Dane Bjork
Joel Eliason

Dr. John Hedengren
Brigham Young University
350 Clyde Building
Provo, UT 84602

April 14, 2016

Dear Dr. Hedengren,

We are pleased to submit to you our final project, which is an analysis of trajectory planning methods, with the specific application of unmanned autonomous vehicles (UAVs). UAVs are becoming a more common subject of study. With problems such as privacy and restricted areas, the autonomous control of UAV models in denfense situations needs to be more fully explored.

Our project investigates the modeling of UAVs based on natural flight patterns of birds. In particular, we focus on the prediction of a target UAV's position in order to intersect that position with a chasing UAV. Such a prediction method could be used in multiple scenarios, including using UAVs to destroy other UAVs that are in restricted zones, capturing UAVs that may contain useful information, etc.

In our project we have two UAVs, each of which flies independently of the other. One UAV takes off initially, and follows a set path given to it. The second UAV then observes the first, and estimates the first UAV's next position. It then takes that estimation and sets it as its target and attempts to collide or intersect with the first UAV at its predicted point. The following items are the focus of our project and hope to explore further through the use of dynamic optimization and model predictive control.

- Solve a system that contains controller lag.
- Analyze different methods of estimating and planning trajectories, such as using first-order and higher-order polynomial fitting and ARMA time-series modeling.
- Create a system that can be easily implemented in real UAV scenarios.
- Model natural systems of bird flight.
- Compare L1-norm and squared error objectives.

In investigating these prediction methods, we have started to uncover further problems and avenues of research, and hope that that these problems will continue to be explored in the future.

Yours faithfully,

Dane Bjork
Joel Eliason

Attached: project report

The Eagle and the Pigeon: An Exercise in Trajectory Planning and Model Predictive Control

Dane Bjork, Joel Eliason, Hasan Sildir

April 2016

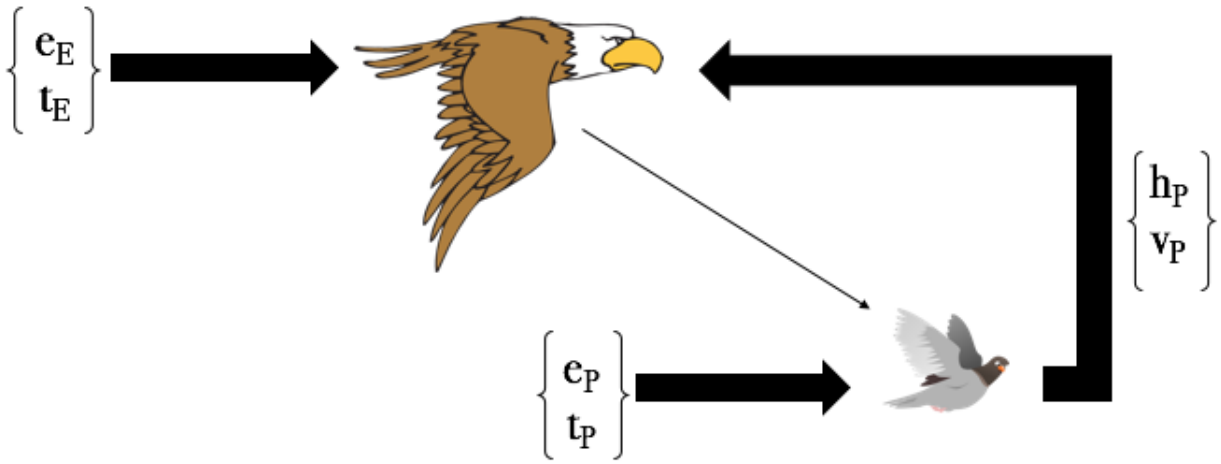

Figure 1: Eagle Vs. Pigeon Model

1 Abstract

Many autonomous systems are currently being developed in order to track and follow targets. Such systems, however, are fundamentally unable to intersect the current state of their dynamic target, simply because they are unable to accurately predict the future states of their target. These systems are inherently reactive to the movement of their target, rather than actively predicting the target's movement. Traditionally, control theoretic models have relied on such methods as system identification in order to predict future states. However, without accurate measurements of input and/or rich enough measurements of the target's position, results from system identification become much less accurate. Thus, autonomous systems for which the primary objective is to intersect the current state of their target require fast, accurate methods that rely on relatively few measurements. We present our results in developing such methods in the following, utilizing a toy model of an eagle following a pigeon as representative of one control process predicting the state of another.

