Electronics for a Robotic Quadrotor-Top Inverted Pendulum (Q-TIP)

25th of June, 2019

Jee Yong Park

Year 4, Electrical and Electronic Engineering (MEng)

Supervisor: **Dr. Eric C. Kerrigan**

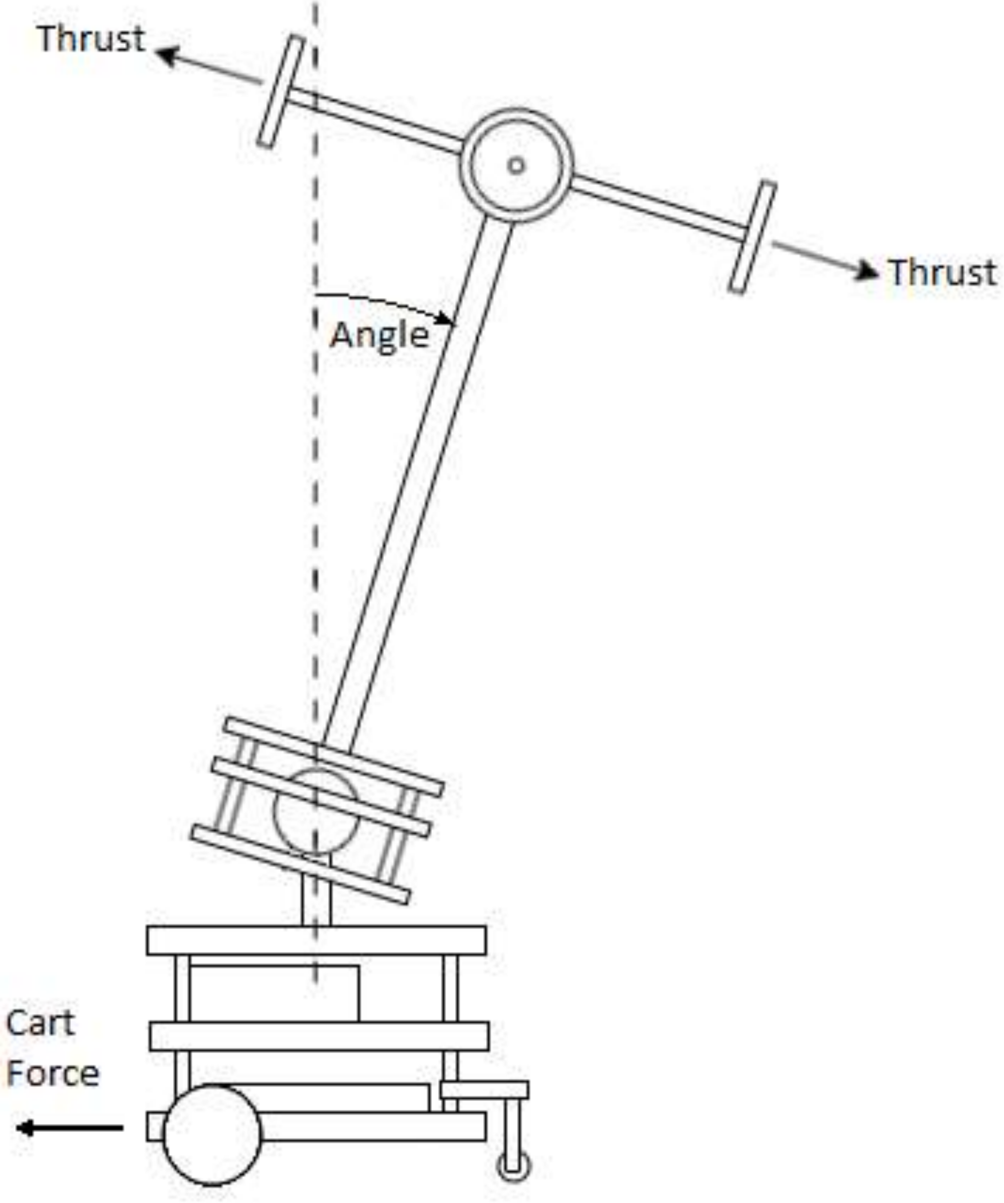
Second Marker: **Dr. Àdria Junyent-Ferré**

1.1. QTIP: overview

- Inverted pendulum, actuated by quadrotors
  - Mounted on cart via ball joint
  - 3DOF (yaw, pitch, roll)
- Cart driven by DC motors
  - 3DOF (translation in x, y, and yaw rotation)

- Good platform to test/demonstrate controllers for nonlinear multivariable system
  - Students learning robust linear control
- *"It's cool."*

– Dr. E.C. Kerrigan



1.2. Preceding work

- Based on previous M.Sc. project by Bell

- Controlled by Arduino Mega
- Stable PID controller
- Cart driver (Bluetooth)

- Messy, dangerous, and difficult to fix!

2. Project objectives

- Address the problems that exist in the previous version of QTIP
  - Mechanical and electrical safety
  - Low hardware modularity – difficult to repair / modify
  - Robustness to EM noise
  - Bugs in source code
  - Source code comprehensibility
- Derive system dynamics
- Design and implement effective linear controllers

