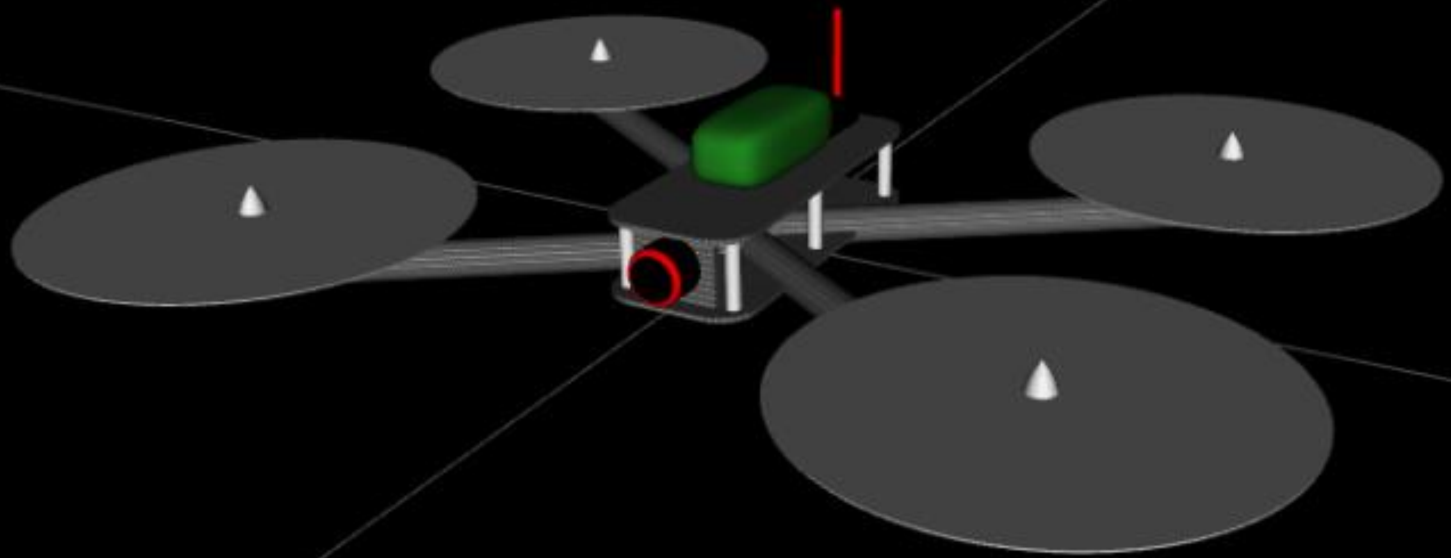
Pegasus

Multi-Rotor Simulation, Control, and Estimation

April 24th, 2018

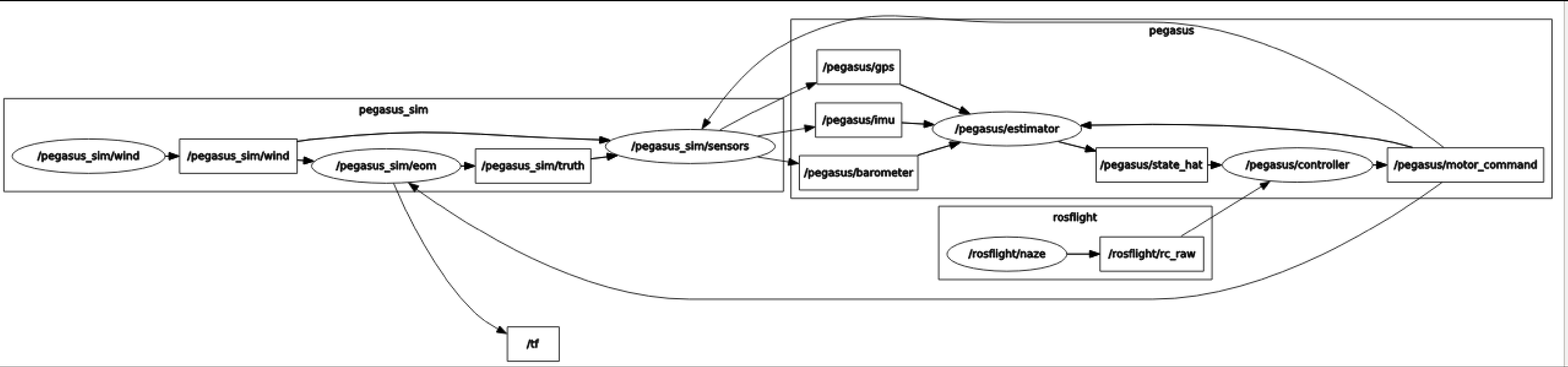


Austin Hurst

Major Developments

- C++ and ROS implementation
- Flat plate aerodynamic model
- Motor and propeller model
- Control for manual flight
- Control for autonomous flight
- Extended Kalman Filter