2 Introduction

Model predictive control (MPC) is an extremely useful technique for using the process model for a system (which is, at its most coarse grained, an input-output model) to determine the optimal inputs in order to follow a trajectory and achieve a desired output. Often, though, model predictive control is not enough, particularly if process set points are dynamic and changing with time. In these cases, it is often more economical and productive to predict future set points and converge to them. Otherwise, by the time the controller converges to the current set point, the process has not remained constant and the desired set point has moved, relegating the controller to always lag behind the process. Of course, one could utilize model predictive control techniques to understand how the process is going to move. This is an oft-used method to overcome such controller lag. However, without a model of the process and/or rich enough data to estimate the process model, alternative methods must be used instead. It is these alternative methods that will be the particular focus of this project.

In this work, we combine the strategies of model predictive control along with these trajectory planning methods and showcase these efforts in a toy model of an eagle and a pigeon (representative of one UAV attempting to catch another). The eagle is tasked with determining the current and future position of the pigeon and then determining the optimal manipulated variable moves in order to converge with the pigeon, all while the pigeon is moving. Thus, the position of the pigeon is continuously changing, implying that the process set points of the eagle are changing concurrently. A comparison of optimal methods for intersection of the pigeon's path are our main focus in this paper, and we present numerical results from simulation of the aforementioned eagle and pigeon to showcase these methods. An immediately apparent application for this work is in the development of UAVs and drones designed to catch other flying objects, though the results are much more broadly applicable.

In reviewing the literature and recent advances in autonomous UAV development, there are examples of UAVs that are able to catch other drones. However, such work as has been done by Rastgaar [5] or by scientists at Malou Tech [1] still rely heavily on tracking of the target UAV and are less concerned with the intersection of the current position of the target UAV. These developments are more particularly focused on tracking the target UAV long enough to be able to catch it with a net while it remains more or less stationary. Such projects, while applicable, are less informative of methods of overcoming controller lag. While there are many research articles and projects dealing with collision avoidance with UAVs [6], we were unable to find any dealing with UAV collision. It seems that most UAV research deals with protecting and maintaining UAVs, such as the commonly cited Rathbun's algorithm to aid in a UAV's navigation in uncertain environments [7]. These are all very different from our approach where we exhibit the need for a collision between UAVs.

However, outside of the development of UAVs, there are many examples of extrapolation methods that are used in prediction. Such strategies include various methods as polynomial fit by least squares regression and ARMA models fit by maximum likelihood estimation. These predictive methods are the focus of the following section. A method that we did not test, but would be tested in the continuation of this project, is the Kalman filter [4]. As this is a very widely used estimator, comparing our results with that of the Kalman filter would be sure to prove fruitful.

3 Methods

As mentioned previously, in beginning this project, we initially were inclined to rely on system identification methods, as these are quite widely used and are often quite accurate. However, as the project progressed, we realized that such methods typically rely on a knowledge of the inputs and outputs of the target. In designing our methods with UAVs in mind, we knew that the inputs of the target UAV would not be available to the chasing UAV. There are, of course, additional system identification methods that do not require knowledge of inputs; however, such methods usually require a large amount of time-series output data. Our project mentor, Hasan Sildir, initially pointed us in the direction of such a method, called ARMA modeling [2]. However, in testing the predictive value of such a method on our quite constrained data sets (which often contained 10-20 data points), we were quick to realize that an ARMA model would give us very poor results when fit on such a small data set. Therefore, we decided to focus our efforts on extrapolation by polynomial fitting. It is these methods that are discussed below.

