Ryerson University

Three Degrees of Freedom Helicopter Simulation

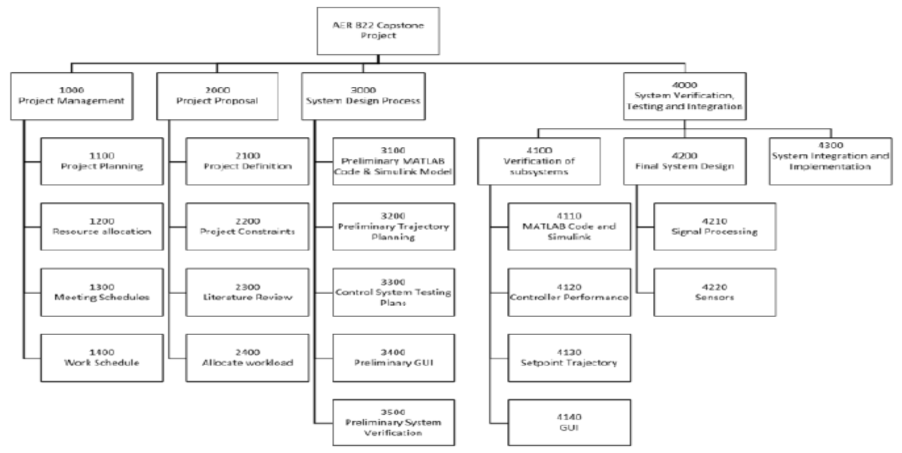
Authors: Pyotyr Young, Lior Saprikin, Muhammad Farooq, Justin Chan, Jeffrey Lee

Department of Aerospace Engineering

Motivation

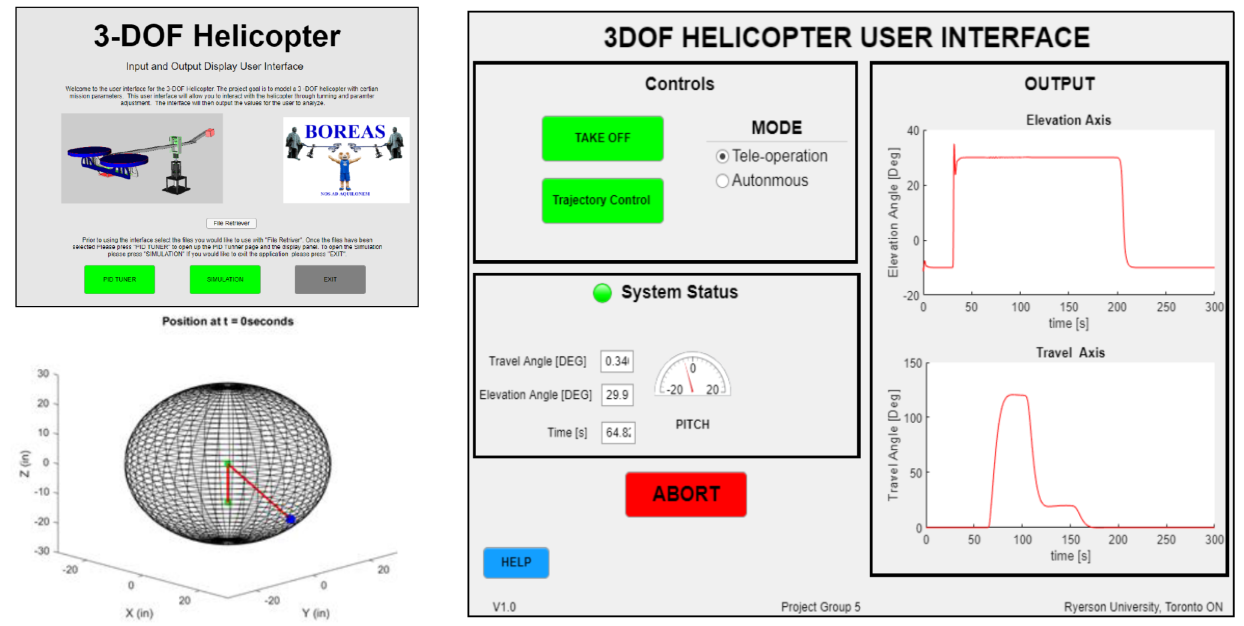
The goal of this project is to design and create a system to effectively control a 3 D.O.F helicopter on certain mission parameters. The team demonstrated that the final design meets all system requirements with acceptable risk and within the cost and schedule constraints and establishes the basis for proceeding to a prototype design. It shows that the correct design options have been selected, interfaces have been identified, and verification methods have been described. The project management and systems engineering methods were developed with the NASA Systems engineering methodology. The sub-teams accomplished all project tasks. The modeling and simulation of the system were finalized through MATLAB and Simulink. Values were finalized through analytical methods and experimental data to build the controllers and PID controls. The mission profile and trajectory were finalized and integrated with the modeling and simulations. The Graphical User Interface (GUI) was developed using the best practices of HMI principles. The GUI has been successfully integrated with the simulations and modeling systems and with the mission profile. The project team has accomplished all the tasks required and is ready for the final design phase.

