Valbal Trajectory Planning

Joan Creus-Costa and John Dean

Stanford Student Space Initiative

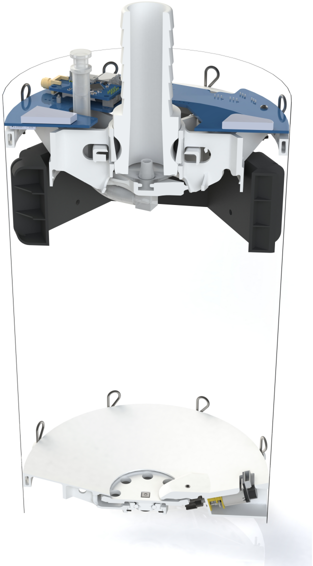
December 7, 2018

Outline

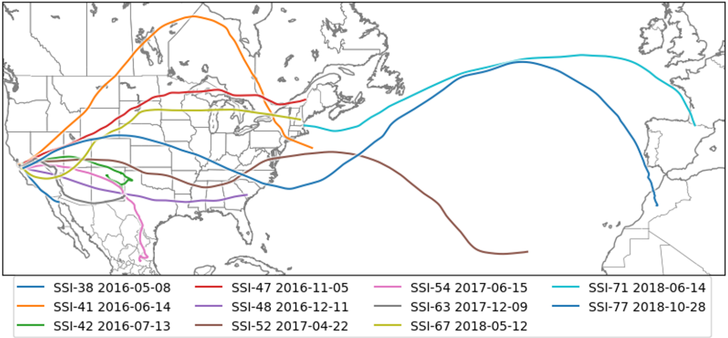
ValBal

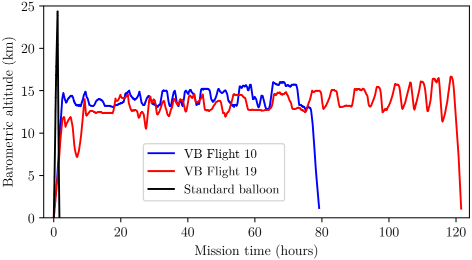
- Research project by the Stanford Space Initiative (undergrad student club).
- High altitude latex balloon platform that controls its altitude by venting lifting gas and dropping ballast mass.
- Very cheap (sub thousand dollars), long endurance (5 days demonstrated).
- Potential applications: hurricane data collection, radar probing of Greenland ice, lightning research, data relay...
- Control: in altitude (remain between bounds while minimizing control effort), in space (pick altitude to get good winds).

1. A. Sushko, A. Tedjarati, J. Creus-Costa, S. Maldonado, K. Marshland and M. Pavone, "Low cost, high endurance, altitude-controlled latex balloon for near-space research (ValBal)," 2017 IEEE Aerospace Conference, Big Sky, MT, 2017, pp. 1-9.
2. A. Sushko et al., "Advancements in low-cost, long endurance, altitude controlled latex balloons (ValBal)," 2018 IEEE Aerospace Conference, Big Sky, MT, 2018, pp. 1-10.



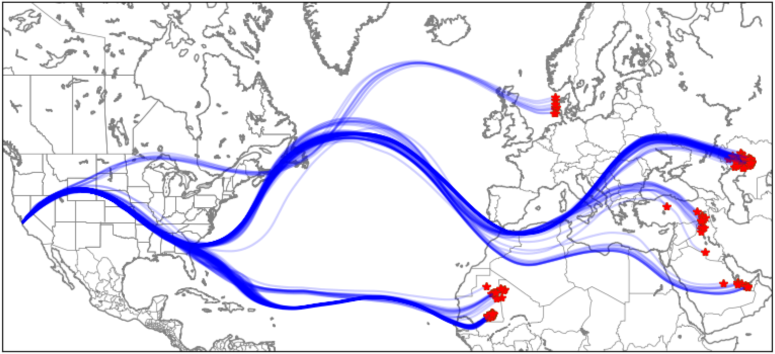

SSI-63


SSI-38 2016-05-08
SSI-41 2016-06-14
SSI-42 2016-07-13
SSI-47 2016-11-05
SSI-48 2016-12-11
SSI-52 2017-04-22
SSI-54 2017-06-15
SSI-63 2017-12-09
SSI-67 2018-05-12
SSI-71 2018-06-14
SSI-77 2018-10-28


Barometric altitude (km)
Mission time (hours)
VB Flight 10
VB Flight 19
Standard balloon
0
5
10
15
20
25
0
20
40
60
80
100
120

