AHRS Project proposal

Anubhav Vandkar(av3336) | Jaxson Natalini Robinson (jnr2154) |
Zongyang Li (zl3621) | Darshan Ramakrishnaiah (dr3412)

Spring 2026 | CSEE4840

The following project proposes an hardware accelerator for a basic AHRS (Attitude heading and reference system) designed to process Inertial measurement unit data, consisting of the accelerometer, gyroscope and magnetometer data to compute roll, pitch, and yaw of an aerial vehicle and animate a basic version of a primary flight display on a VGA display as shown in the figure below.

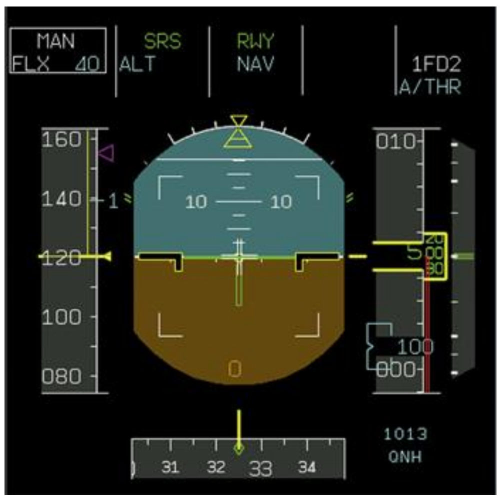

Figure1: Representational figure of a primary flight display

We will generate IMU (Inertial Measurement Unit) data entirely in software by modeling the gyroscope, accelerometer, and magnetometer in C on the HPS. This allows us to inject controlled noise, drift, and offsets while maintaining full control over the data characteristics. The raw simulated sensor values will then be written onto the Avalon bus, mimicking a real IMU data stream without requiring physical hardware sensors. Specific definitions and some corresponding errors they generally read are given below:

- **Gyroscope**: measures rotational rate around each axis in degrees per second, It is subject to a bias error which is a tiny accumulating error in its angle. Over time, it becomes noticeable and looks to be of constant nature.
- **Accelerometer**: measures the acceleration of the aircraft in all three axes in meters per second. It is almost always non-zero (due to earth's gravity on the z-axis). Subject to bias errors due to temperature and some high-frequency disturbance due to the engine.
- **Magnetometer**: measures the direction of local magnetic fields, from which can estimate the yaw of the aerial vehicle with respect to true north can vary based on nearby false magnetic fields.

All three of these sensors are also subject to some white gaussian noise. At each time frame, we will hence generate 9 values for each sensor.

The hardware accelerator will then take these raw values from the Avalon bus, and estimate the Euler angles (Roll, Pitch and Yaw). For the estimation algorithm we are currently looking into two ways

1. **Complementary filter**: Fast, less hardware resources, less accurate

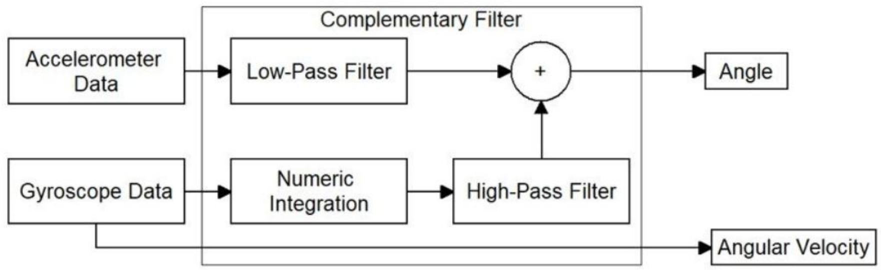

Figure 2 Complementary filter-based estimation

2. **Kalman filter**: Relatively slow, More hardware resources, more accurate

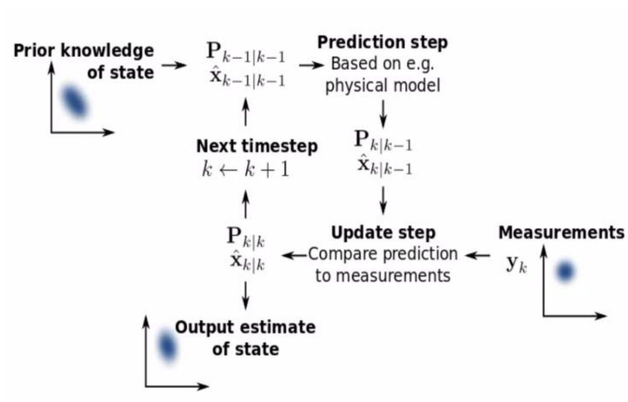

Figure 3: Kalman filter based estimation

The system renders a horizon line on the VGA display that rotates left or right about its center based on the estimated roll angle, visually indicating the vehicle's bank. The same horizon line also shifts vertically according to the estimated pitch angle, moving against a pitch ladder marked with pitch angles to show the aircraft's nose-up or nose-down attitude. Additionally, a horizontal heading strip with angle markings is displayed to represent yaw, with a moving cursor indicating the current yaw value, as illustrated in Figure 2.

The VGA signal generation will follow the standard 640×480 at 60Hz timing. Display state will be updated by the HPS as IMU frames are consumed from the shared memory region, translating sensor values into the corresponding visual elements on screen.

Major tasks:

- *Design Document*
  Define the complete system architecture by creating detailed block diagrams, designing the memory-mapped register interface, selecting fixed-point formats, determining data rates and timing constraints, and specifying hardware software integration
- *Implement Software IMU Data Generator*
  - Generate accelerometer + gyroscope samples that mimic real IMU behavior.
  - Include drift, noise, and realistic motion patterns (tilt, oscillation, rotation).
  - Stream this data to the FPGA at a fixed sample rate
- *Build Hardware Kalman/ complementary Filter on FPGA*
  - Fuse the software-generated gyro + accel data to compute roll and pitch.
  - Implement fixed-point arithmetic and pipelining for real-time performance.
  - Output stable angle estimates for every sample.
- *Implement VGA Rendering Engine for Attitude Indicator*
  - Convert roll/pitch angles into graphical elements.
  - Render the horizon line, pitch ladder, bank angle ticks, and center marker.
  - Maintain 640×480 @ 60 Hz with smooth motion.
- *Integrate Data Pipeline (Software → Hardware → VGA)*
  - Send IMU samples from software to the hardware filter via memory-mapped registers or FIFO.
  - Feed the filter's roll/pitch outputs into the VGA renderer.
  - Ensure synchronized updates so the display reflects the latest angles.
- *Develop Control & Debug Software*
  - Write C code to configure the pipeline, send IMU samples, and read filter outputs.
  - Add test modes (constant angle, sweep, step response, noise injection).
  - Log angle outputs for verification.
- *System Testing & Performance Evaluation*
  - Validate angle accuracy using known synthetic motion patterns.
  - Measure latency from IMU sample → rendered horizon.
  - Verify stable 60 FPS VGA output with no tearing or jitter.