Toward Introspective Human Versus Machine Learning of Simulated Airplane Flight Dynamics

Dan Tappan and Matt Hempleman
Department of Computer Science
Eastern Washington University
{dtappan,mhemple}@ewu.edu

Abstract

This paper presents the preliminary results of an extensible Java architecture for modeling, simulating, visualizing, and analyzing modularized, plug-and-play machine-learning strategies applied to instrument-based airplane flight control. A set of basic flight maneuvers challenged the machine to learn how to fly unsupervised by trial and error, from which the learning module attempted to introspectively determine interdependencies among the many inputs and outputs. For baseline comparison, this work also included a pilot study on human subjects who conducted the same experiments. The overarching goal was to determine how, and how well, both groups learned to solve the same flight-related problems on their own, which could be useful to refine and expand the learning strategies.

Introduction

Flying an airplane by reference to its cockpit instruments alone—no external visual cues—is a complex, multi-dimensional, real-time task that maps a small set of inputs to a large set of dynamically changing outputs in a continuous feedback loop. Formally learning to understand and manipulate such a system is mostly a top-down directed process, whereby a teacher explains problems and how to solve them, and then the learner repeatedly practices variations on the solution process under different conditions until achieving consistent, satisfactory performance (Guralnick and Levy 2009). A problem with this approach for machine learning is that the teacher's investment and oversight may become so extensive that they are almost explicitly programming the solution (Poli, Langdon, and McPhee 2004).

Although impractical in real life, learning to fly in a predominantly unsupervised bottom-up manner by trial and error may also be effective. In a simulated environment with no real consequences for failure, the unsupervised learner may be able to develop their own model of how the system operates with far less hands-on involvement from the teacher. Not only may it be possible for this reinforcement approach to achieve the same goals, but if done strategically, it could also introspectively show how it learned to do so for insight into the process of both flying and learning to fly (Haykin 1994; Harrington 2012).

This work focuses on an extensible architecture for the modeling, simulation, visualization, and analysis of instrument-based airplane flight control, with a plug-and-play module for the learning strategy. The long-term application is to investigate and compare various machine-learning strategies. This paper describes the architecture, a straightforward proof-of-concept learning strategy, and a pilot study of human subjects for comparison. The primary goal is to determine how, and how well, both groups learn to solve the same flight-related problems on their own.

Pedagogical Foundation

Any nontrivial system has complex interrelationships among its components. The continuous mapping of inputs to processing to outputs is based on countless direct and indirect dependencies, correlations, causes and effects, stimuli and actions, and so on (Haykin 1994; Jones 2008). The framework for learning here is based on first decomposing the problem space of flight data into its constituent W5H question words (i.e., *who*, *what*, *when*, *where*, *why*, and *how*), and then trying to establish a richly interconnected associative DIKW structure for it hierarchically from superficial to deep understanding as follows (Bloom 1956; Dorn 1989; Irish 1999; Rowley 2007):

- ***D**ata*: raw values with no associativity or context; *what* questions.
- ***I**nformation*: values in one context; *how* questions.
- ***K**nowledge*: values in multiple contexts; *when*, *where*, and *why* relationships.
- ***W**isdom*: creation of generalized principles by connecting a network of contexts from different sources for predictive, anticipatory, proactive understanding.

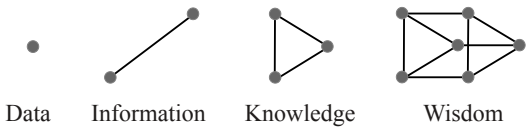

Figure 1: Learning Associativity

An accomplished learner (the *who*) can generally indicate *what* happens *when* and *where*, and *how* it happened or *how* to make it happen, but they do not necessarily understand *why*. The introspective aspect of this work allows for postanalysis by a subject-matter expert to glean insight into the rationale behind decisions. Such insight could be used to refine teaching and learning processes.

System Architecture

The system consists of 327 Java classes, with Swing and Java 3D for the graphics. The human test subjects were using this code base primarily for developing an unmanned aerial vehicle simulator as the project in their undergraduate software-engineering course, so much of this code is not directly related to this work yet. The main components of interest here are the flight-dynamics model, machine-learning engine, instrumentation, and data logger.

