

Reinforcement Learning in Aerospace (Final Exam)

Steve Barriault

December 13, 2009

Contents

1	Introduction	1
2	The Challenges Addressed by the Reviewed Papers	2
2.1	Description	2
2.2	Common Technical Challenges to All Situations	3
3	How can Reinforcement Learning be used to Solve these Challenges	3
3.1	Genetic Algorithm Reinforcement Learning (GARL)	4
3.2	Associative Search Element and Adaptive Critic Element	4
3.3	Co-Evolutionary Perception	5
3.4	Galerkin Sequential Function Approximator (SFA)	5
3.5	Recapitulation of the Methods Used	5
4	Results	6
5	Analysis of the Results	9
5.1	Common Points that are Mutually Self-Supporting in Most or All Papers	9
5.2	Points that Contradict Each Other	9
6	Conclusion	9

1 Introduction

This paper reviews four recent peer-reviewed articles discussing the application of machine learning techniques in Aerospace. More precisely, they propose different implementations of the reinforcement learning paradigm to address different challenges.

The reason why I chose this topic is first and foremost by pure personal interest. I work in embedded software, especially with companies creating safety (or mission) critical systems, many of them being in Aerospace and Defense industries. Also, because these applications are critical in nature, they offer both an opportunity and a challenge to machine learning algorithms such as reinforcement learning. As we saw throughout the semester, these can offer optimizing solutions to problems. But are they efficient enough, fast enough, and especially reliable enough to become a viable solution in the Aerospace industry, even in a safety-oriented, conservative-leaning environment such as civilian aerospace (which uses one of the strongest software quality standard, DO-178B, for certification purposes)? This survey is a first step toward answering this question.

The papers were selected based on a simple search in the IEEEExplore Digital Database.[1] I used keywords such as "reinforcement learning", "machine learning" and "aerospace" in conjunction with one another to retrieve a list of articles, which I then listed in decreasing chronological order so as to choose the most recent

ones. I also briefly reviewed the abstracts linked to each paper to determine whether these articles were appropriate (some were not, as they were linked to other academic disciplines). I then made a final choice among the papers remained. I selected four of them so as to highlight different challenges in the Aerospace industry, ranging from the "mundane" task of maintaining altitude (and a smooth, safe ride) in a Boeing 747 to the futuristic goal of optimizing the flight of a morphing vehicle.

This paper is divided in several parts. In Section 2, the challenges addressed by each paper will be explained, and their common characteristics highlighted. Then, in Section 3, I will explain the different techniques that were chosen to implement the reinforcement learning algorithm. Section 4 will resume the results and Section 5 will analyze them, highlighting where results are compatible with one another and where they are not. The Conclusion will explain what I learned from this exercise and the questions that are left unanswered.

2 The Challenges Addressed by the Reviewed Papers

2.1 Description

The first paper (by Jiang, Gong, Liu, Xu and Chen) applies reinforcement learning to altitude control for airplanes, in particular the Boeing 747 - one of the largest, highest performance civilian planes in the world.[2] This is no small challenge, as the altitude of the plane is determined by a number of factors, both internal and external to the aircraft. Pressure, wind direction, speed and gusts are examples of environmental impacts on the plane that will, *ceteris paribus*, modify the altitude of the B747. To compensate, the plane may use its ailerons, flaps, elevators and rudder - usually in conjunction with one another.

To make matters worse, obtaining a predictable mathematical model for both the endogenous and exogenous factors is either very difficult or plainly impossible. As explained by the authors, not only *an aircraft is a non-linear and high dimension system* for which *it is difficult to obtain the precise mathematical model*[2], but the environment will often change in a 15-hour flight to Australia, most of the time suddenly and without warning. And since flying protocols instructs eastbound planes to fly at an altitude with an odd multiple of 1000 feet, and westbound planes to fly at an altitude with an even multiple of 1000 feet, the *error of altitude control of an aircraft should be less than a hundred feet* in order to be safe.[2] Not mentioning that a constantly changing altitude may result in passenger discomfort, extending to the frequent use of certain plastic-lined paper bags.