3.1. Hardware improvements

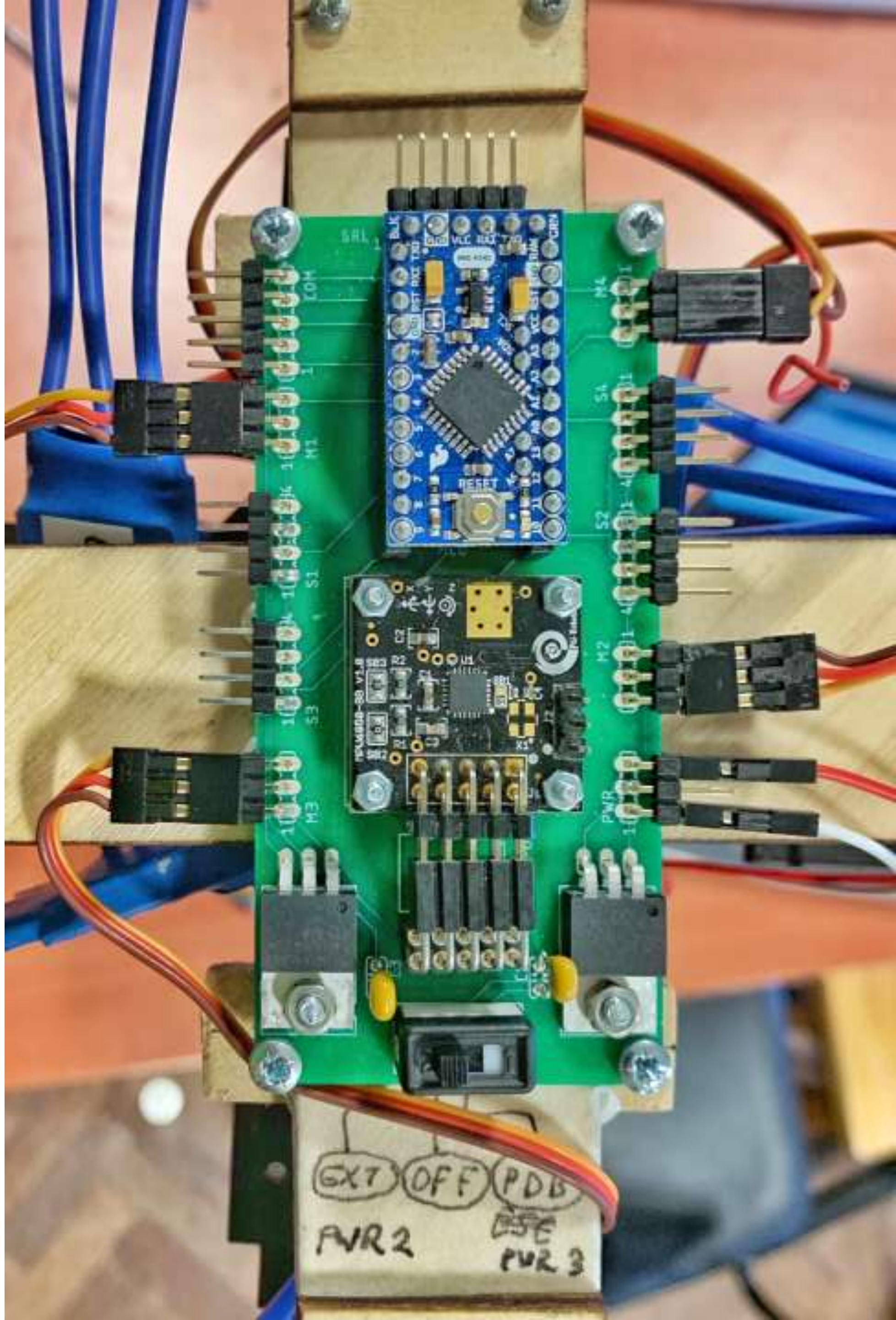
- **Power supply** adjustment
- **New MCU** to control pendulum actuation separately
- **Custom PCB** to mount MCU and MPU

3.1.1. Hardware: Power supply adjustments

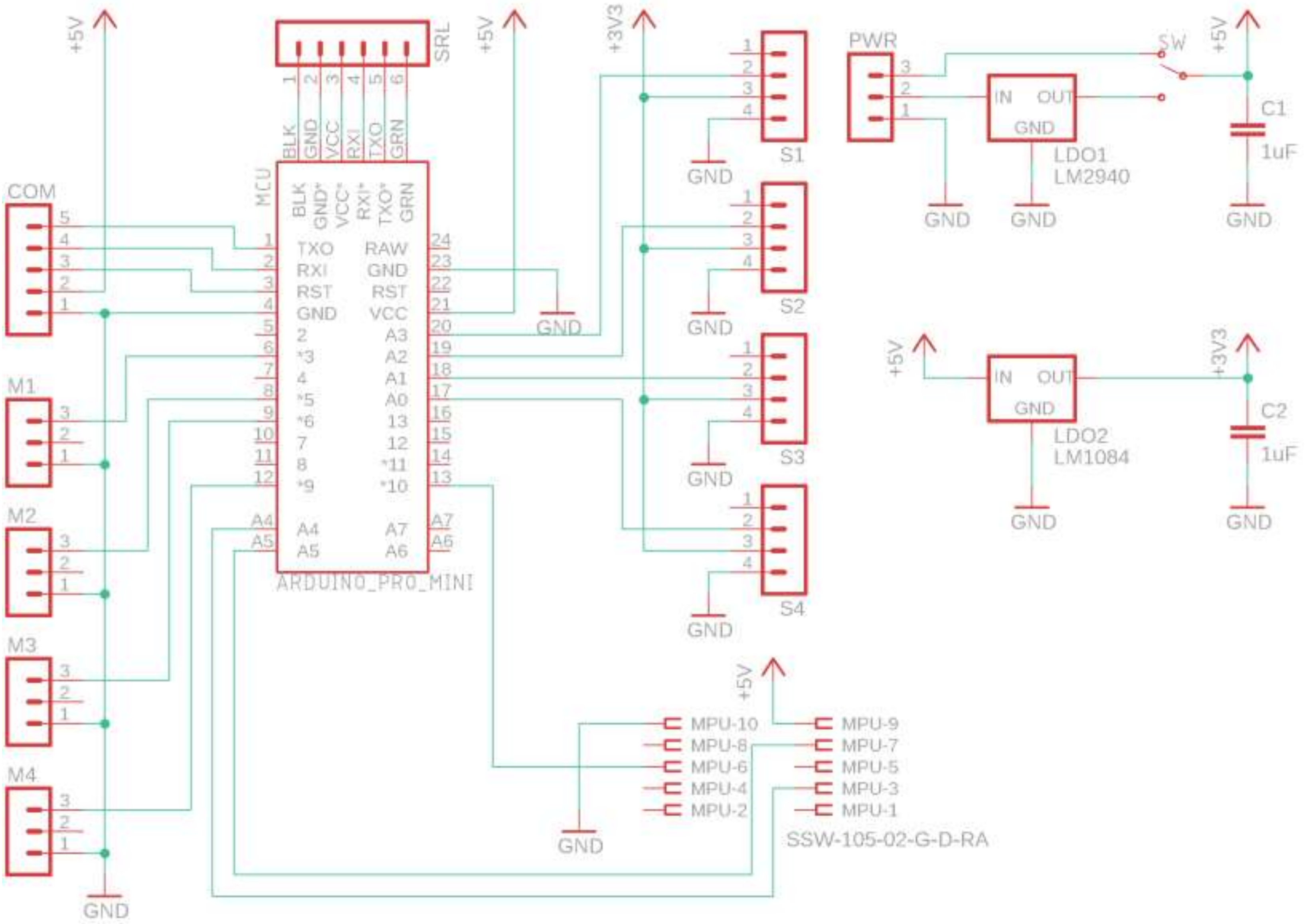
- Added **another PDB** on pendulum
  - Less power cables running along pendulum
  - Shorter wires for power supply to upper components
- Higher gauge **power cables** between PDBs
- Clip connectors: easier fix/mod