ROS Nodes



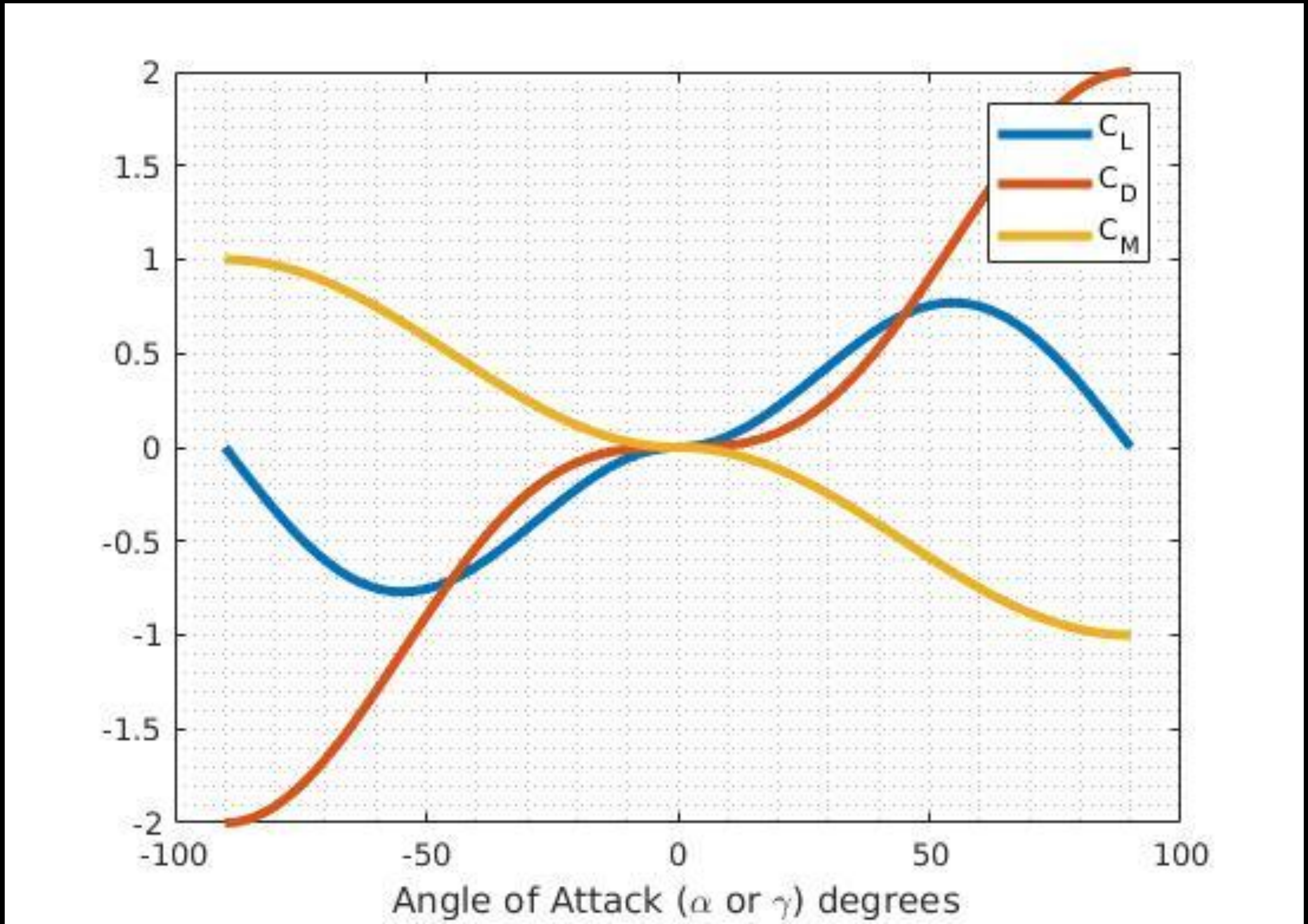
pegasus
/pegasus/gps
/pegasus/imu
/pegasus/estimator
/pegasus/barometer
/pegasus/state_hat
/pegasus/controller
/pegasus/motor_command
pegasus_sim
/pegasus_sim/wind
/pegasus_sim/wind
/pegasus_sim/eom
/pegasus_sim/truth
/pegasus_sim/sensors
rosflight
/rosflight/naze
/rosflight/rc_raw
/tf

Flat Plate Aerodynamic Model

$$C_L(\alpha) = 2\sin^2(\alpha)\cos(\alpha)\,\mathrm{sgn}(\alpha)$$
$$C_D(\alpha) = 2\sin^3(\alpha)$$
$$C_M(\alpha) = -\sin^2(\alpha)\,\mathrm{sgn}(\alpha)$$

R. F. Stengel, *Flight Dynamics*. Princeton, NJ: Princeton University Press, 2004.

Flat Plate Aerodynamic Coefficients



Flat Plate Aerodynamic Model

Define

u_r = component of V_a in the i^b direction

v_r = component of V_a in the j^b direction

w_r = component of V_a in the k^b direction

$$\alpha = \mathrm{atan}\left(\frac{w_r}{u_r}\right)$$

$$\beta = \mathrm{asin}\left(\frac{v_r}{V_a}\right)$$

$$\gamma = \mathrm{atan}\left(\frac{w_r}{v_r}\right)$$

Flat Plate Aerodynamic Model

Forces and moments due to airspeed at angle of attack α and γ in wind frame

$$f_{L,\alpha} = \frac{1}{2}\rho S(u_r^2 + w_r^2)C_L(\alpha) \qquad f_{L,\gamma} = \frac{1}{2}\rho S(v_r^2 + w_r^2)C_L(\gamma)$$

$$f_{D,\alpha} = \frac{1}{2}\rho S(u_r^2 + w_r^2)C_D(\alpha) \qquad f_{D,\gamma} = \frac{1}{2}\rho S(v_r^2 + w_r^2)C_D(\gamma)$$

$$f_{M,\alpha} = \frac{1}{2}\rho Sc(u_r^2 + w_r^2)C_D(\alpha) \qquad f_{M,\gamma} = \frac{1}{2}\rho Sb(v_r^2 + w_r^2)C_D(\gamma)$$