The second paper, written by C-K. Lin, deals with a much faster, more mission-oriented aircraft: missiles.[3] Here, gross performance, not safety, is the primary goal. The author explains that bank-to-turn (BTT) missiles have been widely studied, because they have greater maneuverability and aerodynamic acceleration than traditional skid-to-turn (STT) missiles. A skid-to-turn missile or aircraft does not roll its entire body when turning - it merely use the rudder or a differential in trust to accomplish an horizontal turn.[4] Bank-to-turn simply describes an aircraft that does rotate on its axis when turning, just like any modern plane.

On the other hand, STT can be performed much quicker than BTT.[4] Given the great speeds achieved by a missile (especially if talking about InterContinental Ballistic Missiles or ICBMs), this may have deleterious effects on the mission. Also, BTT introduces *strong coupling between pitch and yaw motions*, which makes it difficult to control at great speeds.[3]

The third paper (by Berenji, Vengerov and Ametha) deals with Unmanned Aerial Vehicles (UAVs) and the need to maximize their efficiency.[5] A UAV must maximize two goals at once. First, it must make sure to maximize the use of its own sensors to track targets of interest. Second, since it is most often deployed as part of a group, it must also coordinate its action with other UAVs to avoid the situation where all the aircraft track the same subset of targets. It must also do all of this while changing its trajectory and sensor orientation as little as possible, since this can be reduce its flight autonomy. As the authors state, using a centralized, policy-based approach is impossible because the number of possible states exponentially grow with the number of UAVs deployed in the battlefield.

Finally, the fourth paper I reviewed (from Valasek, Doebbler, Tandale and Meade) is, to my humble knowledge, all science fiction. It analyzes how to maximize the performance of a morphing unmanned air vehicle.[6] These have the capability to change forms dynamically, by applying a voltage differential to its alloy, just like the hero of *Batman Begins* can turn its cape into a glider by applying a current to it.[7][8]

Obviously, this is no easy feat - as we saw previously, it is very hard to know the exact equation of a plane's aerodynamics, even when it does not change shape. Add to this a constantly changing environment and an ability to morph its configuration and you are left with a very difficult math problem to optimize.

2.2 Common Technical Challenges to All Situations

These four papers are merely scraping the surface of the number of difficult software challenges in Aerospace, yet many common points already become apparent.

- *Complex environment with lots of inputs.* Each of the challenges described above are made devilishly difficult by the presence of several endogenous and exogenous factors that influence what the optimized choice should be. Among endogenous factors, we have the very shape of the plane itself as well as the aileron, flaps and rudder positions, plus the degree of rotation of the plane on its horizontal axis. Exogenous factors are primarily linked to air pressure, humidity, wind direction and speed, gusts, temperature differentials, and the like.
- *Inputs are usually changing constantly.* The only thing that is remaining relatively stable throughout the flight of a plane is its general shape (unless it is a morphing aircraft). I say relatively here because the configuration of the plane does change for takeoff and landing when flaps are deployed. Likewise, the destination of the aircraft or missile should in theory remain constant, perhaps except when talking about UAVs or combat planes. However, the rest of the factors I cited above will vary, sometimes wildly and without warning. The aircraft will encounter unexpected air pockets or wind gusts that will change both its trajectory and altitude if left uncorrected, and this happens several times during any flight.
- *The state table is sparse.* Because there are a number of inputs and that those are changing constantly, it is impossible or at the very least suboptimal to have a traditional policy mechanism that dictates what to do in every circumstance. Or as Berenji *and al.* explain, *any centralized approach that keeps track of all (aircraft's) states will be defeated by the "curse of dimensionality"*. [5] This points out to the need of having a function to get the state of the plane as it experiences different values of input, or some other way to cope.
- *State function is difficult or impossible to get.* Unfortunately, that "magic" function is not a silver bullet either. Even for a non-morphing plane operating under controlled external conditions, the function describing the aerodynamics of the aircraft is difficult to get and in any case is very complex. In real-life situation, that function will always be an approximation, thus introducing the possibility of error, which must be managed on an ongoing basis.
- *Continuous set of possible actions.* To make matters worse, actions are not a discrete set of function. A regular plane has a number of ways to correct its flight paths which are used in conjunction with one another - rudder direction, pitch, horizontal axis rotation, ailerons and flaps position, and so on.
- *Very stringent performance requirements.* Finally, all of these aircrafts have very stringent performance requirements that must be met, or a major failure may occur. Because of flying regulations and passenger safety, the altitude and trajectory of a plane should be controlled and unexpected changes in either should be minimized. Missile direction should be corrected as soon as possible least it does not reach its intended objective, which itself is likely to create a number of problems (political ones, among others). And for all these systems, there is an imperative that dictates that any necessary corrections must happen in a short time frame.

