Whitter, C.A., Alford, R., Aha, D.W., & Karneeb, J. (2015). Programming a human interaction access point for a virtual air combat simulation environment (Technical Note AIC-015-188). Washington, DC: Naval Research Laboratory, Navy Center for Applied Research on Artificial Intelligence.

Programming a Human Interaction Access Point for a Virtual Air Combat Simulation Environment

Caitlin A. Whitter[1], Ronald Alford[1], David W. Aha[1], and Justin Karneeb[2]

[1] Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington, DC
[2] Knexus Research Corporation; Springfield VA
caitlinwhitter@gmail.com | {first.last}@nrl.navy.mil | {first.last}@knexusresearch.com

22 July 2015

1 Introduction

The first author contributed to the Autonomy for Air Combat Missions (ATACM) project during her Summer 2015 internship. She acquired this position through the Naval Research Enterprise Internship Program (NREIP), sponsored by the Office of Naval Research. Her summer mentors were Dr. Ronald Alford and Dr. David W. Aha. Additionally, she worked closely with Justin Karneeb at Knexus Research Corporation.

2 ATACM Background

The ATACM project is a collaboration between the Naval Research Laboratory, the Naval Air Systems Command, and the Air Force Research Laboratory. The project's overall objective is to create a controller for an autonomous unmanned aerial vehicle. This controller should be capable of flying wingman to a manned vehicle during beyond visual range air combat.

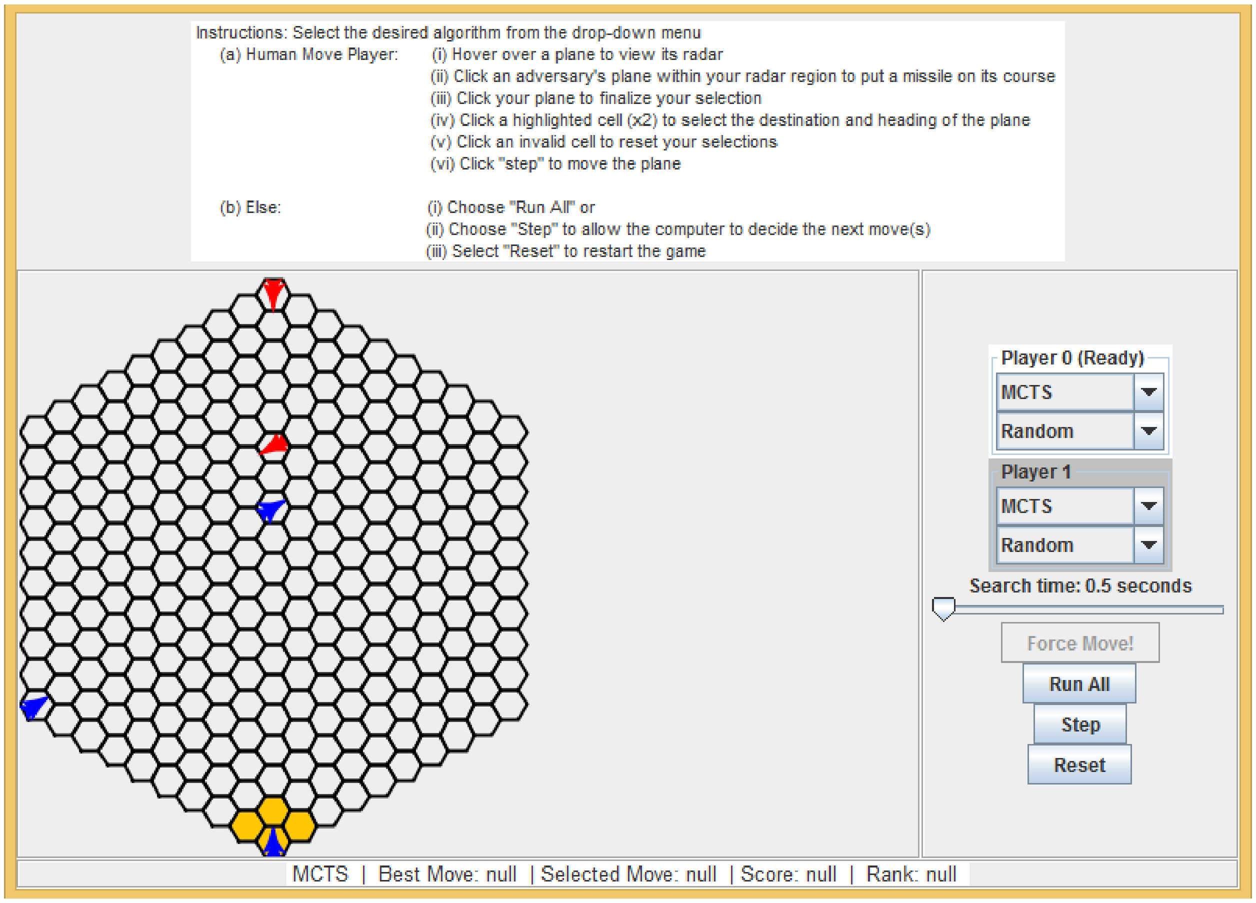
3 Virtual Air Combat Simulation Environment