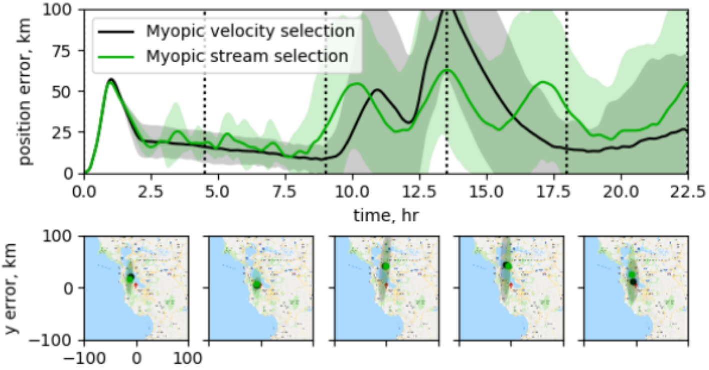
Outline

Trajectory Planning



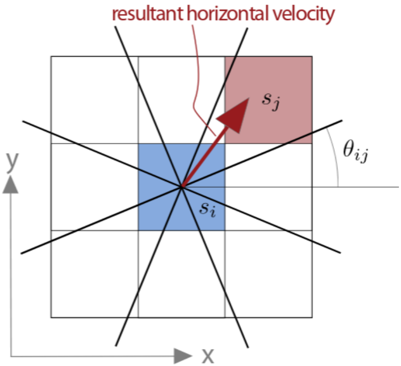
Tree search for station keeping

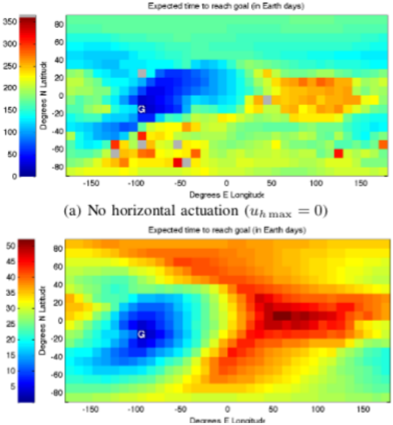
- Born and Schwager, "Riding an Uncertain Wind Field: Receding Horizon Tree Search Planning with Opportunistic Sampling for an Autonomous Weather Balloon" (2019)



MDP

- Wolf et al., "Probabilistic motion planning of balloons in strong, uncertain wind fields" (2010)





(a) No horizontal actuation ($u_{h\,\max} = 0$)

Formulation

- System represented by states $s \in \mathcal{S}$
- $\mathcal{S} = \mathcal{H} \times \Lambda \times \Phi \times \mathcal{T}$ (Time, Altitude, Longitude, Latitude)
  - Physically intuitive to keep majority of state continuous—curse of dimensionality as we discretize.
    - $h \in \mathcal{H}$: altitudes ($[0 \text{ km}, 20 \text{ km}]$)
    - $\lambda \in \Lambda$: longitudes ($[0, 2\pi]$)
    - $\phi \in \Phi$: latitudes ($[-\pi/2, \pi/2]$)
  - $t \in \mathcal{T}$: time, defined in discrete steps of 10 min. Allows us to use simple Euler integration spatially to trace out wind field (altitude handled differently).
- Atmospheric winds act on the balloon $w(h, \lambda, \phi, t) \to (u, v)$. w given by NOAA on 0.25° grid at various altitudes; interpolate in each variable.
- Control policies $\theta(t) = (h_{\text{cmd}}, e_{\text{tol}})$ command the altitude of the balloon and error tollerance around set altitude

Formulation

- Goal: formulate value function V as a differentiable function of a (small) set of policy parameters θ.
  - Avoids having to discretize a huge state space as in most MDP formulations.
- Optimize V via a gradient method, *e.g.* $\theta^{k+1} = \theta^k + \alpha_k \nabla_\theta V(\theta)$. for stepsize a_k
- Allows us to preserve key properties of the spaces we're in:
  - We preserve smoothness of the value function: the wind field is not random, and has a rather small spatial frequency.
  - We preserve the continuity of our variables: altitude, latitude, longitude need small discretization to be realistic.
  - We will be able to handle stochasticity without blowing up the size of the problem.