3 How can Reinforcement Learning be used to Solve these Challenges

All four papers use reinforcement learning as the main strategy to address the unique challenges posed by their slightly different environment and goals. All of them adopted some version of the Q algorithm to do so. The reasons provided by the authors to justify these choice recoup many of the challenge mentioned in the previous section.

- *Reinforcement learning can deal with complex environment with varying inputs.* The reinforcement learning is very good at online learning. Instead of attempting to list exhaustive policies to deal with a very large continuum of possible states and actions, a reinforcement algorithm can be devised to learn about the environment and device an efficient response to different sets of states. Once learning is completed, the system will be likely to provide at least a logical answer to the situations that will be encountered in the field, so as long as the situations are not dramatically different from training, without the need for switching algorithms.[3]
- *Reinforcement learning can deal with uncertainty about underlying physical principles.* As mentioned before, one of the problems in Aerospace is that the precise mathematical formulas determining the motion of an aircraft is difficult to determine. Reinforcement learning does not rely on precise mathematical models of environments - it can deal with functions that approximate the physical reality and adapt through a mechanism of action-reaction to reach optimized outcomes.

However, most papers also identified possible weaknesses of the reinforcement learning approach, mainly the fact that the method may take a long time to explore the solution space in the case of sparse state tables and continuous actions. Authors needed to bring forward specific implementations that would enable reinforcement learning to explore sufficiently the solution space while also complying with the stringent performance requirements stated in the previous section. They also needed to do this while taking into account that the dynamics of the aircraft and its environment could not be modeled with precision.

Therefore, all four papers had a slightly different way to implement reinforcement learning. These implementations were meant to address similar (but not identical) challenges and goals. They are listed below.

3.1 Genetic Algorithm Reinforcement Learning (GARL)

The first paper (altitude control) tries to accelerate reinforcement learning's inefficiencies when learning large state spaces and continuous action spaces by introducing a genetic algorithm to explore the solution space in multiple directions simultaneously.[2]

The problem that genetic algorithm reinforcement learning (or GARL) is meant to solve is to limit the Q-value table growth to something manageable and explore the solution space in a time frame that is practical. In GARL, the selection of chromosomes is determined by the feedback provided by the reinforcement learning algorithm. First, a population pool is created randomly. A first generation is created through crossovers and mutations. The fitness value of each member is then evaluated through the reinforcement mechanism. Two chromosomes are selected in the family with the highest fitness values, a new generation is created and the cycle repeats.

One important advantage of this approach is that the system does not rely on a perfect mathematical representation of its working environment. Also, it can explore a sparse table within a limited amount of time. Finally, it does not need to approximate the underlying physical function of the aircraft.

3.2 Associative Search Element and Adaptive Critic Element

The second paper (missile control) adopts a different approach to accelerate the exploration of a still sparse state space. Instead of using one neural network to control trajectory, it uses two, one for an associative search element (ASE) and another for an adaptive critic element (ACE).[3]

To justify his choice, the author points out the autopilots based on neural networks can effectively control the plane in many flight conditions using the same algorithm, which is not the case with traditional, gain-scheduling systems. However, the learning process may still be long and hard, which is why he introduces the ASE and ACE facilities. The ACE basically generates a reinforcement learning signal that is transmitted to the ASE to condition the choices made during the exploration of solutions. According to Venayagamoorthy and Padhi, this method also removes the need for an exact knowledge of the nonlinear system dynamics (such as the underlying physical equations governing the trajectory of a missile).[9] Lin also mentions that such a system can learn from dynamic environments with less information than supervised learning.[3]

3.3 Co-Evolutionary Perception

The third paper (about UAVs) did not use two neural networks but a two part reinforcement policy to accomplish its two main objectives.[5] Indeed, for all the other papers, there were one single overarching goal that conditioned all the choices that were made by the authors. The first paper dealt with altitude control, the second one trajectory, and the fourth plane configuration optimized for efficiency. In the first two cases, the objective was unique and pre-determined by the operator, while in the fourth case, the optimization was largely driven by the environment (but the objective was still unique).

