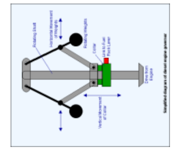
What drives all progress?

Or what enhances self preservation?

Conservation of angular momentum

Watt speed governor



Drive shaft speed

Almost all men are greedy and ambitious

Checks and balances

Legal, social and economic

Mechanics and aerodynamics

Sperry autopilot

Aircraft attitude

Maxwell electrodynamics

PLL, DLL, Power Control

Signal frequency, ID, and power levels

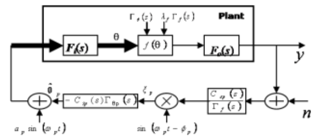
Underlying Model | **System** | **Uncertainty reduced**

What drives all progress?

Or what enhances self preservation?

Wiener-Hammerstein (Ariyur, UCSD)

Extremum seeker



Closeness to optimum (to within sensor/actuator limits)

Thermal-fluid sciences and statistics/estimation (Ariyur, Honeywell)

Gas turbine health monitor

Time of component failure (reduce from 1000s to 10s of hours)

Mechanics and aerodynamics (Ariyur, Purdue)

Coordinated UAV flight

Location, orientation and speeds of all aircraft

Maxwell electrodynamics (Ariyur & Kulatunga, Purdue)

Vehicle traction control

Road traction forces

Underlying Model | **System** | **Uncertainty reduced**

SMALLER ACCOMPLISHMENTS!

Components of Autonomy

- Sensing
- Sensing systems and estimators
- Control systems—adapting to changes
- Optimization—tactics
- Gaming—strategy

Sensing

Collaborators:

Yan Cui

Magnetic Mapping

- GPS system not accessible in:
  - urban and natural canyons;
  - forests;
  - indoor locations.

indoor locations *

urban canyon

- Current geolocation algorithms:
  - received signal strength (RSS), error 3~5m
  - time difference of arrival (TDOA), error 2~3m
  - radio frequency identification (RFID), error 3~5m
  - <u>our algorithm (Magnetic map), error: 1.5~2m</u>

* Figure available from http://www.gpsbites.com/indoor-location-positioning-krulwich-interview

Building a Magnetic Map

- Magnetic Field Model:

$$\forall m(x, y) \in M,$$
$$m(x, y) = m_{IGRF} + m_{Bias}$$

➢ Measurement:

$$z_m = R_{3\times3} \cdot (m_{IGRF} + m_{Bias} + m_{Noise})$$

$$\|z_m\| = \sqrt{z_{mx}^{\ \ 2} + z_{my}^{\ \ 2} + z_{mz}^{\ \ 2}} \qquad \|R_{3\times3}\| = 1$$

M : magnetic map;

$R_{3\times3}$: rotation matrix;

m_{IGRF} : International Geometric Reference Field;

m_{Bias} : local bias;

m_{Noise} : measurement noise.

- # Hardware:

➢ Samsung Galaxy Note II

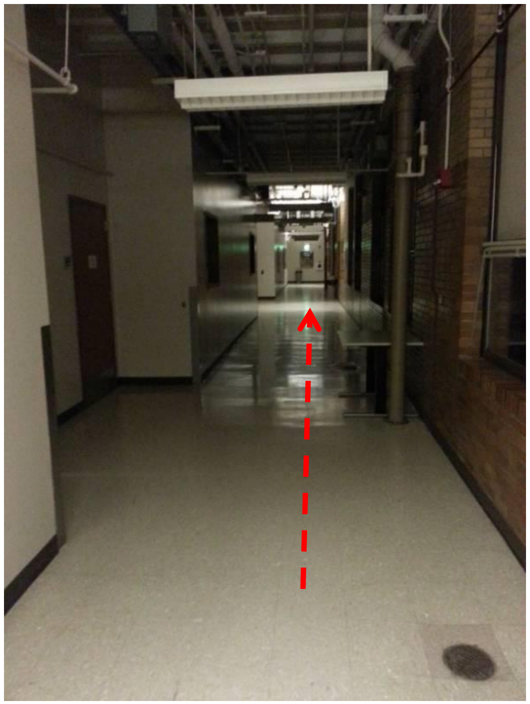
➢ Measurement unit: (μT)

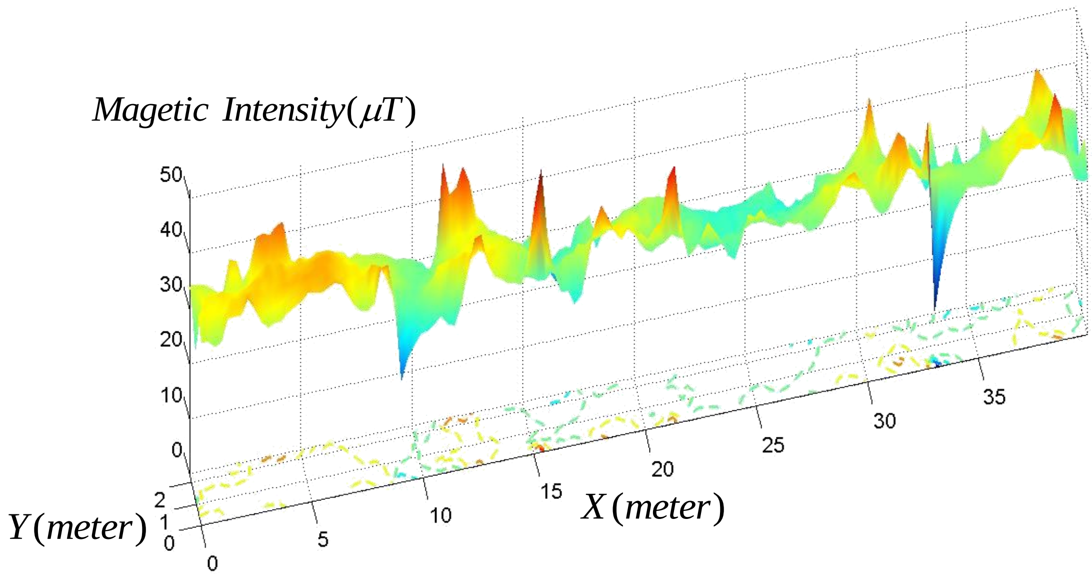
➢ Measurement noise:

$$\sigma^2_{noise} = 0.0734(Gauss^2)$$

- # Floor plan:

➢ Part of Mechanical Engineering Building (2^{nd} floor), Purdue University





Magetic Intensity(μT)
50
40
30
20
10
0
$Y(meter)$
2
1
0
0
5
10
15
20
25
30
35
$X(meter)$

Static Estimation

- Measurement:

$$z_m = R_{3\times 3} \cdot (m_{IGRF} + m_{Bias} + m_{Noise})$$

$$\left\| z_m \right\| = \sqrt{{z_{mx}}^2 + {z_{my}}^2 + {z_{mz}}^2}$$

- Optimization:
  - Cost Function, $c(p_t)$:

$$c(p_t) = \left| \left\| z_m \right\| - m(x, y) \right|$$

$$p = \arg\min_{p \in V} c(p_t)$$

Static Estimation

- Static Estimation Algorithm

  - Definition of Wrapper[6]:

    P: a set;
    IP: subset of **P**;
    IP is a set of *wrappers* for **P**,
    if it satisfies:

    1)P and each singleton of P belong to P;
    2)IP is closed by intersection.

    $\mathbf{P_1}$: a subset of **P**;
    The smallest wrapper [$\mathbf{P_1}$]:

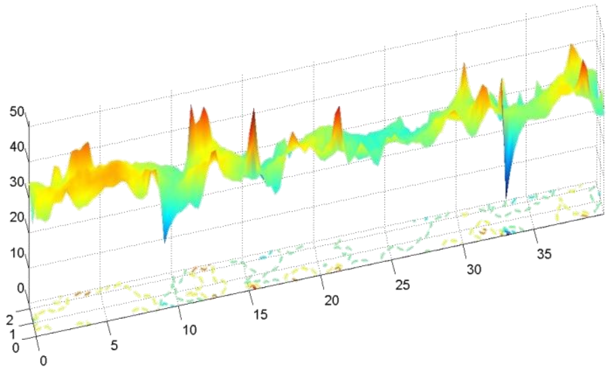
$$[\mathbf{P}_1] = \bigcap \{\mathbf{X} \in \mathbf{IP} \mid \mathbf{P}_1 \subset \mathbf{X}\}$$

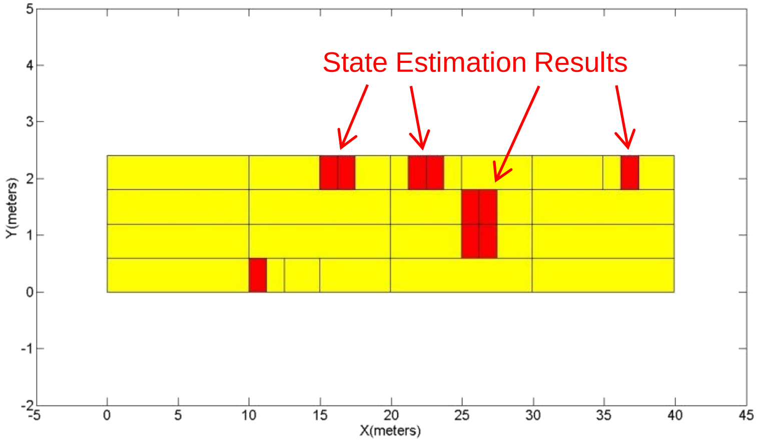
Algorithm:

1. Define the wrapper set $[p]$ for initial guess from $[V]$;
% wrapper definition can be found in [8].
Create an relatively big vector set
$B = \{([p], c(p) = \infty)\}$;
% to store the entire points and cost values
2. Separate $[p]$ into several smaller sets, denoted as $[p_i]$;
3. Iterations:
$[p] = [p_i]$
if ($[p] \neq empty$) then,
if ($w([p]) < \varepsilon$) then, % width of $[p]$
$S(:,i) = ([p], \min(c([p])))^T$
% to store coordinates and minimal
% cost function values
else
bisect $[p]$ into $[p_1]$ and $[p_2]$;
$\{\mathrm{b}_i \in B \mid \mathrm{b}(:,2i-1) = ([p_1], \min(c(p)))^T$
$and\ \mathrm{b}(:,2i) = ([p_1], \min(c(p)))^T\}$
Iterations end until $B = empty$.
4. The estimated interval $[\hat{\mathrm{s}}]$ can be obtained from S
if $c_i \in S < \bar{c}$
% if the cost value is lower than upper bound.
$([p_i], c(p_i)) \in [\hat{\mathrm{s}}]$
% store this into candidate intervals
else
$([p_i], c(p_i)) \notin [\hat{\mathrm{s}}]$ % otherwise, remove it.
end

Experimental Results

- Candidate locations are well bounded within several small 0.6m×1.25m intervals.