Flight Dynamics

The flight dynamics reflect a Cessna 172, which is the world's most popular airplane thanks to its docile handling characteristics and forgiving nature (Cessna 2014). The underlying flight-dynamics model, while a necessary abstraction and simplification of reality, still captures the main elements of any traditional fixed-wing aircraft (FAA 2011). Its six degrees of freedom represent where the airplane is positioned in three-dimensional space, and where it is facing. Specifically, it uses a right-hand coordinate system for x, y, and z, as indicated in Figure 2, where rotation about each axis is respectively roll, pitch, and yaw.

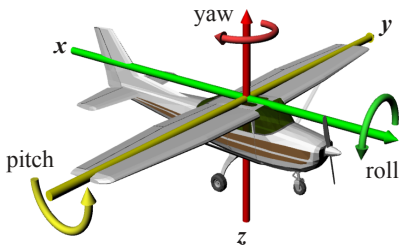

Figure 2: Coordinate System (Sketchup 2014)

In addition, two axes correspond to the main forces of flight. Thrust moves the airplane forward along the x axis, which drag opposes. Lift is always perpendicular to the xy plane, while weight (gravity) is always straight down. The x, y, z and weight components are in the global (world) frame of reference and are independent of the airplane, whereas roll, pitch, yaw, thrust, drag, and lift are in the local frame of reference.

Input

The flight control surfaces in Figure 3 redirect airflow over the airplane to change the roll, pitch, and yaw, which in turn contribute to changes in the (x,y,z) position. The elevator on both sides of the horizontal stabilizer deflects up or down in unison to change pitch. The ailerons outboard on each main wing deflect up or down in opposition to induce roll. The rudder on the vertical stabilizer deflects left or right to coordinate changes in yaw. The flaps inboard on the wings deflect down in unison to increase the wing lift and drag, generally only for landing. Finally, the propeller generates thrust. The *Flight Dynamics Processing* section describes these relationships in detail.

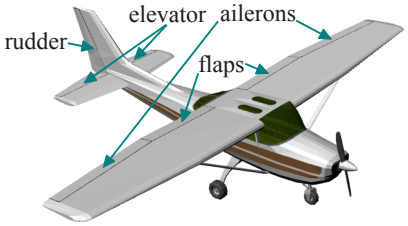

Figure 3: Flight Control Surfaces (Sketchup 2014)

The primary real-world control interface usually involves a wheel, yoke, or stick, as well as pedals. For logistical reasons, the human interface was limited to the keyboard. There were three modes of operation connecting a key press to an action:

- *Instantaneous* changes go to the maximum limit immediately and return to neutral upon release.
- *Incremental auto* changes occur stepwise until reaching the maximum limit or the key is released, then return stepwise to neutral.
- *Incremental manual* changes occur stepwise until reaching the maximum limit or the key is released, then remain there. Opposite action is necessary to neutralize the effect.

The throttle was always in incremental manual mode. Otherwise, this paper consider only instantaneous and incremental auto. The modes remained separate in the experiments for independent analysis. The rationale is that instantaneous inputs are likely tied to determining only *what* the appropriate action is and *when*, whereas incremental inputs also factor in *how much* to apply in terms of time, as well as how to cancel the action.

Output

To fly—and especially to learn to fly—the pilot needs constant awareness of the state of the airplane with respect to the world, known as situational awareness (FAA 2011). The underlying mathematical model, with its 32 variables, is a major simplification of the real world with perhaps several times this number (Napolitano 2011). However, most of these data are not directly accessible to the pilot, who is limited to observing only what is depicted by the instruments. (Visual and kinesthetic [motion] senses play a role in visual flight, but not in instrument flight; in fact,

ignoring kinesthetic inputs, which are dangerously deceiving, is a major challenge.)

Excel

Instruments depict data or information either by directly presenting it (e.g., altitude determined by air pressure) or indirectly computing it from multiple fused sources (e.g., vertical speed as a change in altitude over time). While the focus on learning here by both human and machine is limited to the instrument depiction, it is valuable (from a DIKW standpoint) to see the underlying raw source. An extensive log file conveniently exports directly to Excel, as in Figure 4.

