# The Current State of Autonomous Helicopter Flight

Samuel Moczygemba

December 16, 2009

## Contents



## 1 Introduction

Helicopters are notoriously difficult to fly and have been described as having more complex dynamics than fixed-wing aircraft [1]. These difficulties withstanding, helicopters have played a very important role to society for both defense and non-defense applications. Helicopters perform maneuvers that are impossible with standard fixed-wing aircraft such as hovering and very low speed forward flight. The focus of this survey

will be to examine the current state of autonomous remote-controlled (RC) helicopters. Tools pertinent to autonomous flight, including reinforcement learning, will be briefly defined and their use in this field will be examined. The method used for implementing the algorithms will be discussed, and applicatons demonstrating the use of the conceptual frameworks surrounding the current state of autonomous helicopter flight will follow.

# 2 Applications

The unique flying capabilities of helicopters make them useful for applications such as payload insertion and inspection. Remote-controlled helicopters will be useful for these and other applications in the future. One such area for future use of these helicopters will be crop inspection and spraying [2]. Large organizations such as NASA and the United States Army have already acknowledged their involvement in developing robotic helicopters for such tasks [2]. Farmers are becoming more tech-savvy as they try to increase the yield of their crops with fewer hours of manual labor. It is not uncommon for farming operations to cultivate and harvest thousands of acres with a small number of people. Autonomous helicopters that accurately apply pesticide in a non-destructive manner and perform crop inspections will be welcomed by these farmers if they are affordable and relatively easy to deploy. The ease in deployment could be achieved by developing well-performing autonomous flight methods such as the ones surveyed for this report.

# 3 Tools

An implementation of autonomous flight involves numerous tools ranging from learning models to optimization routines. A brief overview of some of the required tools is presented below.

## 3.1 Reinforcement Learning

Reinforcement learning is based on the concept of an *agent* operating in an *environment* by choosing a *state* from a set of valid *actions* in order to obtain a *reward*. The agent will use the rewards in order to develop a *policy* [3].

### 3.1.1 Agent

The agent controls the movement of the helicopter. The research papers commonly referred to the agent as the "controller" of the helicopter. The actions of the controller are well-defined (see section *Actions* below), but none of the papers described the actual implementation of the controller. It is inferred that the controller is a computer-controlled device that performs the actions of which the helicopter is capable.

### 3.1.2 Environment

Given the agent definition above, the environment of a helicopter-based reinforcement learning system is the actual helicopter and its surroundings. It may seem counterintuitive to define the helicopter as part of the environment, but this classification is in-line with the definition provided with Sutton and Barto [3]. Elements of the environment specifically related to the helicopter include controller latency, payload weight, vibration and fuel level. As is the case with many control systems, there is latency involved with changing a helicopter control and the occurrence of the desired movement. This latency can have a significant impact on the dynamics of the helicopter. Changes in the weight of a helicopter may also dramatically change its performance. Fuel level may be a critical factor in determining whether a helicopter should continue its crop inspection run or return to base in order to refuel. Environmental surroundings of a helicopter include air, ground, structure and other flying objects. Air includes weather, which can drastically impact the dynamics of helicopter flight. Wind speed and precipitation may cause an adverse change in performance for an an autonomous method that performs well in fair weather. The ground may be of varying elevation which could cause the helicopter to crash if it does not have an accurate altimeter. Collisions could occur with structures such as building along with other flying objects. It should be clear the environment in which a helicopter operates is a hazardous area with many variables to consider.

### 3.1.3 State

The state of an RC helicopter can be represented by twelve dimensions comprised of position, orientation, velocity and angular velocity values [4]. The unique case of performing autonomous autorotation, which involves shutting off the engine of the helicopter, requires rotor speed as a thirteenth dimension [6]. Research examples were given where these dimensions were measured by onboard and ground-based sensors [4, 6]. Onboard sensors included Global Positioning Sensors, Inertial Navigation Systems and a digital compass to record the twelve primary dimensions [7]. A tachomoter was used to record the rotor speed in the special case of controlling autorotation [6]. Coates et al. states that using the twelve dimensions is a "crude" approximation for the true state of a helicopter [8], but the practice worked to the satisfication of the other authors [9, 7, 4, 1]. In fact the twelve dimensions can be reduced further and still provide an adequate representation of helicopter dynamics [4].