Formulation: example

- Suppose we want to maximize longitudinal distance of final point after N steps (say $N = 720$, 5 days).
- Final longitude is the result of doing a simulation rollout, using Euler integration in latitude and longitude.
- $V = \lambda_f = \lambda_0 + v_1(t_1, h_1, \phi_1, \lambda_1)\Delta t + \cdots + v_N(t_N, h_N, \phi_N, \lambda_N)\Delta t$.
- Need to parametrize V. We control altitude of the balloon (see later). Simplest formulation: define waypoints $\theta_1, \ldots, \theta_k$ at points $T_1, \ldots, T_k$, and assume balloon goes linearly between waypoints: $h_i = \theta_j + (t_i - T_j)\frac{\theta_{j+1}-\theta_j}{T_{j+1}-T_j}$. Ability to do stochastic formulations (later).

$$h(t) = \begin{cases} \theta_1 + (t - t_1)\frac{\theta_2 - \theta_1}{t_2 - t_1} & t_1 \leq t < t_2 \\ \cdots & \\ \theta_{n-1} + (t - t_{n-1})\frac{\theta_n - \theta_{n-1}}{t_n - t_{n-1}} & t_{n-1} \leq t < t_n \end{cases}$$

$w_{\{\lambda,\phi\}}(\lambda, \phi, h, t) =$ bilinear interpolation in (λ, ϕ), linear in t, h from NOAA

$$(h_0, t_0, \phi_0, \lambda_0)$$

$$\begin{cases} h_1 = h(t_1) \\ \lambda_1 = \lambda_0 + \Delta t \cdot w_\lambda(\lambda_0, \phi_0, h_0, t_0) \\ \phi_1 = \phi_0 + \Delta t \cdot w_\phi(\lambda_0, \phi_0, h_0, t_0) \end{cases}$$

$$\begin{cases} h_2 & = h(t_2) \\ \lambda_2 & = \lambda_1 + \Delta t \cdot w_\lambda(\lambda_1, \phi_1, h_1, t_1) \\ & \scriptstyle = \lambda_0 + \Delta t \cdot w_\lambda(\lambda_0, \phi_0, h_0, t_0) \\ & \qquad \scriptstyle + w_\lambda(\lambda_0 + w_\lambda(\lambda_0, \phi_0, h_0, t_0)\Delta t, \phi_0 + w_\phi(\lambda_0, \phi_0, h_0, t_0)\Delta t, h_(t_1), t_1)\Delta t \end{cases}$$

$$\cdots$$

$\dfrac{\partial \lambda_n}{\partial \theta_i} \neq 0$, follow the computation graph to compute derivative

Stochastic formulation

- Previous formulation does not account for noise—balloon can't hit linearly interpolated altitude waypoints exactly! (Not without wasting tonnes of ballast, anyway.)
- Onboard control algorithm has two parameters: h_{cmd} (setpoint) and e_{tol} (tolerance). Approximate balloon as being somewhere in $h_{\text{cmd}} \pm e_{\text{tol}}$ randomly.
- However, position within the tolerance is correlated between timesteps.
  - If balloon was at $h_{\text{cmd}} - 0.98e_{\text{tol}}$ at $t = 0$, it's unlikely to be at $h_{\text{cmd}} + 0.5e_{\text{tol}}$ at $t = 10$ min.
- Solution: parametrize in terms of a slow-moving random walk $\lambda(t)$.

Stochastic formulation

- Refined differentiable altitude:

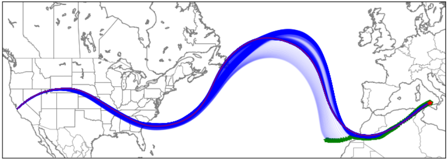
$$h(t;\lambda) = \mathsf{LinInterp}\left(t, t_i \to h_{\mathsf{cmd},i} + \lambda(t_i) e_{\mathsf{tol},i}\right)$$

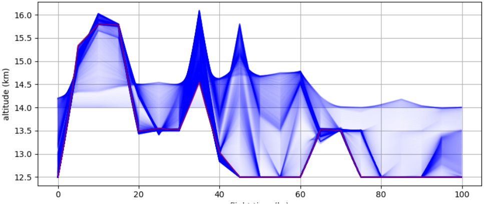
  where $-1 \leq \lambda(t) \leq 1$ is sampled from realistic balloon trajectories (e.g. stochastic altitude controller simulations).
- Note we introduced randomness into the simulation without blowing up the size of the state space!
- If we let $\lambda(t) = 0$, we recover certainty equivalent formulation above.
- If we sample multiple random $\lambda(t)$, we can build a stochastic value function:

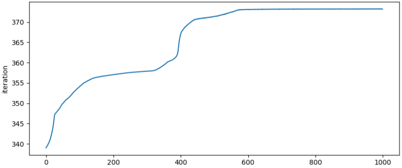
$$\tilde{V} = \sum_{\text{random } \lambda}^{K} V(t_f; \lambda)$$

- If we increase K we capture the variance distribution with more fidelity.