The third paper is the only one that deals with two competing objectives. On one hand, each UAV has an individual mandate to optimize resource utilization (meaning, trajectory and sensor rotation) to track the highest priority target as possible. On the other hand, each UAV also has a group mandate to avoid the focus of all attention on a handful of targets to the detriment of a greater coverage of all targets present.

To do this, the reinforcement function contained two parts. The first one rewards the algorithm for the selection of the highest value target, but taking into account the distance to cover in order to position the UAV so as to track this target. In other words, the UAV would first evaluate the distance to all the targets and the value to the mission of different targets. Selection is made on the basis of target value but modulated for the distance separating the target from the UAV. The second part of the reinforcement function would take into account the position of other UAVs to each targets, so as to reward UAVs for tracking targets that do not have many UAVs around them.[5]

3.4 Galerkin Sequential Function Approximator (SFA)

The fourth paper (about morphing aircraft) focused on the approximation of the underlying physical dynamics function to improve the reinforcement learning algorithm.[6] Since the Q-learning algorithm can be used for continuous or discrete action sets but that the continuous space of morphing capabilities of the aircraft could not be accurately implemented and learned, the authors approximated the underlying physical function. They used the Galerkin Sequential Function Approximator (SFA) method because it is *an adaptive and matrix-free scheme that is useful for interpolating and approximating sparse multidimensional scattered data*.[6]

The Galerkin method enables engineers and scientists to approximate a complex, continuous physical function into a discrete problem.[10][11] Valasek *and al.* used that method to build a relatively simple approximator function that does not explicitly depend on the dimensionality of the approximation, which sidesteps the "curse of dimensionality" that may derail other approximations.[6]

The paper also discussed how the intermediary states are handled when the vehicle is morphing. A Structured Adaptive Model Inversion (SAMI) module is then responsible for maintaining trajectory.

3.5 Recapitulation of the Methods Used

Table 1: Methods Used to Optimize the Performance of Reinforcement Learning

Paper Topic	Method Used	Main Objective
Altitude control in Boeing 747	Genetic Algorithm Reinforcement Learning (GARL)	Explore the solution space faster
Missile control under bank-to-turn (BTT)	Associative Search Element and Adaptive Critic Element	Explore the solution space faster
Mission optimization for group of UAVs	Co-Evolutionary Perception	Modify the reinforcement function to maximize two goals at once
Optimization of morphing aircraft configuration	Galerkin Sequential Function Approximator (SFA)	Create an efficient approximation of the underlying physical constraints

A brief overview of the different methods used can be found in Table 1. All of the methods used revolve around a set of similar goals, all linked to have the reinforcement learning algorithm attain a sufficient level

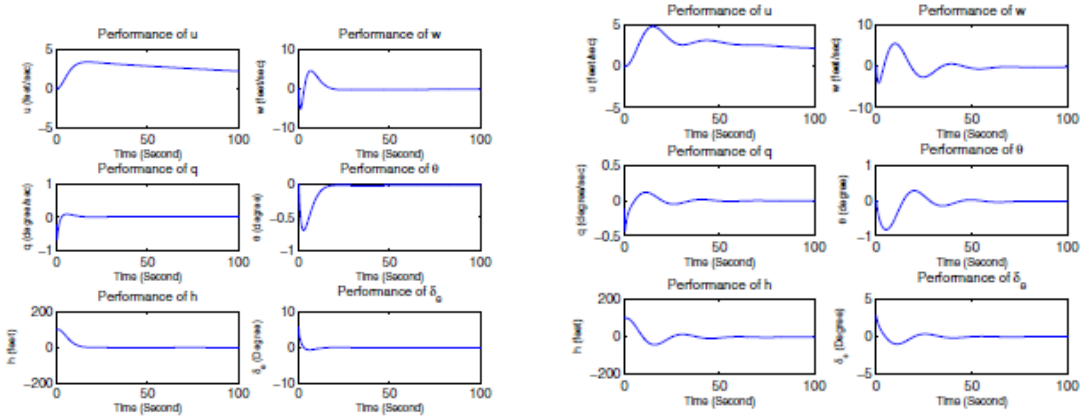


Figure 1: Performance of GARL (right) versus classical state-feedback method (left)[2]

of efficiency in an environment with sparse tables, continuous underlying physical principles and stringent precision/time constraints. However, each paper puts the emphasis on one subgoal in particular, presumably the one most likely to derail the reinforcement learning algorithm in the specific environment analyzed. Thus, meeting timing constraints is the clear objective of the first two papers, while the third paper concentrates on mission delivery and the fourth on the intricacies of controlling a plane that can change shape.