Reducing the number of dimensions is a practice that helps to decrease computational processing time. More than one of the authors used this technique when modeling the state of the RC helicopter. Dimension reduction in this context is achieved by transforming the helicopter from world coordinates to body coordinates [1]. Moving to a body-coordinate system reduces the number of dimensions to eight by eliminating the $x$, $y$ and $z$ variables from the position values along with the *yaw* from the orientation values [1]. These dimensions can be dropped by making certain assumptions about the helicopter. After performing the necessary calculations with reduced number of dimensions, the helicopter is transformed back to the world-space with integration [9].

Additional dimensions beyond the twelve presented comprise the true state of the helicopter environment. These additional variables include: airflow, rotor head speed and actuator dynamics [8]. These variables were not examined further with the exception of using rotor speed in the case of performing an autonomous autorotation maneuver [6].

### 3.1.4 Actions

An RC helicopter commonly has four distinct actions, $(u_1, u_2, u_3, u_4)$, that can be performed by its controller. These actions are comprised of cyclic pitch $(u_1, u_2)$, tail rotor $(u_3)$ and pitch angle $(u_4)$ controls [4]. The total set of actions is therefore $A = [-1, 1]^4$ [1]. The learning model is tasked with choosing a combination of these actions in order to find the maximum policy as directed by the reinforcement learning algorithm. If the algorithm is implemented by a neural network, then the output of the network could be chosen to be the four output values listed above [7].

### 3.1.5 Markov Decision Process

The reinforcement learning papers included in this survey assume they are being used in a Markov Decision Process (MDP). A reinforcement learning task is said to be a finite MDP "if the state and action spaces are finite" [10]. The RL task is therefore defined to have the *Markov property* where the next state can be predicted from the current state [10]. Sutton and Barto suggest that considering non-Markov states as having Markov attributes can lead to better performance of the reinforcement learning system [10]. It is this reason why the authors can use MDP methods even when their control system is not fully Markov-compatible.

### 3.1.6 Reward Functions

Reward functions are instrumental to reinforcement learning policies. A reinforcement learning algorithm uses a reward in order to choose the optimal solution to a problem. The rules for providing the reward to the agent are governed by a reward function. Reward functions identified by various authors in relation to autonomous helicopter flight range from using a previously-defined linear quadratic regulator (LQR) [4] to a unique reward function based on the problem at-hand [7]. Reward functions can be very difficult to implement. They are also commonly subjected to refinement after testing. For example, it has been discovered that some reinforcement learning policies cause helicopter controllers to perform subsequent actions in a manner that causes a physical helicopter to perform erradically due to rapid changes between polarized control values [4, 7]. One method used to fix this erradic behavior was to modify the reward function to

penalize a drastic change in movements between two time steps [4]. Another method for handling the same problem was to implement a quadratic penalty for these actions into the reward function [7].

Sutton and Barto discuss the use of a discount rate which allows us to calculate an estimated return when a system does not have a ending or episode, thus avoiding calculations with extremely large numbers approaching infinity [12]. A discount rate close to one causes the agent to be more "farsighted" in that future rewards receive relatively more importance [12]. A number of examples presented in this research used a discount rate in their implementation even though they could be considered episodic in that they were taught to perform a very specific set of maneuvers [1, 7].

#### 3.1.7 Policy Optimization

The probability that an action will be chosen for the state at a particular time step is known as the agent's *policy* [3]. An agent will change its policy as the reinforcement learning method adapts with the goal of maximizing the reward obtained [3]. Ng et al. discuss the use of a Monte Carlo sampling process to find an approximation for the optimal policy [7]. It is noted that the Monte Carlo process resulted in different values for a policy due to the randomness of the method [7]. This phenomenon caused their controller to be ineffective, thus causing them to try another search algorithm named *Pegasus* [7]. The Pegasus algorithm uses a fixed set of random numbers to search for an optimal policy, thus reducing the problems associated with using Monte Carlo [7]. It has been noted that Monte Carlo methods are better suited to episodic problems [11], and this may have caused some of the problems experienced with using them in this context.

### 3.2 Dynamic Programming

All but two of the main sources for this survey contain references to the use of Dynamic Programming in order to find an autonomous helicopter controller [6, 4, 8]. Coates et al. uses Dynamic Programming whereas the others use Differential Dynamic Programming [6, 4, 8]. Dynamic Programming (DP) is used to find optimal policies through the use of value functions [13]. It is noted that DP is useful for searching "perfect models" based on MDP [13]. Differential Dynamic Programming (DDP) differs from DP in that DDP computes non-global trajectories that may in fact be nonoptimal [14].