Where c is the flat plate length, b is the width and $S = bc$

Motor and Propeller Model

- DC motor model

$$\Omega = K_{\delta_m}\delta_m \quad \text{(RPM) ***}$$

$$i = \frac{\left(V_b - \frac{\Omega}{K_v}\right)}{\mathrm{R_m}}$$

$$Q_m = \frac{i - i_0}{\frac{K_v \pi}{30}}$$

$$P_{shaft} = Q_m \Omega \frac{\pi}{30}$$

*** RPM, ESC's are black boxes of magic

Motor and Propeller Model

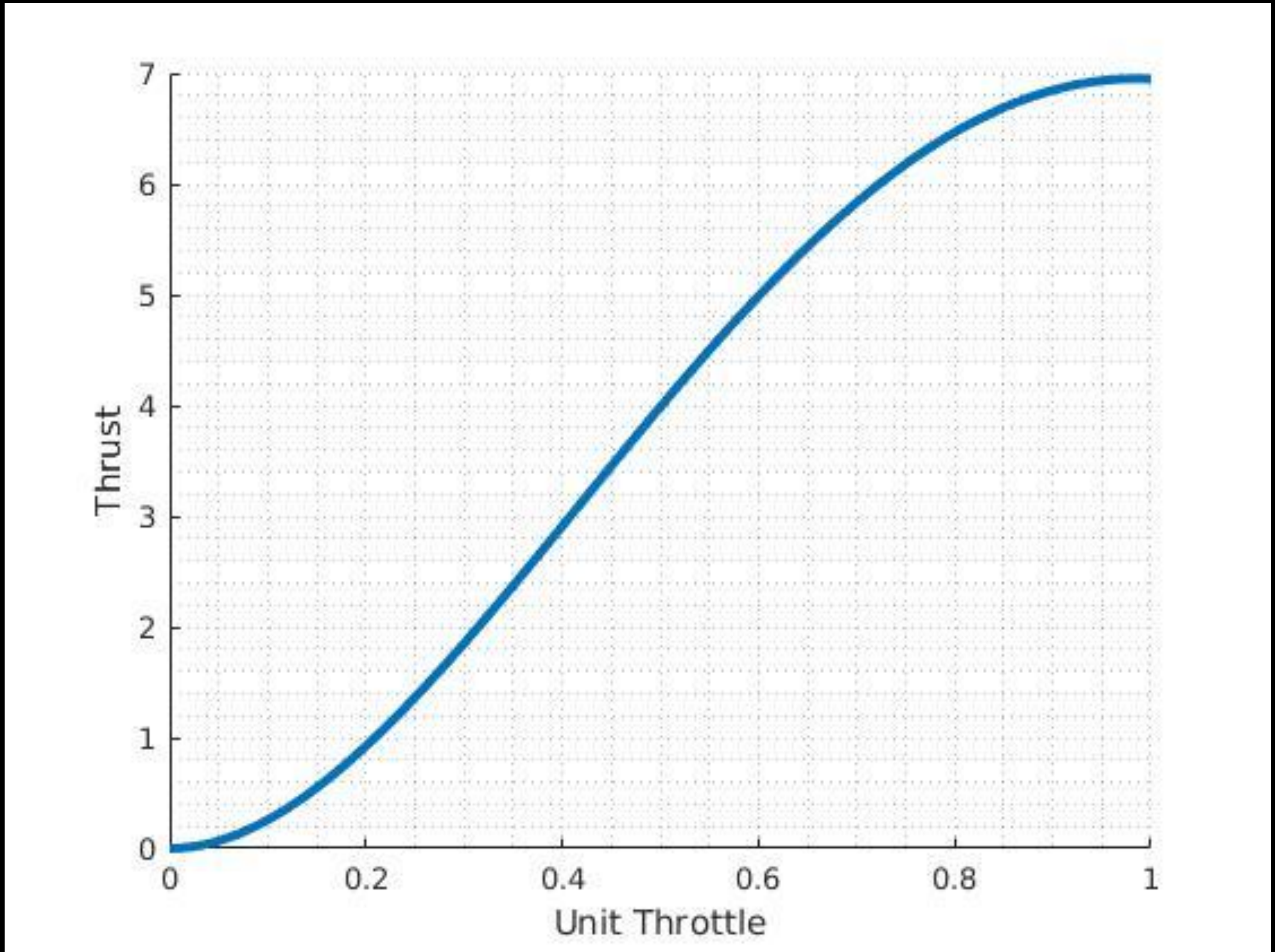
- Momentum Theory

$$F = \left(2\rho A_p P_{shaft}^2\right)^{\frac{1}{3}}$$

Where A_p is the area the propeller sweeps

$$\tau = \frac{F}{6.3}$$

Motor and Propeller Model



Control for Manual Flight

- While manually flying, control roll, pitch, and yaw rate (yaw rate is a misnomer, really control r).

Assume

$$[J] = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}$$

Then the rotational dynamics simplify to

$$\dot{p} = \frac{J_{yy} - J_{zz}}{J_{xx}} qr + \frac{M_x}{J_{xx}} \qquad \dot{q} = \frac{J_{zz} - J_{xx}}{J_{yy}} pr + \frac{M_y}{J_{yy}} \qquad \dot{r} = \frac{J_{xx} - J_{yy}}{J_{zz}} pq + \frac{M_z}{J_{zz}}$$

Control for Manual Flight

- Roll Controller

Assuming rotational rates are small

$$qr = 0$$

$$\dot{p} = \frac{M_x}{J_{xx}}$$

$$\dot{\phi} = p + \sin(\phi)\tan(\theta)\, q + \cos(\phi)\tan(\theta)\, r$$

$$\ddot{\phi} = \dot{p} + \dot{\phi}\cos(\phi)\tan(\theta)\, q + \dot{\phi}\sin(\phi)\sec^2(\theta)\, q + \sin(\phi)\, tan(\theta)\dot{q} - \dot{\phi}\sin(\phi)\tan(\theta) + \dot{\theta}\cos(\phi)\sec^2(\theta)\, r + \cos(\phi)\tan(\theta)\,\dot{r}$$

Which is approximately

$$\ddot{\phi} = \dot{p} = \frac{M_x}{J_{xx}}$$

Control for Manual Flight

Which yields the transfer function

$$\Phi(s) = \frac{1}{J_{xx}s^2} M_x(s)$$

A PD controller yields the following transfer function from commanded roll angle to actual roll angle

$$\Phi(s) = \frac{k_{p_\phi}}{J_{xx}s^2 + k_{d_\phi}s + k_{p_\phi}} \Phi_c$$

$$\omega_n \approx \frac{2.2}{t_r}$$

$$k_{p_\phi} = \omega_n^2 J_{xx}$$

$$k_{d_\phi} = 2\zeta\omega_n J_{xx}$$

Control for Manual Flight

The pitch controller is derived in a similar manner, and yields the following transfer function

$$\Theta(s) = \frac{k_{p_\theta}}{J_{yy}s^2 + k_{d_\theta}s + k_{p_\theta}}\Theta_c$$

Control for Manual Flight

Instead of commanding ψ, yaw rate is typically controlled on multi-rotors

From the kinematics using the simplified inertial matrix,

$$\dot{r} = \frac{J_{xx} - J_{yy}}{J_{zz}} pq + \frac{M_z}{J_{zz}}$$

$$pq = 0$$

$$\dot{r} = \frac{M_z}{J_{zz}}$$

$$R(s) = \frac{1}{J_{zz} s} M_Z(s)$$

$$R(s) = \frac{k_{p_\psi}}{\left(J_{zz} + k_{d_\psi}\right) s + k_{p_\psi}} R_c(s)$$

Control for Manual Flight

Note that this is a 1st order system:

$$\tau = \frac{J_{zz} + k_{d_\psi}}{k_{p_\psi}}$$

$$k_{d_{psi}} = 0$$

$$k_{p_\psi} = 5.0\frac{J_{zz} + k_{d_\psi}}{t_r}$$

Control for Autonomous Flight

- Height Controller

Equations simplify to

$$\ddot{p}_d = \frac{f_z}{m}$$

$$f_z = F - mg$$

$$F_e = mg$$

$$\widetilde{H}(s) = \frac{1}{s^2 m}\tilde{F}$$

$$\widetilde{H} = \frac{k_{p_h}}{ms^2 + k_{d_h}s + k_{p_h}}\widetilde{H}_c$$

Control for Autonomous Flight

- V_g and χ Controller (vehicle-1 frame)

