificial neural network (Smith & Cribbs, 1996b). The prediction of long-term cost reflects a desire to minimize the sum of future (discounted) costs incurred by the agent. Bellman's equation,

$$V(\mathbf{x}_t) = r_t + \gamma \cdot V(\mathbf{x}_{t+1}),$$

expresses the sum of discounted future costs given the current state. This sum of future cost is commonly referred to as *value.* This study uses *Q-values,* i.e., an extension of value relating long-term expected cost as a function of a given state and a specific action (Watkins & Dayan, 1992),

$$Q(\mathbf{x}_t, \mathbf{a}_t) = r(\mathbf{x}_t, \mathbf{a}_t) + \gamma \max_{\forall a \in \mathcal{A}} Q(\mathbf{x}_{t+1}, \mathbf{a}_{t+1}).$$

Here, the feedback signal from the environment, $r(\mathbf{x}_t, \mathbf{a}_t)$, is maximized. This follows the assumption that feedback is reward-based. For cost, one simply changes the maximum operator to a minimum operator, or simply expresses cost as *negative reward.* The latter representation is adopted for this study. For notational convenience, the quantities $r(\mathbf{x}_t, \mathbf{a}_t)$, $Q(\mathbf{x}_t, \mathbf{a}_t)$ and $Q(\mathbf{x}_{t+1}, \mathbf{a}_{t+1})$ shall be referred to as r_t, Q_t, and Q_{t+1} respectively.

In complex environments, the possibility of previously unseen states is a distinct possibility. To accommodate unseen states, the agent needs a mechanism to generalize from past experience. This study uses an ANN to perform its predictions due to the ANN's generalization properties. The ANN approximates Q-values given the current state. The outputs of the ANN are Q-values—one for each action. The aircraft's state variables are:

- LONGITUDINAL: longitudinal stick deflection (δ_{LONG}), Angle-of-Attack (AOA), attitude (Θ), pitching rate (q), forward velocity (U), throttle position (δ_{THR}), Mach number (M), altitude (H), climb rate ($\dot{H}$), and
- LATERAL-DIRECTIONAL: Lateral stick deflection (δ_{LAT}), roll rate (p), bank angle (Φ), Angle-of-Sideslip (AOS), heading angle (Ψ), yaw rate (r), and rudder pedal deflection (δ_{RP}).

This information describes the motion and cockpit state of the aircraft. Available in virtually any aircraft, these variables with an additional constant input of -1.0 compose the input vector to the ANN. To keep the inputs as well conditioned as possible all the variables listed above are scaled to the range $[-1, +1]$.

Table 1: Simple action tableau: Longitudinal control only.

Action	$\Delta\delta_{LAT}$	$\Delta\delta_{LONG}$	$\Delta\delta_{RP}$	$\Delta\delta_{THR}$
0	0	0.5%	0	0
1	0	−0.5%	0	0
2	0	0	0	0.5%
3	0	0	0	−0.5%
4[a]	0	0	0	0

[a] This serves as the "no operation" action.

The ANN refines its predictions through backpropagation of the Bellman residual,

$$\Delta \mathbf{w}_t = \eta \cdot \underbrace{\left(r_t + \gamma \max_{\forall a \in \mathcal{A}} Q_{t+1} - Q_t \right)}_{\text{Bellman Residual}} \cdot \nabla_w Q_t,$$

where r_t denotes the immediate reward at step t, Q_{t+1} denotes the Q-value for the *next step,* $t+1$, and Q_t denotes the Q-value for the action executed at step t (Rumelhart et al., 1986; Watkins & Dayan, 1992).

$Q(\lambda)$ represents a hybrid of the method presented above. Q(λ) modifies the Q-values to consider temporal differences greater than one step ahead, i.e, not only $\gamma \max Q_{t+1}$ but $\gamma \max Q_{t+1} ... \gamma^n \max Q_{t+n}$ (Peng & Williams, 1996),

$$\begin{aligned} \Delta \mathbf{w}_t = \quad \eta \quad & \Bigg[\left(r_t + \gamma \max_{\forall a \in \mathcal{A}} Q_{t+1} - Q_t \right) \cdot \nabla_w Q_t \\ & + \left(r_t + \gamma \max_{\forall a \in \mathcal{A}} Q_{t+1} - \max_{\forall a \in \mathcal{A}} Q_t \right) \\ & \cdot \sum_{k=0}^{t-1} (\gamma\lambda)^{t-k} \nabla_w Q_k \Bigg]. \end{aligned}$$

The second term in this equation forms a weight update in the gradient direction of the maximum Q-value. This is sometimes referred to as an *on-policy* update, since the desired goal of these methods are to exploit the maximum Q-values for each time step. Q(λ) has been shown to accelerate learning in many instances (Peng & Williams, 1996).