Approach



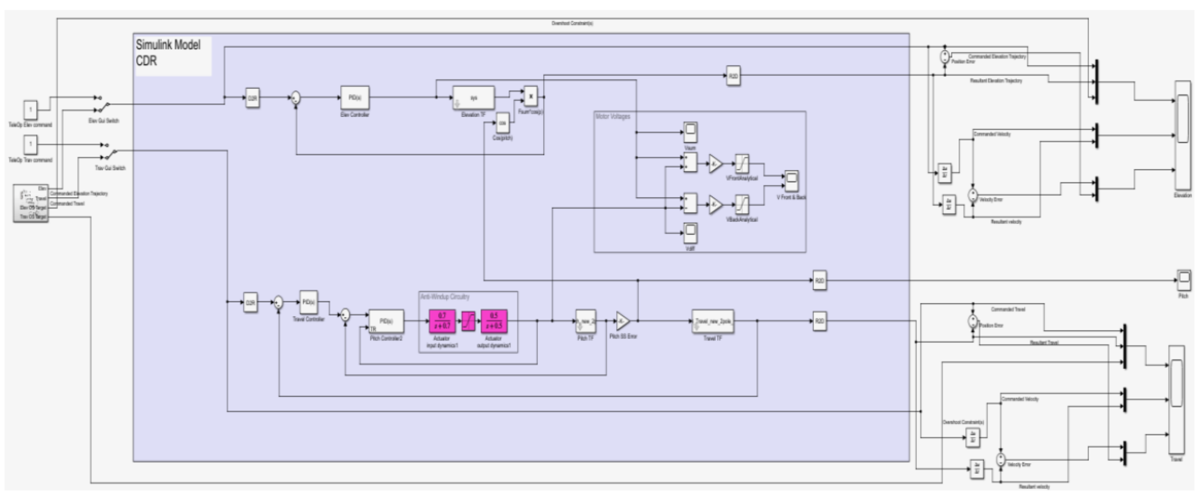
The 3DOF helicopter uses two motors and associated propellers to control the 3 degrees of the helicopter such as pitch, travel and elevation. The models and simulations of the said helicopter are based on the development of experimental models for the three degrees of freedom helicopter. The team first took theoretical models and simulated them in MATLAB. After confirming theoretical values, experimental values from the lab were incorporated. After the simulation was then tuned to the mission profile. The simulation was made into a user friendly graphical user interface a user with no background knowledge or experience with system can operate effectively.