3.1.2. Hardware: New MCU & mount PCB

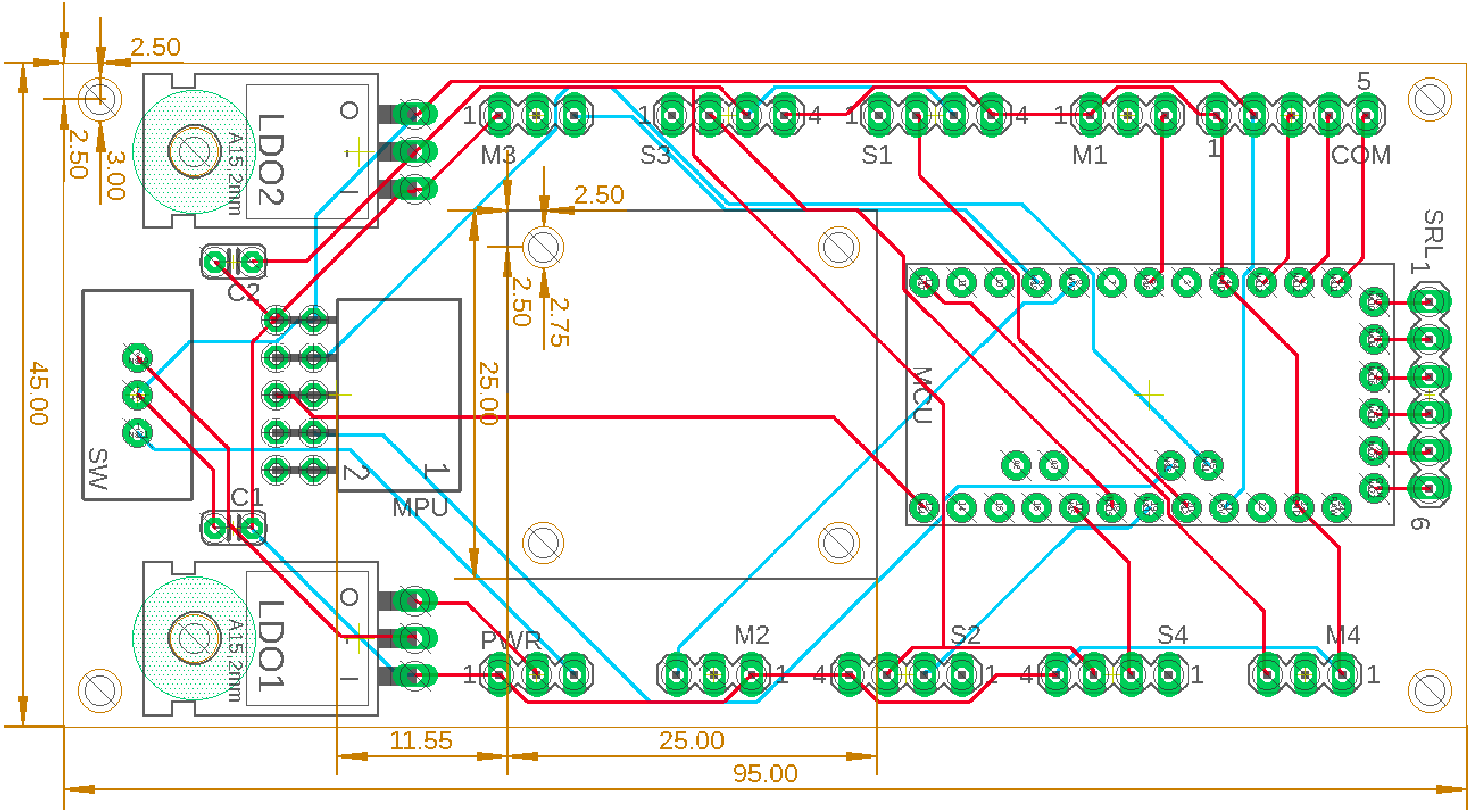
- Arduino Pro Mini 328 5V/16MHz
  - Small and light
  - Enough PWM, digital, ADC, serial I/O
  - Same clock and VCC with Arduino Mega
- MCU & MPU mount PCB
  - Additional **power management** for MCU and MPU
    - Switch between power sources
    - Voltage regulator for MCU, MPU (5V), and IR sensors (3.3V)
    - Capacitive DC coupling: EM noise reduction
  - **Header pin** connections
    - Robust external connection
    - Easy rewiring or parts replacement
  - No more loose wires between MCU, MPU, ESCs