Define

$u_1 =$ Velocity of the multi-rotor along i^{v1}

v_1 = Velocity of the multi-rotor along j^{v1}

$w_1 =$ Velocity of the multi-rotor along k^{v1}

Control for Autonomous Flight

$$\dot{\vec{v}} = \frac{\sum F}{m}$$

$$\begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \dot{w}_1 \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} -\frac{F}{m}\sin(\theta)\cos(\phi) \\ \frac{F}{m}\sin(\phi) \\ -\frac{F}{m}\cos(\theta)\cos(\phi) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

$$U_1(s) = \frac{1}{s}A_1(s)$$

$$V_1(s) = \frac{1}{s}V_1(s)$$

Control for Autonomous Flight

$$U_1(s) = \frac{k_{p_{u_1}}}{(1 + k_{d_{u1}})s + k_{p_{u_1}}} {U_1}_c(s)$$

$$V_1(s) = \frac{k_{p_{v_1}}}{(1 + k_{d_{v1}})s + k_{p_v}} V_c(s)$$

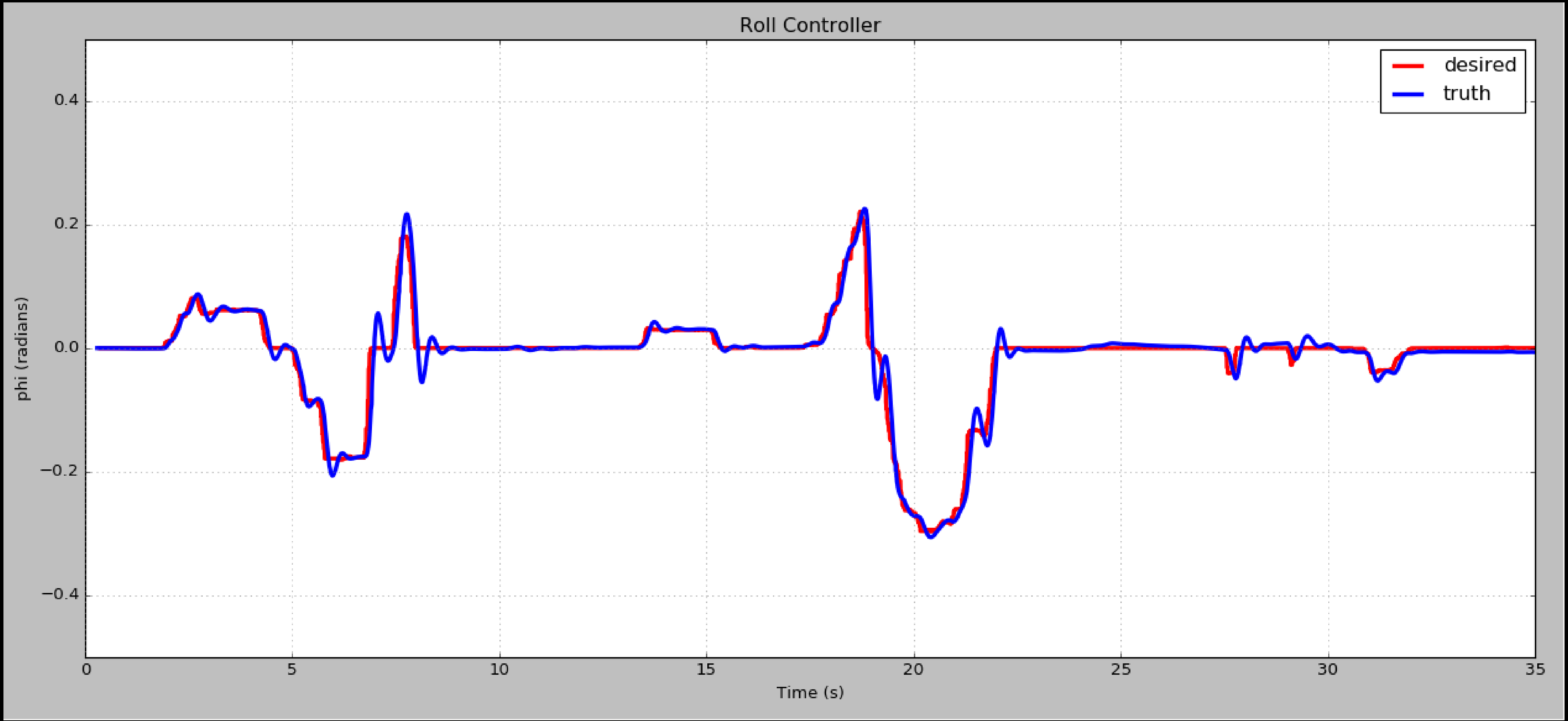
PD controllers output a desired acceleration, model inversion obtains the commanded roll and pitch angles

$$\phi_c = \operatorname{asin}\left(\frac{ma_2}{F}\right)$$

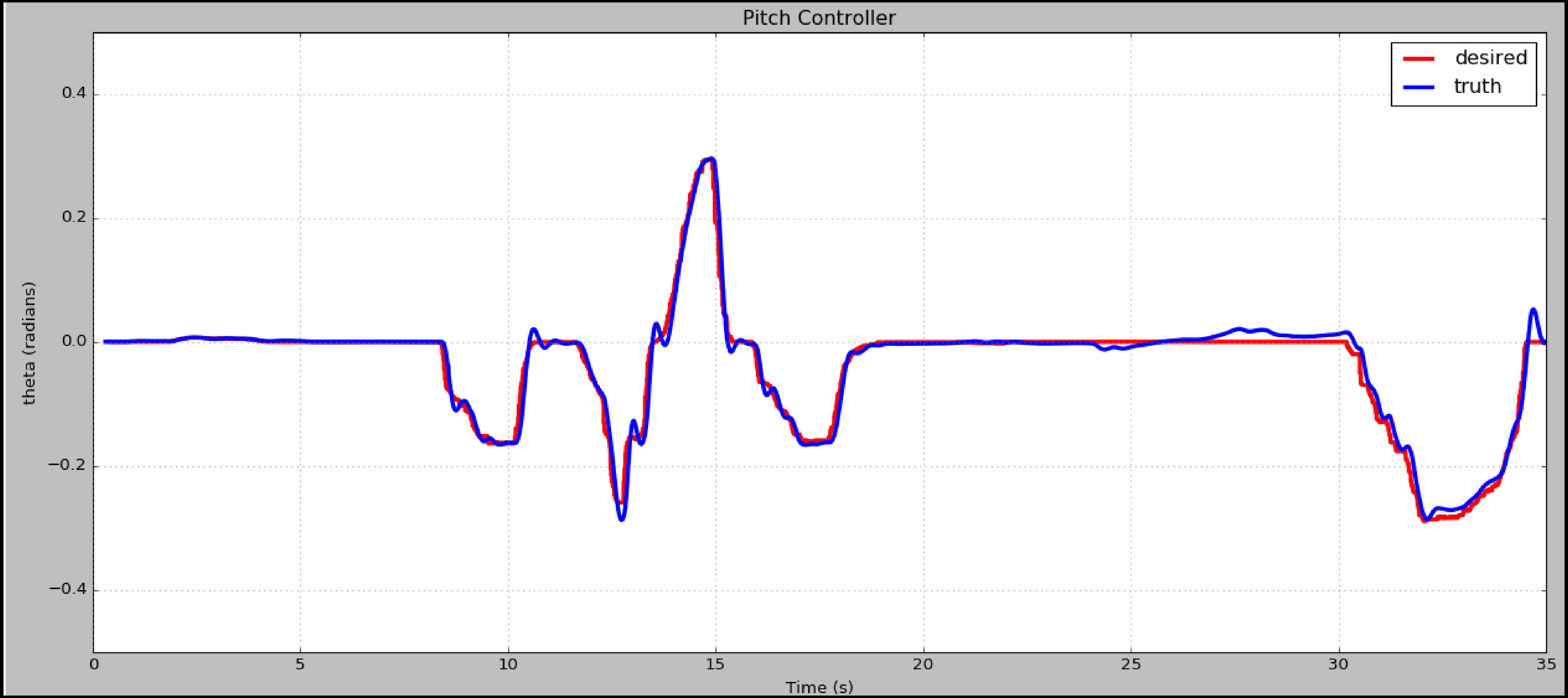
$$\theta_c = \operatorname{asin}\left(-\frac{a_1}{F\cos(\phi_c)}\right)$$

