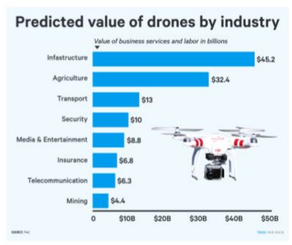
Understanding the Software and Hardware Stacks of a General-Purpose Cognitive Drone

Sam Jijina, Adriana Amyette, Nima Shoghi, Ramyad Hadidi, Hyesoon Kim
Georgia Institute of Technology

Motivation

- Commercial drone industry will reach 805,000 in sales in 2021, a CAGR of 51% [1]
- Increasing use cases of drones from surveying land to emergency services and national security
- Open-source flight stack to promote innovation through collaboration
- Characterizing underlying architecture and flight stack to achieve high reliability, safety, and performance



Intelligence, Business Insider. "Commercial Unmanned Aerial Vehicle (UAV) Market Analysis – Industry Trends, Forecasts and Companies." Business Insider, Business Insider, 10 Feb. 2020, www.businessinsider.com/commercial-uav-market-analysis.

Applications of Drone Technology

- Aerial photography
- Agriculture
- Defense
- Emergency services
- Geographic mapping
- Personal hobby
- Search and rescue
- Shipping

and many more...



Current Technological Shortcomings

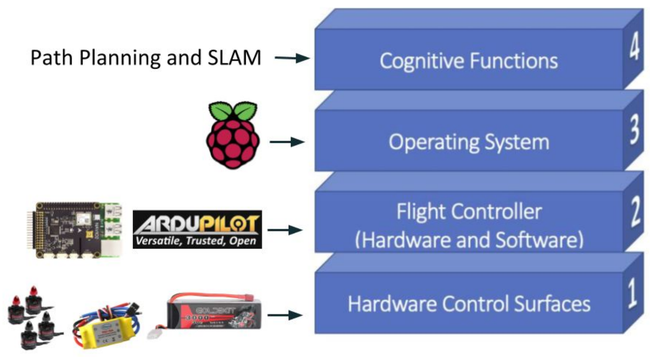
- Most drone flight stacks are not open-source
  - Many popular models are closed due to market forces
- No access to the autopilot code base
  - Since manufacturers are using custom chips
- Weight carrying capacity limitations
  - General hobbyist drones not designed to carry large payloads
- Very difficult to alter hardware due to custom PCBs
  - Difficult to add components and sensors
- Not cost effective for various types of research projects
  - Drones which meet the criteria are too expensive for many research projects

Design Choices

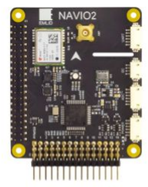
- Open-source drones already exist
  - CrazyFlie [12]
  - PlutoX [13]
- BUT…
  - Limited weight carrying capacity
  - Limited flight time due to battery capacity
  - Microcontroller performance limitation
- We set up a development platform to allow for more sensors and devices to be added in the future
  - Camera for SLAM [14] or OpenCV [15]
  - LIDAR
- So we decided to use a frame kit to build a custom drone



Architecture of Drone Flight Stack

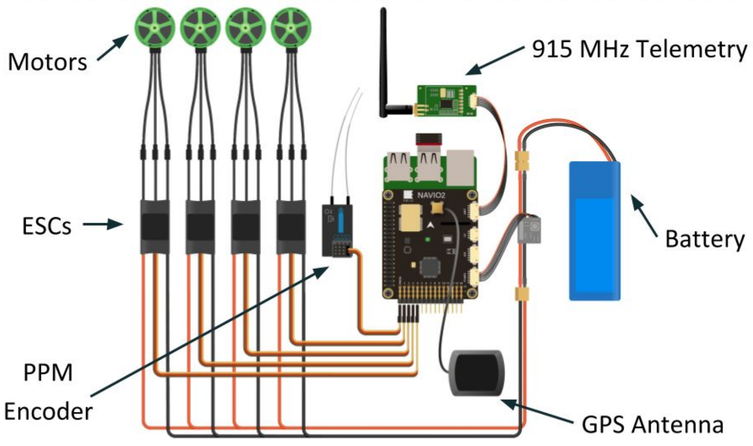


Hardware Overview




- Raspberry Pi 3 Model B +
- Emlid Navio2 HAT for Pi
- ESCs (Electronic Speed Control)
- 935KV motors
- GPS/GLONASS receiver
- 3000 mAh 3S LiPo battery
- 915 MHz Ground-to-Air Telemetry communication
- Features of Navio2:
  - Dual IMU
  - Triple redundant power supply
  - High resolution barometer