Bounded region from static estimation

Sensing Systems and Estimators

Collaborators:

John Barnes and Cheng Liu

Geolocation Approaches

Given the sun vector, moon vector or other celestial reference vectors, how to accurately obtain the location?

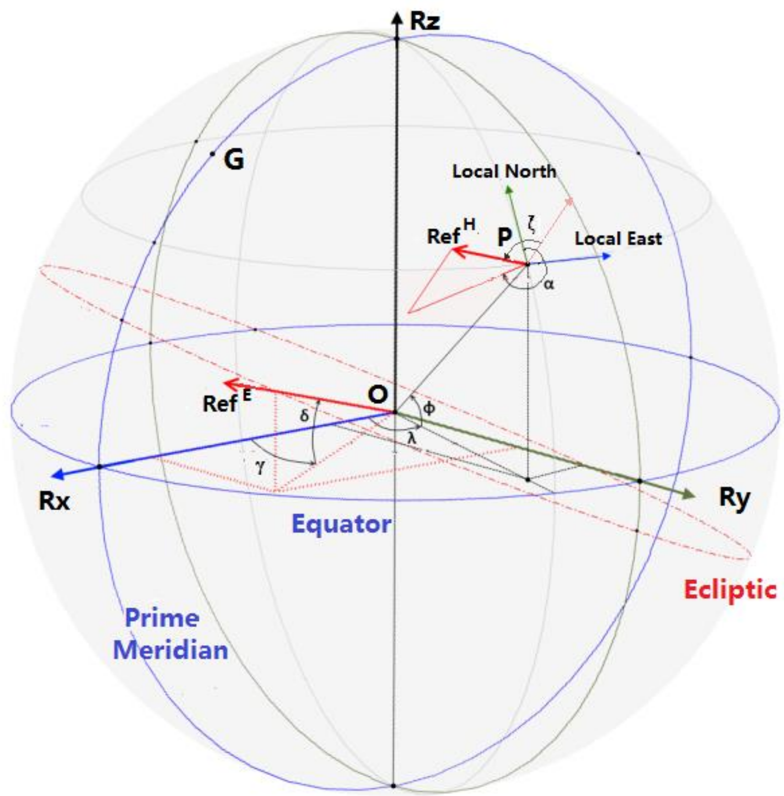
- Approach 1 [10].

$$\mathrm{Ref}^{E} = \begin{pmatrix} \cos(\gamma)\cos(\delta) \\ \sin(\gamma)\cos(\delta) \\ \sin(\delta) \end{pmatrix}, \text{and } \mathrm{Ref}^{\mathrm{H}} = \begin{pmatrix} \cos(\alpha)\cos(\zeta) \\ \sin(\alpha)\cos(\zeta) \\ \sin(\zeta) \end{pmatrix}$$

$$\mathrm{Ref}^{H} = R_{(\phi,\lambda)}\mathrm{Ref}^{E}.$$

Cons:

1. Cumbersome calculation.
2. In practical usage , there is no permanent aid of true north to determine celestial reference vectors.



Mathematical transformations between three coordinates.

Geolocation Approaches

- Approach 2: Intercept Method
  Drawing the Line of Position (LOP) for several celestial objects and find the intersected location [11].

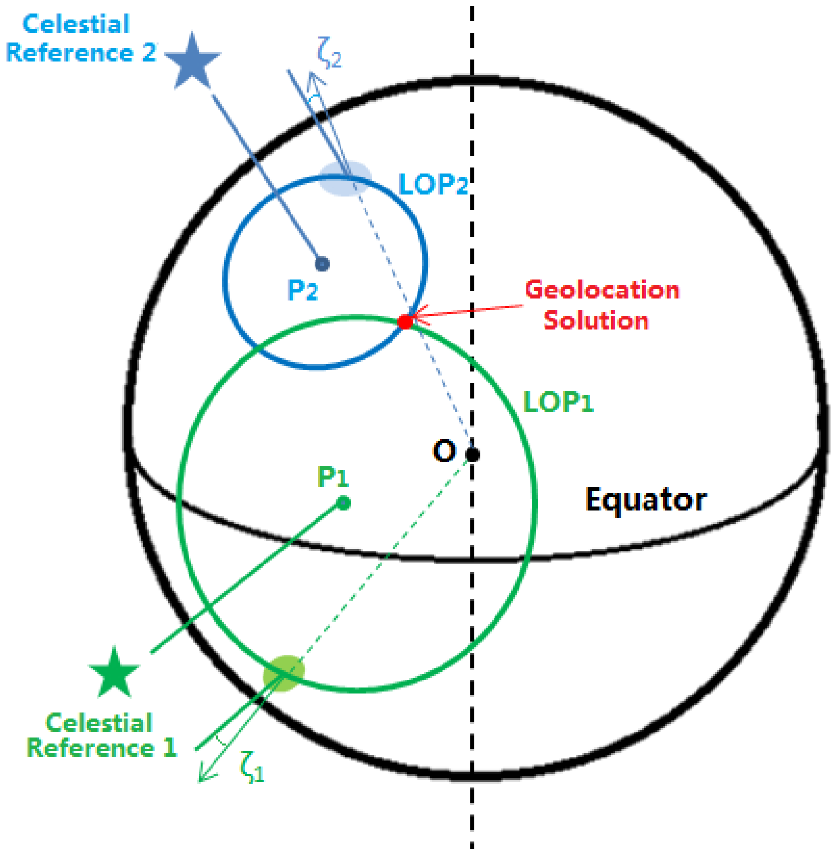
  Pros:

  1.Straightforward.

  2.No need of the direction of true north.

  Cons:

  1. Only applicable when more than two celestial objects can be observed.
  2. As a manual work, accuracy and speed is low.



Intercept Method and Line of Position (LOP).

Generalized Intercept Method

- Define a "Set of Position (SOP)" instead of Line of Position (LOP) as,

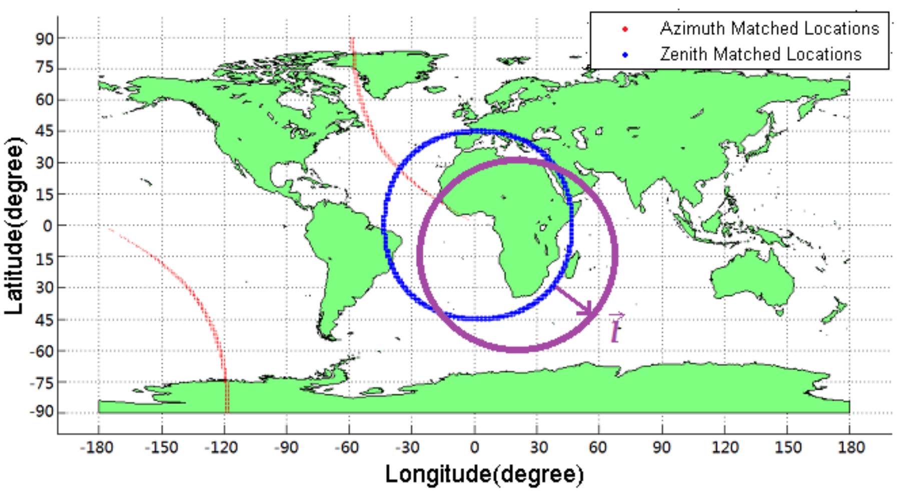
$$\mathrm{SOP}(\mathrm{Ref}\,,t) = \{P(\lambda,\phi,h) \in \mathbb{L} : |\, f(P,t) - \mathrm{Ref}\,| \le \varepsilon\}.$$

ε : expected measurement error

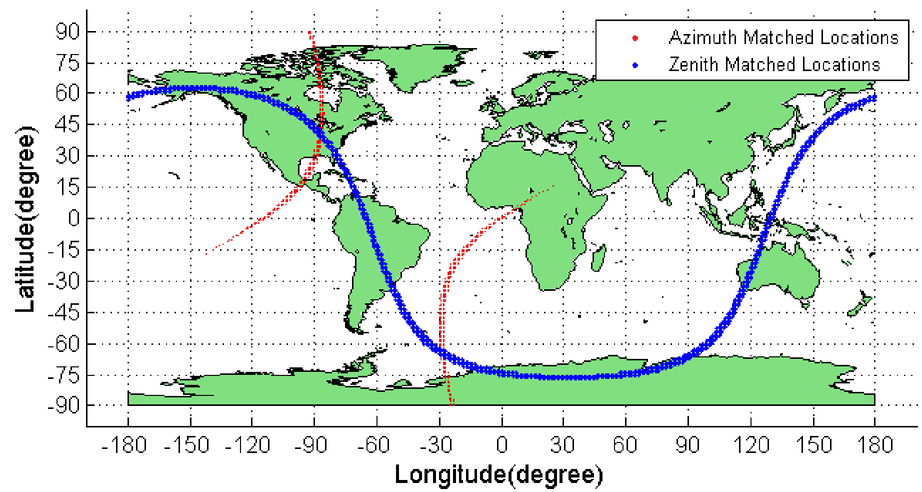
- Define a "successive Set of Position (sSOP)" to apply previous SOPs,

$$\mathrm{sSOP}(\mathrm{Ref}\,,t,\vec{l}\,) = \{P^{'}(\lambda,\phi,h) : \forall P \in \mathrm{SOP}(\mathrm{Ref}\,,t), P^{'} = P + \vec{l}\,\}.$$

- "Ref" can be both azimuth or zenith for drawing SOPs.



a. SOP and sSOP when on solution exists.



b. SOPs for two solutions.