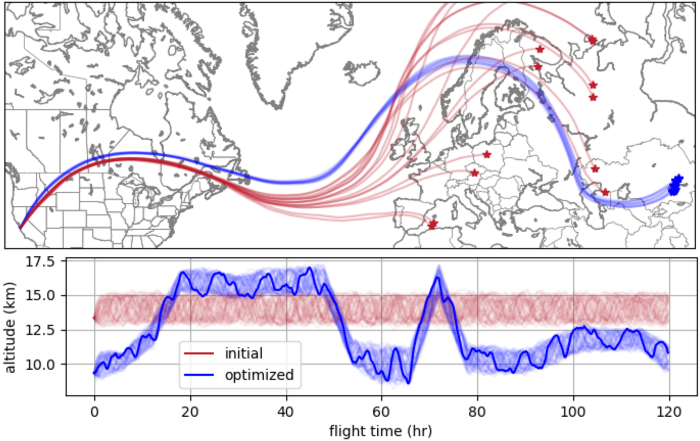
Certainty Equivalent





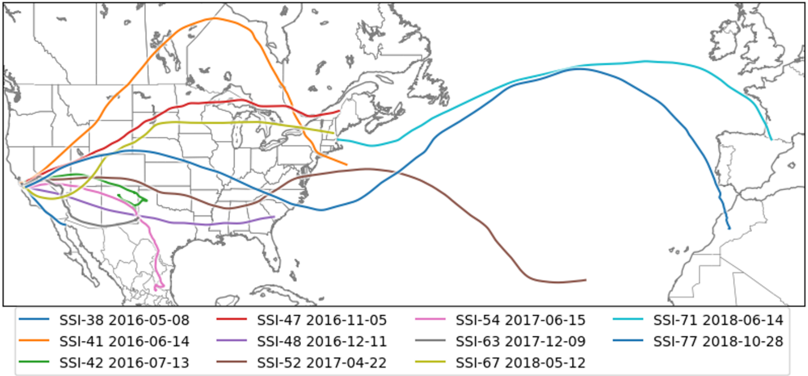


Stochastic



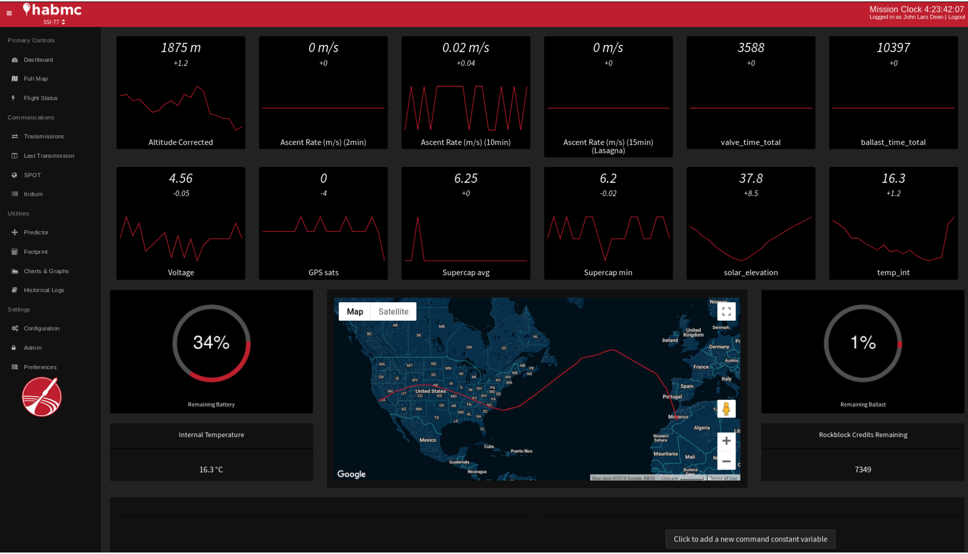
17.5
15.0
12.5
10.0
altitude (km)
initial
optimized
0
20
40
60
80
100
120
flight time (hr)

Valbal Flights

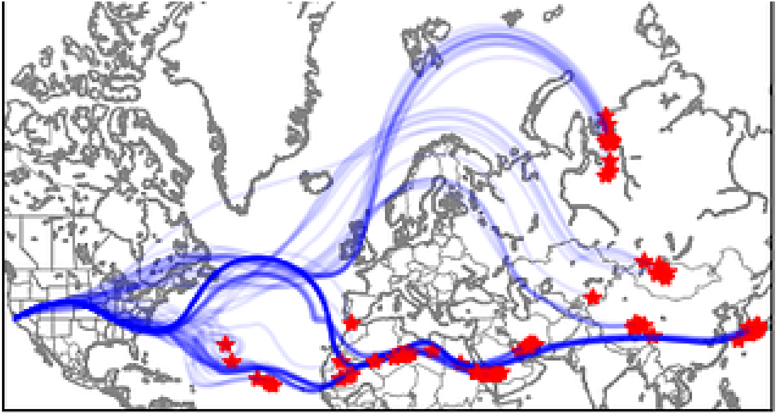


SSI-77 Launch (Oct 28)