4 Results

Since the goals and methods used were different, the way chosen by different authors to validate their ideas also varied. For instance, in the case of the first paper (altitude control based on GARL), the author ran a simulation with, on one side, a classical altitude control system using state-feedback method, and on the other the same simulation using the GARL method (Figure 1).[2]

There are two comments to be made here. First, it seems evident to me that the classical system actually performed better. The goal was to have these graphs settle to zero in the least amount of time, and the traditional method did that faster. However, for some of the factors considered, the amplitude of movement around zero was less for the GARL-based system. Also, the author seems to indicate that not only GARL was meeting the requirements of the system, but that the performance of the latest system was better in the end because a same algorithm could control policies for various tasks, which cuts on the complexity of enumerating these various tasks in a traditional system.

The fourth paper on morphing aircrafts also adopted a similar validation technique, only this time the authors compared actual data recorded in the simulation (in red) with actions (in blue) (Figure 2).[6] These graphs represent the performance time histories recorded, and those are remarkably similar. The authors also report that compared to using KNN to approximate the physical function, Galerkin SFA reduced RMS error between 10% and 20% depending on the dimension being considered. The graphs recorded for both the KNN and Galerkin SFA simulations clearly show a noticeable improvement in the latter case (Figure 3).

The second and third papers do not really paint such a clear before/after picture. Lin’s article is particularly problematic because the author provides ampler details on the results of his simulation (an 18 second flight scenario operating within 0.5 to 4.5 Mach velocity and 0 to 35 km height), but the graphs are clearly meant to be read by a physicist (or rather a rocket scientist) who understands roll angle, yaw rate and other such concepts.[3] One of the figures does provide some clues to the adequacy of the algorithm’s decision versus what the data fed to the simulator should cause, and the graphs show that for roll angle and acceleration in the z dimension the system follows the ”physical” path adequately (although the case for the y dimension looks a lot more murky)(Figure 4). Lin also mentions that his concept would not require a change in parameters under different operating regimes, which is required under traditional gain-scheduling autopilot designs. This conclusion closely matches the conclusion of the GARL paper.[2]

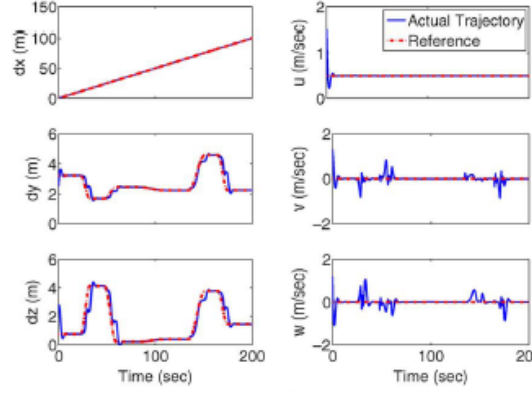


Figure 2: Performance of reinforcement learning for the morphing aircraft versus the actual data of the simulation[6]