SOP and sSOP for both azimuth and zenith.

Generalized Intercept Method

- At an unknown location $P(\lambda,\phi,h)$, if n SOP and m sSOP can be estimated independently, we can find the intersection of them such that,

$$\mathbb{P} = SOP_1 \cap SOP_2 \cap \ldots \cap SOP_n \cap sSOP_1 \cap sSOP_2 \cap \ldots \cap sSOP_m \neq \varnothing,$$
$$P(\lambda,\phi,h) \in \mathbb{P}.$$

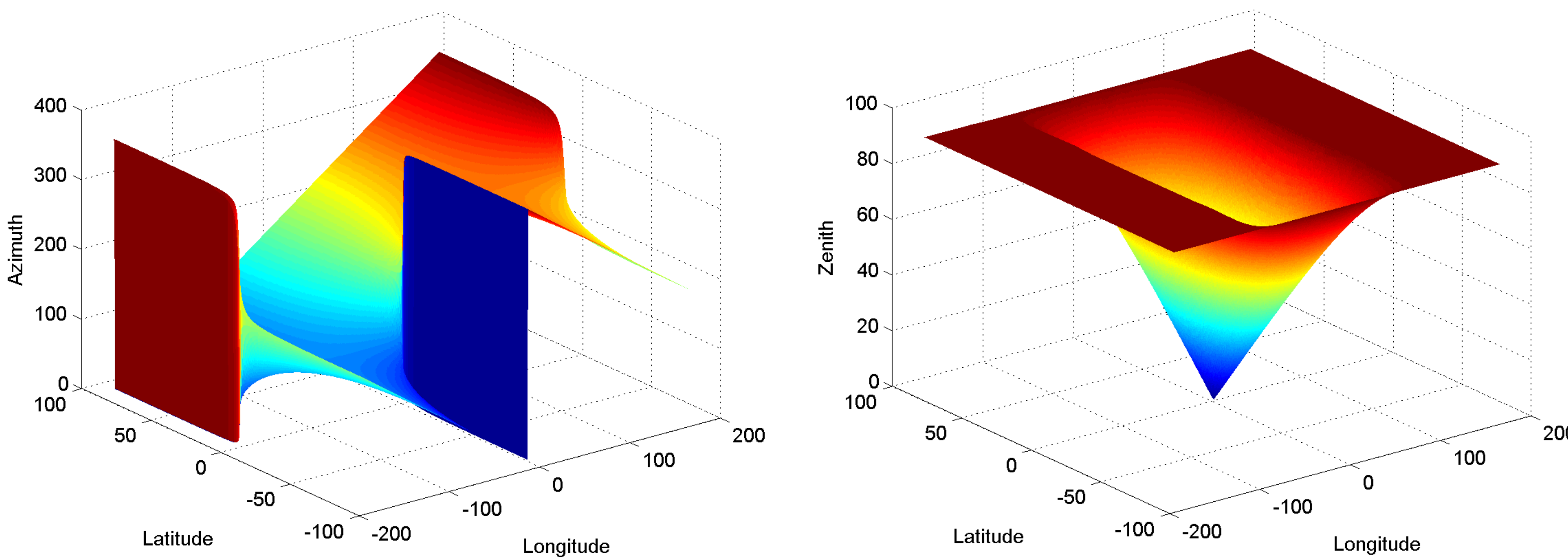
- If the area of $\mathbb{P}$ becomes small enough, it can be treated as the geolocation result: a solution set of area instead of a single location point.

- At least two SOPs or sSOPs are required for geolocation. Possible choices:
  1. SOP(Z1, t1), SOP(Z2, t2);
  2. SOP(A1, t1), SOP(Z1, t2);
  3. sSOP(Ref, t1, *l*), SOP(Ref, t2);

- More and distinct SOP and sSOP combinations are preferred since they can largely reduce the size of $\mathbb{P}$.

Iterative Position Matching Algorithm

- Since there are infinite locations on the earth, finding SOPs via enumeration is impossible.
- Due to the spherical shape of the earth, the distribution of "Ref" should be continuous for nearby locations.
- Apply the principle for geolocation via an iterative position matching algorithm.



a. Distribution of azimuth.

b. Distribution of zenith.

Theoretical global distribution of solar azimuth and zenith for all longitudes and latitudes at noon March 20th, 2012.

Iterative Position Matching Algorithm

- Grid of Locations (GoL).
  For a location area $[\lambda_{min}, \lambda_{max}, \phi_{min}, \phi_{max}]$,
  spliting $[\lambda_{min}, \lambda_{max}]$ by δ_{λ},
  and $[\phi_{min}, \phi_{max}]$ by δ_{ϕ} to form a grid.

- Position Node (PN).
  The intersected location points on the GoL,

- Use a threshold ε' to select qualified PNs to form a smaller GoL.

$$PN' = \{PN(i,j) : | f(PN(i,j),t) - \text{Ref} | \le \varepsilon'\},$$
$$GoL' = [min(\lambda_{PN'}), max(\lambda_{PN'}), min(\phi_{PN'}), max(\phi_{PN'})].$$

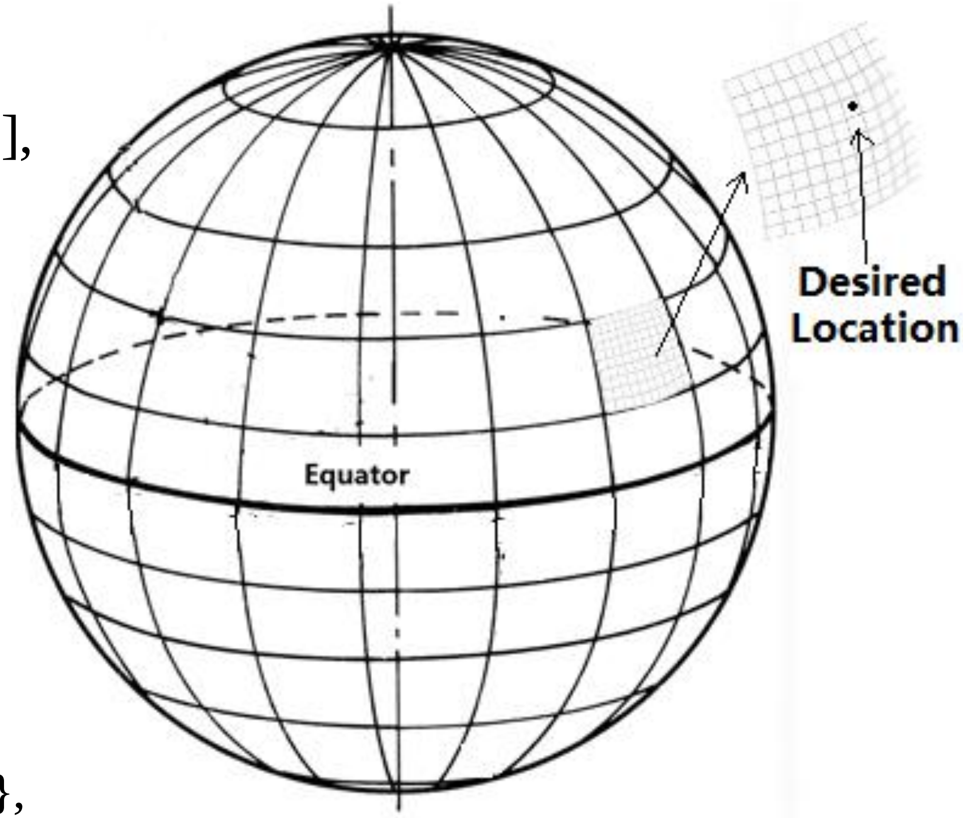


Figure 5.5 Illustration of GoL, PN and the resized new GoL'.

- Repeat the steps for smaller ε' values until the GoL achieves the desired accuracy.