3.1 First-Order Approximation

Inspired by the derivative component of a PID controller, our first method was to view the change between consecutive time steps as linear and thus following a constant slope. Therefore, we predicted that the pigeon's next position would be a linear function of its current and last states. Such a prediction is formulated as the following:

$$P_{t+1} = 2P_t - P_{t-1}, \tag{1}$$

where P_i is the position of the pigeon at time step i and the position is incremented over time steps of unit 1. In utilizing this linear change method, we knew that such a method would be limited in the scope of its extrapolation, as the movement of the pigeon would be typically nonlinear. However, such a method would allow the eagle to converge quickly to a quite close approximation of the pigeon's position and allow it to overcome controller lag.

3.2 Higher-Order Approximation

The above method allowed us to quickly overcome controller lag and get much closer to the pigeon than simply tracking the pigeon's movement did. However, we knew that since the pigeon's path would typically be nonlinear, we were prompted to use higher-order approximations of the pigeon's movement, hoping that such an extrapolation would get us a closer prediction to the pigeon's future state than the first-order method. We implemented this method by utilizing the **polyfit** method found in the Numpy library and fitting our data to second- and third-order polynomials. While sometimes able to give us a better extrapolation than the first-order approximation, higher-order methods were generally much more prone to error and thus a much higher variation in eagle trajectory was seen. In some cases, this resulted in worse predictive value than the linear method. A thorough comparison of the two methods is offered in Section 5.

4 Analysis of Models

The eagle and the pigeon models were created based on the 747c models used in class on a previous assignment, which originated in the textbook *Advanced Textbooks in Control and Signal Processing* [3]. The model we used is shown in equations 2 & 3. However, in simulating the pigeon and eagle, we realized that we needed to add constraints and tune control parameters in order to make our models more realistic. In order to prevent the models from going "underground" we altered the model slightly, with a hard constraint on both the pigeon and the eagle such that their vertical positions could not be negative. There are also different constraints set on the pigeon and the eagle in dealing with thrust. The eagle would be much more capable of thrust, giving it less of a limitation on its range of thrust. When running both the pigeon and the eagle models separately, they are able to reach their given set points, but do so differently based on these constraints. This can be seen in Figures 2 & 3.

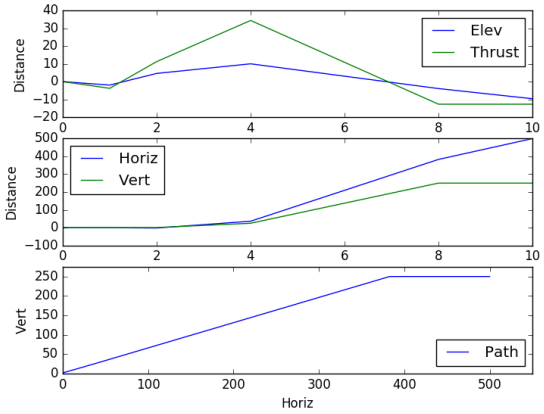

Figure 2: Pigeon model run with a set point of 500 in the horizontal position and 250 in the vertical position. Solved using L1-norm objective.

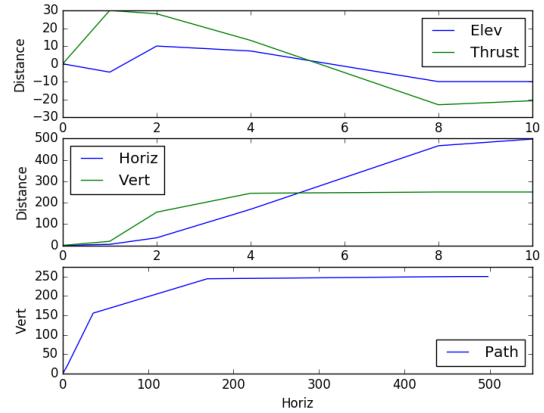

Figure 3: Eagle model run with a set point of 500 in the horizontal position and 250 in the vertical position. Solved using L1-norm objective.