### 3.3 Kalman Filters

Data read from instruments in the helicopter and on the ground were used to measure the twelve dimensions outlined above by passing the raw data through Kalman filters in order to produce less noisy values [6, 8, 1, 9, 4, 7]. Kalman filters are widely used in engineering. They are recursive filters that are able to take a set of noisy measurements and estimate the state of a linear dynamic system from them [16]. It is beyond the scope of this survey to discuss Kalman filters in more detail.

### 3.4 EM Algorithm

Some of the learning methods presented use the Expectation-Maximization (EM) Algorithm [8, 9]. The purpose of the EM algorithm is to find "maximum likelihood estimates of parameters in probabilistic models" [5]. This algorithm is used as an alternative to other algorithms such as Scaled Conjugate Gradient [15]. The iterative nature of EM alternates between computing the log likelihood estimation during an estimation step (E-step) to maximizing the expected log likelihood of the E-step in the maximization step (M-step). Coates et al. combined a Kalman filter with dynamic programming to perform the E-step in order to find "trajectory-specific local models" [8]. This is a good example of combining the various tools presented in order to help find an optimal trajectory for the helicopter.

### 3.5 Neural Network

Neural networks are often used for nonlinear data analysis and function approximation. Ng et al. presents a framework for using an artificial neural network with a reinforcement learning method in the context of controlling an RC helicopter [7]. The output of the neural network corresponds to the four actions listed

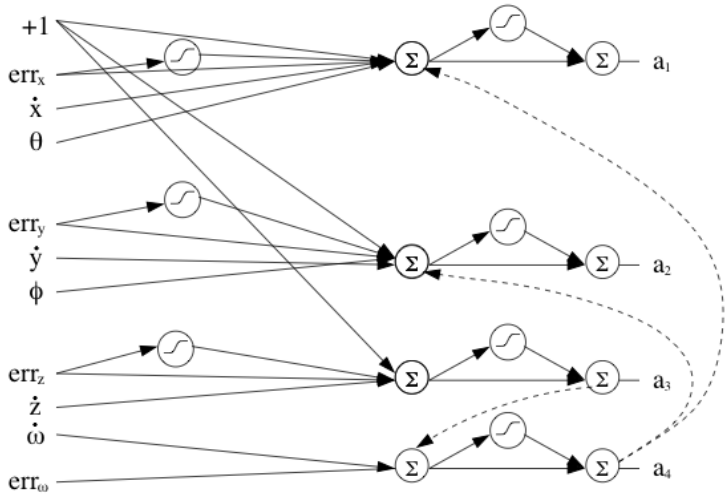


Figure 1: Example neural network for hovering (solid lines) and trajectory-based maneuvers (dotted lines). Figure reproduced from [7].

above, $(a_1, a_2, a_3, a_4)$. Note that $a$ is substituted for $u$, but this is merely a difference in notation between the two papers surveyed.

Figure 1 shows the neural network configurations used in hovering a helicopter and performing more advanced maneuvers [7]. Ng et al. used a similar neural network implementation in their control system which allowed a helicopter to achieve sustained inverted flight [1]. The functions used at each connection were annotated by the symbols in the circles and tunable weight parameters, represented by $w_i$ in the equations, are located at each edge with an arrow [7]. The functions used at the connections are either hyperbolic tangent or summation. The use of the hyperbolic tangent, $tanh()$, is often chosen due to its derivation characteristics [17]. Bishop exlains that the use of $tanh()$ is desirable for an activation function due to fact that its derivative is relatively easy to calculate [17]. The derivative of $tanh()$ also approaches zero for extreme values. Plots of the hyperbolic tangent function and its derivative are shown in Figure 2. The $tanh()$ function is represented by the red line, and its derivative is the solid black line. It can be seen that the derivative approaches a very stable value on the ranges $(-\infty, -3)$ and $(+3, \infty)$. The connections drawn by the solid lines in Figure 1 were used to teach the helicopter to hover [7].

Error measurements were then taken which corrsponded to the helicopter's current state versus its desired state. Following Figure 1, the value of $t_1$ can be computed with $t_1 = w_1 + w_2 err_{x^b} + w_3 tanh(w_4 err_{x^b}) + w_5 \dot{x}^b + w6\theta$. The output value $a_1$, which corresponds to the forward and backward cyclic pitch, is defined as $a_1 = w_7 tanh(w_8 t_1) + w_9 t_1$ [7].