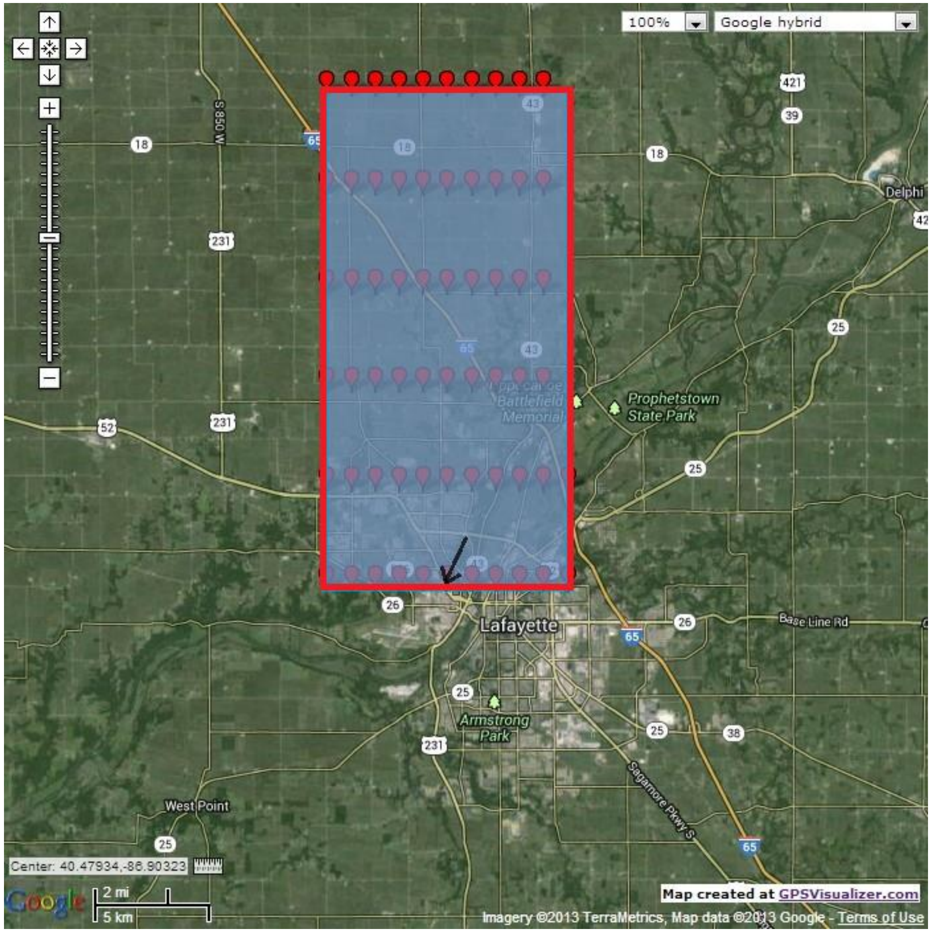
Experimental Results

Inputs:
(A,Z): (185.59°,28.10°) from Magnetic North
Time: UTC 17:49, 04/22/2013
Measurement error: (0.2°,0.1°)

Result:
[-86.975°,-86.85°,40.4297°,40.625°]

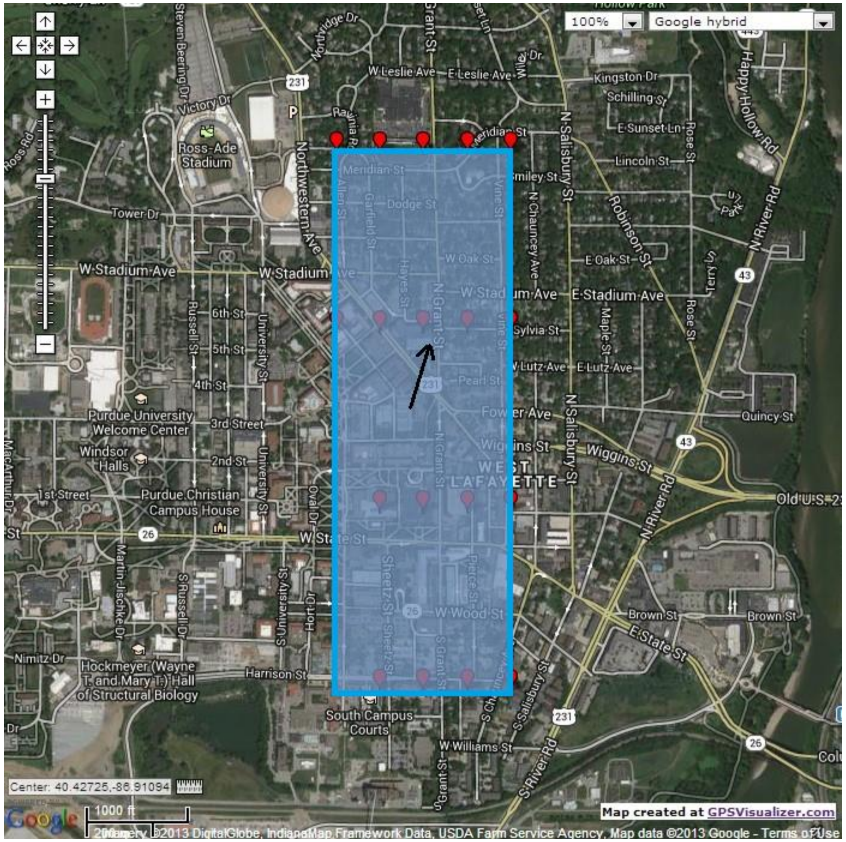
Pros of the method:

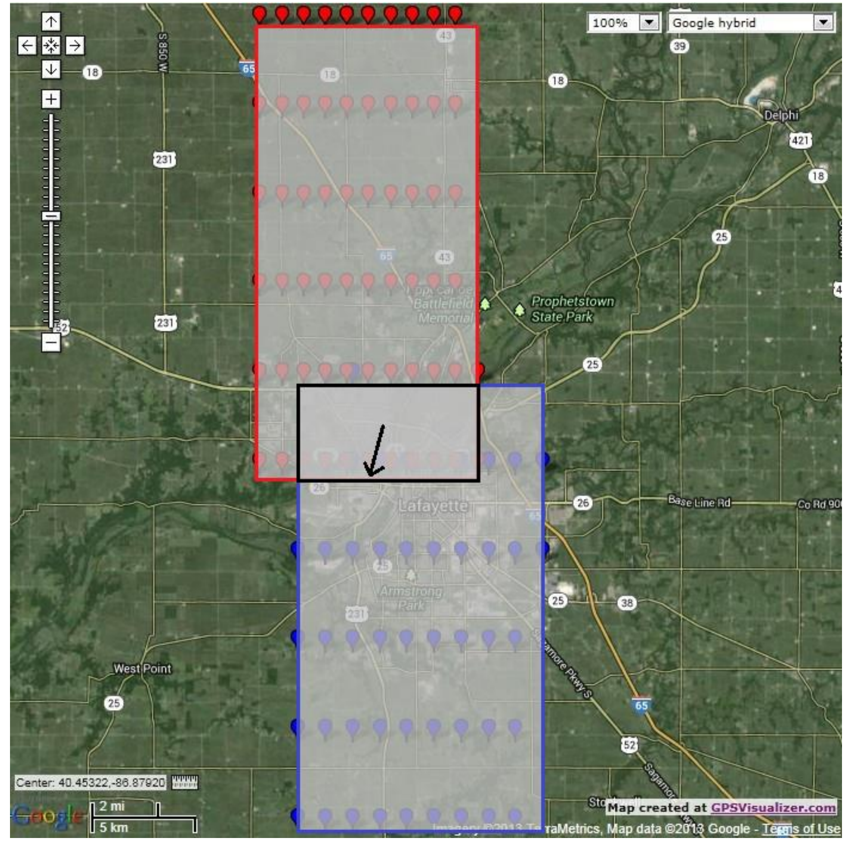
- Fast.
- Accurate.
- Robust to disturbances.
- Fully autonomous and programmable.



Geolocation result of the given sun vector.

Experimental Results





a. Geolocation result for a better estimation within an error of 0.01 °.

b. Improved geolocation result by intersecting two areas.

Approaches for improvements.

Control Systems—Adapting to Changes

Collaborators:

Poorya Haghi and William S. Black

ES-MRAC: A New Paradigm for Adaptive Control

Motivation:

Desired objectives in many engineered systems:

- Controlling the exact trajectories of system
- Dealing with uncertainties

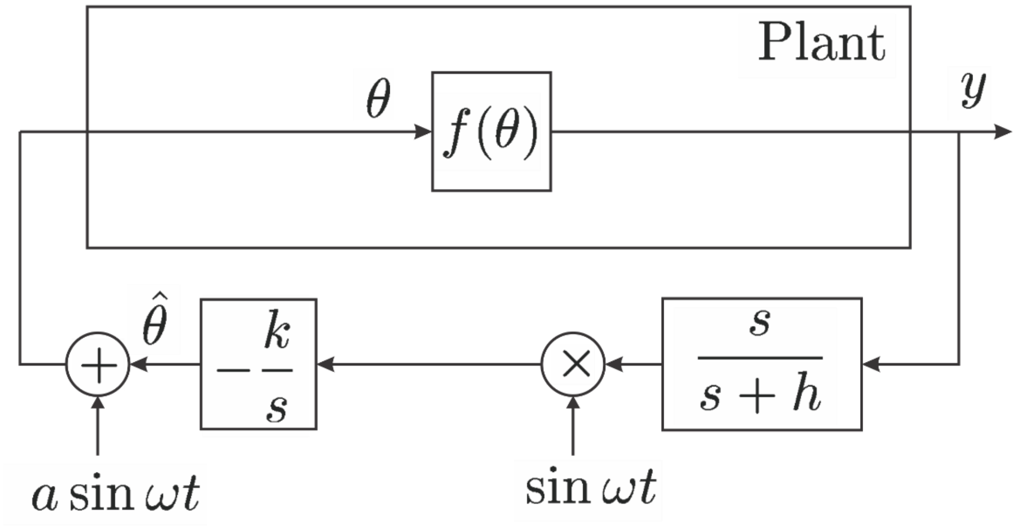
One Solution Category: Adaptive Control

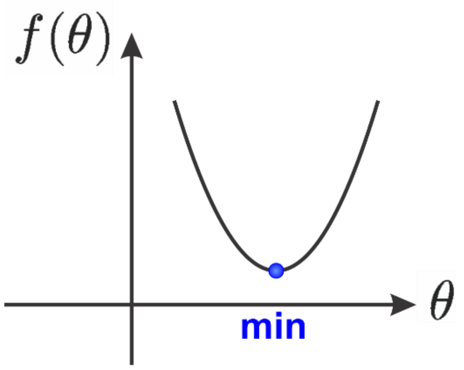
- Model reference adaptive control (MRAC)
- Adaptive feedback linearization
- Adaptive back stepping
- ES-MRAC

ES-MRAC Combines extremum seeking optimization (ES) with MRAC

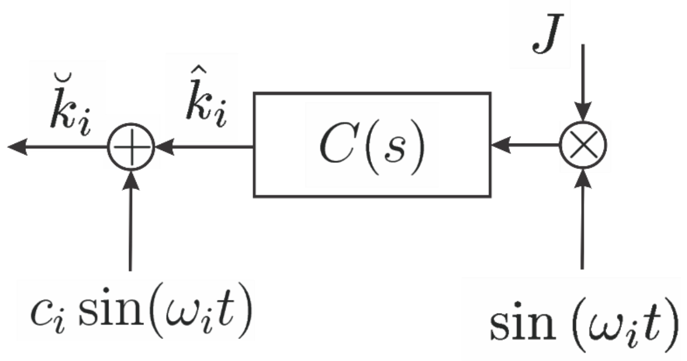
ES-MRAC: A New Paradigm for Adaptive Control

How does ES work?