Because these models are very similar, the sensitivity analysis of both the thrust and elevator positions behave with similar results. For both the eagle and the pigeon the thrust and elevator positions were tested. For the thrust sensitivity analysis we held the elevator position constant at 0, and about halfway through the time series we increased the thrust from 0 to 50 as seen in Figure 4. For the sensitivity analysis of the elevator position we held the thrust constant at 20. The elevator position was initialized at 0, then increased to 5 one third of the way through the time series, then increased again to 10 later, as seen in Figure 5. As expected, our sensitivity analysis shows that our inputs truly do affect our outputs. The thrust is directly related to the horizontal and vertical distances of our model, and the elevator position and aid in direction without effecting the change in distance. The accuracy of our models allows us to assume that we have an accurate and realistic model when using these for flight simulations.

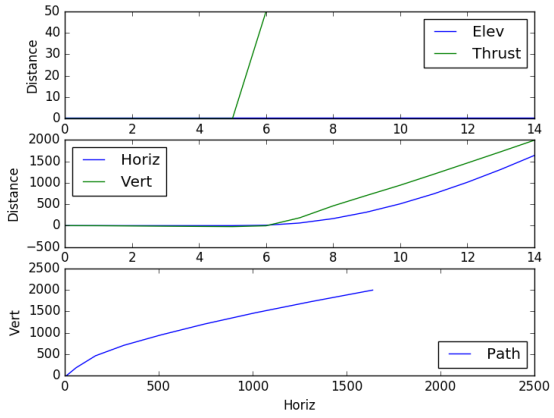

Figure 4: Elevator held constant at 0. Thrust initialized at 0 and increased to 50 half way through the time series.

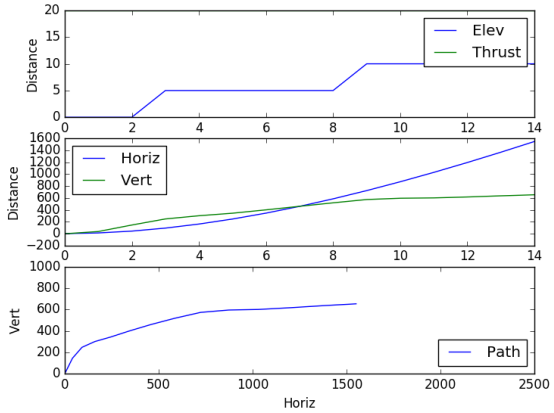

Figure 5: Thrust held constant at 20. The elevator position was initialized at 0, increased to 5, then increased again to 10.

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -0.003 & 0.039 & 0 & -0.322 & 0 & 0 \\ -0.065 & -0.101 & 7.740 & 0 & 0 & 0 \\ -0.020 & 0 & -0.4290 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u - u_w \\ w - w_w \\ q \\ \theta \\ x \\ z \end{bmatrix} + \begin{bmatrix} 0.01 & 1 \\ -0.18 & -0.04 \\ -1.16 & 0.598 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} e \\ t \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ y_4(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 7.74 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u(t) - u_w(t) \\ w(t) - w_w(t) \\ q(t) \\ \theta(t) \\ x(t) \\ z(t) \end{bmatrix} \tag{3}$$

When attempting to reach a set point in our models we used the two common approaches, the squared error and L1-norm objective functions. The squared error method was able to get close to the set point, but had a much harder time ending at the given set point. In Figures 2 & 3 we used the L1-norm method. This method seemed to prove the most accurate of the two. It allowed us to give different weights to upper and lower set points, and allowed the model to be very precise in hitting its targeted set point. When using the squared error, we were not able to compare our different approximation methods due to the inaccuracy of the model with reaching the set point. An example of solving for the set points using the squared error approach is found in Figure 6. This attempt at reaching the set points got very close, but no matter which set point was used, there always seemed to be too much variation in where the model would finish. In Figure 6 the pigeon missed its set points by almost one-tenth of its given value.