The agent interacts with the aircraft by manipulating the cockpit controls, i.e., stick and throttle. The FTMA modifies the aircraft controls at 0.1 second intervals (10Hz). The agent's actions increment or decrement the position of each control by a set percentage of the full range of the control. These per-

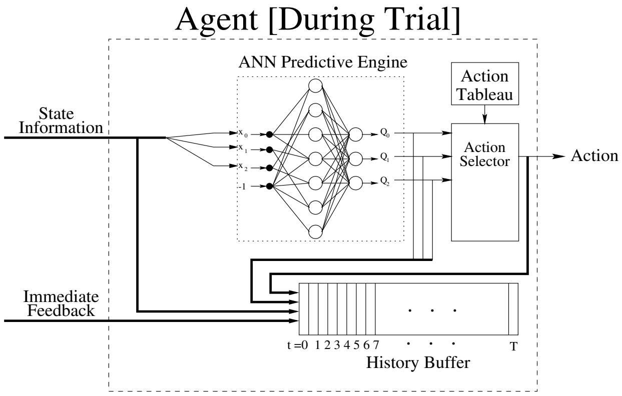

Figure 1: Agent architecture. Note, for simplicity, the agent depicted uses only three state variables and three actions (Q-values). The agent described in the text possesses 17 state variables and five actions (Q-values).

centages compose the agent's *action tableau.* Table 1 shows the action tableau used in this study.

A simple action selection procedure, ϵ-greedy, chooses actions from the agent's action tableau based on the Q-values (Sutton & Barto, 1998). This method selects the action possessing the highest Q-value uniformly, with probability $1-\epsilon$. The probability, ϵ, controls the evaluation of alternative actions (exploration) instead of the highest Q-value.

Figure 1 depicts a the agent architecture. The order of execution for the agent is:

1. scaled state information and feedback from the last action are received from the aircraft,
2. ANN evaluates the state (producing Q-values),
3. all state, feedback, and Q-value information is stored in the history buffer for training at end of trial,
4. action is selected via ϵ-greedy selection and Q-values, and
5. the action sent to the aircraft.

This sequence is executed for each step in a trial. At the end of the trial the sequence is replayed in reverse order to facilitate training via Q(λ).

2.1 Why the GA?

Whitehead and Choate (1996) point out that coverage of an entire, high-dimension space, by regularly spaced radial bases succumbs to the *curse-of-dimensionality* (Albus, 1975; Bellman, 1961). To combat the "curse," Whitehead and Choate suggest GA manipulated bases. GA-based classification also occurs in studies involving RL and boolean classification (Smith & Cribbs, 1996c; Wilson, 1990). Regular spaced grids over multiple dimensions have been suggested in many applications (Sutton & Barto, 1998), but again regular coverage of large spaces may prove impractical.

GA-based input selection allows the topology of the ANN to adapt based on some form of correlation to the networks performance. Hence, learning and evolution are tied together allowing learning to fine tune weights while the GA selects connections.

The GA selects unique input subsets, which dynamically shapes the ANN to aid its predictions (Smith & Cribbs, 1996b). The FTMA differs from Smith and Cribbs' (1996a; 1996b) architecture, where the input-to-hidden layer weights are fixed at $-1,\ 0,$ or $+1$, in favor of *two-layers* of tunable weights with the GA selecting input variables. A modified version of Smith and Cribbs (1996b) fitness function is used here. Fitness is determined by,

$$f_i = D_i \cdot \left[\sigma_i(\mathbf{x}) + (1-\xi) + \sigma_i(\mathbf{x}) \cdot (1-\xi)\right].$$

This measure correlates each hidden layer node's firing state to the ANN's overall accuracy. Each hidden layer node is an individual in the GA population. Thus σ_j denotes the j^{th} node's firing state where,

$$\sigma_j = \begin{cases} 0; & f_j(\mathbf{x}) \leq 0 \\ 1; & f_j(\mathbf{x}) > 0 \end{cases}.$$