Make sure to saturate a_1 and a_2 correctly

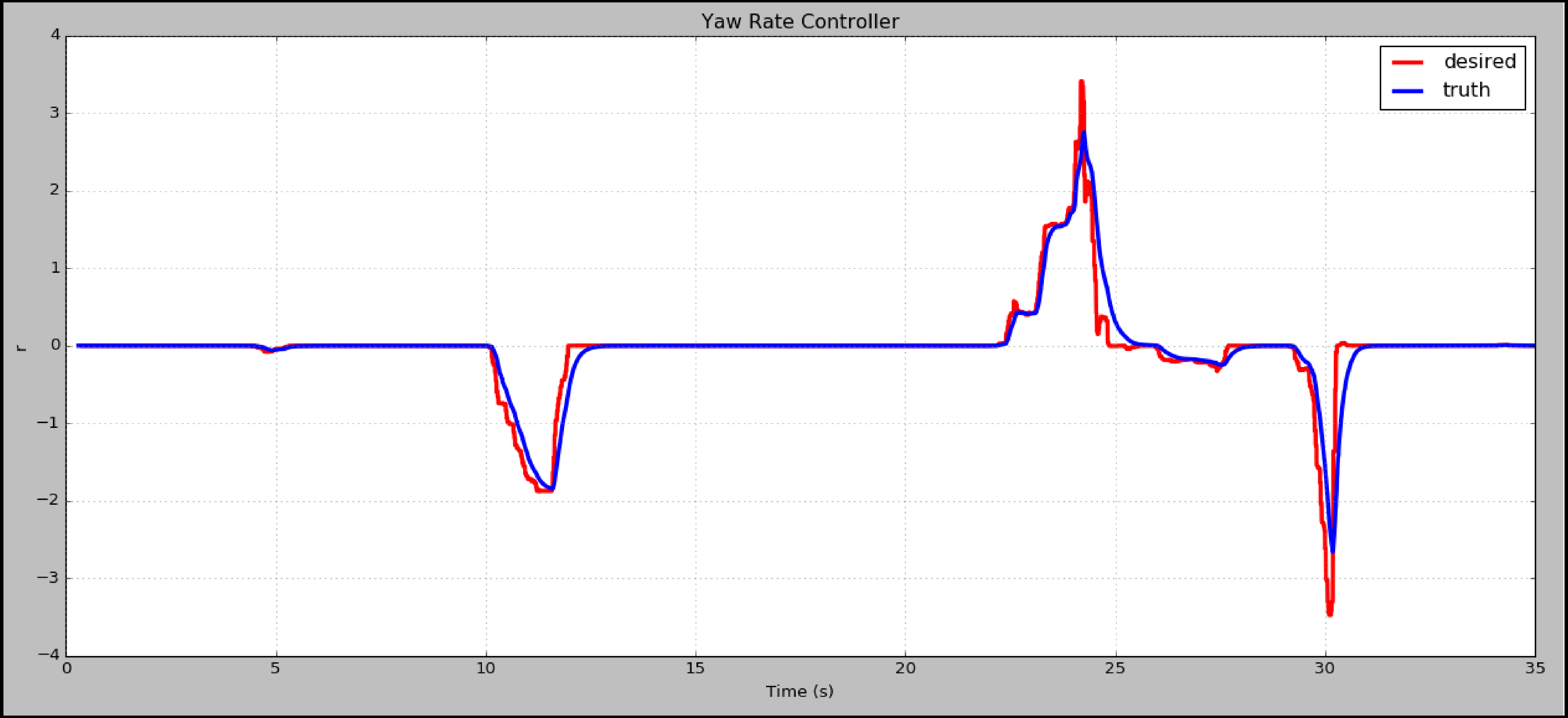
Roll Angle Tracking



Pitch Angle Tracking



Yaw Rate (r) Tracking



Extended Kalman Filter

$$x = \begin{bmatrix} \phi \\ \theta \\ \psi \\ u_1 \\ v_1 \end{bmatrix}$$

Low pass filter p, q, and r from the gyroscopes. Use the accelerometer and GPS for the measurements.

Extended Kalman Filter

$$a_x = \frac{f_{x,p}}{m} - \sin(\theta) g$$

$$a_y = \frac{f_{y,p}}{m} + \cos(\theta)\sin(\phi) g$$

$$a_z = \frac{f_{z,p}}{m} + \cos(\theta)\cos(\phi)\, g$$

$$f = \begin{bmatrix} p + \sin(\phi)\tan(\theta)\, q + \cos(\phi)\tan(\theta)\, r \\ \cos(\phi)\, q - \sin(\phi)\, r \\ \sin(\phi)\sec(\theta)\, q + \cos(\phi)\sec(\theta)\, r \\ \cos(\theta)\, a_x + \sin(\theta)\sin(\phi)\, a_y + \sin(\theta)\cos(\phi)\, a_z \\ \cos(\phi)\, a_y - \sin(\phi)\, a_z \end{bmatrix}$$