3.1.2. Hardware: MCU & MPU mount PCB



3.1.2. Hardware: MCU & MPU mount PCB



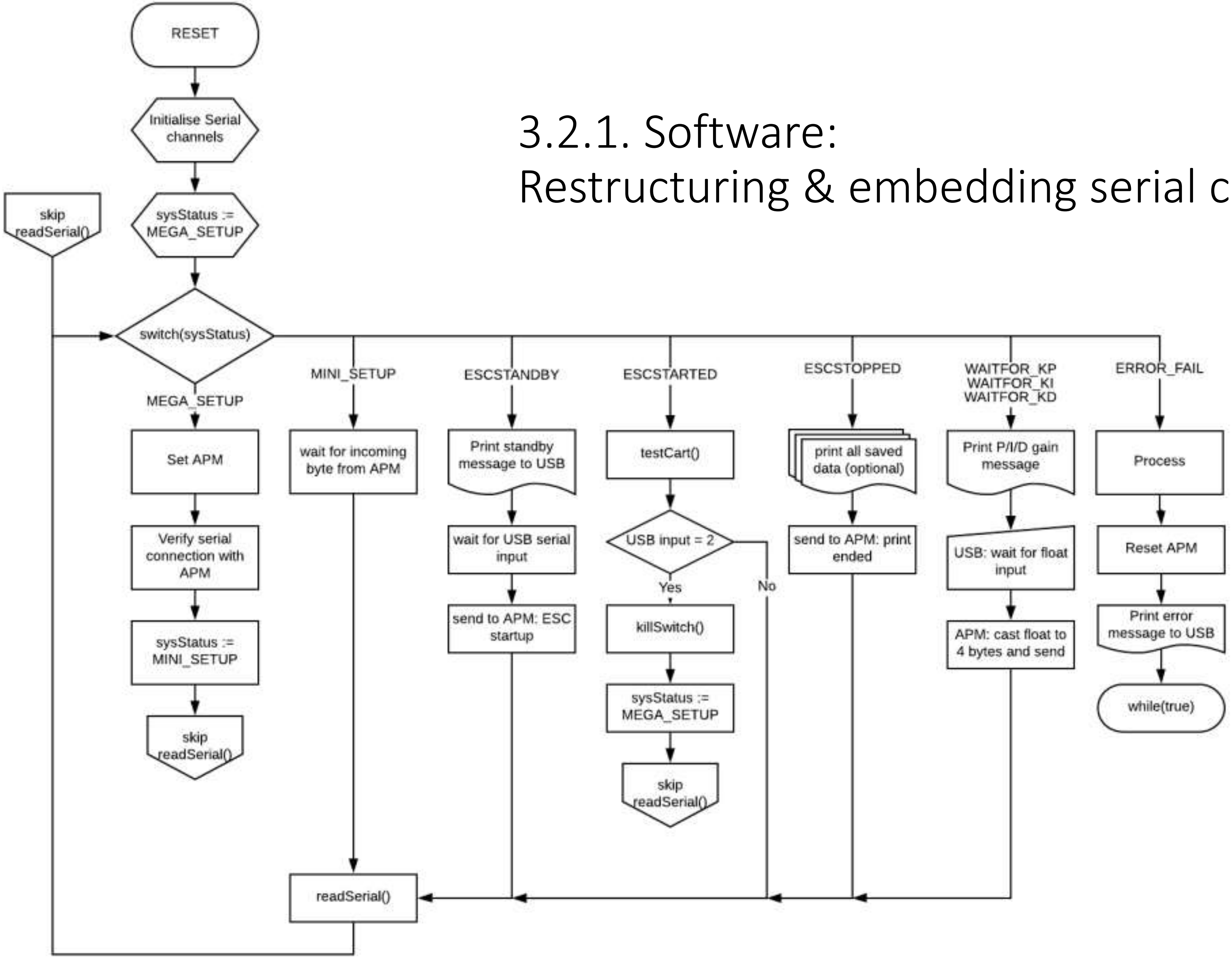
3.1.3. Hardware: Overall changes

- 21 → 5 wires along the pendulum arm
- Less likely to be electrocuted
- Less loose wires hanging around the top
- Can power up MCU and MPU only without powering ESCs

3.2. Software improvements

- Optimisation for Arduino Pro Mini
    - Remove unused variables and lines (memory/performance)
- Bug fixes
    - Incorrect DMP information fetch
    - Incorrect DMP offset calibration source code
- Consistent sampling time and control delay
    - Added while statement to wait until designated Ts has passed since last iteration
- Implement serial communications between two MCUs
    - Exchange 1-byte flags to communicate operation status
    - One USB connection to PC from Arduino Mega (1 less cable)
    - Send and receive 4B float and 2B short int data (roll, pitch, dt)
- Soft kill-switch for emergency
    - Mega can reset Pro Mini via digital pin connection to reset pin