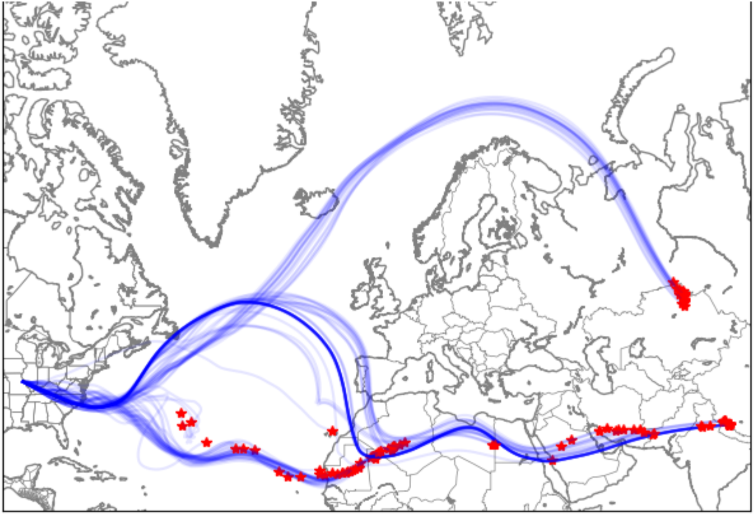
Mission Control



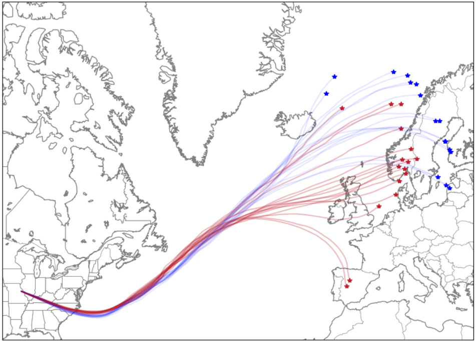
Flight Optimization

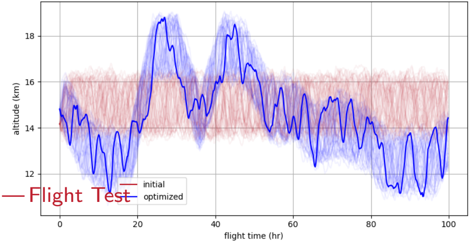


Flight Optimization

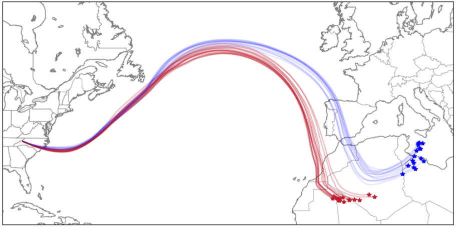


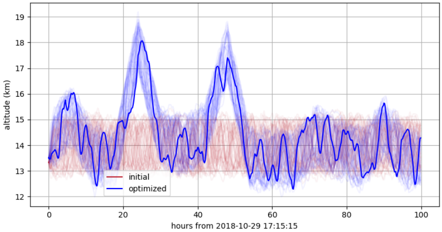
Flight Optimization



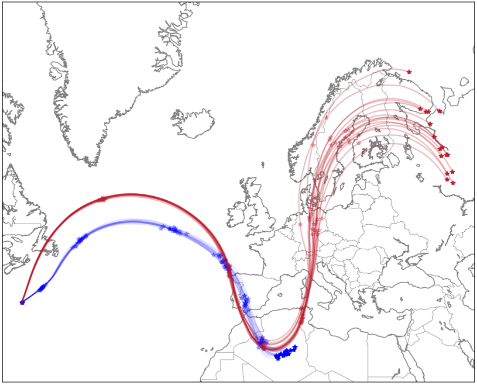


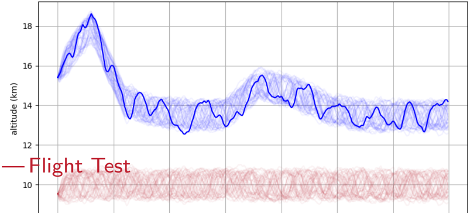
Flight Optimization



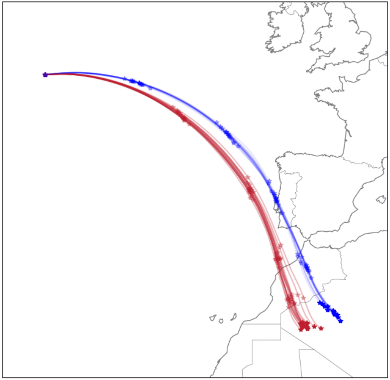


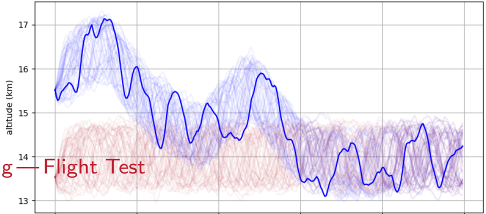
Flight Optimization



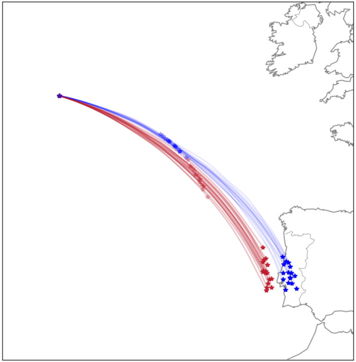


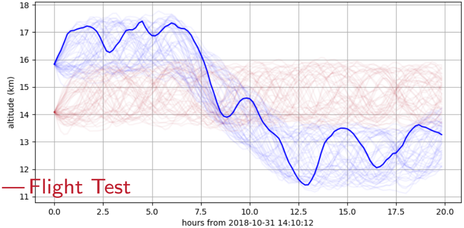
Flight Optimization



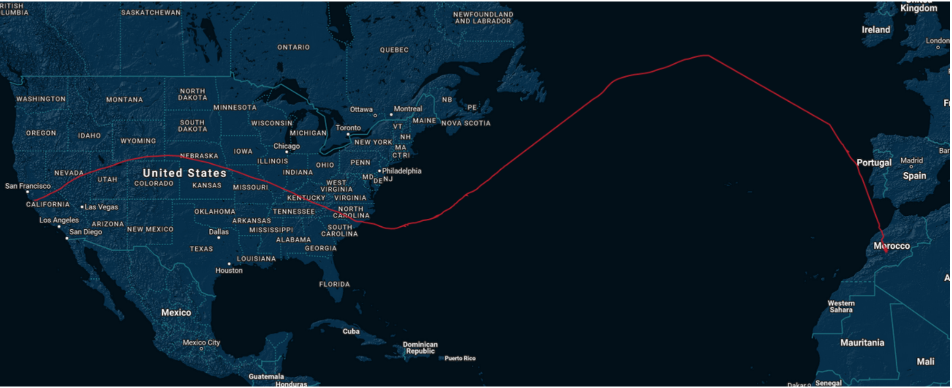


Flight Optimization





Flight Optimization



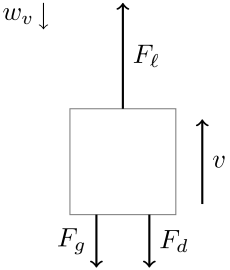
Architecture

- core of simulator written in C++ for efficiency
- automatic differentiation with Adept
- simulation classes template defined for standard computation use as well as differentiable computation
- processed atmosphere data efficiently cached using Linux `mmap`
- 300,000 stochastic 100hr sims per second
