Autonomous Cognitive Agent Flying Aircraft in X-Plane

Team Members:

Avery Mackey:	amackey2023@my.fit.edu
Dylan Ostolaza:	dostolaza2024@my.fit.edu
Zachary Reese:	zreese2023@my.fit.edu
Brianna Marshall:	bmarshall2022@my.fit.edu

Faculty Advisor:

Dr. Bhattacharyya: sbhattacharyya@fit.edu

Client:

Dr. Bhattacharyya, Department of Electrical Engineering and Computer Science

Date(s) of Meeting(s) with the Advisor and Client:

- August 27 at 10:00 a.m.
- August 28 at 2:00 p.m.

Goal and motivation:

Goal: The goal of this project is to extend the existing X-Plane and SOAR framework by developing a cognitive agent that can adapt the flight plan in response to multiple aircraft equipment failures.

Motivation: The project aims to create a safer and more adaptable autonomous flight system that can handle a wider range of unexpected situations within a single flight. Current models are adapted to specific problems, such as inclement weather or population density, and are not currently able to adapt the flight plan in a significant manner when aircraft equipment failures occur.

Approach (Key features of the system):

- **The user can define the original departure and destination airports for the cognitive agent's flight route.**
    i. The selected airports establish the starting conditions and overall goal of the flight. The user can then observe how the agent determines an appropriate route between the selected locations, allowing different flight setups to be tested.
- **The user can create and simulate an aircraft equipment failure.**
    i. The user can place the aircraft under abnormal operating conditions and observe the agent's response to the selected failure. This allows the user to evaluate whether the response is appropriate for the current flight situation and provides a functional way to test the agent's dependability during unexpected events.
- **The user can create weather and human-population-density scenarios.**
    i. These scenarios represent environmental conditions that the cognitive agent must consider when planning the flight. The user can observe how weather and population density influence the originally selected route and compare different scenarios to understand how environmental changes affect the resulting flight path.
- **The user can observe the flight route change in response to new conditions.**
    i. When conditions such as an equipment failure or changing weather affect the original plan, the user can observe the agent's revised route and evaluate whether the adaptation is appropriate. Comparing the original and revised routes provides a clear view of how the agent changes its decisions as the flight environment changes.

Novel features/functionalities:

1. **Equipment failure handling.** The agent recognizes an equipment failure, adjusts the aircraft's performance, and changes the route when necessary. This extends the system beyond normal flight operations by allowing it to operate under degraded aircraft conditions.
2. **Dynamic flight routes.** The planned flight route can change when new conditions, including weather or equipment limitations, affect the original plan.

Algorithms and tools:

- **SOAR Production Rules / SOAR Architecture**
    i. Defines how the cognitive agent reacts to perceived information and how its working memory is updated as conditions change.

- **IntelliJ**
    i. Java integrated development environment used to develop and maintain the project code.
- **Java**
    i. Primary programming language used by the existing framework that the team is extending.
- **X-Plane Datarefs**
    i. Interfaces to X-Plane flight data that allow the agent and user interface to read or manipulate simulator state. For example, a simulated engine failure can be generated by updating the relevant engine dataref.
- **Maven**
    i. Java project configuration and dependency-management tool.
- **XPC Plugin (X-Plane Control)**
    i. NASA's standard plugin for allowing external software to control and communicate with the X-Plane flight simulator.

Technical Challenges:

The project presents three primary CSE-related technical challenges:

1. **SOAR architecture.** SOAR is a new cognitive architecture for the team, so the group must learn its production-rule system, memory model, and development workflow from the beginning.
2. **X-Plane and aircraft behavior.** None of the team members are pilots, so the details of aircraft systems, controls, physics, and engineering behavior will be new. The team will need to understand enough of these concepts to model failures and interpret the aircraft's response correctly.
3. **Existing codebase.** A substantial amount of code already exists. The team must first understand the current implementation and the communication among SOAR, Java, and X-Plane, then extend the code without breaking the existing framework.

Milestone 1 (Sep 28):

- Compare and select technical tools for the SOAR agent, X-Plane, and Secant.
- Provide small (“hello world”) demos to evaluate the tools for the SOAR agent, X-Plane, Secant, and related components.
- Resolve initial technical challenges:
    i. Develop an understanding of how the SOAR architecture works.
    ii. Understand how X-Plane works and develop the aircraft knowledge needed for the project, with assistance from a flight instructor.
    iii. Review the existing codebase in depth and understand how SOAR, Java, and X-Plane communicate with one another.
- Compare and select collaboration tools for software development, documents and presentations, communication, and the task calendar:
    i. Software development: GitHub
    ii. Documents/presentations: Google Drive
    iii. Communication: group chat and email
    iv. Task calendar: Jira
- Create Requirement Document.
- Create Design Document.
- Create Test Plan.

Milestone 2 (Oct 26):

- Implement, test, and demo a system for triggering and communicating at least one aircraft equipment failure from X-Plane to the SOAR agent.
- Implement, test, and demo the agent’s ability to follow a complete flight plan in the X-Plane simulation with no complications.
- Implement, test, and demo SOAR rules that recognize the selected failure and determine an appropriate response based on the aircraft’s remaining capabilities.
- Implement, test, and demo the agent issuing the appropriate commands in response to the failure.

Milestone 3 (Nov 23):

- Implement, test, and demo the failure-generation system with multiple component failures occurring at the same time.
- Implement, test, and demo the GUI showing the various failures and solutions as they are implemented.
- Implement, test, and demo the agent responding to multiple component failures in parallel.
- Implement, test, and demo a complete flight plan with one or more equipment failures.

Task Matrix for Milestone 1:

Task	**Dylan**	**Zach**	**Avery**	**Brianna**
Compare and select Technical Tools	25%	25%	25%	25%
“Hello world” demos	30%	30%	20%	20%
Resolve Technical Challenges	25%	25%	25%	25%
Compare and select Collaboration Tools	Documents / Presentations	Software	Communication	Task calendar
Requirement Document	20%	20%	20%	40%
Design Document	20%	40%	20%	20%
Test Plan	40%	20%	20%	20%

Approval from Faculty Advisor

“I have discussed with the team and approve this project plan. I will evaluate the progress and assign a grade for each of the three milestones.”

Signature: _____ Date: _____