3.2.1. Software:
Restructuring & embedding serial comms

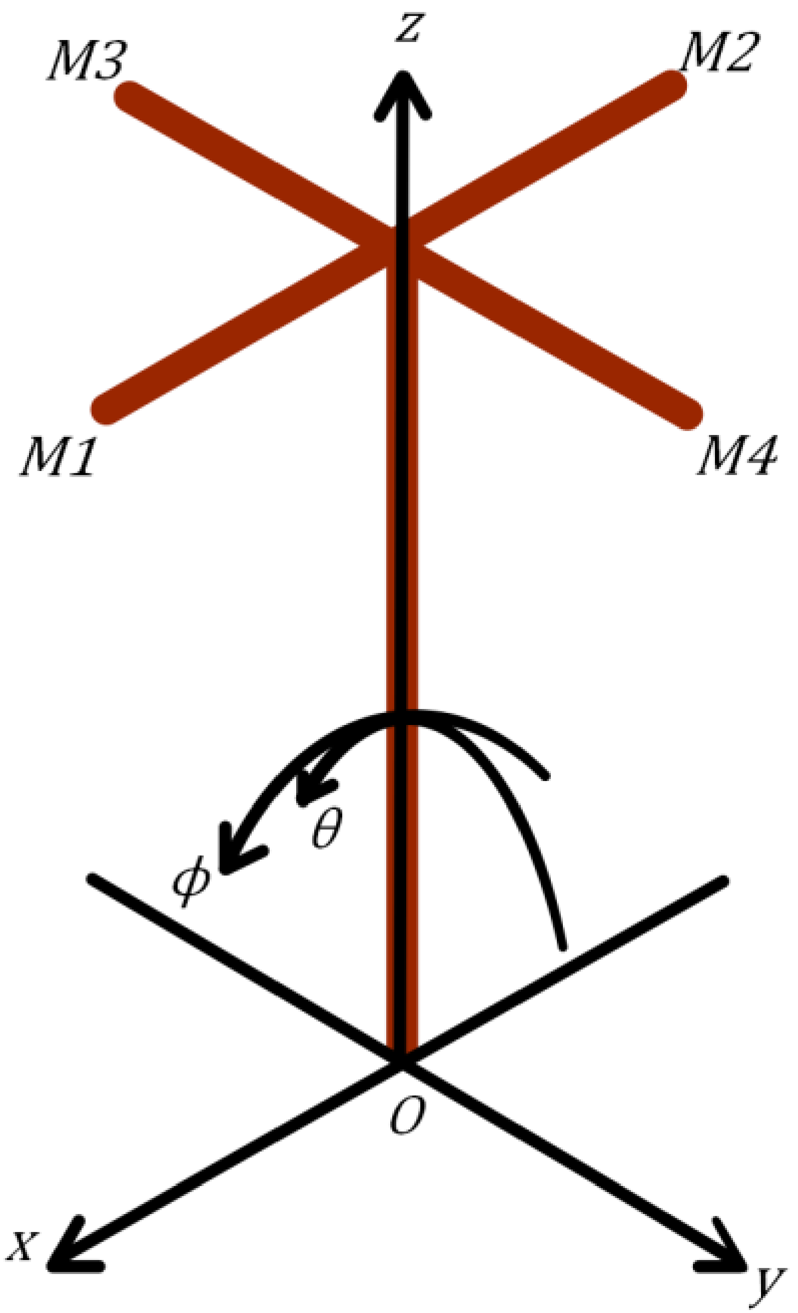


4. Modelling and controller design

- State-space system dynamics
  - Derive
  - Linearise
  - Discretise
- Design effective linear feedback controllers
  - PID
  - LQR
  - H_∞ optimal
- Verify controller performance

4.1.1. Model derivation: Frame of reference

- Consider pendulum as sole 2DOF system
- Consider base of pendulum (ball joint) as O
- x-axis along M1 and M2
- y-axis along M3 and M4
- z-axis upwards along the pendulum
- Roll (ϕ) around y-axis
- Pitch (θ) around x-axis
- Total thrust $f_x = f_2 - f_1$ along x-axis
- Total thrust $f_y = f_3 - f_4$ along y-axis



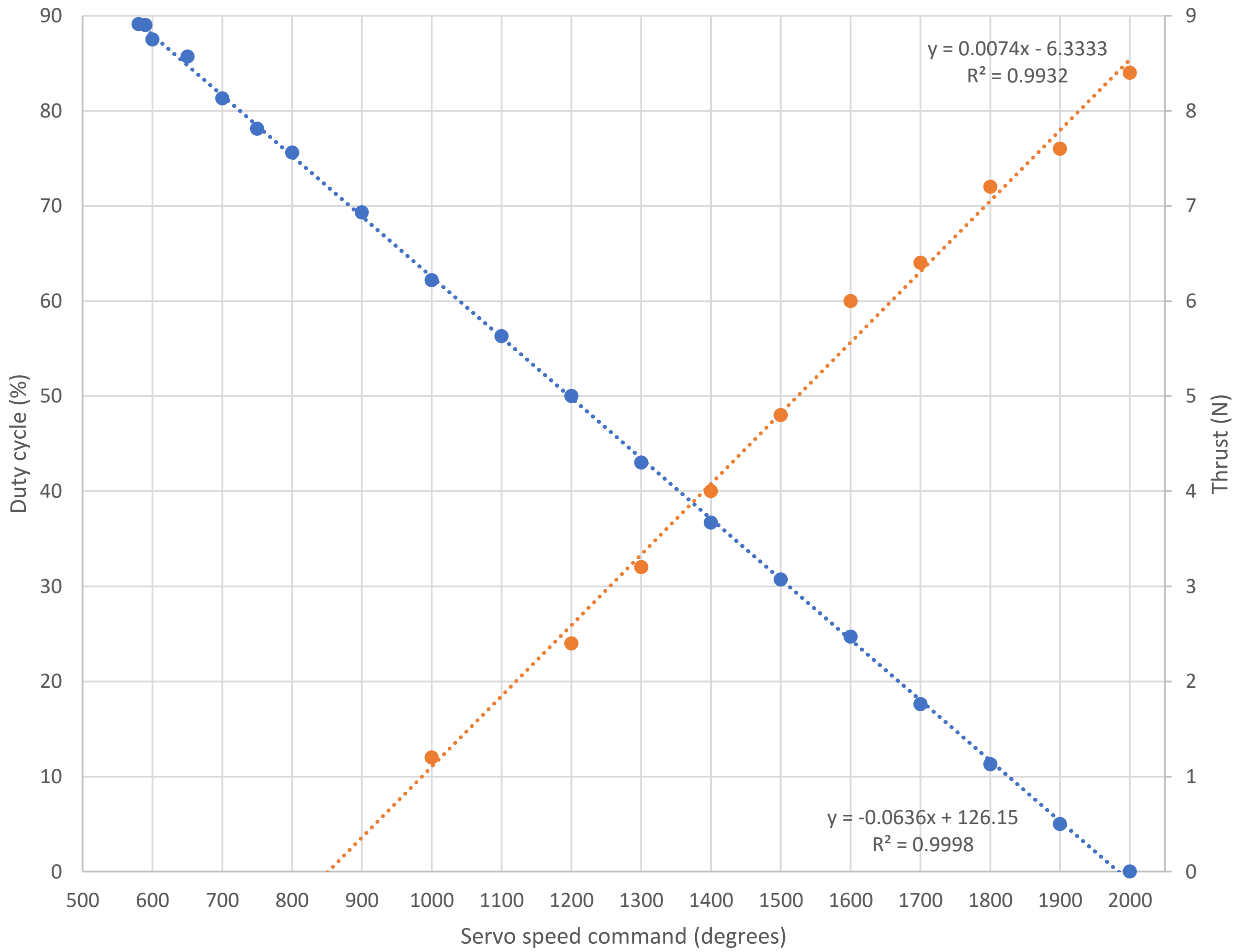
4.1.2. Model derivation: System prameters