The connections shown by the dotted lines in Figure 1 were changed after evaluating the performance of the system for more advanced maneuvers than hovering [7] The specific change involved linking the collective pitch controls, $a_1$ and $a_2$, to the tail rotor control, $a_4$ [7]. This link logically represents the connection between the tail rotor and pitch controls as these two systems work in tandem to control the trajectory of a helicopter [7].

The aforementioned tools are useful in developing a method for allowing a computer to control a helicopter. The tools must be logically organized and combined into a method in order to perform well.

# 4 Method

The basic method for implementing an autonomous helicopter algorithm involves the following steps: training, modeling, learning and testing.

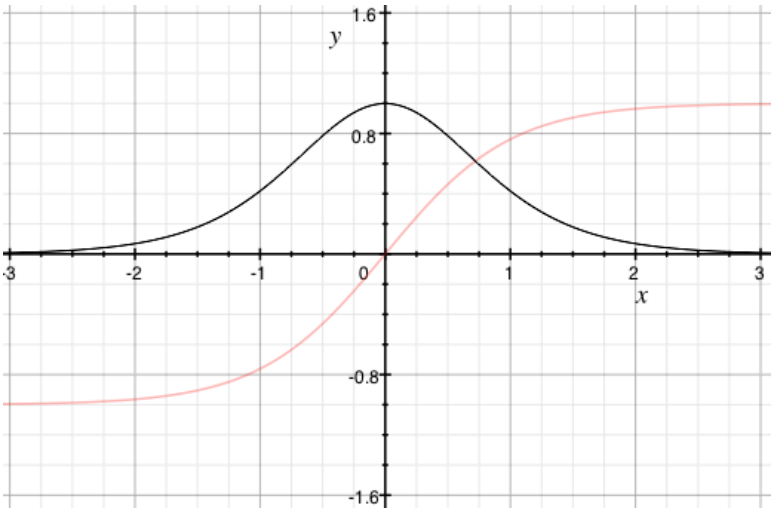


Figure 2: Plot of $y = tanh(x)$ (red) and its derivative (black)

### 4.1 Training

An apprentice-based approach is often used to train autonomous helicopter systems. This approach usually employs a supervised, expert human pilot who performs the desired set of maneuvers while the dynamics of the helicopter are recorded [1, 8, 6, 4]. The recordings are used in conjunction with a dynamics model and learning algorithm to develop an autonomous flight method.

### 4.2 Modeling

The dynamics of the helicopter being used in the implementation must be adequately modeled in order to train a computer to fly the aircraft. Problems related to a myriad of items such as the design of the model, including the dimensions chosen, or malfunctioning sensors can cause modeling problems to be difficult to troubleshoot. Modeling helicopter dynamics has been performed with linear models such as Comprehensive Identification from Frequency Response (CIFER) and non-linear models such as acceleration prediction or lagged-error criterion [9]. The availability of an adequate approximation of helicopter dynamics using the twelve-dimension state described above has proven to provide an adequate system for achieving autonomous helicopter control. Many of the surveyed papers employed this approximation in modeling the dynamics of the helicopter.

Modeling the dynamics of the system must also be combined with modeling trajectories and having the system follow the desired trajectories in order to develop a successful autonomous flight plan. Autorotation is a technique employed by helicopter pilots to utilize rotor speed in a manner to safely land a helicopter in the event of main or tail rotor failure [6]. Developing a flight plan for this maneuver is very difficult due to the techniques involved, combined with the loss of power to the helicopter's engine [6]. The actual autoration maneuver is subdivided into three maneuvers: glide, flare and landing [6]. The glide and landing maneuvers were relatively easy to model, but the flare movement was difficult to quantify into a trajectory [6]. A sufficient model was created by using an "idealized version" of the expert pilot's demonstration trajectory [6].

### 4.3 Learning

The next step in developing a successful autonomous flight method involves the implementation of a chosen learning algorithm. Reinforcement learning was common in the surveyed material and has been described above, but other learning methods could be used. Reinforcement learning was combined with supervised learning in some cases to fine-tune parameter selection [9]. These "optimal" parameters could then be used

in a reinforcement learning method if desired. Learnings time varied with methods chosen. The paper that compared various methods for learning vehicular dynamics provided a wide spectrum of performance results both in terms of algorithm accuracy and runtime [9]. For example, the acceleration-based learning methods were more accurate than the linear methods, but their runtimes were as high as one hour twenty minutes. Linear models working with the same data ranged in processing time from one second for linear regression to five minutes for CIFER [9].