How we can use this for adaptation:
(the extremum seeker!)



ES-MRAC: A New Paradigm for Adaptive Control

- There are many optimization methods. Why ES?
  - A very strong real-time optimization tool
  - Can change J and C(s)
  - Can use non-sinusoidal perturbations
  - Can create forms and terms that are impossible for other methods to implement
- There are many control methods. Why mix ES with MRAC?
  - MRAC controls both transient and steady state responses
  - Thus, we build a very strong adaptive control method (ES-MRAC)!

ES-MRAC: A New Paradigm for Adaptive Control

LTI Systems

- The system: $a_n y^{(n)} + a_{n-1} y^{(n-1)} + \ldots + a_0 y = u,$
- The reference (desired) model:

$$a_{mn_m} y_m^{(n_m)} + a_{m(n_m-1)} y_m^{(n_m-1)} + \ldots + a_{m0} y_m = r(t)$$

- Plant Assumptions:
  - P1. States are measurable
  - P2. Plant parameters are unknown
- Reference Model Assumptions:
  - M1. The model is stable
  - M2. The plant and the model have the same order

ES-MRAC: A New Paradigm for Adaptive Control

Theorem (Stabilization):

Let the cost function and compensator be

$$J = \frac{1}{2}\left[\mathbf{q}^T\mathbf{e}\right]^2 = \frac{1}{2}\left[\sum_{k=1}^{n} q_i e_i\right]^2 \qquad C_i(s) = -g_i\left(\frac{1+d_i s}{s}\right)$$

Assume that plant assumptions P1 and P2 and model assumptions M1 and M2 hold.

Furthermore, assume that

- The probing frequency for each loop is $\omega_i = n_i\omega$
- Probing frequencies are large and distinct $n_i \neq n_j$ for $i \neq j$
- $O(d_i\omega_i) = O(g_i) = 1$

Then, this setup will guarantee global asymptotic convergence of the tracking error vector, to an O(1/ω) neighborhood of the origin.

ES-MRAC: A New Paradigm for Adaptive Control

Features of the method:

Pros:

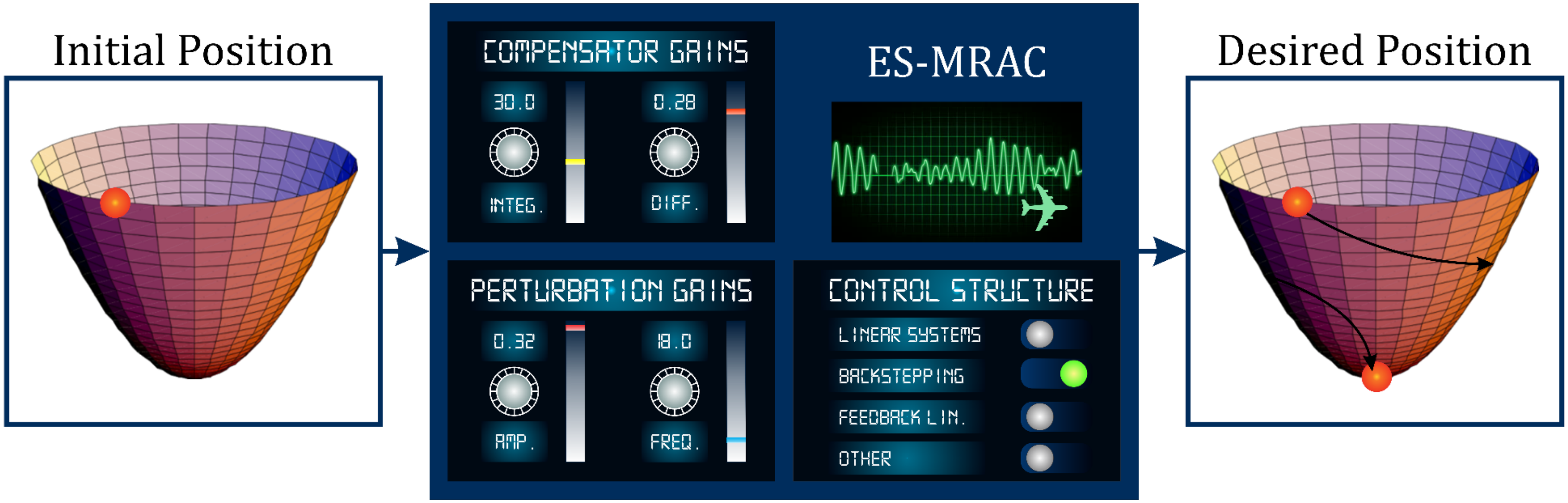
- Real-time adaptation
- Extendible to virtually all other control methods and nonlinear systems
- PE conditions can be explicit

Cons:

- Perturbation frequencies can get very high if the system is large
- Challenging numerical problems in nonlinear systems

ES-MRAC: A New Paradigm for Adaptive Control

Visualizing the Concept of ES-MRAC:



ES-MRAC: A New Paradigm for Adaptive Control

Hypersonic Vehicle Example:

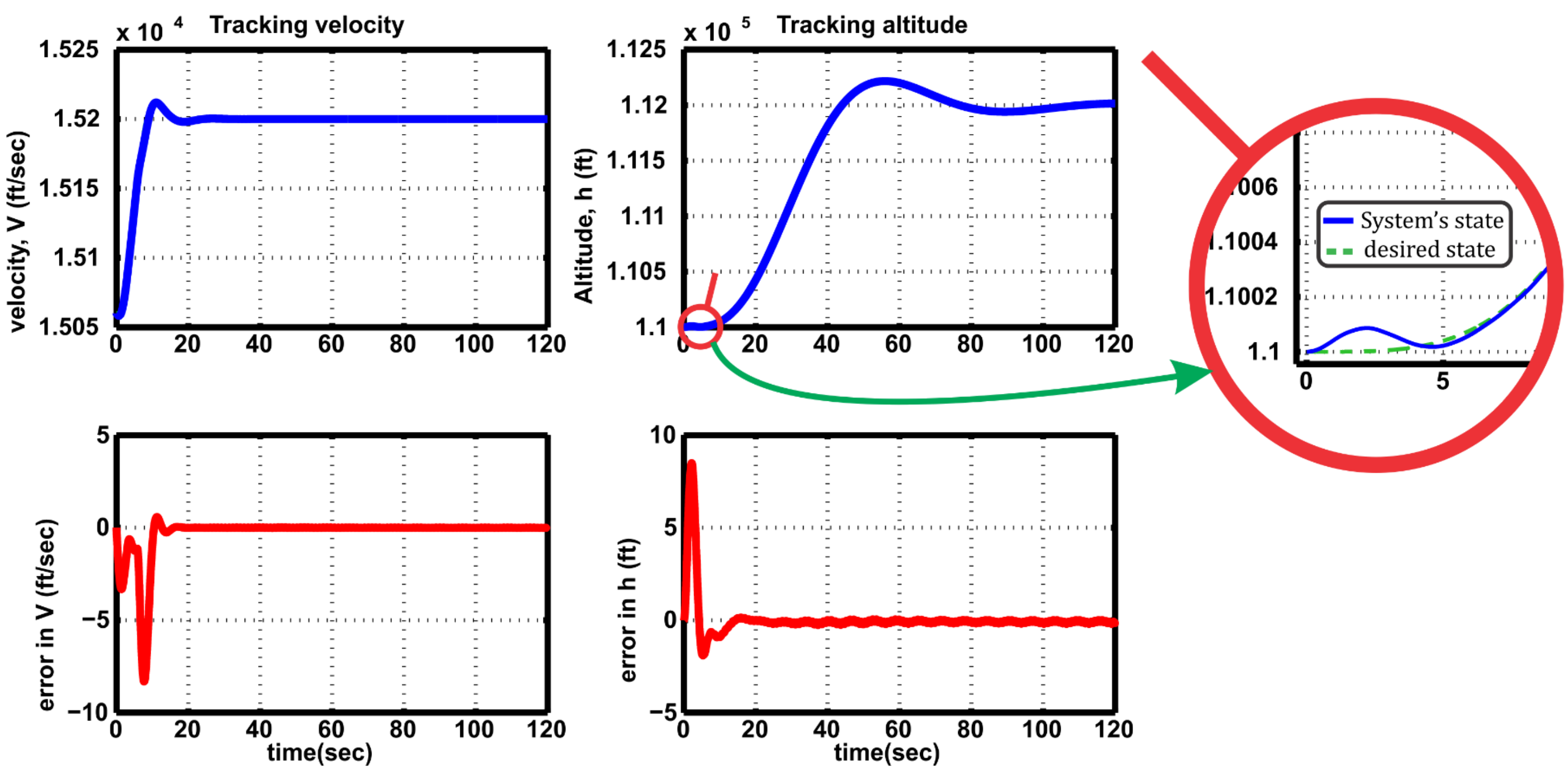
– Objective: Track velocity and altitude under uncertainties

Challenges:

•Highly nonlinear system
•Numerical values range from 10e– 11 to 10e +13
•Relative degree of 7
•After 7 differentiations, each equation is 50 pages long! (Very difficult to linearly Parameterize.)

ES-MRAC: A New Paradigm for Adaptive Control

Hypersonic Vehicles
Simulation Results On a Computer:



Optimization—Tactics

Collaborators:

Sunghun Jung (PhD Fall 2013)

Scalable UAV Operations

•Automated Integrated Surveillance Reconnaissance (ISR) algorithm for multiple UAVs by converting mTSP* to *m* TSP**.