The function, $f_j(\cdot)$, denotes the output value of hidden-node j. The ANN's prediction accuracy, $(1-\xi)$ is derived from,

$$\xi = \left| \frac{r_t + \gamma \max_{\forall a \in \mathcal{A}} Q_{t+1} - Q_t}{r_t + \gamma \max_{\forall a \in \mathcal{A}} Q_{t+1}} \right|.$$

D_i represents the node's *decisiveness rating* which corresponds to the variance of a given node's output weights normalized by the largest output weight variance in the hidden layer. Through D_i individual features that relate to specific action, i.e., connections that bias few outputs, are favored due to the desire for a set of distinct feature detectors to base predictions. This caveat could be handled by either evolving the output connectivity along with the input connectivity or by working out an arbitrary connectivity scheme. The D_i method avoids complex issues due to the possibility of recurrent connections in the arbitrary connectivity case, and avoids long genomes which could drastically affect the GA's search efficiency.

Table 2: Agent architecture summary.

Component	*Description*
RL Method	Q(λ) as defined in Rummery & Niranjan (1994) and backward replay (Lin, 1991). • Discount Parameter, $\gamma = 0.99$ • Trace Decay Parameter, $\lambda = 0.7$
Predictive Engine	Genetics-based ANN: • Multi-layer Feedforward ANN composed of *bipolar sigmoidal* activation neurons in the hidden layer and *linear sum* output neurons. – Each neuron possesses unique input connectivity similar to *macro-classifiers* (Wilson, 1994). – Learning Rate, η (varies with Aircraft) – Momentum Parameter, $\alpha_m = 0.8$ • Steady-State GA using 10% of population: – Tournament Selection (tourney size 2), – two-point crossover ($p_c = 0.9$), – point mutation ($p_m = 0.01$), – deterministic replacement of worst individual(s).
Action Selection	ϵ-greedy action selection ($\epsilon = 0.1$)

3 The Problem: Capture Altitude and Mach Number

The agent's assignment is to learn to maneuver the aircraft from level flight at 8000 feet and a 0.6 Mach number to 10,000 feet and a 0.65 Mach number. This sort of maneuver takes place numerous times during flight test and is generally used for staging more complex maneuvers.

To accomplish this task the agent receives feedback from the environment in the form of immediate cost. The cost function uses a distance to target scheme for both altitude and Mach number. The functional,

$$r_t = G_H \cdot \left| \frac{H - H^*}{H_{tol}} \right| + G_M \cdot \left| \frac{M - M^*}{M_{tol}} \right|,$$

depicts the immediate cost. The gain terms, G_H and G_M, bias the distance to weight the two goals. Additionally, the tolerances, H_{tol} and M_{tol}, provide a sensitivity bias. Table 3 specifies the values used here.

Table 3: Cost function parameters. Note that the gain factor sign reflects cost (negative reward).

Parameter	*Value*
Desired Altitude, H^*	10,000 feet
Altitude Tolerance, H_{tol}	100 feet
Altitude Gain, G_H	–8
Desired Mach Number, M^*	0.65
Mach Number Tolerance, M_{tol}	0.001
Mach Number Gain, G_M	–16

4 Results and Discussion

Training the FTMA involves repeatedly subjecting the agent to the aircraft. In each trial the FTMA maneuvers the craft until either the end of the trial (120 seconds) or until the aircraft crashes or departs. Training parameter sensitivity definitely factors into the move from aircraft to aircraft. For the purposes of this study, the Q(λ) parameters were fixed. Table 2 summarizes the values used. The only value changed was the ANN *learning rate*, which was $\eta = 0.009$ for the

X-31A, and $\eta = 0.008$ for the F-106.

4.1 X-31A Behavior

Training the X-31 FTMA for 1495 simulated flights produced an agent with prediction accuracy of approximately 70%. Figure 2 shows the learning curve in *absolute, mean relative error* for each trial. Figure 2 also shows the number of steps the FTMA takes in each trial. During Evaluation the exploration probability, ϵ, is set to zero, i.e., pure exploitation. The X-31 FTMA was evaluated in trials 1495 through trial 1500.

Figure 3 shows an interesting behavior observed near the end of training. The agent noses the aircraft towards the ground to increase speed. The agent then uses the added speed to "zoom" skyward. With additional thrust (more throttle, see Figure 4) the agent begins an oscillation that takes it through the altitude goal, but overshoots the altitude goal. Interestingly the Mach number goal occurs several times in the maneuver.