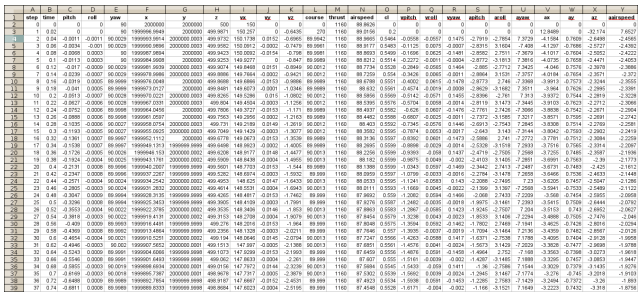

Figure 4: Excel Log Data

While these values represent the discrete states of the simulation in every pertinent detail, no human—even a subject-matter expert—could make intuitive sense of them in this form, which continues for thousands of entries for most maneuvers. Basic visualization as line plots, however, as in Figure 5, can be very revealing. While this representation is beyond the scope of this paper, it is relevant and worthwhile to mention because the key aspect in their value is in deciding *which data* to plot: meaningful relationships are only apparent when presented as appropriate combinations of independent and dependent variables.

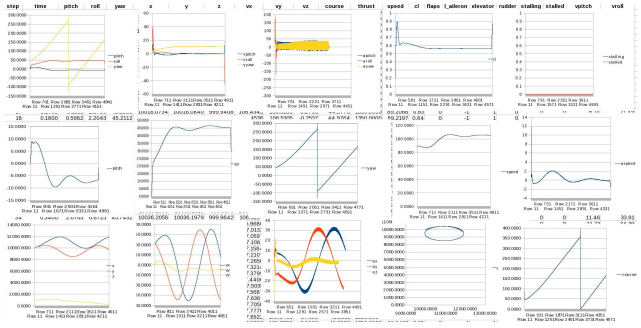

Figure 5: Excel Log Plots

Humans, lacking any insight into the raw data at all, would not be able to decide wisely which plots to generate. Most combinations would be meaningless, although a human would likely find many baseless correlations. Indeed, in an earlier assignment, students were seriously confused by extraneous data and drew wildly incorrect conclusions. A similar situation commonly occurs with machine learning by overfitting the data, among other causes (Conway 2012). Although a machine can easily consider countless combinations, very few of them would truly reflect meaningful correlative and causative behaviors of the unknown system. Therefore, any brute-force approach on the raw data would need to be selective. This foresight played an important role in deciding how to set up the machine learning to operate on the instrumentation data, as discussed in the *Machine Learning* section.

Instrumentation

The nine instruments in Figure 6 depict the refined state of the airplane derived from the raw data. Students in another earlier assignment had already researched their basic form and function, but until this assignment had never seen them in operation. The only difference between the student and machine perspectives was that the students saw this visual representation, whereas the machine saw the equivalent variable representation (e.g., needle position).

A. *Airspeed Indicator* (ASI): shows airspeed in knots.
B. *Attitude Indicator* (AI): shows pitch and roll via an artificial horizon.
C. *Altimeter*: shows altitude in feet above sea level (which is the ground here); the caret, thick needle and thin needle are 10,000, 1,000, and 100 feet, respectively.
D. *Turn Coordinator* (TC): shows rate of turn in degrees per second via the bar, as well as nose-to-tail alignment in a turn via the ball; the *Preliminary Results and Discussion* section elaborates on this relationship
E. *Directional Gyro* (DG): serves as a compass, where the numbers rotate around the stationary airplane.
F. *Vertical-Speed Indicator* (VSI): shows change in altitude in positive or negative feet per minute.
G. *Clock*: serves as an ordinary clock; the caret and reset button were not in play.
H. *Tachometer*: shows propeller revolutions per minute.
I. *Stall Warning*: shows when the wings have ceased to provide lift, resulting in imminent loss of control.

This set of primary instruments, minus G, H, and I, is often called the "six pack" because together they minimally depict the state of the airplane. Loss of one or more, known as a partial panel, may be accommodated with significantly more difficulty by interpreting the others in combination, but such a condition was not part of this work. Nevertheless, the general approach should still apply, although likely with degraded results.