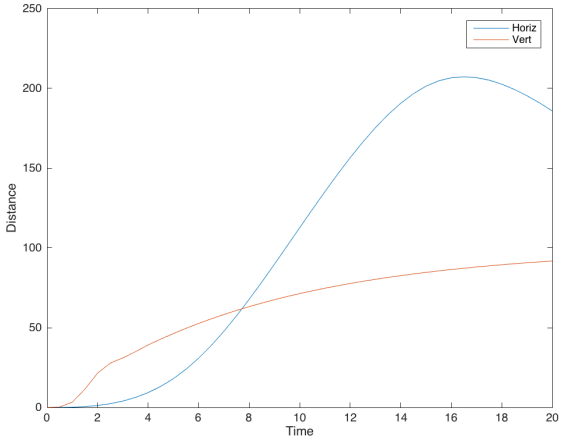

Figure 6: Pigeon model run with set point of 200 in the horizontal position and 100 in the vertical position. Solved using squared error approach.

Once we made the transition to the L1-norm for controlling our models, we were capable of precisely moving the pigeon, and trust that our eagle model will be capable of reaching the next estimated value for the pigeon's position. In Figure 7 the pigeon was able to hit each set point given to it. However, the pigeon still needed to be kept within the physical bounds of its model when determining the set points. If the set points were set too high, the pigeon would not be capable of reaching it. This evidence only further proves that our model is closely related to a realistic situation and is accurate in evaluating our different approximation methods.

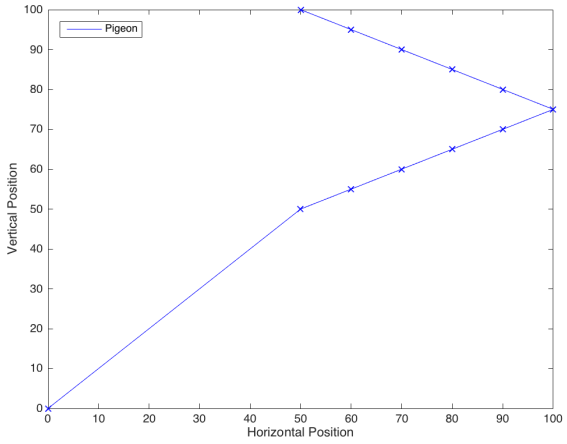

Figure 7: The pigeon hit its first set point of 50 in the horizontal position and 50 in the vertical position. It then continued to hit each incremented/decremented set point thereafter using the L1-norm objective

Once the pigeon was able to move to the designated set points, we were able to run our model, feed the last known position of the pigeon into the set point of the eagle, and watch the eagle trail the pigeon. The eagle trailed the pigeon because we forced it to be one time step behind the pigeon. This allowed for us to evaluate the different approximation methods and watch as the eagle tries to learn from the pigeon's previous movements and try and intersect it.

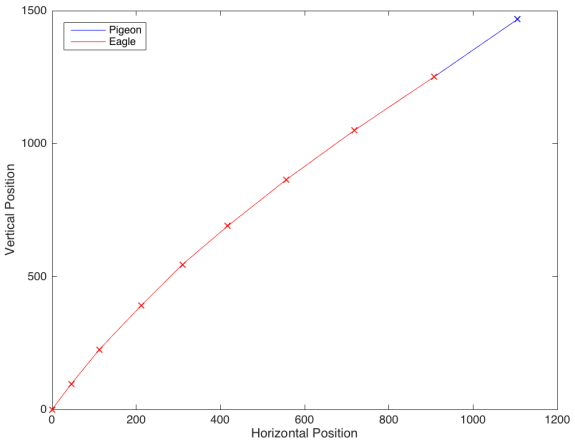

Figure 8: An example of controller lag. The eagle lags behind the pigeon by exactly one time step.