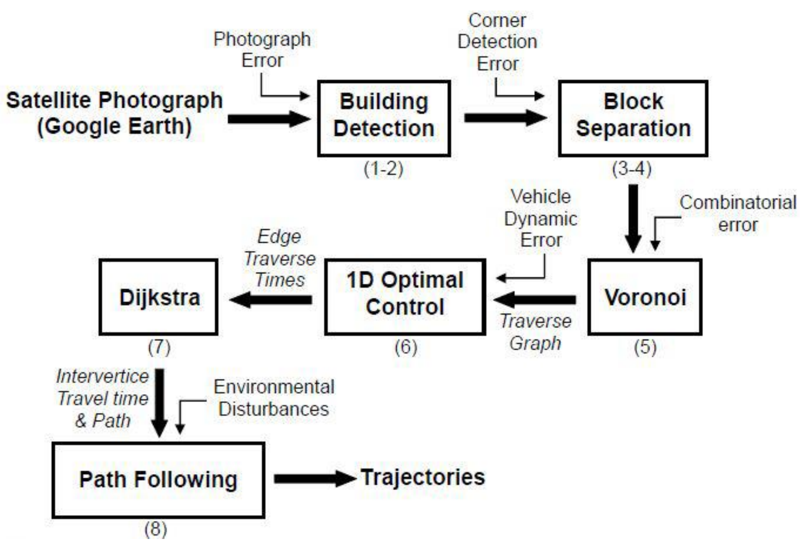


Figure 1.1: Hierarchy of mission planning.

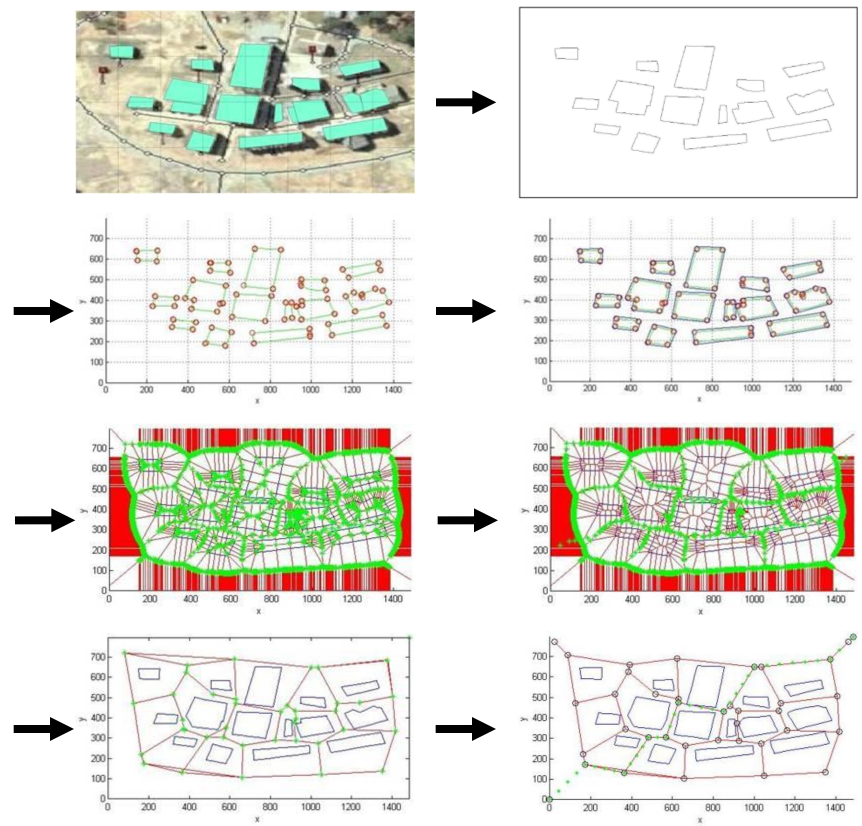


Figure 1.2: Processes to generate a trajectory of a single UAV.

mTSP*: multiple Traveling Salesman Problem, TSP**: Traveling Salesman Problem.

Optimization Problem Formulation

$$\min \quad E\left[t_{mission} = \sum_{k=1}^{m} \left(\left(\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ijk} x_{ijk} \right) + w_k + o_k \right) \right], \quad (5.1)$$

$$\text{s.t.} \quad \begin{aligned} & \sum_{p=1}^{m} \sum_{q=1,\, p \neq q}^{m} \left| t_p - t_q \right| < T_t, \\ & \sum_{j=2}^{n} x_{ijk} = 1 \text{ for any } k, \\ & \sum_{j=2}^{n} x_{jik} = 1 \text{ for any } k, \\ & \sum_{i \neq j} x_{ijk} = 1 \text{ for any } k, \\ & \sum_{i=j} x_{ijk} = 0 \text{ for any } k, \\ & c_{ijk} = G_t(i, j) \quad \text{for } i \neq j, \\ & w_k, o_k \geq 0, \end{aligned}$$

where, T_t = threshold,

c_{ij} = time taken between i^{th} and j^{th} node,

x_{ij} = integer between $\{0,1\}$,

$k = k^{th}$ UAV,

n = number of vertices,

w = time increase due to wind,

o = time increase due to unexpected obstacles,

m = total number of UAVs,

G_t = time array among nodes $(n \times n)$.

Converting mTSP to m TSPs

•Break down mTSP* into *m* TSPs** using two region division methods:

1. Uniform Region Division (URD),
2. K-means Voronoi Region Division (KVRD).

•TSP** with GA:

TSP with *GA* algorithm

---

for $i = 1 : num\_UAV$ **do**
  while $min\_dist > max\_dmat$ **do**
    for $iter = 1 : num\_iter$ **do**
      min_dist = calculated minimum distance from total distance lists;
      if $min\_dist < global\_min$ **then**
        $global\_min = min\_dist$;
        GA: flip, swap, slide operations;
      end if
    end for
  end while
end for

Algorithm 1.1: Algorithm structure of the TSP** with GA***.

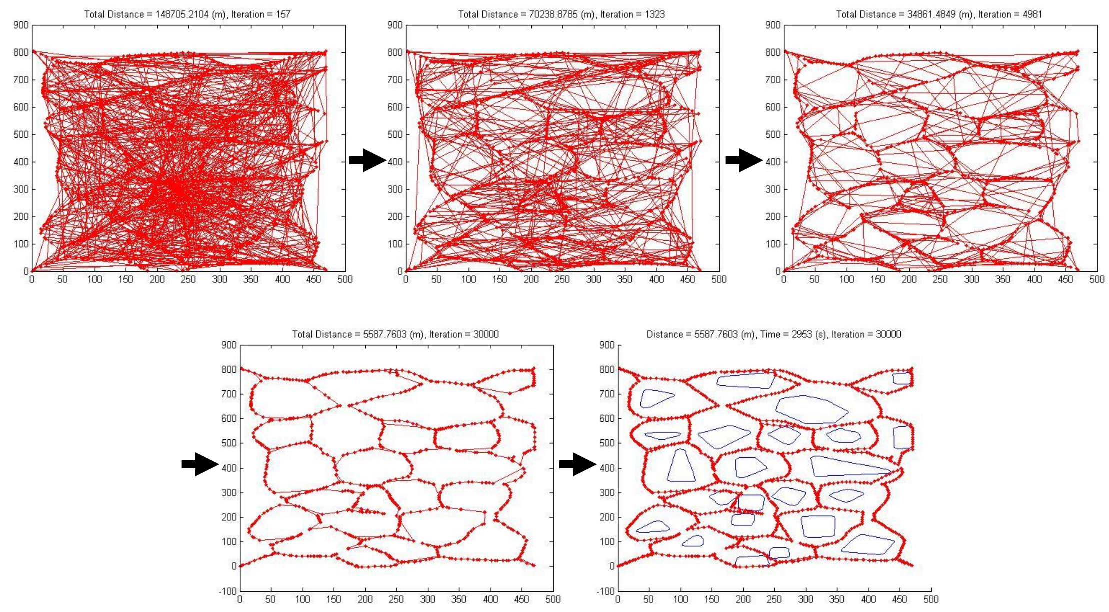
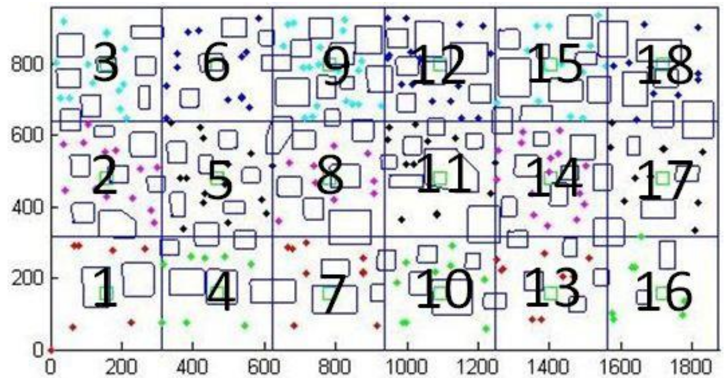
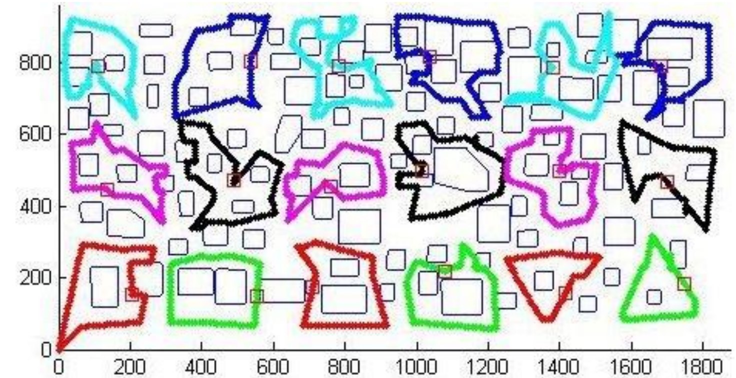


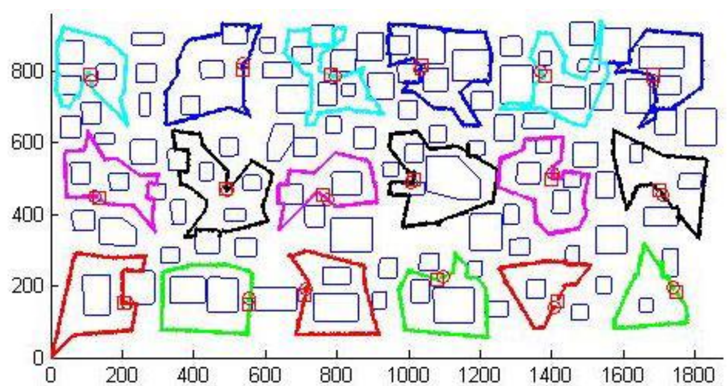
Figure 1.3: Simulation of TSP.

