Valbal Trajectory Planning

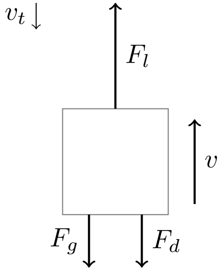
Joan Creus-Costa and John Dean

Stanford Student Space Initiative

December 6, 2018

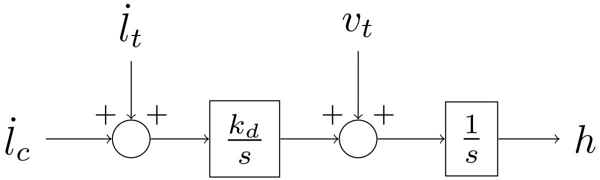
System Dynamics

- Assumptions
  - v_t is small
  - $F_d \propto v$ i.e. drag is linear.
  - $F_l - F_g = F_d$ i.e. the balloon is always at terminal velocity
- Equations of motion
  - let $l = F_l - F_g$ be the net lift on the balloon
  - $\dot{l}$ is commanded by controller
  - $v = k_d \int \dot{l}\, dt$
  - $h = \int v\, dt$
  - $\mathcal{L}\{\cdot\} = k_d/s^2$



F_d: Force of drag
F_g: Gravity
F_l: Buoyant force
v: vertical velocity of balloon
v_t: vertical velocity of surrounding air

Open Loop Block Diagram

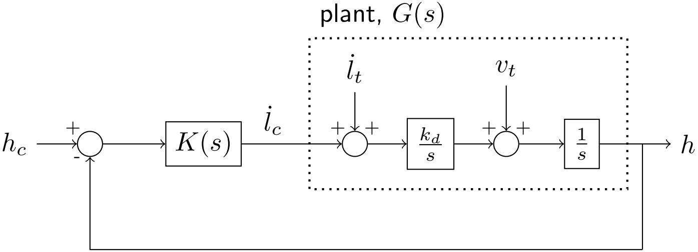


$\dot{l}_c$: commanded change in lift (valve and ballast actions)
$\dot{l}_t$: atmospheric lift disturbance
v_t: atmosphereic velocity disturbance
h: altitude

Spaghetti Controller Motivation

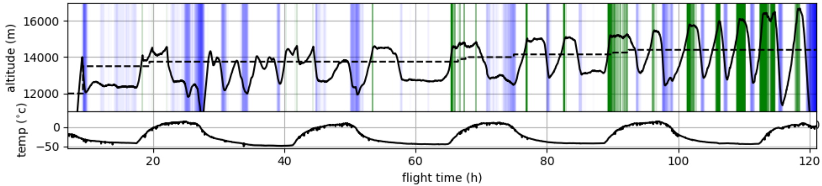
- Use a simple linear compensator to stabilize altitude with robust stability margins
-

Spaghetti Block Diagram



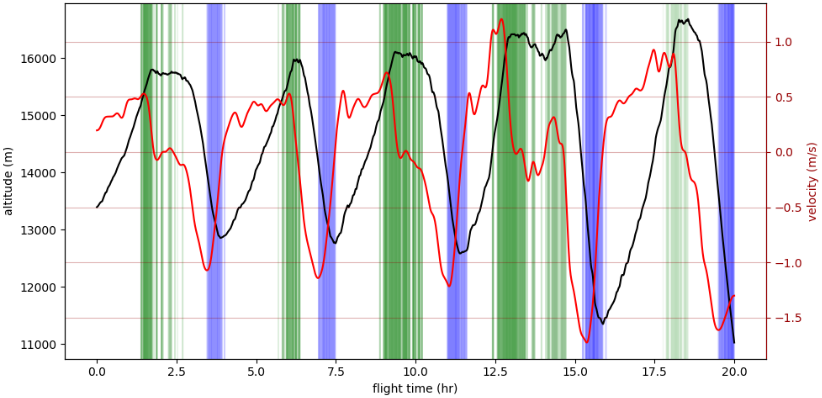
$K(s)$: First order lead compensator.

Spaghetti Flight



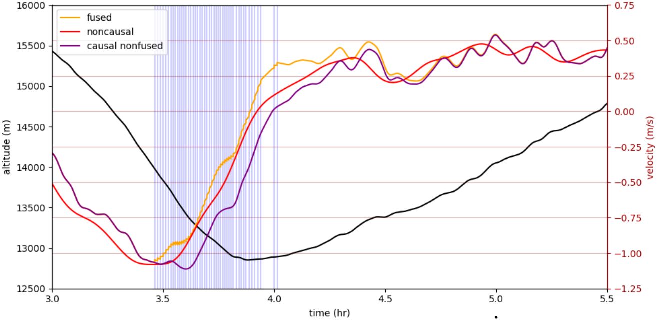
- 120hr flight from December with spaghetti as controller
  - blue shows ballast events, green shows vent events
  - temperature shows sunset/sunrise, large effect on ballast use
- issues durring flight:
  - valve controller had software bug, instead of changing duty cylce, one threshold met valve was repeatedly opened
  - At end of flight, balloon has low overpressure–opening valve has no effect until balloon rises high enough