Graphical User Interface



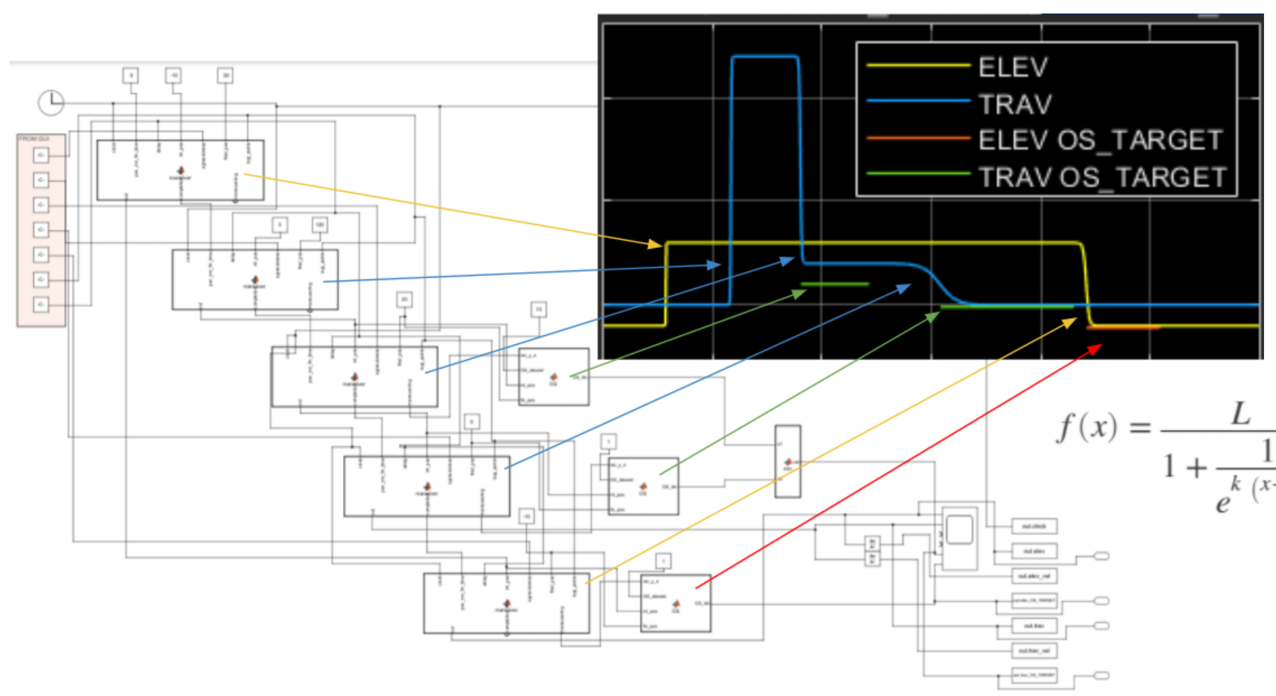
The GUI connects to the Simulink backend, by sending the user-defined commands and inputs (GUI variables) in addition to the trajectory variables. These user-defined inputs as outlined in the phase II GUI workflow figure get send to the Simulink backend. Then these input variables to the Simulink are inputted into the trajectory formulation which was developed by the trajectory sub-team. The trajectory formulation then outputs the desired response which then is inputted to the controllers and plants of the system, which were designed and tuned by the control sub-team. The outputs of the Simulink back end (theta angle, phi angle) are then inputted into the GUI backend post-processing software which then generates the resultant cartesian position vectors used to output the 3D simulation visualizer. Other Simulink outputs such as the simulation time and pitch angles are also sent to the GUI and are displayed in the system status panel in the Figure above. In addition, the scope outputs are used to display the trajectory graphs seen in the conclusion.

Controller



The helicopter control system was simulated using MATLAB's Simulink application. The system model was designed and integrated by incorporating the plant models and controllers into the complete system architecture. Once flight test data was obtained, the experimental model was estimated using the MATLAB's 'iddata' and 'tfest' commands. The final system model of the control system shown in the Figure above displays the internal dynamics inside the box and the trajectory input and output analysis scopes on the outsides. The two switches left of the box allow flight operations to switch between set point control and a mission profile. The Trajectory subsystem block simulates the specified mission profile and outputs the respective elevation and travel signals. The upper subsystem represents the elevation dynamics which include a basic feedback loop with one controller and one plant. Earlier versions of the model assumed that pitch and elevation were independent of one another, but due to the decrease in vertical lift, the assumption was canceled to imitate the true system. To implement the pitch dynamics into the elevation loop, the pitch angle signal passed a 'cos' block then multiplied by the elevation angle. This was a motivating factor for saturating the pitch angle to within ±20° as to not lose elevation stability. The lower subsystem incorporates the travel dynamics in a cascading system. The most inner loop is used for anti-windup tracking with the intermediate loop used for a quick pitch response. The outer loop converts the pitch angle into the travel angle and the signals are recorded in the scopes. A Motor Voltage section was also included to track the voltages driving the motors during the flight mission.