•Uniform Region Division (URD)



(a) Region division.


(b) Preplanned trajectories.


(c) Simulated flight trajectories with fixed wings UAVs.

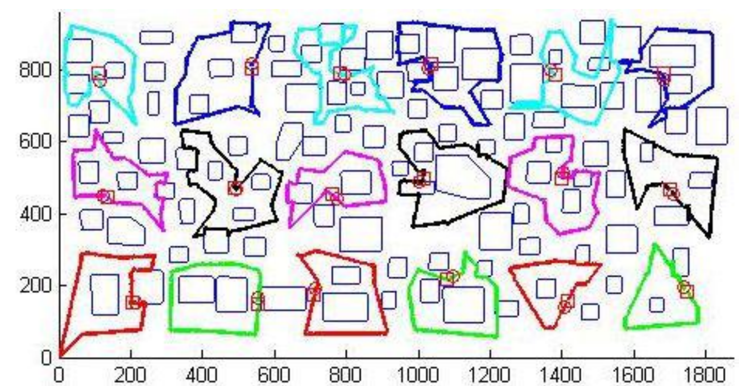
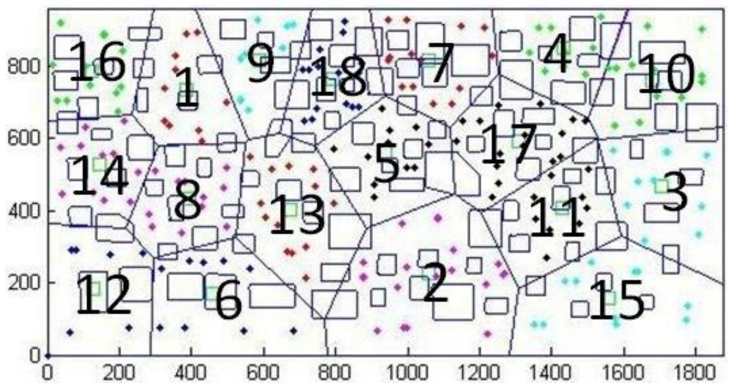
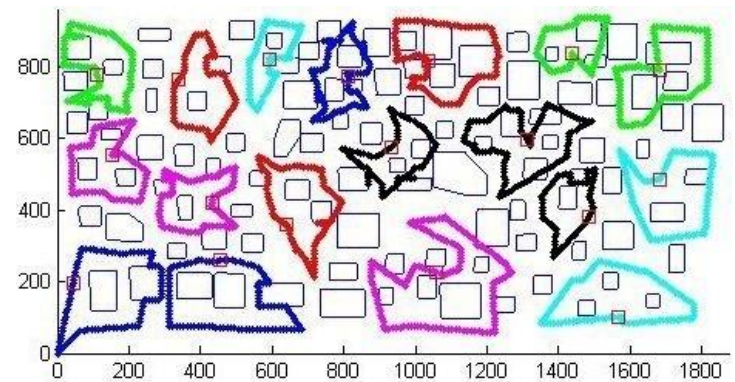

(d) Simulated flight trajectories with hover capable UAVs.

Figure 1.4: Application of URD (103 buildings & 18 UAVs).

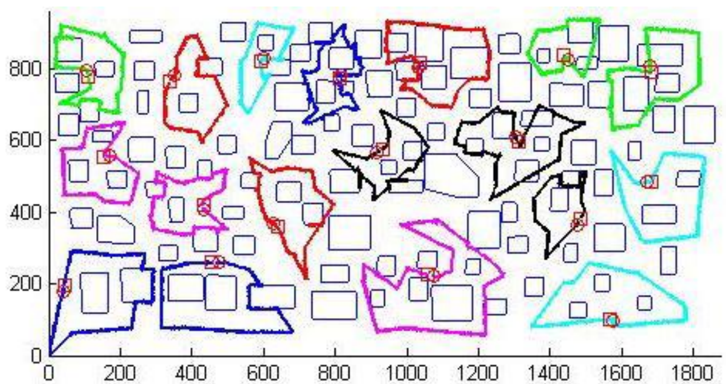
•K-means Voronoi Region Division (KVRD)



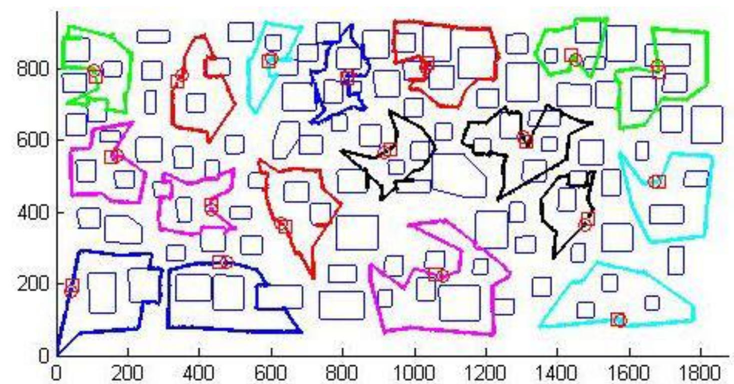
(a) Region division.



(b) Preplanned trajectories.



(c) Simulated flight trajectories with fixed wings UAVs.



(d) Simulated flight trajectories with hover capable UAVs.

Figure 1.5: Application of KVRD (103 buildings & 18 UAVs).

Robustifying the Planner

•Robustness of the automated UAV mission planning can be proved by analyzing disjointed error propagations at each step since each algorithm runs independently, in series, and in one direction.

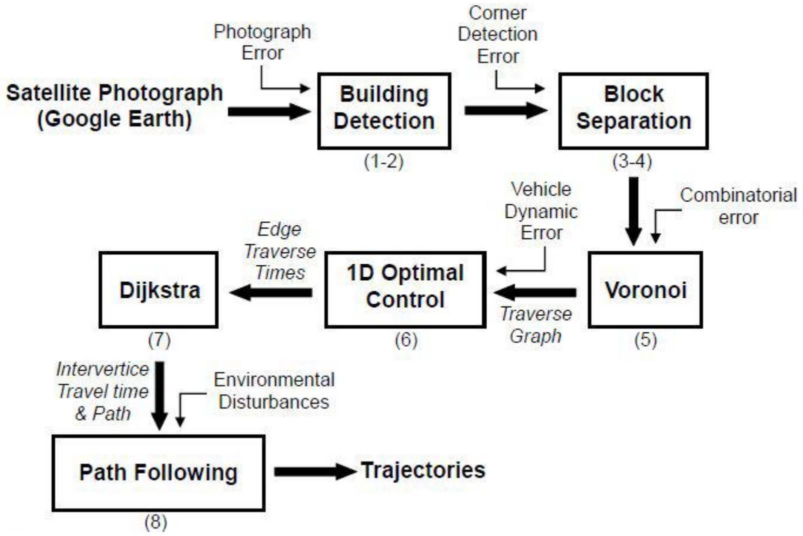


Figure 4.1: Hierarchy of mission planning.

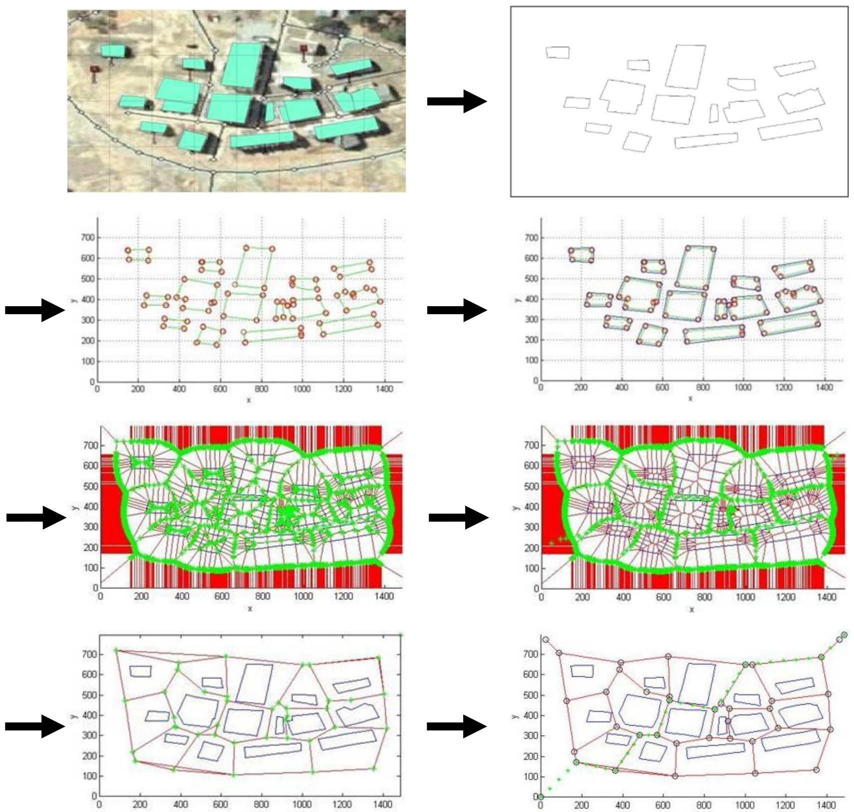


Figure 4.2: Processes to generate a trajectory of a single UAV.