Evaluation revealed an FTMA that uses a "pure stick" control policy. Figure 5 shows the control trajectories that produced in an evaluation trial of the X-31A. Figure 6 shows the X-31 FTMA passing through the altitude goal 4 times and an oscillating Mach number.

4.2 F-106 Behavior

Training the F-106 FTMA produced a 75% accurate agent. Unlike the X-31 FTMA, all evaluation trials flew the allotted 120 seconds. Figure 7 shows the training history of the F-106 experiment. As in the X-31 experiment, the F-106 FTMA was trained in 1495 simulated flights. Exploratory steps (during training) occurred with probability $\epsilon = 0.1$.

Trial 1493 (Figure 9) characterizes the F-106 FTMA during the last stages of training. The agent achieves the altitude goal near step 1200 (the end of the trial). The Mach number behavior is oscillatory, but upon examination of the graph's scale the oscillations are bounded between 0.54 Mach and 0.6 Mach. In light of non-zero exploration (this is a training trial), the behavior is encouraging due to small oscillation in Mach number and the trend toward the altitude goal.

Evaluation shows the same trend toward the altitude goal and stabilized Mach number at approximately 0.56 Mach. The similarity in behavior both during training (Figure 9) and in evaluation (Figure 10) show the FTMA to use predominantly longitudinal stick combined with minor throttle adjustments. Figure 10

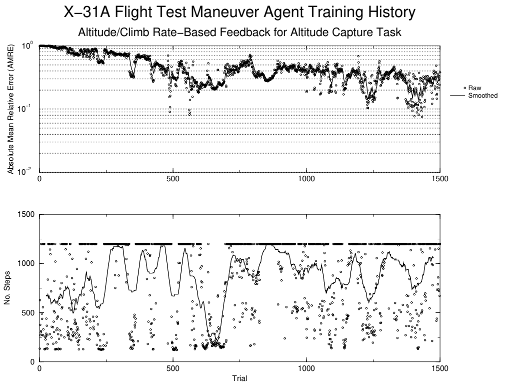

Figure 2: X-31A FTMA performance metrics. The upper graph relates the absolute mean relative error during training. The lower graph shows the number of steps (decisions) the agent made in each trial. The thick black line in the lower graph is the windowed average of the last 25 trials.

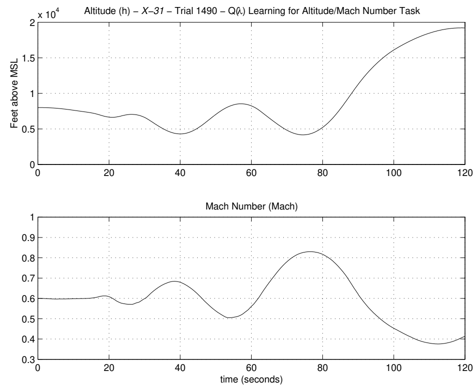

Figure 3: X-31A altitude and Mach number behavior near end of training.

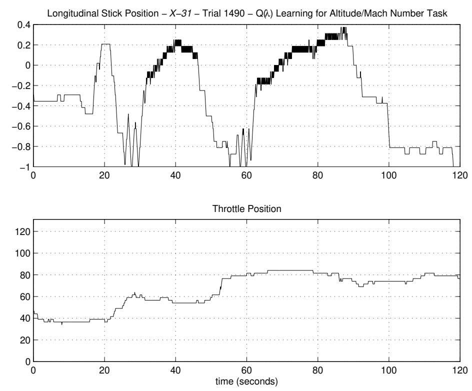

Figure 4: X-31A longitudinal stick and throttle lever position near the end of training.

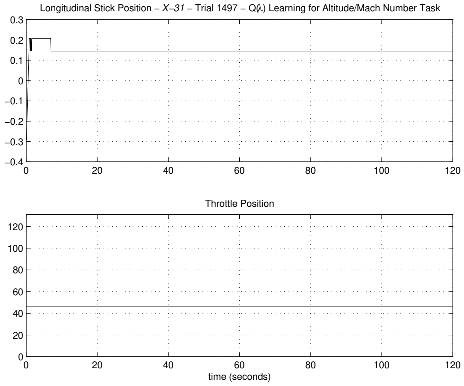

Figure 5: X-31A longitudinal stick and throttle lever position during an evaluation trial.

