EE 183DB Midterm Presentation

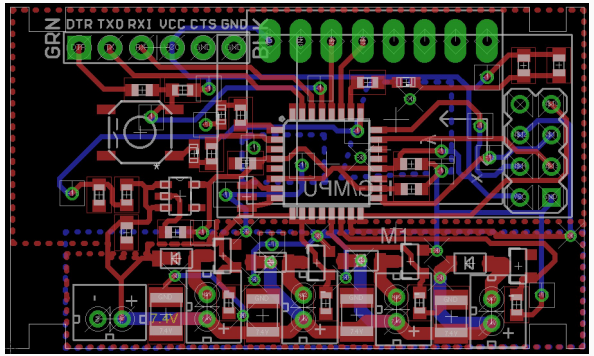
Team Daphne
Jerry Duan
Kevin Fan
Likai Wei

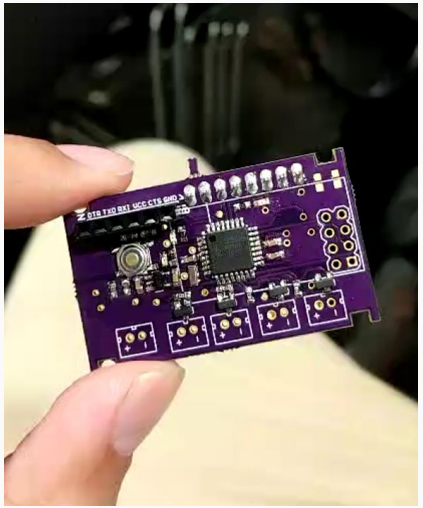
Reevaluation of our project

- We are building a quadcopter that works underwater
- For first half of this quarter, we focused on the small quadcopter plan and prove it is practical and doable.
- The key challenges:
  - Correctly assembled the PCB
  - Wireless control and interfacing of the quadcopter
  - Tuning control parameters for control algorithm
  - Waterproofing and testing

What’s done during week 1-4

- Learn SMD soldering
- Finished PCB soldering and tested it for function



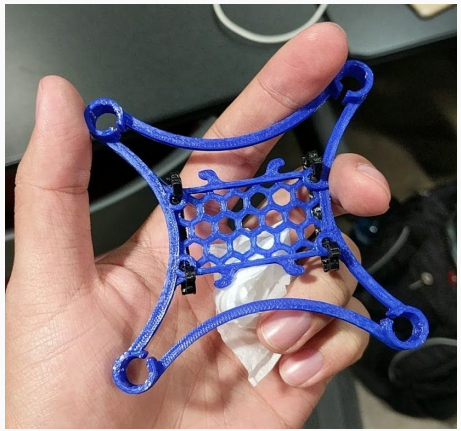


What’s done during week 1-4

- Tested micromotor works underwater
- Thus, we only have to worry about waterproofing for PCB and connections

What’s done during week 1-4

3D printed quadcopter frame for holding PCB, battery and motors.



What’s done during week 1-4

- Finished writing control code
- The controller sends out setpoints
- MCU reads IMU input and compute PID
- Adjust motor speed accordingly

```
getProcessVariable();

if(rxValues.roll == 255 && rxValues.pitch == 255)
{
  idle_state = false;
  delay(1000);
}


if(!idle_state)
{
  getProcessVariable();
  gyro_yaw_input = gyro_yaw;
  gyro_pitch_input = gyro_pitch;
  gyro_roll_input = gyro_roll;

  pid_yaw_setpoint = rxValues.yaw;
  pid_pitch_setpoint = rxValues.pitch;
  pid_roll_setpoint = rxValues.roll;

  calc_PID();

  pwm_fl = rxValues.throttle + pid_output_pitch - pid_output_roll + pid_output_yaw;
  pwm_fr = rxValues.throttle + pid_output_pitch + pid_output_roll - pid_output_yaw;
  pwm_rl = rxValues.throttle - pid_output_pitch - pid_output_roll - pid_output_yaw;
  pwm_rr = rxValues.throttle - pid_output_pitch + pid_output_roll + pid_output_yaw;

  if(pwm_fl < minPWM) pwm_fl = minPWM;
  if(pwm_fr < minPWM) pwm_fr = minPWM;
  if(pwm_rl < minPWM) pwm_rl = minPWM;
  if(pwm_rr < minPWM) pwm_rr = minPWM;
```

Conformal Coating

Techspray Urethane Conformal Coating

- The controller sends out setpoints
- Free of acidic erosion

iOS Remote Controller

Blynk iOS library

- Controls IoT with iPhone
- Compatible with Arduino and Raspberry Pi IDE
- Well-documented on Github

Schedule Reevaluation

Time	Status	Metrics
Week 1-4		Quadcopter fully functional in air
Week 4-7		Waterproof implementation & Underwater Control finished
Week 7-10		Testing

Schedule Reevaluation

- Currently behind the original schedule
- However, waterproof implementation doesn’t need 3 weeks (1 week tops)
- Need extra time to prepare for backup components

Schedule for Remaining of Quarter

- Week 5-7: Test Quadcopter, Prepare for backup components
- Test Routine
  - Stable Hovering
  - Move up/down, forward/backward
  - Bend left/right
  - Rotate clockwise/counterclockwise
- Week 8: Waterproofing
- Week 8-10: Underwater testing

Milestones

- We should have a fully working quadcopter and a backup by the end of Week 7
- Finish the waterproofing and start testing underwater by Week 8
- Finish implementing the underwater quadcopter by Week 10

Project Management

Task	Owner	Status	Notes
Camera Streaming Configuration	*Likai*		Put aside as we're focusing on the small quadcopter right now
Chassis Building	*Kevin* *Likai* *Jerry*	Complete	
Circuit Soldering	*Likai*	Complete	
Coding and Adjustment	Kevin *Jerry*	In Progress	Finished writing code, still need tuning
Waterproof Coating	Kevin		
Underwater Testing	*Kevin* *Likai* *Jerry*		
Air-to-water Landing Test	*Kevin* *Likai* *Jerry*		

Challenges

- Will the PID control parameters be different underwater? Need to test underwater and find out (the answer is most definitely yes)
- 2.4GHz signal propagation underwater
- Waterproofing for battery and USB connections
- Weight and frame adjustment