5 Results

Our optimizing efforts were focused on which approximation method would best minimize distance between the eagle and the pigeon. The eagle does not know the next position of the pigeon, but only knows its past positions. With this knowledge it estimates the next position of the pigeon using a first, second, or third-order approximation method.

Each of these approximation methods were used with three separate pigeon movement models. The first movement type is a constant positive movement in both the horizontal and vertical positions. However, to each position move is added a random perturbation. The horizontal position can increase anywhere between 30 and 60 between time steps, and the vertical positions can increase anywhere between 30 and 45. The random pigeon model is implemented in Figures 9, 12, and 15. The second movement type of the pigeon simulates takeoff and flight. It follows a square root function and levels out the further the pigeon moves along the horizontal distance. This movement model can be found in Figures 10, 13, and 16. The third pigeon movement model is similar to the previous. It follows a square root model to the middle time step, where it then reflects its vertical movement and slowly descends. This simulates a pigeon taking off and landing and is used in Figures 11, 14, and 17.

These three pigeon models attempt to simulate a realistic pattern that a pigeon would follow, and should allow for a more accurate measurement of the approximation method's error (eagle's distance from the pigeon).

5.1 First-Order Approximation

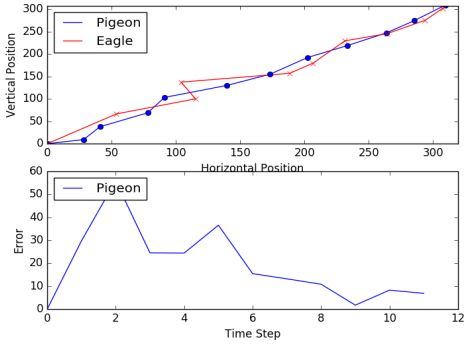

Figure 9: The eagle tracking the pigeon on a randomized, upward course using the first-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

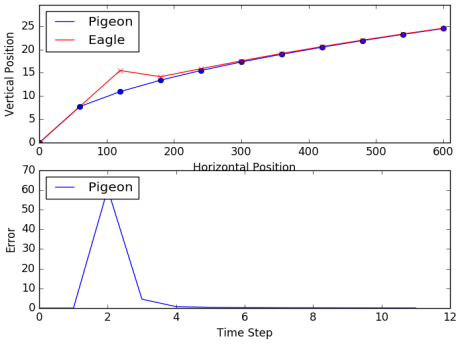

Figure 10: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The eagle estimates the pigeon's next point using the first-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

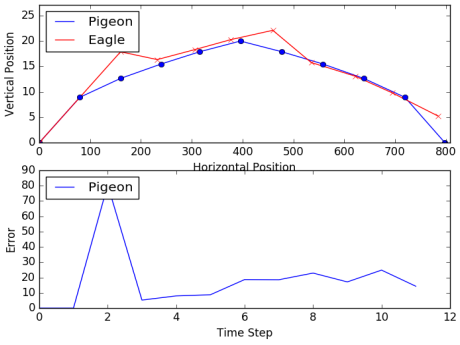

Figure 11: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The flight is reversed half-way through the timestamps to simulate the pigeon landing. The eagle estimates the pigeon's next point using the first-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

5.2 Second-Order Approximation

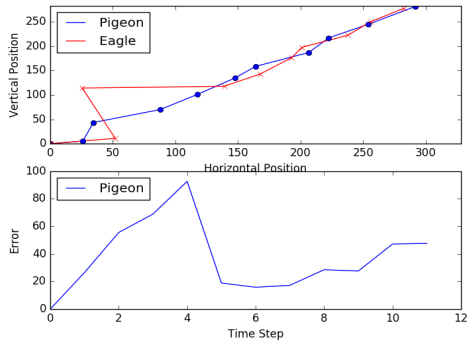

Figure 12: The eagle tracking the pigeon on a randomized, upward course the second-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