Extended Kalman Filter

$$\frac{\delta f}{\delta x} = \begin{bmatrix} \cos(\phi)\tan(\theta)\,q\ -\sin(\phi)\tan(\theta)\,r & \dfrac{qsin(\theta) + rcos(\phi)}{\cos^2(\theta)} & 000 \\ -qsin(\phi) - rcos(\phi) & 0 & 000 \\ \cos(\phi)\sec(\theta)\,q\ -\sin(\phi)\sec(\theta)\,r & \left(qsin(\phi) + rcos(\phi)\right)\tan(\theta)\sec(\theta) & 000 \\ \sin(\theta)\cos(\phi)\,a_y - \sin(\theta)\,a_z & -\sin(\theta)\,a_x + \cos(\theta)\sin(\phi)\,a_y + \cos(\theta)\cos(\phi)\,a_z & 000 \\ -\sin(\phi)\,a_y - \cos(\phi)\,a_z & 0 & 000 \end{bmatrix}$$

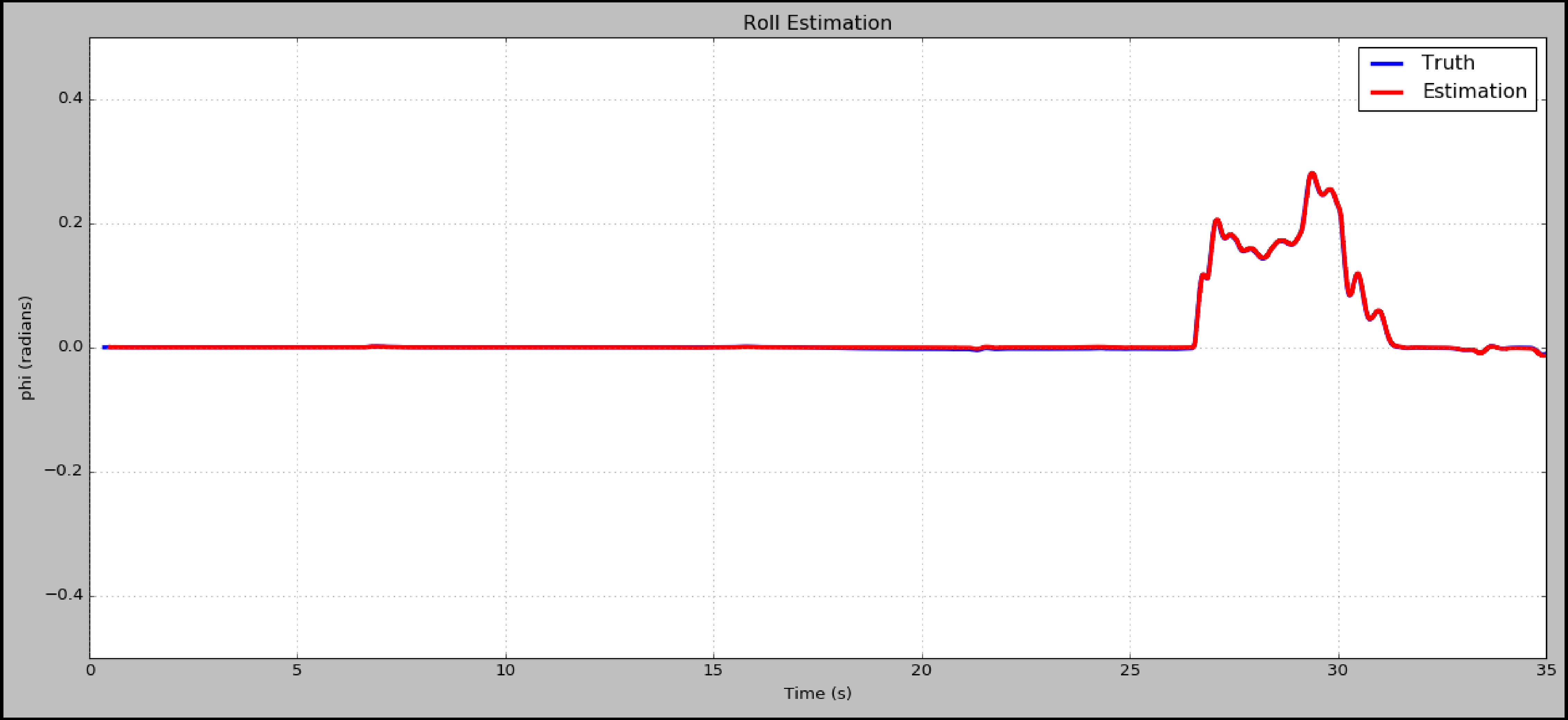
Extended Kalman Filter

$$h = \begin{bmatrix} \dfrac{f_{x,p}}{m} + \sin(\theta)\, g \\ \dfrac{f_{y,p}}{m} - \cos(\theta)\sin(\phi)\, g \\ \dfrac{f_{z,p}}{m} - \cos(\theta)\cos(\phi) \\ \sqrt{u_1^2 + v_1^2} \\ \operatorname{atan}\left(\dfrac{(u_1 \sin(\psi) + v_1 \cos(\psi))}{u_1 \cos(\psi) - v_1 \sin(\psi)}\right) \end{bmatrix}$$