- Assumptions:
  1. No mechanical friction at ball joint
  2. Thrust ∝
     a) PWM signal
     b) Arduino Servo command given to ESC
  3. No drag from rotors of orthogonal directions

Parameters	Symbol	Value	Unit
Mass of pendulum	m	1.066	kg
Distance to centre of gravity	r	0.35	m
Length of pendulum	l	0.45	m
Thrust constant*	K	0.0074	N
Gravitational constant	g	9.81	m/s^2

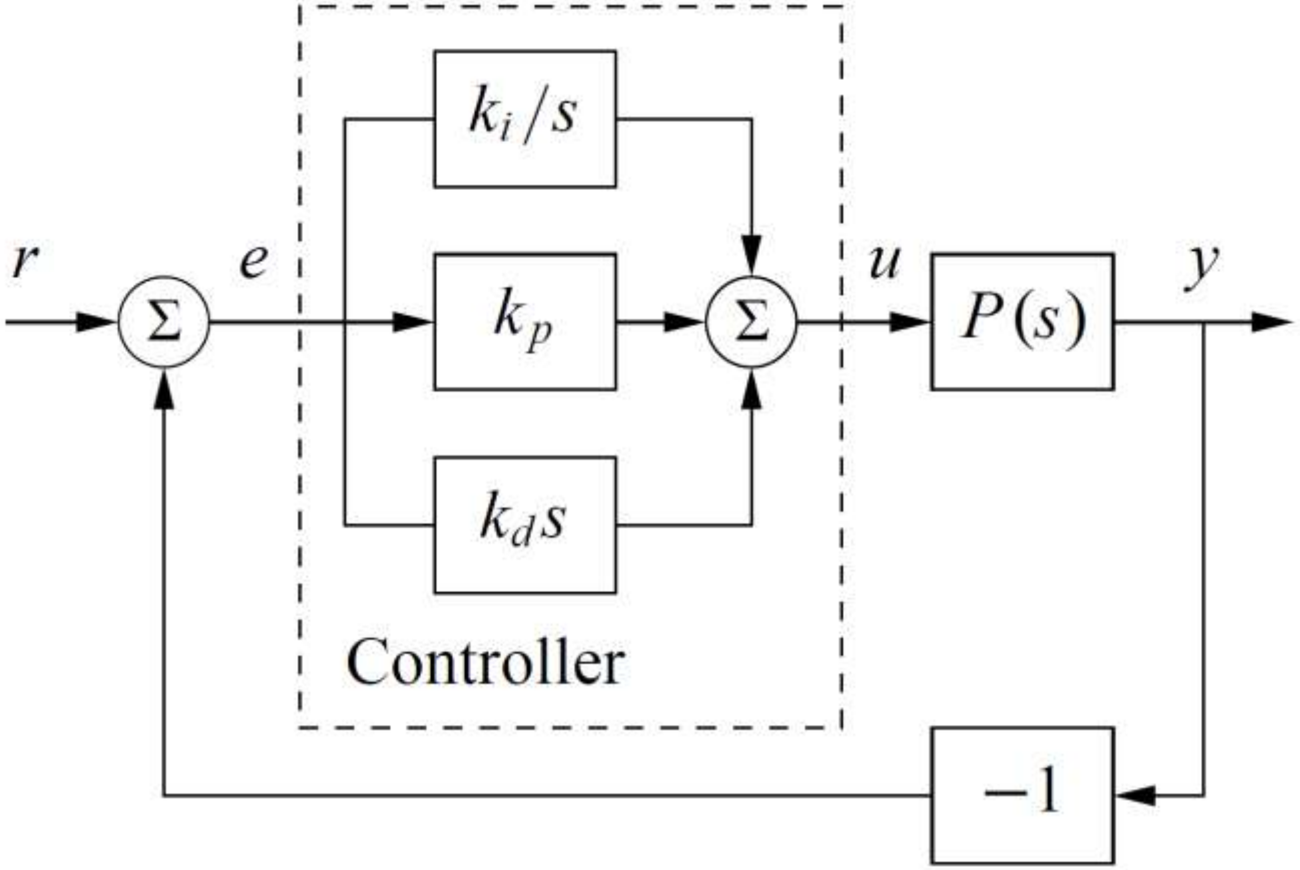
4.1.2. Model derivation: System parameters (individual rotor thrust)

- ESC control
  - not by PWM (analogueWrite)
  - by Arduino Servo angle commands (Servo.write)
- Measured force: effective force provided by individual rotor thrust at centre of gravity orthogonal to pendulum arm
- Trendlines
  - Rotor thrust ∝ servo command
  - Blue: PWM duty cycle of 3-phase power supply from ESC
  - Orange: Thrust



4.2.1. Controller design: Proportional-integral-derivative (PID)

- Apply linear gains to error, integral of error, and derivative of error
- Pros:
  - Thoroughly researched and widely implemented
  - Easy implementation
- Cons:
  - Need to balance all three gains
  - Need accurate model
  - Low robustness to uncertainties & disturbances
- Already there, but gains retuned experimentally