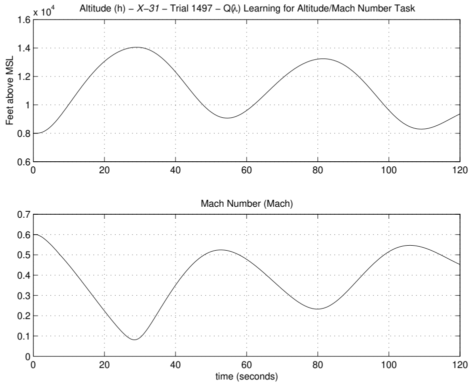

Figure 6: X-31A altitude and Mach number behavior during the evaluation.

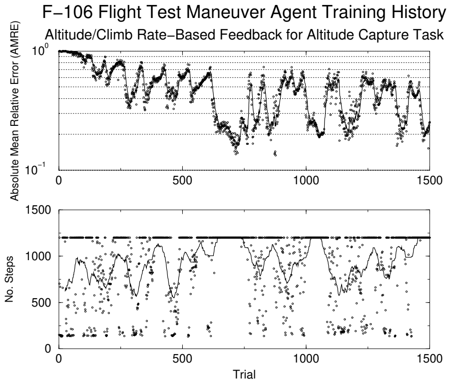

Figure 7: F-106 predictive performance. The upper graph relates the absolute mean relative error during training by trial. The lower graph shows the number of steps (decisions) the agent made in each trial. The thick black line in the lower graph is the windowed average of the last 50 trials.

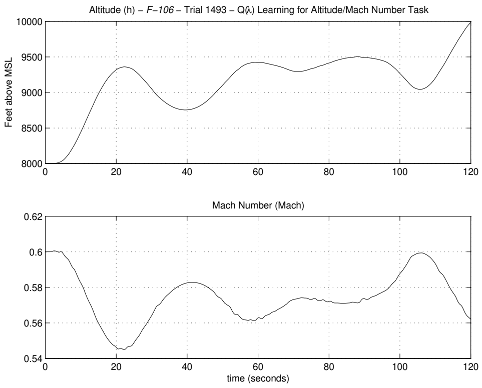

Figure 8: F-106 FTMA altitude and Mach number behavior near the end of training.

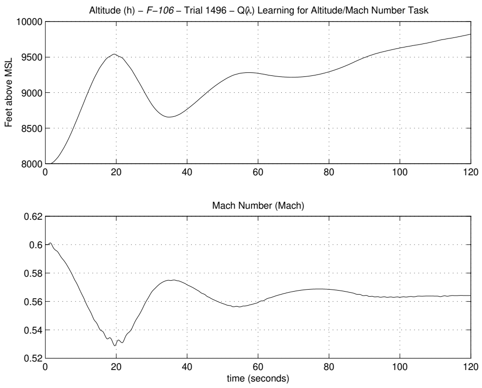

Figure 9: F-106 FTMA altitude and Mach number behavior during an evaluation trial.

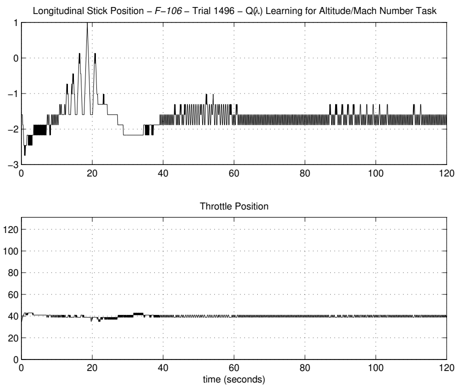

Figure 10: F-106 control trajectories during evaluation.

shows the control trajectories during evaluation (trial 1497). The max frequency oscillation in controls relate back to the fixed increment/decrement action scheme adopted in this study.

5 Conclusions and Future Work

The basic FTMA discussed here provides some adaptive qualities to aircraft maneuvering. While an initial investigation into genetics-based agents for flight test, the potential to move from one aircraft to another in a general way has great potential. The agent described here found several unique departures, mainly dealing the simulations' look-up tables.

The FTMA shows an ability to work with the dynamics of different aircraft. The 32,000 pound, F-106, weighs roughly twice that of the X-31. The additional mass most likely contributes to the more stable behavior of the F-106 FTMA. This stems from the additional mass reducing the overall agility of the aircraft. Also the control laws and control surfaces dramatically differ in each aircraft. The varying policies adopted in each of the simulations show that the FTMA adapts to its aircraft and its assigned goals.