Navio2 HAT Setup



Flight Controller - GCS

- Two types of software:
  - Ground Control Station
  - Autopilot firmware
- Ground Control Station (GCS)
  - Executes from laptop
  - Real-time data (altitude, speed, location, battery)
  - Telemetry communication
  - Remote commands to override erroneous behavior
- Most popular GCS is MissionPlanner [20]
  - Open-source
  - Actively maintained

Flight Controller - Autopilot

- ArduCopter (fork of ArduPilot) [21]
- MAVLink [22]
  - **M**icro **A**ir **V**ehicle **Link**
  - Data packet protocol for standardized communication between drones
  - DroneKit API [23]
  - Python and C++ APIs to issue flight commands easily
  - Converts commands to MAVLink protocol

Drone Operating System

- Real Time Operating System (RTOS)
  - Minimal, if any, latency in response
  - Kernel tasks can be pre-empted
  - Most not open-source
- We had the choice of using Linux [25] or Robot Operating System (ROS) [24]
  - ROS is a specialized OS for robotics
  - Due to community support and documentation, we chose Linux
- Setting up Linux for drone hardware
  - Built Linux with PREEMPT_RT patch to achieve nearly identical performance to a RTOS
  - PREEMPT_RT patch alters kernel scheduler to preempt all processes
  - Used for incoming MAVLink packets

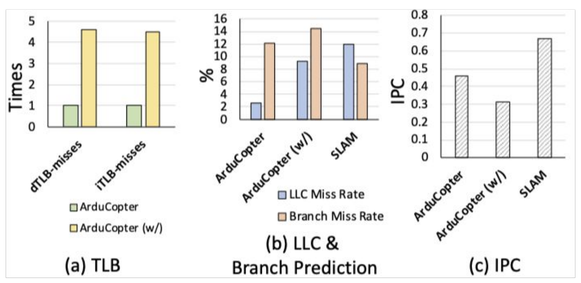
Firmware Switching

- Commercially available drones from Boeing, DJI, and SkyDio are capable of changing their missions mid-flight [26]
  - They are unable to completely shut down their autopilot binary and load a different one since access to their autopilot architecture is limited
- Achieving this ability would open up the field of general purpose drones to the mass consumer and industrial markets

Flight Testing Methods

- Manual Flying and Testing
  - Weather dependent
  - Battery limitation
  - Approval Process
- Simulations
  - Software in the Loop (SITL)
  - Hardware in the Loop (HITL)
- SITL simulations used to test flight code
  - ArduCopter natively compiles for SITL simulation
  - Less system resource heavy
- Microsoft AirSim [27] for HITL simulation
  - Open-source
  - System resource heavy
  - Provides environment simulation (neighborhood, city)

Performance Metric Measurements



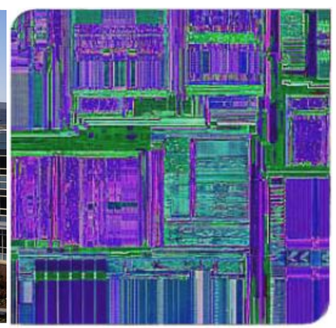
(a) ArduCopter with SLAM dTLBand iTLB misses normalized to only execute ArduCopter. (b) LLC and branch prediction miss rates for ArduCopter, ArduCopter with SLAM, and SLAM.(c) IPC for ArduCopter, ArduCopter with SLAM, and SLAM.

TABLE I
POWER CONSUMPTION MEASUREMENTS

Experiment	ArduCopter	ArduCopter +SLAM	ArduCopter +SLAM +Fly
Avg. Power Consumption	3.39 W	4.05 W	4.56 W
Peak Value	4.21 W	4.18 W	5.40 W
σ/SD	0.13	0.09	0.34

Conclusions & Next Steps

- Conclusion:
  - Running additional workloads would decrease drone flight time, but increase response time from the autopilot software, underscoring the need for better optimizations
- Next steps:
  - Use the obtained metrics to further develop the autonomous drone
  - Conduct further research into optimizing the flight-stack and drone collaboration for deep learning tasks [11], [12]
  - Continue building a baseline model for a general-purpose drone capable of switching between firmware versions and changing missions mid-flight

NSF CSR 1815047