  - Ryzen 2700x 8C16T 3.9Ghz CPU
  - about $10\times$ slow for differentiable sims

Outline

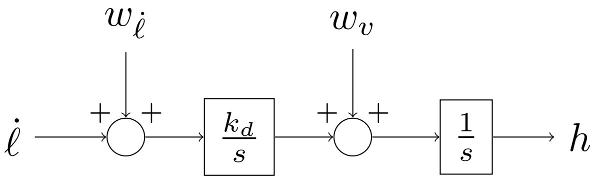
System Dynamics

- Assumptions
  - $F_d \propto v$ i.e. drag is linear.
  - $F_l - F_g = F_d$ i.e. the balloon is always at terminal velocity
- Equations of motion
  - let $\ell = F_\ell - F_g$ be the net lift on the balloon
  - $\dot{\ell}$ is commanded by controller
  - $\dot{v}(t) = k_d(\dot{\ell}(t) + w_{\dot{\ell}}(t))$
  - $\dot{h}(t) = v(t) + w_v(t)$
  - $\mathcal{L}\{h(t)/\dot{\ell}(t)\} = k_d/s^2$



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

Plant Block Diagram

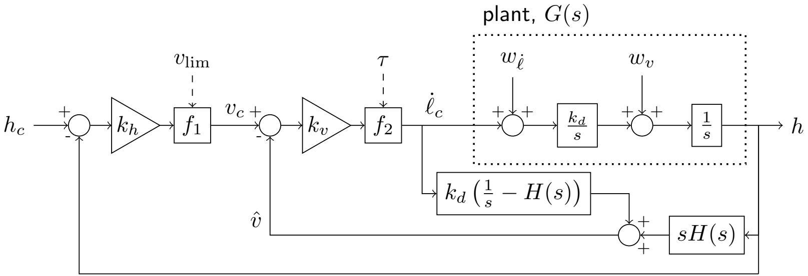


$\dot{\ell}$: commanded change in lift (valve and ballast actions)
$w_{\dot{\ell}}$: atmospheric lift disturbance
w_v: atmosphereic velocity disturbance
h: altitude

$$x = \begin{bmatrix} h \\ v \end{bmatrix} \qquad u = \dot{\ell}$$

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ k_d \end{bmatrix} u + \begin{bmatrix} w_v \\ k_d w_{\dot{\ell}} \end{bmatrix}$$