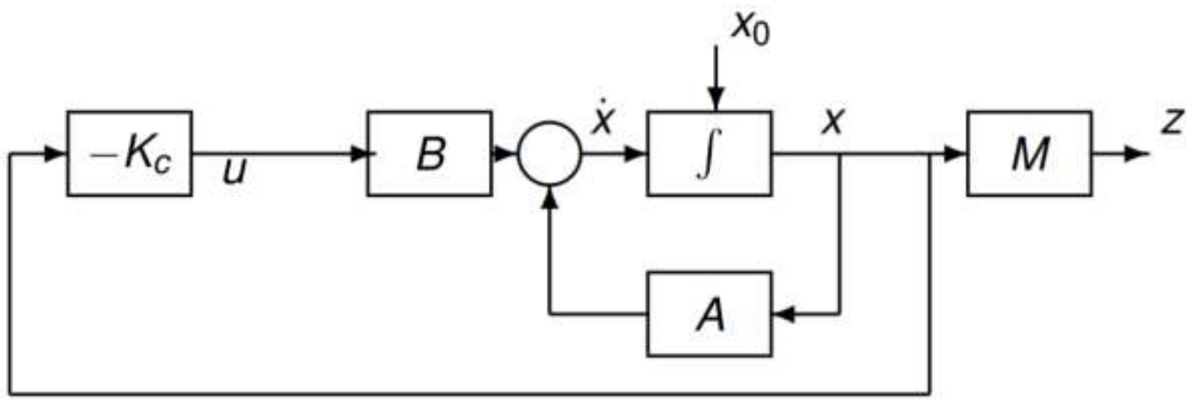


4.2.2. Controller design: Linear Quadratic Regulator (LQR)

- Find linear state feedback gain that minimises cost function:

$$J(u) = \int_0^\infty (z'Qz + u'Ru)\, dt$$

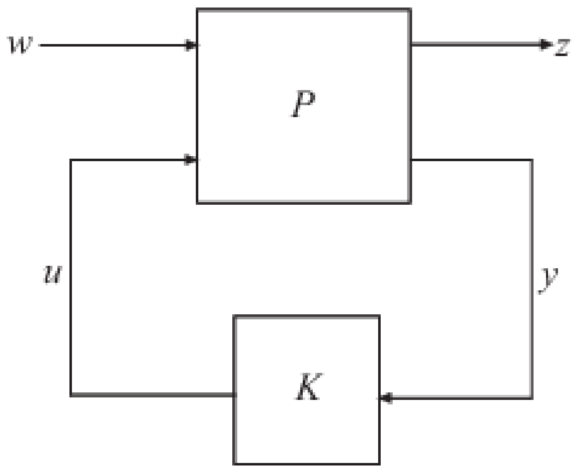
- Pros:
  - Automatically compute controller by choosing Q, R
  - Fast control input computation (single K_c)
  - Guaranteed stability
  - Guaranteed stability margins
  - Robustness against output uncertainty
- Cons:
  - Require all state variables or good estimation
  - Require good model
- MATLAB functions: dlqr



4.2.3. Controller design: H_∞ optimal control

- Find output feedback that makes optimal trade-off between
  - Sensitivity
    - Good tracking performance
  - Complementary sensitivity
    - Noise attenuation
    - Robust (multiplicative uncertainty)
  - Control effort
    - Robust (additive uncertainty)
- Pros:
  - Automatically computed (optimisation problem)
  - Weights can be chosen to minimise effects of noise, disturbance, and plant-model mismatch
- Cons:
  - Require good model
  - Need to choose relevant cost function
- MATLAB functions: augw, hinfsyn

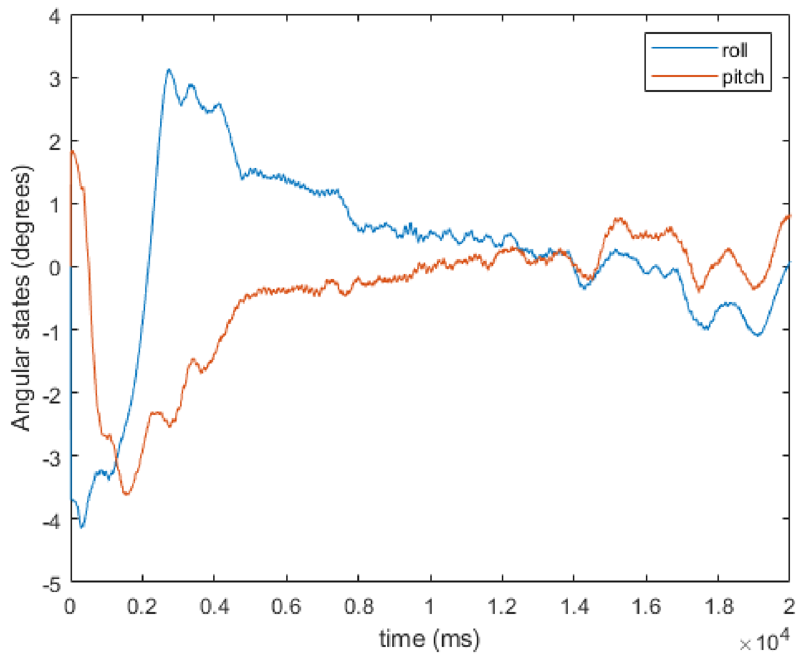
$$P(s) = \begin{bmatrix} W_1 & -W_1 G \\ 0 & W_2 \\ 0 & W_3 G \\ I & -G \end{bmatrix}$$

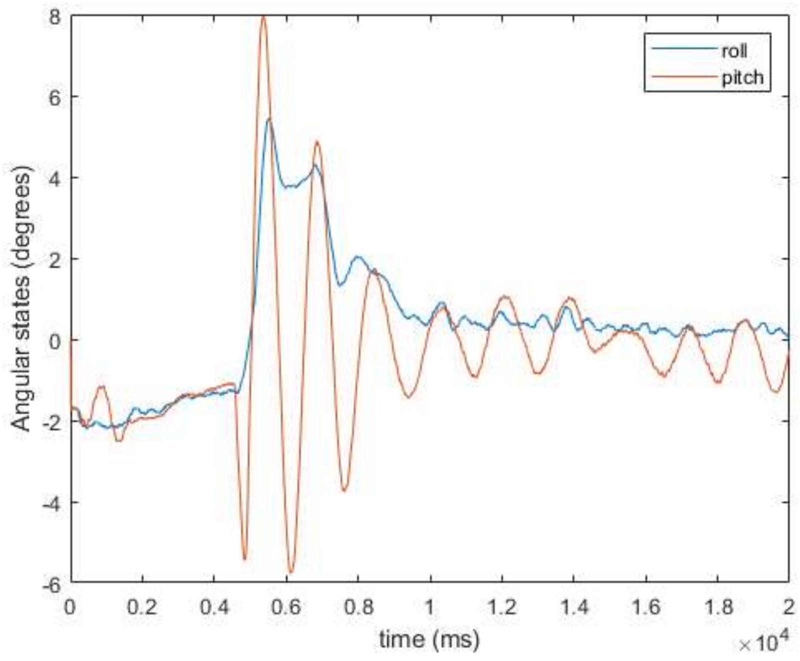


$$J := \|z\|_2^2 - \gamma^2 \|w\|_2^2 \leq 0, \ \forall w \text{ such that } \|w\|_2^2 < \infty$$