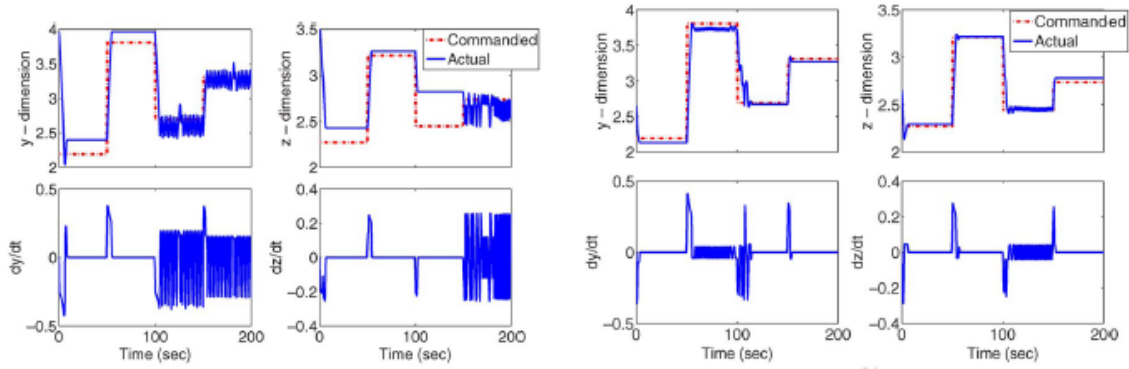


Figure 3: Performance of reinforcement learning for the morphing aircraft using KNN (left) and Galerkin SFA (right) function approximation methods[6]

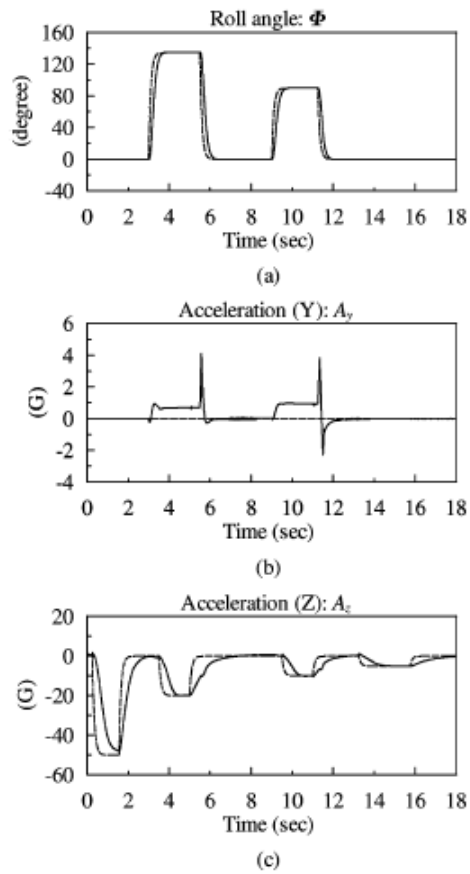


Figure 4: Performance of reinforcement learning for the missile guidance system under BTT versus simulated test values[3]

	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 0.9$
Before learning	1.1	1.1	1.1
After learning	2.55	2.52	2.25

Figure 5: Performance of reinforcement learning for the UAV paper - reinforcements experienced before and after learning[5]

Finally, the paper on UAVs did not provide any basis for comparison. It merely stated that the average reward experienced was booted from 1.1 before learning to between 2.25 and 2.55 after learning, the exact number depending on the lambda parameter, which is the discount parameter for Q (Figure 5).[5] The authors then merely explained that co-evolutionary reinforcement learning is a promising approach which needs to be further investigated.

5 Analysis of the Results

As stated above, it is difficult to really compare the four papers side by side, as their emphasis were slightly different, and their implementation also differed. However, the following points were observed.

5.1 Common Points that are Mutually Self-Supporting in Most or All Papers

- *Reinforcement learning, if used with the right support functions, can overcome the challenges of a sparse action/state table and the difficulty to model precisely a complex physical environment.* All of the papers concluded that their approach to reinforcement learning overcame these potential pitfalls - but all used a different way to do it. One used a better way to approximate the physical function ([6]), another used genetic algorithm to explore the solution space ([5]), and others used either two neural networks ([3]) or two sets of reinforcement policies ([2]). The underlying message is that reinforcement learning can be used in Aerospace so as long as it is implemented with the right strategies.
- *It is possible to reduce the time taken to learn and react to the environment to an acceptable level.* Once again, this is done through selecting the right support functions and a bit of fine-tuning. All of the papers concluded that the performance of reinforcement learning could be made adequate for the stringent requirements of Aerospace. Lin, perhaps inadvertently, made the point all the more evident by mentioning he was able to achieve his satisfactory results using a C language program (less optimized for complex calculations than, say, R) running on a relatively poor environment (Pentium 4 with 128MB of RAM, and since this was a simulator a significant part of the memory would be occupied by the OS).[3]

5.2 Points that Contradict Each Other

The only point that is really in contradiction is whether the reinforcement learning algorithm should attempt to approximate the physical properties present in the system. Indeed, all papers indicated that modelling the underlying physical principles at work was hard or impossible, so at best the only real option is to approximate what the function is. This is precisely what some of the papers attempted ([6]), while others tried to bypass the whole process altogether ([2]). It would be interesting to learn which approach is best, as it seems to be the only two approaches that cannot be used in conjunction with one another.