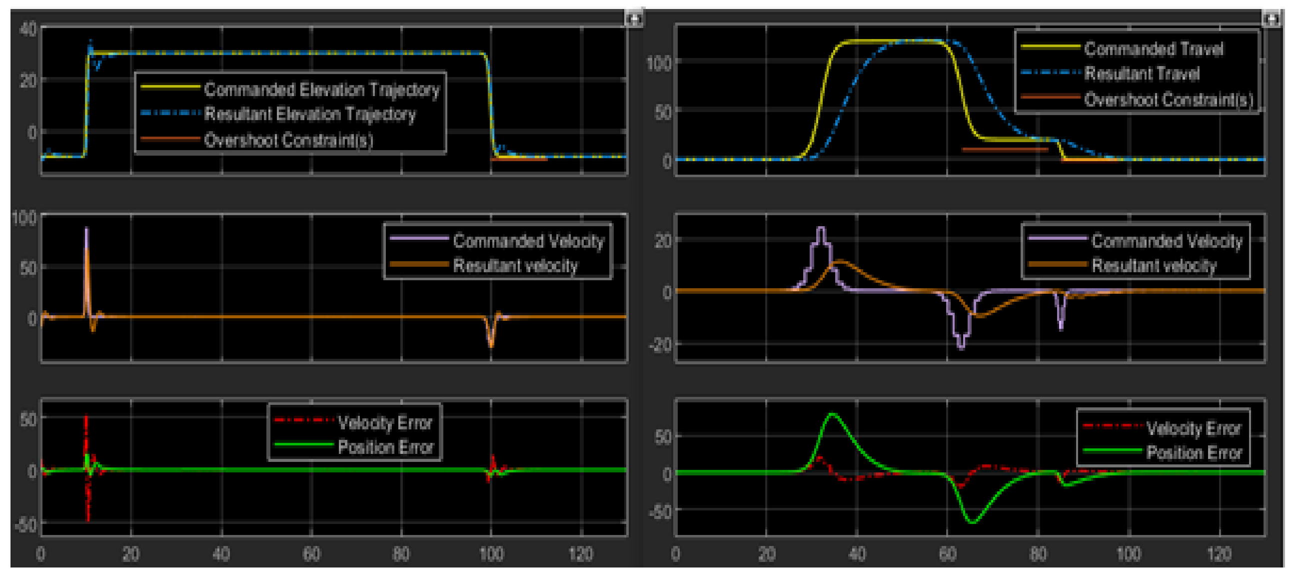
Trajectory



$$f(x) = \frac{L}{1 + \frac{1}{e^{k(x-x_0)}}}$$

The commanded trajectory was based on the final mission profile that was assigned to the whole AER822 class. Each maneuver seen in this image is actually a logistic curve. The logistic curve has a zero-initial velocity that then smoothly increases then decreases in velocity until it reaches its final position. Its equation can be seen as the function f(x). The logistic curve is useful in that it helps in overcoming the static friction in the joints slowly and in a controlled manner before beginning to move more drastically. The biggest difference between the maneuvers are the slopes of their respective curves. In addition to the trajectory, overshoot targets were added in the graph to assist the user in meeting the mission requirements while turning the controller.

Conclusion



The obtained transfer functions from the experimental data were tuned via the root locus editor. Pre-defined requirements were established for the design of each controller and these requirements were derived based on the general operating principle of the helicopter and expectations of the helicopter standard level of performance. In this case, the elevation and travel controller were tuned as a PID while the pitch controller was tuned with a PD configuration. The gains were significantly lowered and performance such as settling time was improved and verified through a comparison conducted with the controllers used in the preliminary design report. The plant models and tuned controllers were implemented into the system model architecture using MATLAB's Simulink application. The system model incorporated negative feedback loops and signal distribution to represent the helicopter's control system. The PDR compared the analytically and experimentally derived transfer functions and concluded that the flight data derived system performed more accurately. The refinements included coupling the pitch and elevation dynamics and incorporating the anti-windup circuitry for the CDR. The final system model simulated the helicopter's dynamics with a high performance, meeting all the mission profile requirements.