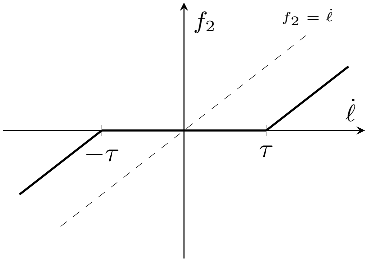
Controller Block Diagram



$H(s)$	low-pass filter
$f_1(v\,;v_{\lim})$	clamp on the velocity commanded by the altitude loop set by $v_{\lim}$
$f_2(\dot{\ell}\,;\tau)$	deadband on the controller effort set by τ
h_c	commanded altitude (set by Flight Controller)
v_c	commanded velocity (output of position loop)
$\dot{\ell}_c$	commanded change in lift per unit time (output of velocity loop)
$w_{\dot{\ell}}$	atmospheric disturbances that change balloon lift (heating/cooling)
w_v	atmospheric disturbances that change balloon velocity (turbulence)
h	balloon altitude
$\hat{v}$	estimate of velocity

Controller Deadband

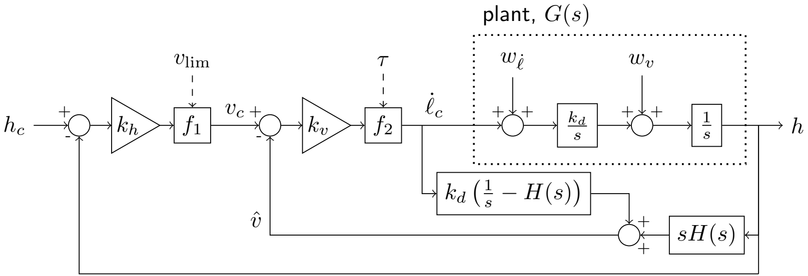
Since we typically command a target altitude and an allowable region, we add a deadband to the controller output. Let $\dot{l}_o$ be the output of the nonlinearity. Deadband:



$$f_2(\dot{\ell}\,;\tau) = \begin{cases} 0 & |\dot{\ell}| < \tau \\ \textbf{sign}(\dot{\ell})(|\dot{\ell}| - \tau) & |\dot{\ell}| > \tau \end{cases}$$

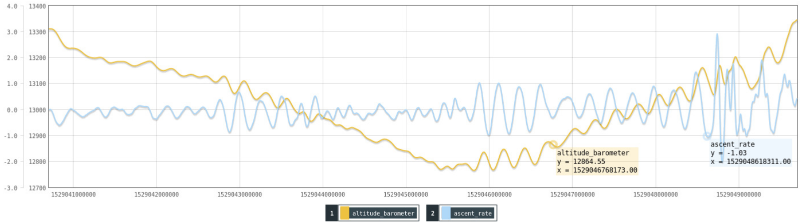
To set bounds on the altutude, we set $\tau = e_{\text{tol}} k_v k_h$, where e_{tol} is the allowable distance from the altitude command.

Controller Block Diagram



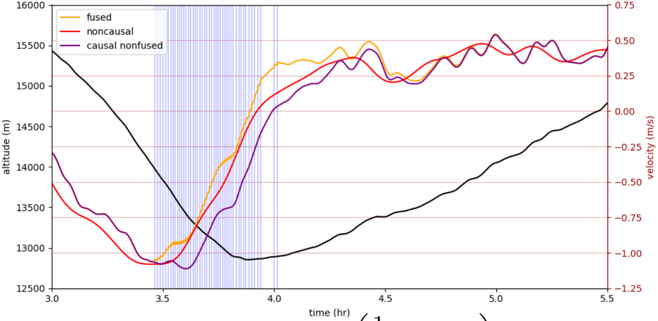
$H(s)$	low-pass filter
$f_1(v\,;v_{\text{lim}})$	clamp on the velocity commanded by the altitude loop set by v_{lim}
$f_2(\dot{\ell}\,;\tau)$	deadband on the controller effort set by τ
h_c	commanded altitude (set by Flight Controller)
v_c	commanded velocity (output of position loop)
$\dot{\ell}_c$	commanded change in lift per unit time (output of velocity loop)
$w_{\dot{\ell}}$	atmospheric disturbances that change balloon lift (heating/cooling)
w_v	atmospheric disturbances that change balloon velocity (turbulence)
h	balloon altitude
$\hat{v}$	estimate of velocity