Oscillations



Velocity Estimator

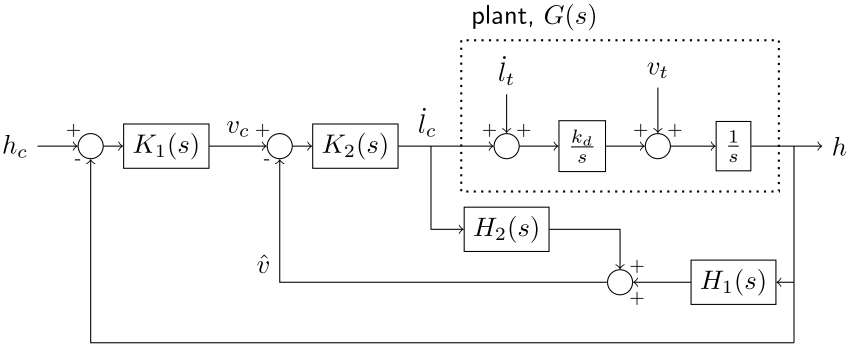
Lowpass filtered velocity estimate that fuses information on actions from the controller.



$$\mathcal{L}\{\hat{v}\} = H_1(s)\mathcal{L}\{h\} + H_2(s)\mathcal{L}\{\dot{l}_c\}$$

$H_1(s)$ is differentiation and 2nd order lowpass filter
$H_2(s)$ is integration with decay (estimate of effect of actions, decays to 0 over time)

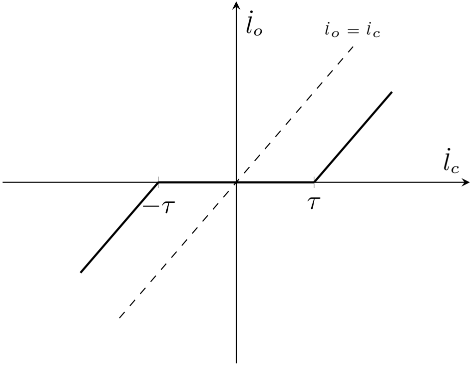
Lasagna Block Diagram



$K_1(s)$: Position loop compensator
$K_2(s)$: Velocity loop compensator
$H_1(s), H_2(s)$: Velocity estimator
With just proportional control, we have $i_c = ((h_c - h)k_h - \hat{v})k_v$

Lasanga Nonlinearities

Since we typically command a target altitude and an allowable region, we add a deadband to the controller output. Let i_o be the output of the nonlinearity. Deadband:



To set bounds on the altutude, we set $\tau = e_{\text{tol}} k_v k_h$, where e_{tol} is the allowable distance from the altitude command.

Picking gains

note: while the deadband makes the controller non-linear, it still peicewise linear, thus linear analysis can be used.
Transfer function for the linear system is

$$\frac{k_v k_h k_l}{s^2 + k_l k_v s + k_l k_v k_h}$$

So damping ratio is $\zeta = \frac{1}{2}\sqrt{\frac{k_l k_v}{k_h}}$.

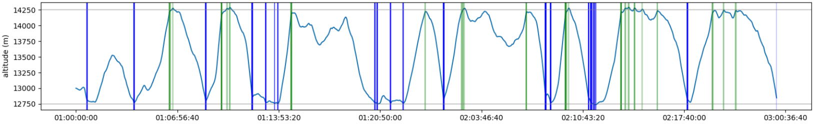
- We choose gains such that $\zeta = 1$ and we have critical damping.
- This gives ratio between k_v and k_h, but what about magnitiude?
- high gain $\rightarrow$ controller waits and acts agressively near e_{tol}
- low gain $\rightarrow$ controller acts cautiously before e_{tol}

demonstraited on next slide

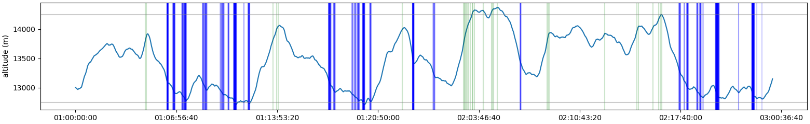
High vs Low Gain

Plots of simulation shown

high gain

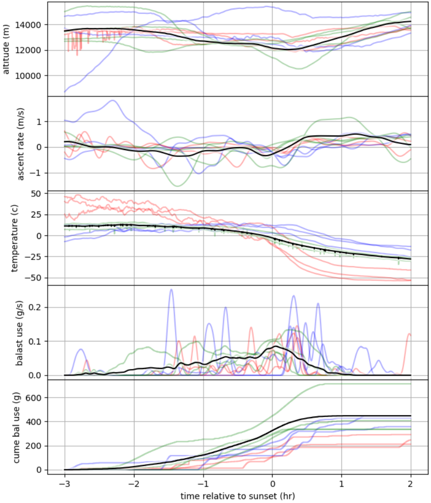


low gain



High gain performs better but can't tolerate uncertainty, low gain is worse but performs better under uncertainty

Nightfall



- Left plot shows 10 sunsets across various flights (each flight different color).
- plot blow shows a fit to the data using convex regularization and contraints