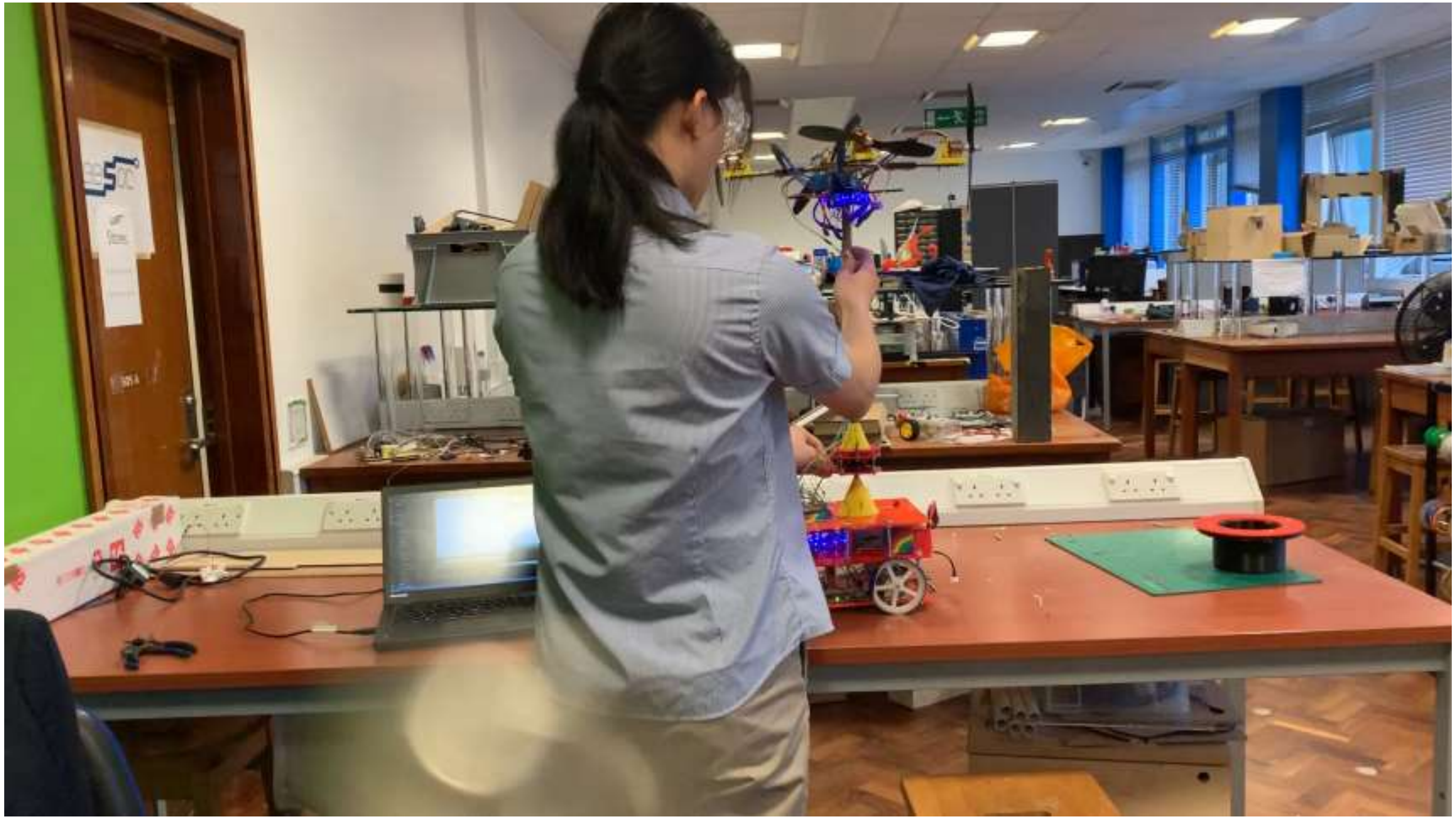
5.1 Controller performance: Results

- PID
  - Steady-state
    - Slight oscillation towards the end
    - Offset (within ±1°)
  - External disturbance injection
    - Stable (within certain amount)
    - Stabilised from peak-to-peak 14° to 1° within 6 sec
- LQR
  - No stability
- H_∞
  - No stability





5.1. Controller performance: PID (in action)



5.2. Controller performance: Evaluation

- Plant-model mismatch
  - Too many assumptions
    - Friction and drag probably have significant effect
  - Inaccurate parameter measurement
- Factors unaccounted for
  - Rotor thrust as battery drains
  - Saturation for control inputs
  - Tension from cables
- Others
  - MPU calibration: sensor offset
  - No yaw: pendulum rotates (yaw) while stabilising itself

6.1. Project outcomes & remarks

- Hardware:
  - Safe, robust, easy to fix / modify
- Software:
  - Easy to read / debug / modify
  - Successfully untethered Arduino Pro Mini from PC
- Controller performance:
  - QTIP can be a reliable platform to test computationally simple controllers
  - Needs more accurate model of the system

- Importance of developing code in small steps + many debugging
- Real machines almost never behave as you expect

6.2. Future works

- Additional modification to QTIP
  - Mechanical safety: wire fence around the rotors
  - Electrical safety: relay switch circuit on main power supply for remote power-off
  - Enhanced pose estimation: incorporate IR distance sensors into sensor fusion algorithm
  - Totally untethered system: se HC-05 Bluetooth module to remotely give user inputs to Arduino Mega
- Derivation of accurate model
- Model QTIP as 6DOF system: embed cart actuation into the controller design
- Verification by implementing control designs based on the model

Thank you