6 Conclusion

This quick survey taught me that, indeed, machine learning concepts can be used at the heart of mission or safety-critical systems in Aerospace. In fact, the very nature of the challenges present in that industry (i.e. the number of environmental and endogenous factors influencing the trajectory and altitude of an aircraft) makes reinforcement learning and other machine learning algorithms prime candidates to deliver systems that are more performant, have more functionalities or simply do things that were not possible before.

However, since most Aerospace applications - at least those that are flying aboard embedded systems in planes - have stringent performance requirements in terms of timing, the adequacy of reinforcement algorithm heavily depends on its implementation. How is the reinforcement done, how or whether to approximate the underlying physical function, etc, all of these factors play an important role.

The question I have left then is whether this algorithm can be applied to other critical embedded industries. I still have a doubt on this point. For instance, in Automotive, applications need to have their timing calibrated down to the nearest nanosecond, according to talks I had with researchers at Toyota. In

the Aerospace industry, we are talking about miliseconds adjustment. This may not seem much, but we are still talking about a major jump in requirements. After all, a plane rarely has to take decisions down to the split second, but in the case of an ABS or air bag system, this is fairly common. Although I have no doubt that machine learning algorithms have their place in less critical functions (such as GPS applications), I am unsure whether they would bring significant advantages in more critical applications embedded in a car Electronic Control Unit (ECU). Or, more to the point, I am uncertain whether these can be made fast enough so the dream of having an autopilot at the car level can be realized using an algorithm such as reinforcement learning. This is the question I am leaving open for future research.

Finally, I would like to thank the professor Charles Anderson for his help throughout the semester.

References

- [1] *IEEEExplore Digital Library*, available at CSU Library (<http://0-ieeeexplore.ieee.org.catalog.library.colostate.edu/Xplore/dynhome.jsp>).
- [2] J. Jiang, H. Gong, J. Liu, H. Xu and Y. Chen, "Altitude Control of Aircraft Using Coefficient-Based Policy Method", *Electrical and Computer Engineering*, 2008, CCECE 2008 (Canada).
- [3] C-K. Lin, "Adaptive Critic Autopilot Design of Bank-to-Turn Missiles Using Fuzzy Basis Function Networks", *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, Vol. 35, No. 2, April 2005.
- [4] *Wikipedia on Skid-to-Turn*, <http://en.wikipedia.org/wiki/Skid-to-turn>.
- [5] H.R. Berenji, D. Vengerov and J. Ametha, "Co-evolutionary Perception-based Reinforcement Learning For Sensor Allocation in Autonomous Vehicles", *The 12th IEEE International Conference on Fuzzy Systems*, 2003.
- [6] J. Valasek, J. Doebbler, M.D. Tandale and A.J. Meade, "Improved Adaptive-Reinforcement Learning Control for Morphing Unmanned Air Vehicles", *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, Vol. 38, No. 4, August 2008.
- [7] *Defense Sciences Office - Morphing Aircraft Structures*, <http://www.darpa.mil/dso/archives/mas/index.htm>.
- [8] *Wikipedia on Batsuit*, <http://en.wikipedia.org/wiki/Batsuit>.
- [9] *Applications of Adaptive Critic Design*, The University of Memphis, Computational Neurodynamics Laboratory, <http://cnd.memphis.edu/ijcnn2009/tutorials/pathi.pdf>.
- [10] *Wikipedia on Galerkin Method*, http://en.wikipedia.org/wiki/Galerkin_method.
- [11] Y. Zhan and N. Ma, sets of slides in Computational Engineering, Galerkin Method, *Ruhr Universitt Bochum - Lehrstuhl fr Statik und Dynamik*, ftp://melmac.sd.rub.de/shortpres/Galerkin_method.pdf.