The linear cost functional used embodies a simple distance-to-goal method. The overshoot of the goals prevalent in both simulations suggest that an integrative cost function, or a cost function that considers the momentum characteristics of aircraft, may provide better information to the FTMA.

Because RL can have sparse reward schemes a different reward approach might yield better results. One thought is a point-based scheme where way-points of a maneuver encourage trajectory following behavior.

Comparison of RL methods in the FTMA may assess the benefits of differing methods. Two areas of inspiration involve TD(λ) hybrids and on-policy versus off-policy learning (Rummery & Niranjan, 1994; Sutton & Barto, 1998). These issues are currently under study in the FTMA.

Acknowledgments

This work was supported by the NASA Graduate Student Research Program (GSRP) [grant NGT 4-52403]. I want to thank the outstanding personnel at NASA Dryden Flight Research Center; especially, Joe Barnicki, Bob Clarke, Steve Jensen, John Kelly, Joe Pahle, and Keith Schweikhard for their contributions and patience.

I would also like to thank my advisor, Rob Smith, who helped me find funding for my graduate education. I do not believe I would ever have stuck out the Ph.D. without his advice and friendship.

Finally, I thank my parents who have had to put up with my "professional student" status for far too long.

References

Albus, J. S. (1975). A new approach to manipulator control: The cerebellar model articulation controller. *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 220–227.

Bellman, R. E. (1961). *Adaptive control processes: A guided tour.* Princeton, NJ: Princeton University Press.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial intelligence through simulated evolution.* New York: John Wiley.

Lin, L.-J. (1991). Programming robots using reinforcement learning and teaching. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, Vol. Two, 781–786.

Norlin, K. A. (1995, October). *Flight Simulation Software at NASA Dryden Flight Research Center* (Tecnical Memorandum 104315). Edwards, CA: NASA Hugh L. Dryden Flight Research Center.

Peng, J., & Williams, R. W. (1996). Incremental multi-step Q-Learning. *Machine Learning,* **22**, 283–290.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representation by backpropagating errors. *Nature*, *1986*(323), 533–536.

Rummery, G. A., & Niranjan, M. (1994). *On-line Q-learning using connectionist systems* (Tech. Rep. CUED/F-INFENG/TR 166). Cambridge CB2 1PZ, England: Cambridge University Engineering Department.

Ryan, G. W. (1995, August). *A genetic search technique for identification of aircraft departures* (Contractor Report 4688). Hugh L. Dryden Flight Research Center, Edwards, CA: NASA.

Smith, R. E., & Cribbs, H. B. (1996a). Combined biological paradigms: A neural, genetics-based autonomous systems strategy. In *Conference Proceedings of Biologically Inspired Autonomous Systems: Computation, Cognition, and Control*, Durham, NC.

Smith, R. E., & Cribbs, H. B. (1996b). Cooperative versus competitive system elements in coevolutionary systems. In *FROM ANIMALS TO ANIMATS 4: Proceedings of the 4th International Conference on Simulation of Adaptive Behavior*, Cape Cod, MA.

Smith, R. E., & Cribbs, H. B. (1996c). Parsimonious neural, genetics-based Q-learning for autonomous systems. In *Proceedings of An International Workshop on Learning for Autonomous Robots (ROBOLEARN-96)*, Key West, FL.

Smith, R. E., & Dike, B. A. (1995). Learning novel fighter combat maneuver rules via genetic algorithms. *International Journal of Expert Systems*, *8*(2), 247–276.

Stroud, P. D. (1998). Adaptive simulated pilot. *Journal of Guidance, Control, and Dynamics*, *21*(2), 352–354.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction.* Cambridge, MA: MIT Press.

Watkins, C. J. C. H., & Dayan, P. (1992). Q-Learning. *Machine Learning,* **8**, 279–292.

Whitehead, B. A., & Choate, T. D. (1996). Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction. *IEEE Transactions on Neural Networks*, *7*(4), 869–880.

Wilson, S. W. (1990). Perceptron redux: Emergence of structure. In S. Forrest (Ed.), *Emergent Computation: Proceedings of the Ninth Annual International Conference of the Center for Nonlinear Studies on Self-Organization, Collective, and Cooperative Phenomena in Natural and Artificial Computing Networks*, 249–256, Amsterdam. North-Holland.

Wilson, S. W. (1994). ZCS: A zeroth level classifier system. *Evolutionary Computation*, *2*(1), 1–18.