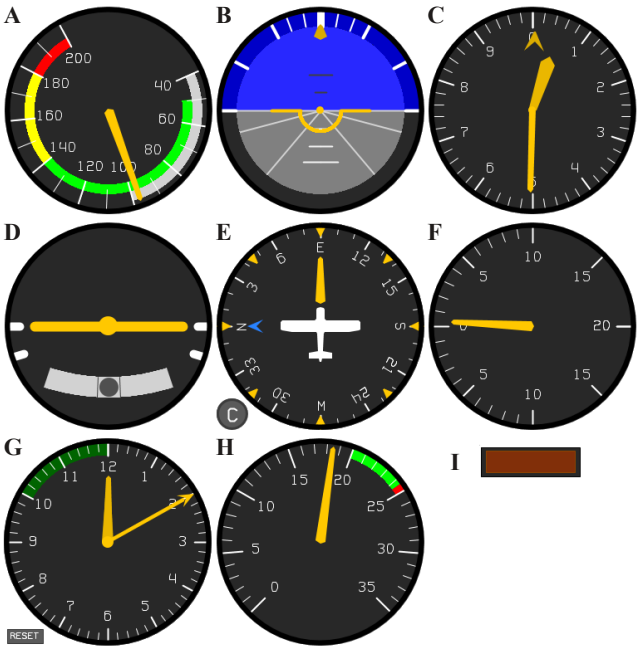

Figure 6: Instrument Panel

The architecture also supports six navigational instruments, but the panel omitted them for these experiments. None of the tests addressed a global frame of reference that required the pilot to know where the airplane was with respect to the world (except in altitude).

3D Viewer

Although the scope of this work was limited to the internal cockpit view of the instruments, for reference after tests, an external view was available. Not only was it entertaining to review both the successful and spectacularly disastrous results, but the discussion proved to be very informative to both students and instructor on why students made their decisions. Such rich reflective and introspective interaction with the machine-learning aspect would be an ideal goal for future work beyond this limited approach .

Figure 7 shows three-dimensional visualizations for two attempts at a counterclockwise turn. This visualizer has seen extensive use in the first author's artificial intelligence courses, related pedagogical research, and industry work as a general-purpose world viewer (Tappan 2008, 2009, 2012).

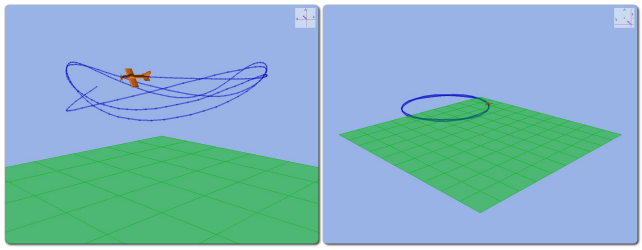

Figure 7: Turn Visualizations

Flight Dynamics Processing

The flight-dynamics model is a Java port of the C++ code by Bourg (2002). The main differences are in the input mechanism to account for the instantaneous and incremental modes, the extensive logging capability, and changes to the flight characteristics to model a Cessna 172. Higher-fidelity models are available, but the internals of this one are especially accessible for inspection and logging (Allerton 2009; Napolitano 2011).

While the complex differential equations of flight involve countless intricate interactions, the main objectives of this study were to elicit an understanding of at least the following representative cause-and-effect relationships, which are generalized here for aerodynamic reasons beyond the scope of discussion (FAA 2011):

- An increase in elevator deflection (up) causes an increase in pitch (depicted in the AI), which causes an increase in lift (in the VSI and altimeter) and a decrease in speed (in the ASI) until a stall occurs (in the stall warning); the opposite holds for a decrease in elevator deflection, except for the stall, and the propeller speed increases (in the tachometer).
- An increase in left aileron deflection (up), and therefore down on the right, causes a roll to the left (in the AI), which causes a turn to the left (in the DG and TC bar and ball), as well as a loss of lift (in the VSI and altimeter); the opposite holds for a decrease in left aileron.
- An increase in rudder (right) causes a yaw to the right (in the TC ball), which causes a roll to the right (in the AI), which causes a turn to the right (in the DG and TC bar), as well as a loss of lift (in the VSI and altimeter); the opposite holds for a decrease in rudder. The *Preliminary Results and Discussion* section discusses this relationship further.
- An increase in flap deflection (down) causes a decrease in pitch (in the AI) and speed (in the ASI), but an increase in lift (in the VSI and altimeter); the opposite is dependent on the initial state.
- An increase in throttle causes an increase in propeller speed (in the tachometer), which increases thrust (not depicted on any instrument), which results in an increase in speed (in the ASI) and therefore an increase in lift (in the VSI and altimeter); the opposite holds for a decrease in throttle.

Machine Learning

The long-term purpose of this plug-and-play architecture is to investigate various machine-learning strategies applied to this problem space. At this preliminary stage, only a proof-of-concept module is in play.