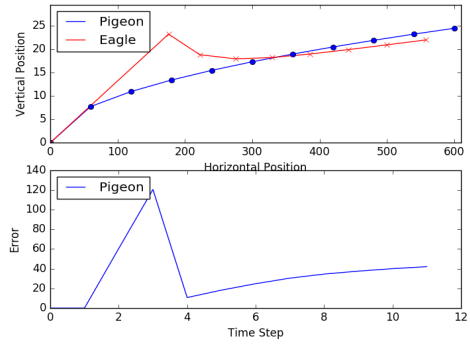

Figure 13: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The eagle estimates the pigeon's next point using the second-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

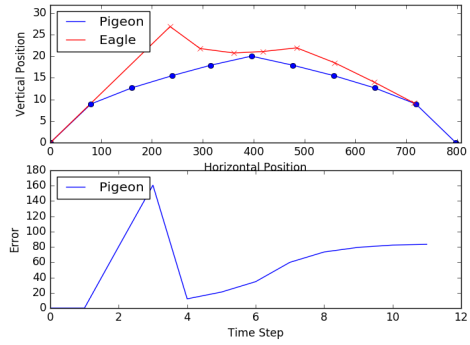

Figure 14: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The flight is reversed half-way through the timestamps to simulate the pigeon landing. The eagle estimates the pigeon's next point using the second-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

5.3 Third-Order Approximation

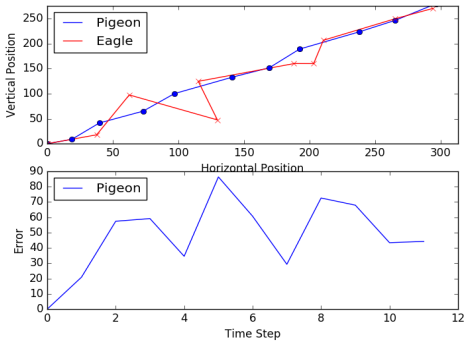

Figure 15: The eagle tracking the pigeon on a randomized, upward course the third-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

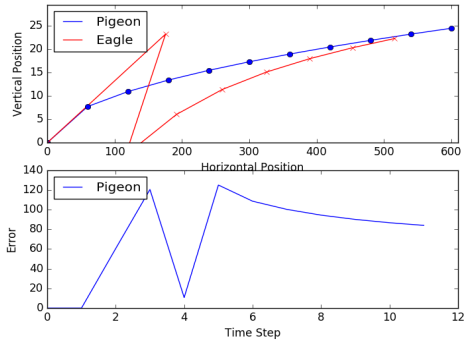

Figure 16: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The eagle estimates the pigeon's next point using the third-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

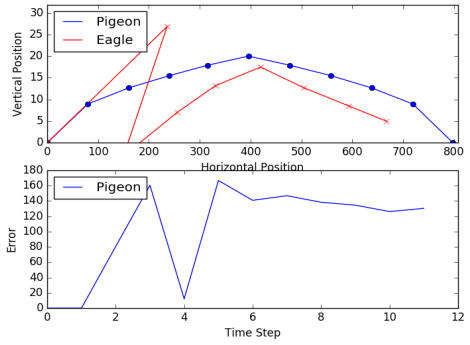

Figure 17: The eagle tracking the pigeon on a flight based on the square root value of the given horizontal distance. The flight is reversed half-way through the timestamps to simulate the pigeon landing. The eagle estimates the pigeon's next point using the third-order approximation. Below is the distance (error) of the eagle from the pigeon at the given time step.

6 Discussion

When comparing the three approximation methods and their errors based on the pigeon movements, we see that the first-order model is consistently more accurate than the other models. The average error rates are difficult to calculate accurately. This is due to the delay of the hawk. Because the hawk is required to delay, the distance from the pigeon spikes initially and becomes a sort of outlier. However, averages can be approximated by evaluating the error graphs underneath each of the pigeon models.

By estimating the error values based on the graph, it becomes clear that the linear approximation method was the most capable of predicting the pigeon's movement. While the pigeon model may not be linear, the first-order model averaged an error rate, or distance from the pigeon, of less than 50 for each of the pigeon movement types. The eagle was even capable of capturing the pigeon (intersecting the pigeon at the same time point) during the in-flight simulation of the pigeon, and came very close in both of the other simulation.