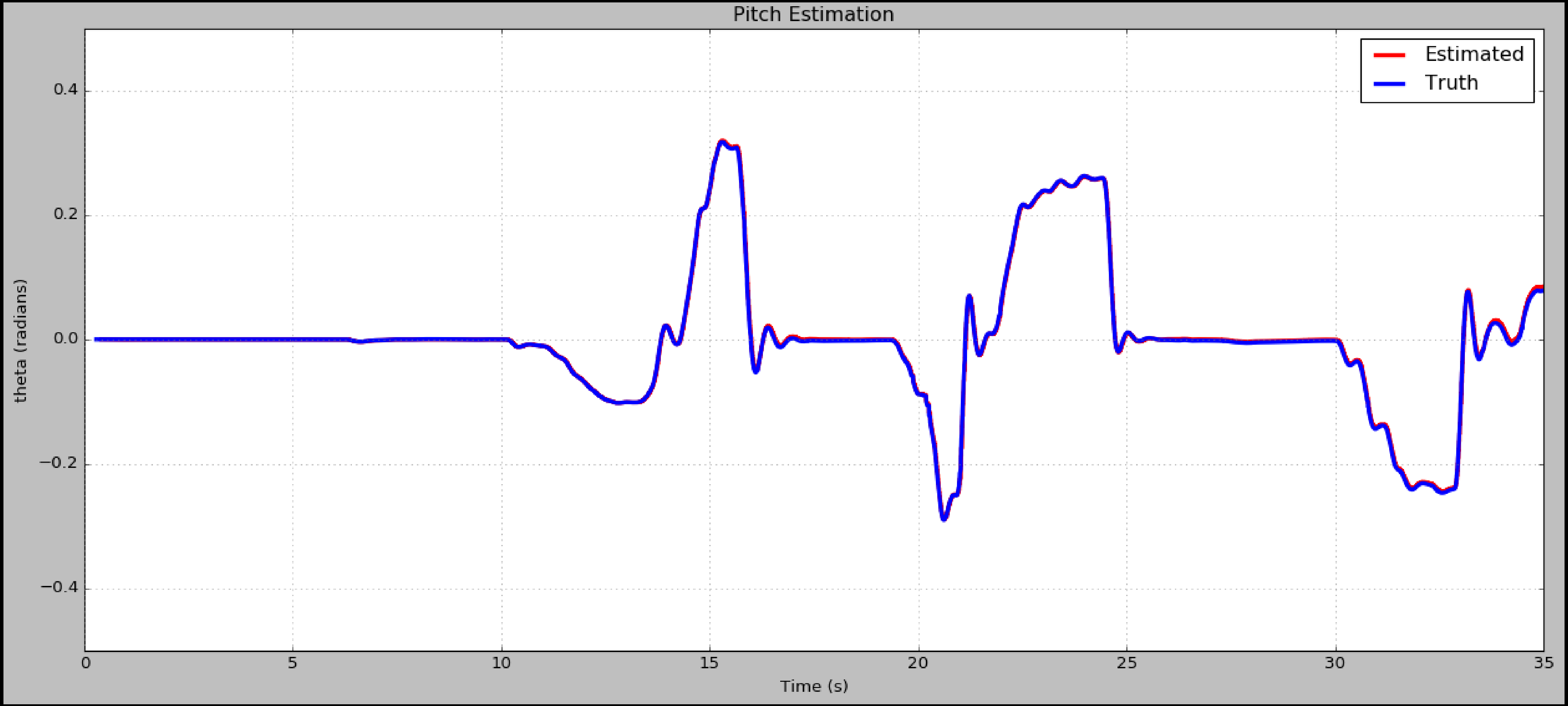
Extended Kalman Filter

$\frac{\delta h}{\delta x} =$ left to the interested reader, or in kalman_filter.cpp

