Writing Control Law Code for the UAV platform
UAV research group
University of Minnesota, Twin Cities

The control laws for the UAV platform are implemented in C code. The same control code is used by the Software in the Loop simulation (SIL), Processor in the Loop simulation (PIL) and flight computer of the UAV. The control code must implement the function prototypes defined in control_law_interface.h, which can be found in the folder /Software/FlightCode/control/. The functions are listed as follows:

extern double ***get_control_outputs**(**double** * feedback, **double** *reference , **double** time, **double** sample_time)

This function returns a double array containing the outputs of the controller. The array should contain the actuator commands [aileron, elevator, rudder, throttle]. See UAV_controllaw_ICD.pdf for a full description of the control law I/O.

double *feedback : Double array that contains the feedback signals necessary for the controller. In the UAV platform, it contains the body frame angles, angular rates, and accelerations. See UAV_controllaw_ICD.pdf for a full description of the control law I/O.

double *reference: Double array containing the reference signals needed by the controller. For the current UAV platform it contains the pitch and roll reference angles.

double time: Current time

double sample_time: Controller sample time.

extern void reset_control()

This function resets the controller’s internal variables to their initial values, such as an integrator state. The exact form of this function will vary depending on the controller.

Example Control Codes

Examples and baseline code can be found in the folder Software/FlightCode/control/; a brief explanation about these implementations is given as follows:

empty_control.c: Provides a basic layout for the control code. The position where the control code needs to be written is indicated in comments.

baseline_control.c: Baseline PI controller which uses angles and angular rates feedback. It has an integral action on the position errors (angles) and an integrator anti-windup strategy to handle actuator saturations. Has a simple yaw damper for the directional axis.

lqr_control.c: LQR controller that uses angles and angular rates feedback and an integrator anti-windup strategy to handle actuator saturation. Lateral-direction axes (aileron-rudder) are coupled.

Compiling a Control Law Code

The compilation process for software in the loop and hardware are explained in the following sections.

SIL simulation

Requirements: Matlab - mex utility linked to a C compiler. Type `mex -setup` in the Matlab command line.

The C code utilized by the SIL simulation is compiled from the Matlab environment by running the setup.m script where the path and name of the desired controller code must be specified in the variable control_code_path. This variable is a string with the file location and name. Thus, a desired control law can be selected from this script without changing either the code of the flight computer or the SIL simulation C code. The setup.m file is located in the SIL simulation folder : Simulation/SIL_Sim/. Errors and warnings of the control code implementation are displayed in the Matlab console after running the setup.m script.

The control code must be in the same folder used for the flight computer code (Software/FlightCode/control/), so the same file will be used for both hardware implementation and software in the loop simulation.

The file control_manager_SIL.c located in the SIL_Sim folder links the inputs and outputs from the Simulink model with the control functions. This file includes the header file control_law_interface.h which serves as interface connecting the function definitions and the control implementation specified in the compilation process. Note that the control file declaration does not have to be changed in control_manager_SIL.c. The source code and header files included in control_manager_SIL.c are located in the Software/FlightCode/ directory and are used for the hardware implementation as well.

Hardware implementation:

To compile the Flight computer software the make tool is used. A makefile written for the UAV platform compiles the source code. To change the selected flight controller, the makefile must be edited to specify which controller is compiled. Currently this is Line 40 of the makefile, which is as follows:

```
OBJ = L_avionics.c control/lqr_control.c
```

With this code, the controller in “control/lqr_control.c” will be compiled. The software requirements for compiling the flight computer code for the MPC5200b-tiny system are explained in the PIL simulation guide.