Atmosphere Waves



Velocity Estimator

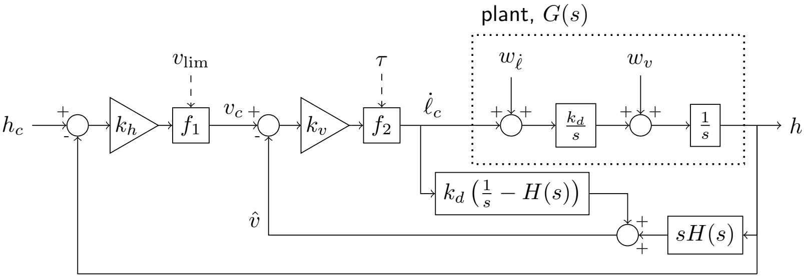
Low pass filtered velocity estimate uses a 2nd order filter to remove the effect of atmospheric waves



$$\mathcal{L}\{\hat{v}\} = sH(s)\mathcal{L}\{h\} + \left(\frac{1}{s} - H(s)\right)\mathcal{L}\{\dot{l}_c\}$$

$H(s)$ a 2nd order lowpass filter

Controller Block Diagram



$H(s)$	low-pass filter
$f_1(v\,;v_{\lim})$	clamp on the velocity commanded by the altitude loop set by $v_{\lim}$
$f_2(\dot\ell\,;\tau)$	deadband on the controller effort set by τ
h_c	commanded altitude (set by Flight Controller)
v_c	commanded velocity (output of position loop)
$\dot\ell_c$	commanded change in lift per unit time (output of velocity loop)
$w_{\dot\ell}$	atmospheric disturbances that change balloon lift (heating/cooling)
w_v	atmospheric disturbances that change balloon velocity (turbulence)
h	balloon altitude
$\hat v$	estimate of velocity

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

$$T(s) = \frac{k_h k_v k_d}{s^2 + k_v k_d s + k_h k_v k_d}.$$

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

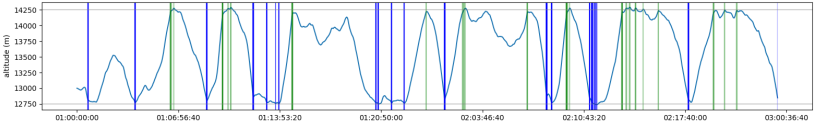
- We choose gains such that $\zeta > 1$ and we have over damping.
- This gives ratio between k_v and k_h, but what about magnitiude?
- high gain $\rightarrow$ controller waits and acts agressively near e_{tol}
- low gain $\rightarrow$ controller acts cautiously before e_{tol}

demonstraited on next slide

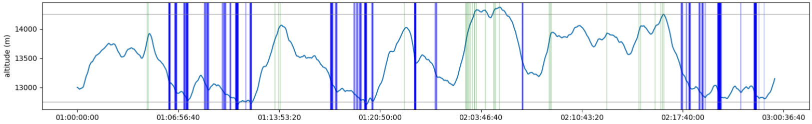
High vs Low Gain

Plots of simulation shown

high gain

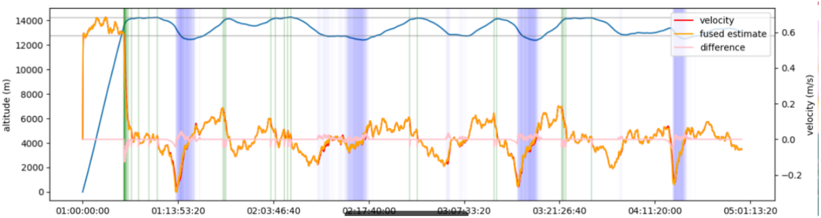


low gain

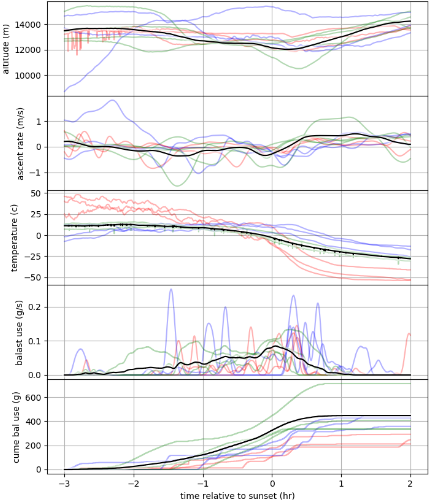


High gain performs better but can't tolerate uncertainty, low gain is worse but performs better under uncertainty

Simulations



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