### 4.4 Testing

Many of the papers compared the expert demonstration to the autonomous runs and judged the success of the autonomous flying method in terms of its deviation from the trajectories created by the expert. One of the papers used a unique approach in that the expert demonstration was analyzed prior to beginning autonomous flight in order to find a more optimal trajectory [8]. This method was used to perform very advanced maneuvers in a manner that was significantly better than the expert helicopter operator. Testing often requires a repeat of the other steps above in order to fine-tune parameters and algorithm choices. There are many options when implementing any of the learning algorithms presented and experimentation is often the only way they can be adequately tested. Helicopter dynamics are difficult to simulate and some learning algorithms that perform well on a simulator may not perform well in the physical world [9].

## 5 Examples

Numerous real-world examples were provided in the surveyed research. These examples were based on experiments performed with real (non-simulated) remote-controlled helicopters. The results of these experiments were sometimes surprising. The examples have been separated into those that performed one basic maneuver and those that performed multiple maneuvers.

### 5.1 Single Maneuvers

The experiments that performed a single maneuver should not be considered basic compared to the multiple-maneuver experiments. It will be seen that these single maneuvers are complex examples of what can be achieved with autonomous flight.

#### 5.1.1 Hover

Hovering is an example of a basic flight maneuver performed by a helicopter that is surprisingly difficult to control. The neural network shown in Figure 1 demonstrated a different wiring for hovering versus trajectory-based maneuvers [7]. Autonomous implementations of hovering can perform very favorably compared to expert human pilots. Figure 3, which was reproduced from [7], shows a performance comparison between an autonomous hovering helicopter versus a Yamaha-licensed expert pilot [7].

#### 5.1.2 Inverted Flight

Ng et al. were able to achieve sustained inverted flight of an RC helicopter with a reinforcement learning method [1]. Flying a helicopter in an inverted manner involved turning the helicopter over and flying it upside-down with the main rotor closest to the ground. The experiment was performed by having a human operator launch the helicopter and turn it upside down [1]. The controller was subsequently engaged and achieved the goal of inverted flight over a sustained time period [1]. Performance measures relative to an expert were not provided for this experiment.

#### 5.1.3 Autorotation

Autorotation refers to a method of performing a safe helicopter landing after losing power to the main or tail rotor [6]. The pilot uses a three-step process consisting of glide, flare and landing steps in order to land the helicopter [6]. Twenty-five successful autorotation maneuvers were performed by the autonomous method implemented by Abbeel et al. [6]. Comparisons of the real-world autonomous runs were presented along

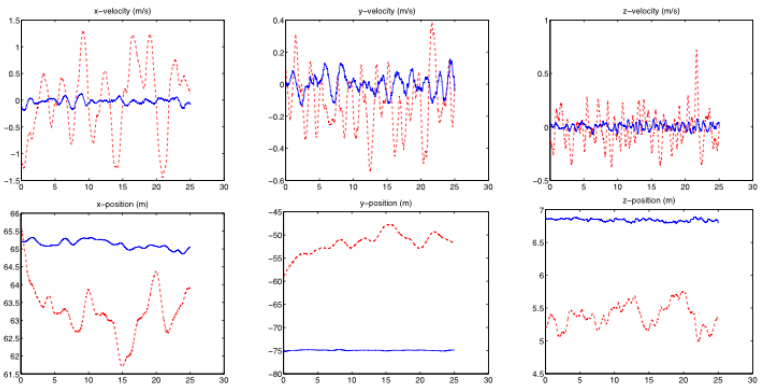


Figure 3: Autonomous helicopter hovering (solid line) versus Yamaha-licensed operator hovering (dotted line). The helicopter position is shown in the top three plots and velocities are shown in the bottom three plots. Figure reproduced from [7]

with the results provided by a simulator [6]. The simulator and experiment performed in a similar manner [6].

### 5.2 Multiple Maneuvers

Successful experiments performed with multiple autonomous maneuvers are relatively recent in the field of RC helicopters. Three maneuvers of Class III-level, according to Academy of Model Aeronautics contest rules, were successfully performed by an antonomous helicopter as reported in a paper published in 2004 [7]. A complete air show consisting of all maneuvers and the necessary transitions was completed by an automomous helicopter and presented at ICML in 2008 [8]. The air show consisted of over ten required maneuvers that had to be performed in quick succession. The autonomous helicopter presented in this research developed its own optimal trajectories based on the analysis of expert trajectories and was considered a significant advancement in the state of autonomous remote-controlled helicopter flight [8].