Evaluation of learning (machine and human) was not through the traditional crossvalidation approach of learning on a training set, then performing on a withheld test set.

Rather, the goal was simply to reach the objectives however possible reactively, and then for a subject-matter expert to analyze these steps qualitatively to gain insight into how the subjects presumably learned. For now, there is no way to repeat the actions proactively based on this experience, but this capability will be added eventually for rigorous quantitative analysis. Specifically, the steps are:

1. *Acquisition*: receive data from sensors
2. *Transformation*: convert data into usable form
3. *Fusion*: combine data into coherent, unified views
4. *Inference*: derive unstated data
5. *Reasoning*: make sense of data
6. *Prediction*: anticipate trajectory of data

It is fair to characterize the provisional approach here as pure brute force and very restrictive, but it does reasonably reflect the students' approach of developing their own generalized principles through trial and error without understanding the underlying aerodynamic principles. It is an enumerative approach of trying an input, seeing its effects, and continuing if the trajectory toward the objective appears promising, or discontinuing otherwise and trying something else.

The objectives are declarative statements defining the form of an acceptable solution (with some freedom). For humans, English sufficed (e.g., climb at 80 knots); for the machine, it was equivalent hardcoded conditional statements. A priori knowledge was necessary to constrain the solutions to reasonable flight characteristics and avoid undesirable states like flying upside down (Mitchell 1997). Students had acquired this background from earlier research; the machine required additional logic.

The reinforcement signal for evaluating trajectory was crude: converging, diverging, or no effect. It functioned somewhat like a myopic feed-forward neural network with no or few hidden layers and a three-state linear activation function (Haykin 1999; Bourg and Seemann 2004). Each of the four inputs (elevator, aileron, rudder, and throttle) mapped to the 11 accessible values in the instruments (roll, pitch, yaw, speed, etc.). Flaps were initially considered but quickly discarded due to their overwhelmingly destructive effect on the other inputs. The direct mapping considered 44 combinations (4×11); the indirect mapping had a second layer with 440 (4×11×10), and a third layer with 3,960 (4×11×10×9), for a grand total of 4,444 combinations. This network captures relationships of input→output, input→(output$_1$∧output$_2$), and input→(output$_1$∧output$_2$∧ output$_3$), respectively. The decreasing count reflects no need to map to the same instrument output twice. This approach addresses steps 1 through 3 above.

Experiments

A suite of rudimentary experiments provided a rich basis for discovering relationships. Each experiment consisted of a task to perform, which could be attempted any number of times. The logger kept track of the performance data.

Tasks

The 14 tasks considered are basic flight maneuvers that demonstrate a recognition of the current state of the airplane and some understanding of what needs to be done to achieve the desired next state repeatedly toward the final objective (FAA 2012). Each attempt at satisfying a task started in the air with the same initial conditions and was independent of any others. The attempt ended upon reaching the objective or significantly exceeding the specifications. The tasks could be performed in any order.

- *Straight and level*: fly in a straight line with no change in course (0 degrees), altitude (3,000 feet), or speed (80 knots), which are the initial conditions.
- *Indefinite climb*: increase altitude indefinitely at any sustainable vertical rate, where sustainable means stall or loss of control is not imminent.
- *Definite climb*: increase altitude to 4,000 feet at any sustainable vertical rate, then level off.
- *Indefinite constant-rate climb*: increase altitude indefinitely at 500 feet per minute (FPM).
- *Indefinite constant-speed climb*: increase altitude indefinitely while holding speed at 80 knots.
- *Indefinite descent*: decrease altitude indefinitely at any sustainable vertical rate.
- *Definite descent*: decrease altitude to 2,000 feet at any sustainable vertical rate, then level off.
- *Indefinite constant-rate descent*: decrease altitude indefinitely at 500 feet per minute.
- *Indefinite constant-speed descent*: decrease altitude indefinitely while holding speed at 80 knots.
- *Left turn*: perform a 360-degree left turn while holding altitude at 3,000 feet.
- *Climbing constant-rate left turn*: perform a 360-degree left turn while climbing at 500 FPM.
- *Descending constant-rate left turn*: perform a 360-degree left turn while descending at 500 FPM.
- *Descending constant-speed left turn*: perform a 360-degree left turn while descending at 80 knots.
- *Landing*: synchronize a descent with flaps with no change in course (0 degrees) such that altitude is 0 feet when rate of descent is 0 FPM and airspeed is 40 knots (stall). There was no actual runway to target.