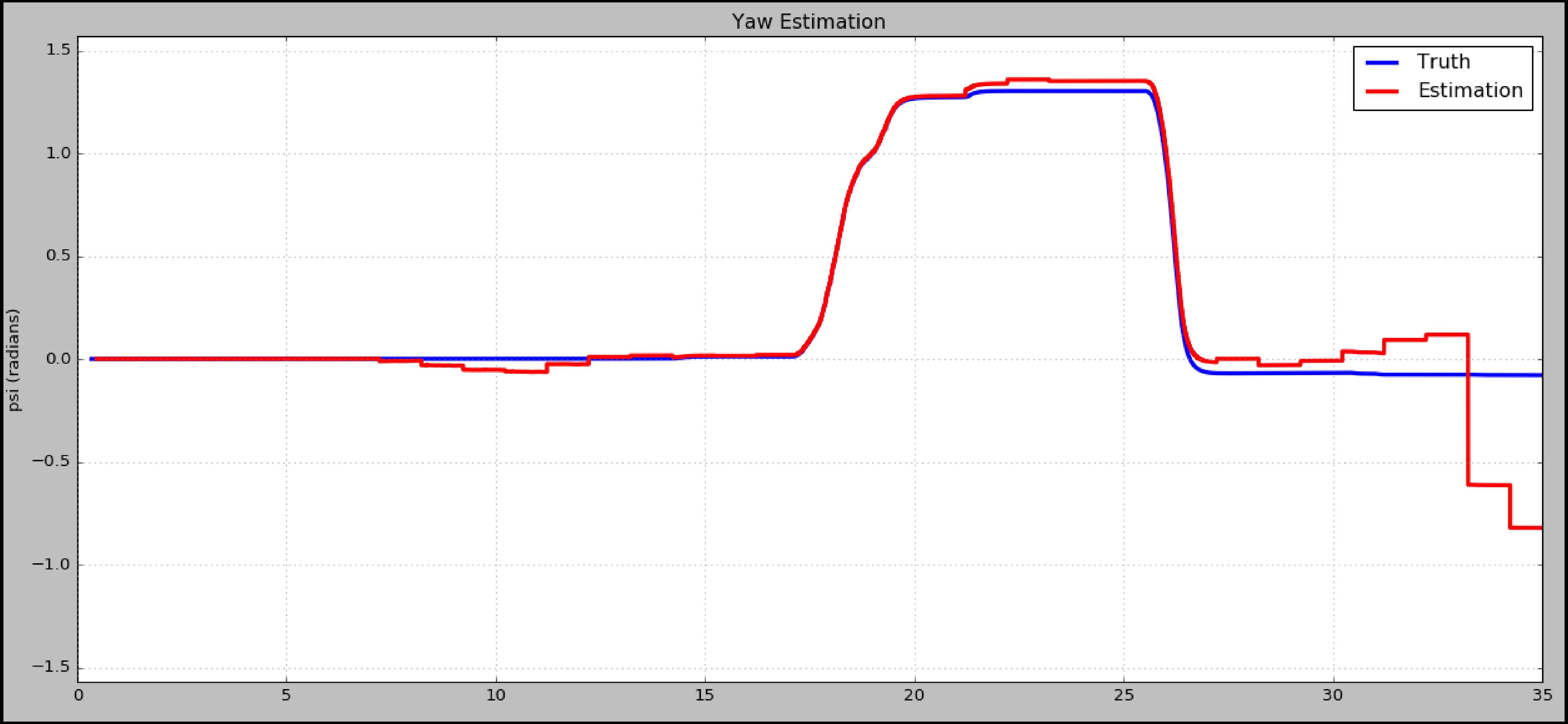
Roll Estimation



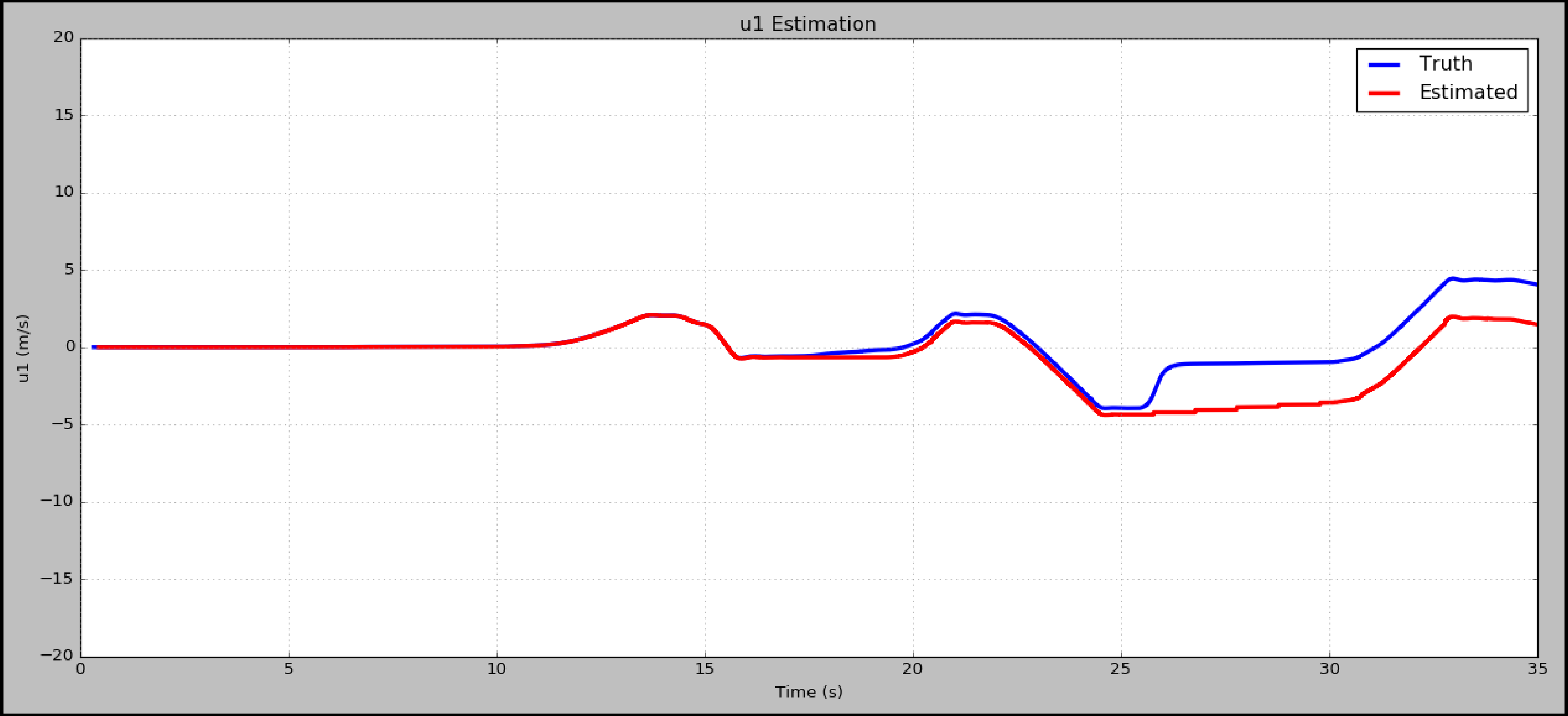
Pitch Estimation



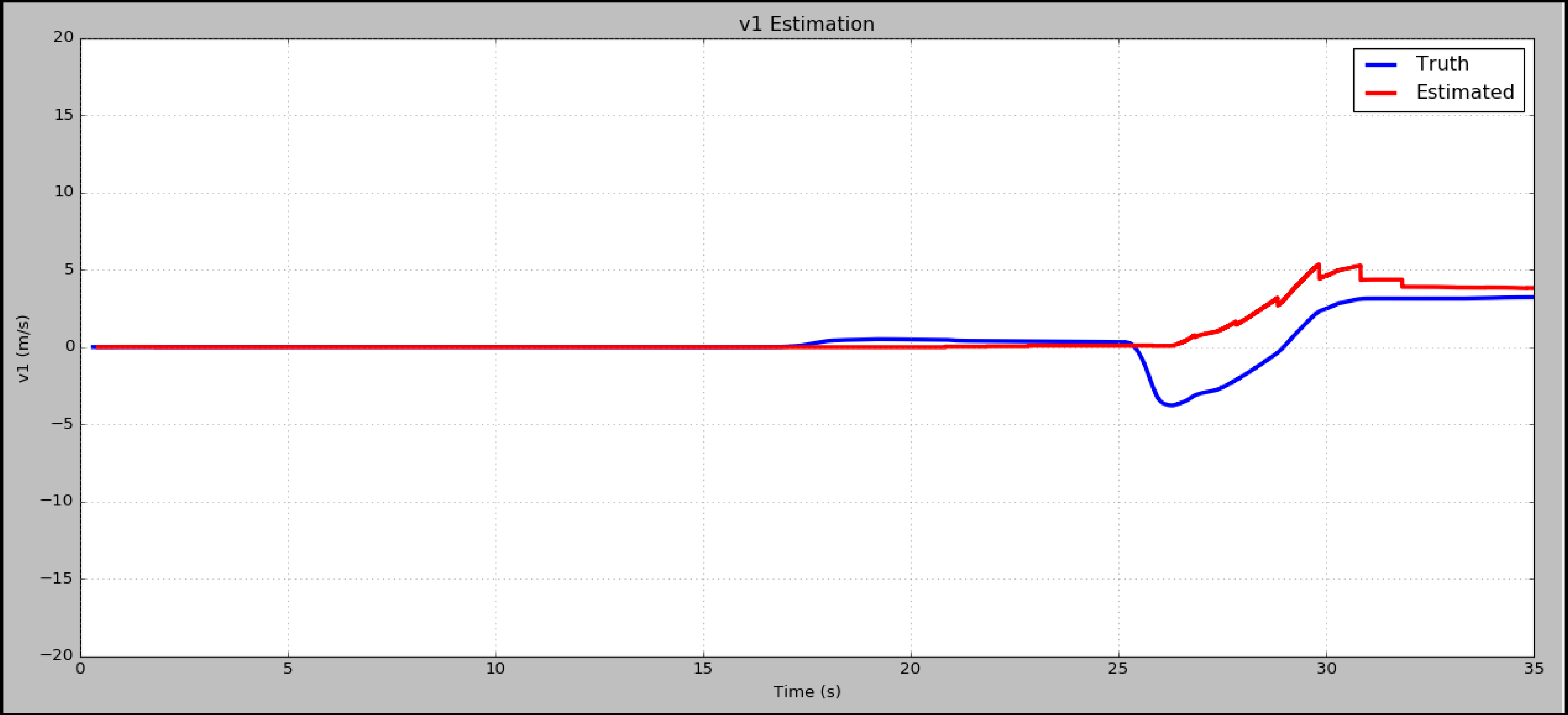
Yaw Estimation



U1 Estimation



V1 Estimation



Other Bells and Whistles

- RC transmitter to commanded states
  - Uses ROSflight to get raw RC PWMs.
  - Channel mapping and PWM to commands (RC rate, expo and super rate)
- Equations of motion are expressed for a full inertia matrix and motors in any orientation or placement
- Flight modes (manual and autonomous)
- GPS to NED and NED to GPS converter (oblate spheroid WGS84 model)
- Multi-rotor CAD model