Caitlin's contribution was to a virtual air combat gameplay system (Figure 1). This environment allows for simulation of air combat scenarios with unmanned aerial vehicles. It enables users to simulate a discrete form of air combat by selecting an algorithm that uses a scoring metric to decide the movements and targets of the aircrafts. The following is an overview of most of the components of the system's graphical user interface (GUI) that existed prior to Caitlin's internship:

Player Manager: The boxes on the right side of the GUI switch from gray to white to indicate whose turn it is. Once users have selected an algorithm from the drop-down menu, they can select "run all" to playout an entire game under that search algorithm, "step" to playout one move for the active player, or "reset" to move all aircraft and missiles back to their original state.

Hex Grid: On the hexagonal grid, there are a configurable number of aircraft and missiles of opposing sides in play. Each aircraft has eight missiles and is capable of firing up to two at opposing aircraft on each turn. Aircraft have imperfect information about their environment; they are only aware of enemy aircraft that are within their radar range. Missiles move a predetermined number of hex cells each turn. Once an aircraft fires a missile at a target, the missile will move toward its target, simultaneously with the movement of the target. In this way, it is possible for the targeted aircraft to "outfly" the missile, depending on the range of both the aircraft and the missile. Additionally, aircraft can move off of the hexagonal grid to dodge missiles (for a deduction of points). However, once an aircraft is struck, it is removed from the environment. A game is won once all of the aircraft of one side are out of play.

MCTS Algorithm: MCTS **(**Monte Carlo Tree Search algorithm) is an option among possible algorithms to use for the simulation to playout the combat scenario on the drop-down menu. It uses a combination of exploration, exploitation, and scoring of playouts to make decisions on where the aircraft should move and fire missiles.



Instructions: Select the desired algorithm from the drop-down menu

(a) Human Move Player: (i) Hover over a plane to view its radar
(ii) Click an adversary's plane within your radar region to put a missile on its course
(iii) Click your plane to finalize your selection
(iv) Click a highlighted cell (x2) to select the destination and heading of the plane
(v) Click an invalid cell to reset your selections
(vi) Click "step" to move the plane

(b) Else: (i) Choose "Run All" or
(ii) Choose "Step" to allow the computer to decide the next move(s)
(iii) Select "Reset" to restart the game

Player 0 (Ready): MCTS, Random

Player 1: MCTS, Random

Search time: 0.5 seconds

MCTS | Best Move: null | Selected Move: null | Score: null | Rank: null

Figure 1: The ATACM virtual air combat simulation environment after adding Caitlin's Human Move Player.

4 GUI Components – Human Move Player

We wanted to develop a GUI to allow for human-controlled gameplay. This system would enable users to override a computer-determined action or simulate an entire game through only human commands. Additionally, we wanted to integrate the system's Monte Carlo Tree Search-based planner (MCTS) into the human player system to allow the user to make informed decisions based on the outcome of the algorithm.

Over the summer, Caitlin wrote this human-interactive player in the Eclipse integrated development environment using the Java programming language. It allows users to move and orient their aircraft, as well as to aim and fire missiles by their command. Moreover, once users have selected a move, the GUI displays to the users the scoring and ranking of the user's prospective move, as well as the suggested best next course of action, according to the MCTS algorithm's calculation.