The other two approximation methods seemed to only get worse in estimating the future point of the pigeon. These errors grow substantially from the first-order approximation to the second, where average distance grows from 50 to roughly 80, depending on the movement type. In the third-order approximation, that error average increase to over 100 for each of the model types.

The above results showcase the two methods outlined in Section 3 of this paper. In particular, we have seen that the linear method is generally the more accurate prediction method, due to a smaller magnification in error. This typically compensates for it being a lower-order approximation of a higher-order path.

In simulating our pigeon and eagle, we did not introduce measurement or process noise into our final simulations. In more concrete terms, measurement noise could be thought of as imperfect measurement or estimation of the pigeon's position, while process noise could exist as atmospheric disturbances such as wind. Initial simulations that included these noise processes indicated that they typically did not affect the error distribution of our first-order models; knowing beforehand that higher-order models are more prone to noise, we decided to not include noise processes in our final model. Therefore, a better analysis of the effects of noise would be included in a continuation of this project. Furthermore, our model is inherently limited to two dimensions; an extension of simulation to three dimensions would allow a better analysis of more realistic circumstances. In three dimensions, hard constraints could be introduced as well that represent physical objects in the space, such as trees and rocks. Therefore, in planning trajectories, the eagle would have to account for such physical constraints that it would not just be able to fly through. Lastly, feedback methods for the pigeon were not included in this project. Feedback would allow the pigeon to evade the eagle and would therefore add much more realism to the project. Additionally, it would force the analysis of these prediction methods in greater depth.

7 Conclusion

The focus of this work was overcoming the problem of controller lag. Ultimately, this problem boiled down to a problem of prediction and forecasting, especially under the constraint of limited data, and the utility of polynomial fitting as an estimation technique in control. We were able to successfully show that our methods were able to overcome this controller lag and much more closely approximate the position of the pigeon. However, we recognize that there still exists a literal gap between how closely the eagle approximates the pigeon based on these methods and the actual intersection of the pigeon's current state. Such a gap invites the investigation of other methods for extrapolation and prediction. In particular, we advise the investigation of linear extrapolation methods; as noted above, higher-order extrapolation more often magnifies error than reduces it.

A fitting next step would be exploration of Kalman filtering methods, as the Kalman filter is the optimal estimator of linear dynamical systems [4]. Further testing of other statistical time-series methods (including deeper investigation of ARMA modeling, which we unfortunately were not able to explore deeply due to time constraints) would be quite beneficial in the design of fast and accurate methods for prediction. Lastly, it is almost certain that there is no "one size fits all" method for prediction; therefore, an investigation of using multiple forecasting strategies for the solution of one prediction problem would certainly be quite fruitful.

References

[1] France tests interceptor drones. *UAS VISION*, Feb 2015.

[2] George E. P. Box and Gwilym M. Jenkins. *Time series analysis: forecasting and control*. Holden-Day, 1976.

[3] Eduardo F Camacho and Carlos Bordons Alba. *Model Predictive Control, 2nd Edition*. Springer Science & Business Media, 2004.

[4] Ramsey Faragher. Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]. *IEEE Signal Process. Mag. IEEE Signal Processing Magazine*, 29(5):128132, 2012.

[5] Marcia Goodrich. Drone catcher: "robotic falcon" can capture, retrieve renegade drones. *Michigan Tech News*, Jan 2016.

[6] Luis Mejias, Scott McNamara, John Lai, and Jason Ford. Vision-based detection and tracking of aerial targets for uav collision avoidance. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 87–92. IEEE, 2010.

[7] David Rathbun, Sean Kragelund, Anawat Pongpunwattana, and Brian Capozzi. An evolution based path planning algorithm for autonomous motion of a uav through uncertain environments. In *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, volume 2, pages 8D2–1. IEEE, 2002.