Right turns were not considered because in this simplified flight model, they would be mirror images of the left turns. In real airplanes, the characteristics would often be different for reasons beyond the scope of this discussion (Phillips 2009).

All attempts started from straight and level. The first maneuver therefore was to transition to the intended flight maneuver, then to hold it. Tasks with definite targets then

transitioned back to straight and level, whereas indefinite ones simply terminated. For the machine, there is no planning of any sort to carry out tasks. Students were not asked about how they carried them out.

Data Acquisition

The protocol for performing each task was the same for human and machine. The task was indicated, and the state data through each attempt were recorded from start to end. Any number of attempts was possible; only the best was considered here.

The human subjects consisted of three groups. Two were students in different offerings of fundamentally the same upper-division undergraduate software-engineering course, 41 subjects in total. According to a preassignment survey, none had any background in aviation, although some had relevant gaming experience. It was not a goal of this work to compare these groups to each other, so they were considered together as the student subjects.

The third group consisted of a single person, the instructor and principal investigator, with over 20 years of relevant real-world flight experience in both airplanes and helicopters. These results served as a control to verify that the tasks could be performed to the specifications. They also provided some indication of the maximum variation to expect on each task. Even a subject-matter expert exhibits some learning curve and performance inconsistencies, especially due to the unorthodox keyboard input mechanism. The results of the control group were not part of the analysis due to obvious biases. A better control group would consist of real pilots with no role in the development of the project, but for this pilot study, such objective baseline performance was not critical.

Humans subjects had the option of discarding the data from an attempt if they deemed it too unrepresentative of a valid attempt. For example, mistakes in keyboard commands were common. Without this option, the data would subsequently record the process of regaining control, which was not under study.

Data acquisition from the machine-learning process was identical, except that it could not opt to discard its results itself. For both groups, there was selective manual postprocessing for consistency. A common example was removing data from a protracted initial straight-and-level configuration to the start of the attempt, and then after achieving the objective, if the attempt did not terminate on its own.

Preliminary Results and Discussion

Despite working on a graded assignment requiring substantial effort, students by and large enjoyed the exercise, even going so far as to write in the postassignment analysis that they had “serious fun” with it. Moreover, their results were quite consistent with the relationships expected in the *Flight Dynamics Processing* section.

A typical subset of students did not take all or parts of the assignment seriously and submitted unusable results, but these were easily culled by inspection of the three-dimensional visualizations. The remaining results of primary interest are characterized here, but due to space limitations, this discussion addresses only the highlights. Unless otherwise indicated, the student and machine actions were fundamentally the same, although the performance of the former group was collectively always much better.

Instantaneous input mode (i.e., neutral and full control-surface deflection only) was surprisingly much better than incremental auto mode (i.e., smooth stepwise actions) for both groups for all tasks. While instantaneous mode produced choppy (vomit-inducing) results, on average they were more consistent with the expected trajectory. Unfortunately, there was no postassignment survey question that addressed this aspect, so the reason cannot be substantiated. Anecdotally, it appears related to uncertainty in how much force was being applied to the controls, which a real pilot is usually aware of by feel (Napolitano 2011). No instrument depicts this feedback.

Elevator operation was partially intuitive: push forward to go down, and pull back to go up. However, it was not immediately clear that the pitch remains set even when the elevators return to neutral (i.e., input changes elevator, which changes pitch), so the climb and descent continue for some time until aerodynamic effects level the pitch. As a result, the definite tasks often overshot their altitude targets. The machine approach never began this transition early because it is purely a reactive process.

Maintaining a constant speed or rate in climb or descent requires coordination between the elevator and the throttle. The climb and descent, once established, were acceptable, but the transitions usually deviated and required substantial corrections to converge on the appropriate trajectory. Landing was an outright disaster because the flaps and minimal airspeed radically changed the flight characteristics, reducing the margin for error. The target conditions were also the most complex. Flaps were not under machine control as an input, so they were already deflected as part of the initial conditions.

Turning via ailerons was not intuitive. In a car, the driver turns the steering wheel to the desired angle and holds it, returning to neutral to cancel the turn at the end. This relationship is therefore direct between the input and output. In an airplane, it is indirect: the ailerons change the bank, which causes the turn. If the wheel is held as in a car, the bank continues to increase and rolls the plane over. Having to neutralize the ailerons after establishing the bank surprised the students. The machine never figured it out consistently, usually due to an inadequate or excessive

bank angle. Thirty degrees is typical in a Cessna 172, with 45 degrees considered steep.