Instructions: The instructions for generating aircraft movement via computer algorithm or human movement are located at the top of the GUI.

Drop Down Menu: Users must select “Human Move Player” from the drop-down menu on the right side of the GUI to indicate to the program that the user will decide their aircraft’s action, not one of the other algorithms in the drop down menu.

Target Selection: The orange hexagonal cells surrounding the aircraft indicate which player’s turn it is to move. Users can hover their mouse over that aircraft to see what enemy aircraft are in their aircraft’s radar. Subsequently, users can select these visible enemy aircraft to set a missile on their course. The enemy aircraft’s hexagonal cell will turn dark gray to indicate that the user has selected a valid aircraft. The cap on the number of enemy aircraft an aircraft can target is configurable. To deselect the targets and start over, users may click any cell that is not a valid target cell. Once users are satisfied with their choice of targets, they may click their own aircraft (surrounded by orange hex cells) to confirm their decision.

Aircraft Movement: To designate their aircraft’s prospective movement, users must select a hexagonal cell to move to. The orange hexagonal cells indicate the cells to which the aircraft can move. Once users have chosen a valid destination cell, another ring of cells will form in yellow. These cells represent the possible headings the aircraft may have. The heading the aircraft will have after it moves is the directional relationship between the destination cell and the orientation cell. Once users have selected a valid orientation cell, it will highlight in pink. Users may click on any invalid cell to reset the entire movement selection process.

MCTS Algorithm and Human Player Applications: After users have selected their aircraft’s actions, the coordinates, as well as the MCTS scoring and ranking of their selected move compared to all other possible moves, will appear at the bottom of the GUI. Additionally, the GUI will display what the MCTS algorithm determines is the best next action to take.

Move Generation: Once users are satisfied with their action selection, they may press the “Step” button on the right panel of the GUI. This action will generate aircraft movement and missile placement on the hex grid.

5 Project Extensions

The following are additional features that would benefit the human-interactive player system that have not yet been implemented:

Multiple Opening Scenarios: Users could customize each game by selecting the quantity, location, and orientation of the aircraft and missiles, as well as radar capability.

Simultaneous Movement: Users would no longer move their aircraft one-by-one, alternating with the other opponent, but move their aircraft simultaneously. This movement pattern would more closely resemble a realistic combat scenario. In simultaneous play, users would choose the actions of all of their aircraft and the execution of those movements would happen at once.

Movement off the Hexagonal Grid: The human move system would support movement to a position off the hex grid in order to dodge a missile. The MCTS algorithm accommodates movement of the board (for a penalty) already.

Additional Human Interaction Features: The human move system would have additional options for human interaction with the system. For example, users indicate their desired actions by entering their coordinates and other identifying information into a field on the GUI instead of clicking on hex cells.

Further Applications of MCTS to Human Move Player: We could devise additional ways for the human move player to interact with the MCTS algorithm, such as by displaying to users how much the MCTS scoring of the "best move" is due to aircraft leaving the board and how much is due to aircraft being hit by a missile. Further applications would allow users to make even more comprehensive decisions.

6 Conclusions

Participating in this internship has done much to improve Caitlin's Java programming skills, as well as her knowledge of data structures and search algorithms. Additionally, she has gained additional experience in working in a professional environment and reading, understanding, and contributing to other people's code. This summer, she experienced what it might be like to work as a software developer and researcher in the artificial intelligence field. This experience and the skills she has developed will serve her well in her schoolwork and technical career, and she is thankful to have had the opportunity to work at the Naval Research Laboratory this summer.

7 Acknowledgements

Caitlin would like to thank the Office of Naval Research, the Naval Research Enterprise Internship Program, and the Naval Research Laboratory for providing the funding and research facility access that enabled her to participate in this program.

Caitlin also thanks Dr. Ronald Alford, Dr. David W. Aha, and Justin Karneeb for inviting her to work on the ATACM project, mentoring her throughout the summer, and being welcoming and supportive of a student at the beginning of her computer science career.