## 6 Conclusion

Autonomous remote-controlled helicopter flight is a burgeoning field of research with promising applications for the future. Autonomous helicopter controllers are already exceeding the performance of human experts in very complicated maneuvers. Research surveyed in this area covers a diverse range of topics such as dynamics, engineering, control systems and machine learning. The cost of entry is relatively small and there is a chance to make a significant difference in this research area.

## References

[1] Ng, Andrew Y., Adam Coates, Mark Diel, Varun Ganapathi, Jamie Shulte, Ben Tse, Eric Berger and Eric Liang, *Autonomous Inverted Helicopter Flight via Reinforcement Learning*, International Symposium on Experimental Robotics, 2004, `http://cs.stanford.edu/groups/helicopter/papers/iser04-invertedflight.pdf`.

[2] National Aeronautics and Space Administration, *Robotic Helicopter Offers New Option for Public Safety*, Aerospace Technology Innovation, Volume 5, Number 2, March/April, 1997, `http://ipp.nasa.gov/innovation/Innovation52/robhelic.htm`.

[3] Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 1998, pages 51 to 52.

[4] Abbeel, Pieter, Adam Coates, Morgan Quigley and Andrew Y. Ng, *An Application of Reinforcement Learning to Aerobatic Helicopter Flight*, Advances in Neural Information Processing Systems 19, B. Scholkopf and J. Platt and T. Hoffman, MIT Press, Cambridge, MA, 2007, pages 1 to 8 `http://books.nips.cc/papers/files/nips19/NIPS2006_0845.ps.gz`.

[5] Various, *Expectation-Maximization Algorithm*, Wikipedia, `http://en.wikipedia.org/wiki/Expectation-maximization_algorithm`.

[6] Abbeel, Pieter, Adam Coates, Timothy Hunter and Andrew Y. Ng, *Autonomous Autorotation of an RC Helicopter*, International Symposium on Robotics, 2008, `http://cs.stanford.edu/groups/helicopter/papers/AbbeelCoatesHunterNg_aaoarch_iser2008.pdf`.

[7] Ng, Andrew Y., H. Jin Kim, Michael I. Jordan and Shankar Sastry, *Autonomous Helicopter Flight via Reinforcement Learning*, Advances in Neural Information Processing Systems 16, Sebastian Thrun and Lawrence Saul and Bernhard, MIT Press, Cambridge, MA, 2004, `http://books.nips.cc/papers/files/nips16/NIPS2003_CN07.pdf`.

[8] Coates, Adam, Pieter Abbeel and Andrew Y. Ng, *Learning for Control from Multiple Demonstrations*, ICML, 2008, `http://cs.stanford.edu/groups/helicopter/papers/coatesabbeelng_icml2008.pdf`.

[9] Abbeel, Pieter, Varun Ganapathi and Andrew Y. Ng, *Learning Vehicular Dynamics, with Application to Modeling Helicopters*, Advances in Neural Information Processing Systems 18, Y. Weiss and B. Scholkopf and J. Platt, MIT Press, Cambridge, MA, 2006, pages 1 to 8, `http://books.nips.cc/papers/files/nips18/NIPS2005_0824.pdf`.

[10] Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 1998, pages 61 to 70.

[11] Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 1998, pages 111 to 115.

[12] Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 1998, page 58.

[13] Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 1998, pages 89 to 90.

[14] Jacobson, D., *Differential Dynamic Programming Methods for Solving Bang-Bang Control Problems*, IEEE Transactions on Automatic Control, Volume 13, Issue 6, December 1968, pages 661 to 675, `http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1099026`.

[15] Anderson, Charles, *CS545 Lecture: Gradient Descent*, Department of Computer Science, Colorado State University, October 1, 2009, `http://www.cs.colostate.edu/~anderson/cs545/Lectures/week6day2/week6day2.pdf`.

[16] Various, *Kalman filter*, Wikipedia, December 15, 2009, `http://en.wikipedia.org/wiki/Kalman_filter`.

[17] Bishop, Christopher, M. *Pattern Recognition and Machine Learning*, Springer Science+Business Media, LLC, 2006, pages 225 to 246.