The bank diverts some of the lift perpendicularly away from gravity in order to force the turn, which results in a loss of altitude. Students realized that they required some additional elevator up for pitch to account for this loss. The machine tried, but it could not coordinate the amount well and generally increased in altitude or entered an unrecoverable spiral descent (known as a "graveyard spiral" when done by human pilots) (FAA 2011). A few students attempted to increase speed (which is aerodynamically valid because it also increases lift), but the lag in acceleration is too difficult to manage. The machine never came close to figuring out this relationship, although it tried.

Rudder usage was an utter failure. Initially both groups tried to turn the airplane with it, which appears deceptively intuitive because it indeed affects the vertical axis and initially appears to have the expected result. However, this approach is completely wrong. Its true purpose is to coordinate the nose-to-tail angle through a turn, in the same way the front wheel on a bicycle maintains the appropriate arc of travel for the amount of lean (bank), where critically the lean/bank *comes first*. Attempting to steer with the handle bars first would result in an upset at any appreciable speed. The only difference in mechanics between these two systems in where the vertical axis is located. On a bike, it is over the rear wheel, whereas on an airplane, it is usually over the main wings, as in Figure 8 (Phillips 2009).

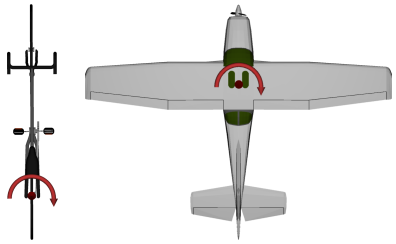

Figure 8: Bicycle Versus Airplane Yaw Axes (Sketchup 2014)

The ball in the turn coordinator is the only instrument reflecting this coordination. It is based on centrifugal force, which is actually not even in the flight-dynamics model. Rather, the virtual instrument uses an ad hoc approach to derive a good approximation by calculating the turning arc based on the bank angle and appropriate subarc that corresponds to the nose-to-tail yaw angle based on the rudder deflection. This information was not accessible to the machine.

Worse is that neither group was even aware that the rudder played a role once they discarded it as an option for directly turning the airplane. The airplane appears to turn with or without rudder input, leaving both groups to disregard its value. Even real pilots are often sloppy with the rudder for the same reasons (Langewiesche 1990). Its aerodynamic effects, while subtle, are still substantial. Figure 9 demonstrates the difference between a coordinated turn with appropriate rudder (A) and ones where there is respectively not enough (B), called slipping, and too much (C), called skidding. On a bicycle, the awkward sideways force would be immediately noticeable and corrected, but in this type of airplane, it mostly affects the passengers in the back, not the pilot in the front, due to the position of the vertical axis, and can easily be ignored with no apparent consequence. This discovery was unexpected and warrants separate investigation.

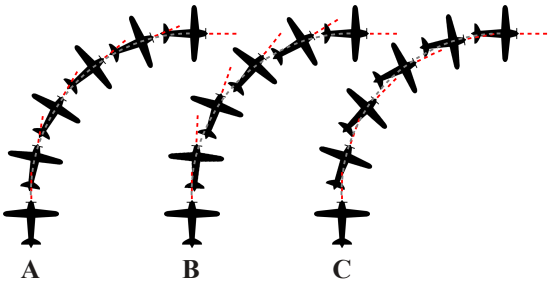

Figure 9: Normal, Slipping, and Skidding Turns

Finally, the bar in the turn coordinator registers rate of turn (normally not to exceed three degrees per second), which is the amount of arc covered in a fixed amount of time. Neither group associated the change in heading with the change in time. Rather, both groups treated the bar as a roll indicator apparently providing the same information as the attitude indicator, despite the depictions rarely agreeing.

Future Work

This plug-and-play architecture was designed for investigating machine-learning strategies, so immediate follow-on work will integrate others beyond the current simplistic one. Moreover, so far the system has considered only the lowest three AI processing levels (acquisition, transformation, and some fusion). Inference, reasoning, and prediction are where higher-level understanding and action occur (Russell and Norvig 2009). Experiments with navigation (both wide-area and local airport approach/departure operations), which the architecture already supports in great detail, offer ample opportunities (FAA 2007). Finally, at all levels, the expressiveness and objectivity of the introspection needs improvement.

Rudder coordination can stand as its own independent investigation. The fact that neither human nor machine could even recognize the situation adequately suggests that it involves many or all of these AI processing levels.

The flight-dynamics model needs to be more flexible in accommodating other test configurations. The current implementation involves significant trial and error to tune. Baseline performance is also difficult to establish, so it could benefit from calibration with real-world airplanes. It

also needs to accept input from a proportional joystick and pedals instead of the keyboard.

Conclusion

As a work in progress, this system has only begun to demonstrate its usefulness. Nevertheless, the flexibility of the plug-and-play modularization of the learning strategy clearly shows promise. The baseline strategy successful captured actions and learning processes of the student group. The de facto machine strategy, while hardly elegant in its application of sheer brute force, showed that it can indeed process many aspects of flight simulation with some semblance to reality. Replacing it with more advanced learning strategies should produce far better results. Finally, the introspective nature of the learning process demonstrated that it can provide valuable insight into how it operates, which was the primary goal of this work.

References

Allerton, D. 2009. *Principles of Flight Simulation*. Chippenham: Wiley.

Bloom, B. 1956. *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*. David McKay: New York.

Bourg, D. 2002. *Physics for Game Developers*. Sebastopol: O'Reilly.

Bourg, D. and Seemann, G. 2004. *AI for Game Developers*. Sebastopol: O'Reilly.

Cessna. www.cessna.com/single-engine/skyhawk. Last accessed 12 Feb. 2014.

Conway, D. 2012. *Machine Learning for Hackers*. Sebastopol: O'Reilly.

Dorn, D. 1989. Simulation Games: One More Tool on the Pedagogical Shelf. *Teaching Sociology* 17:1–18.

FAA. 2007. *Instrument Procedures Handbook, FAA-H-8261-1A*. San Bernadino: Skyhorse.

FAA. 2011. *Airplane Flying Handbook, FAA-H-8083-3A*. New York: Skyhorse.

FAA. 2012. *Instrument Flying Handbook, FAA-H-8083-15B*. Washington: ASA.

Guralnick, D. and Levy, C. Putting the Education into Educational Simulations: Pedagogical Structures, Guidance and Feedback. *International Journal of Advanced Corporate Learning* 2(1).

Harrington, P. 2012. *Machine Learning in Action*. Shelter Island: Manning.

Haykin, S. 1994. *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs: Macmillan.

Haykin, S. 1999. *Neural Networks and Learning Machines*. Upper Saddle River: Pearson.

Irish, R. 1999. Engineering Thinking: Using Benjamin Bloom and William Perry to Design Assignments. *Language and Learning Across the Disciplines* 3(2):83–102.

Jones, M. 2008. *Artificial Intelligence: A Systems Approach*. Hingham: Infinity Science.

Langewiesche, W. 1990. *Stick and Rudder: An Explanation of the Art of Flying.* McGraw-Hill.

Mitchell, T. 1997. *Machine Learning*. Boston: McGraw-Hill.

Napolitano, M. 2011. *Aircraft Dynamics: From Modeling to Simulation*. Hoboken: Wiley.

Phillips, W. 2009. *Mechanics of Flight*. Hoboken: Wiley.

Poli, R, Langdon, W., and McPhee, N. 2004. *A Field Guide to Genetic Programming*. Creative Commons.

Rowley, J. 2007. The wisdom hierarchy: representations of the DIKW hierarchy. *Journal of Information Science* 33(2):163–180.

Russell, S. and Norvig, P. 2009. *Artificial Intelligence: A Modern Approach*. Upper Saddle River: Prentice Hall.

Sketchup, adapted from public-domain models on Google Sketchup: www.sketchup.com. Last accessed 12 Feb. 2014.

Tappan, D. 2008. A Pedagogical Framework for Modeling and Simulating Intelligent Agents and Control Systems, Technical Report, WS-08-02, AAAI Press.

Tappan, D. 2009. A Pedagogy-Oriented Modeling-and-Simulation Environment for AI Scenarios. In Proceedings of WorldComp International Conference on Artificial Intelligence, Las Vegas, NV.

Tappan, D. 2013. Student-Friendly Java-Based Multiagent Event Handling. In Proceedings of Association for the Advancement of Artificial Intelligence, Bellevue, WA.