Disjunction of Algorithmic Error

- Step 1: Photograph Error
  1) Google Earth has positional accuracy of $39.7m$ RMSE ($0.4m < \text{error} < 171.7m$).
  2) Some research works propose a method to use georeferencing to increase the data point accuracy to a positional accuracy of $5-6m$.
  3) I will be able to enhance the accuracy using same method up to $1.5m$ by using a GPS device with higher accuracy (ex. Sokkia GSR2700 ISX).
  4) I can set a constant buffer size, c, as $1.5m$ in the mission hierarchy in Fig 5.1 to avoid possible vehicle crashes.
  5) Since there are still 50% of chance to have GPS data collected outside of a circle with $1.5m$ radius, so I set the buffer zone size as, $c = 1.5N_{ph}$, where N_{ph} is a safety factor for the photograph error.

- Step 2: Building Detection Error
  1) The latest automated building extraction algorithm using IKONOS images has 83.2% building detection rates. By running n times on the same surveillance region, I can get

$$(1-0.832)^n = 0.168^n.$$

Disjunction of Algorithmic Error

- Step 3: Corner Detection Error
  1) The corner detection algorithm works greatly to detect buildings except the one with curvatures and it results about 96.16% correctness to wrap buildings (Fig. 5.3).

$$\text{Corner detection rate} = \frac{\text{Area of the correctly detected buildings}}{\text{Area of the total buildings}}.$$

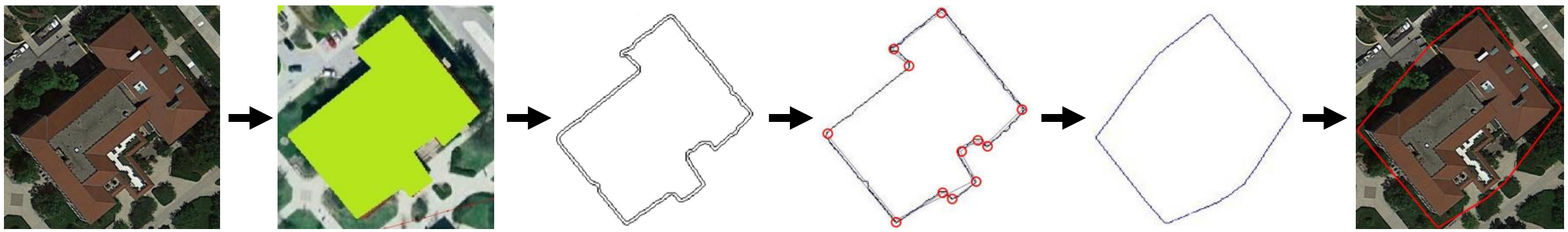


Figure 4.3: Processes to detect buildings with 2m buffer zone size.

Figure 4.4: Wrapping the detected buildings with 2m buffer zone size.

Disjunction of Algorithmic Error

•Step 5: Combinatorial Error

1)There are three sources of the combinatorial error in the Voronoi diagram algorithm; distance error (due to incorrect depth comparison of a pixel); resolution error (due to coarse discrete sampling); Z-buffer precision error (due to precision limitations of bits in graphic systems).

2)With an assumption that there is no Z-buffer precision error, the error bound can be expressed as,

$$dist(P,A) \le dist(P,B) + 2\varepsilon, \quad (4.3)$$

where, $dist(P,A)$ = distance from the center of pixel P to the site A,

ε = maximum distance error.

3)According to the paper [21],

$$\varepsilon = R\left(1-\cos\left(\frac{\alpha}{2}\right)\right) = R\left(6.83\cdot 10^{-4}\right), \qquad (4.4)$$

where, α = acute angle of the isosceles triangle

(1024×1024 has 85 triangles),

R = radius of the cone (max distance

between site and sample point).

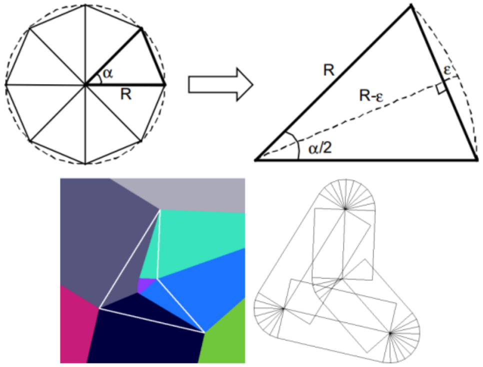


Figure 4.5: Calculation of α and R [21].

Since ε is small enough, I ignore the combinatorial error in step 5.

Disjunction of Algorithmic Error

•Step 6: Vehicle Dynamic Error

1) In vehicle UAV dynamic model,

$$x_c(k+1) = x_c(k) + Tv_c(k),$$

$$v_c(k+1) = -\frac{T}{\tau_x \tau_v} x_c(k) + \left(1 - \frac{T}{\tau_v}\right) v_c(k) + \frac{T}{\tau_x \tau_v} x_c^{ref}(k),$$

where, x_c = position vector of the UAV ($\in R^3$),

v_c = *velocity* vector of the UAV ($\in R^3$),

T = sampling time,

x_c^{ref} = tracking reference points ($\in R^3$),

τ_x = position tracking time constant,

τ_v = velocity tracking time constant.

we can introduce dynamic errors in position and velocity as,

$$x_c(k+1) = x_c(k) + Tv_c(k) + \left[\delta x_c(k) + T\delta v_c(k)\right],$$

$$v_c(k+1) = -\frac{T}{\tau_x \tau_v} x_c(k) + \left(1 - \frac{T}{\tau_v}\right) v_c(k) + \frac{T}{\tau_x \tau_v} x_c^{ref}(k) + \left[-\frac{T}{\tau_x \tau_v} \delta x_c(k) + \left(1 - \frac{T}{\tau_v}\right) \delta v_c(k)\right].$$

With $\tau_x = 0.25s$, $\tau_v = 0.5s$, and $T = 0.01s$, the amount of errors in position becomes,

$$\begin{aligned} e_p &= \delta x_c(k) + 0.01\delta v_c(k), \\ &= 3N_{po} + 0.01 \cdot 0.015, \\ &\approx 3N_{po}. \end{aligned}$$

Disjunction of Algorithmic Error

- Step 8: Environmental Disturbances (wind effect)

$$x_c(k+1) = x_c(k) + Tv_c(k) + T^2 \frac{1}{2} a_w(k),$$

where, a_w = amount of UAV acceleration caused by the wind ($0.1m/s^2$).

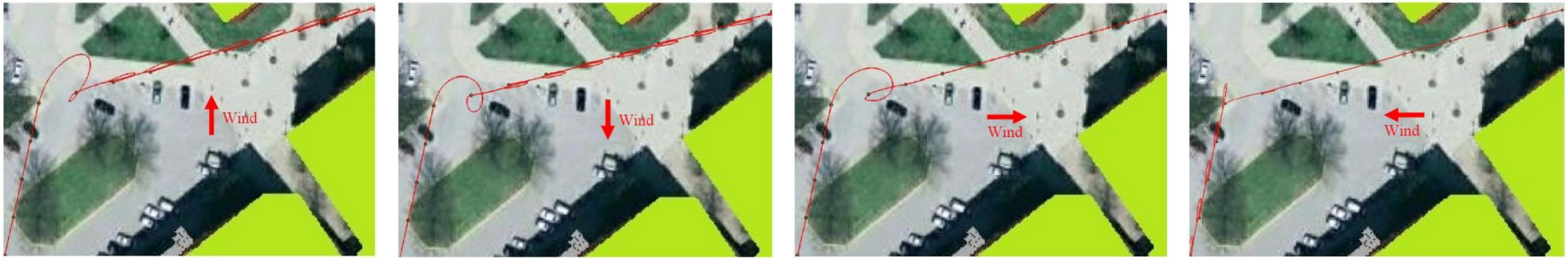


Figure 4.6: UAV trajectories when wind causes the UAV to accelerate with magnitude of $0.1m/s^2$.

- Overall Algorithmic Error
  - Step 1 (photograph error): $1.5N_{ph}$ RMSE photograph position error (unit: m),
  - Step 2 (building detection error): $100 \cdot 0.168^n$ % building detection error,
  - Step 3 (corner detection error): 3.84 % corner detection error,
  - Step 5 (combinatorial error): Negligible,
  - Step6 (vehicle dynamic error): $3N_{po}$ position error (unit: m),
  - Step 8 (environmental disturbances): $3.3 \pm 0.23m$.

Disjunction of Algorithmic Error

•With $N_{ph} = 3$ and $N_{po} = 2$, total error can be calculated as, $e_{t1} = 1.5 \cdot 3 + 3 \cdot 2 + 3.3 = 13.8m.$

•However, not only the overall algorithmic error, but I also need to incorporate UAV system constraints such as $v_{\min}$, $v_{\max}$, $a_{\max}$, and $r_{\min}$ to achieve much safer operation.

$$e_{t2} = b_1 + b_2, \quad (4.9)$$

where, $b_1 = v_{\max} t_d - \frac{1}{2} a_{\max} t_d^2 = \frac{v_{\max}^2 - v_{\min}^2}{2a_{\max}}$,

$$b_2 = r_{\min} \tan\left(\frac{\pi}{2} - \frac{\alpha}{2}\right) = \frac{v_{\min}^2}{a_{\max}} \tan\left(\frac{\pi}{2} - \frac{\alpha}{2}\right).$$

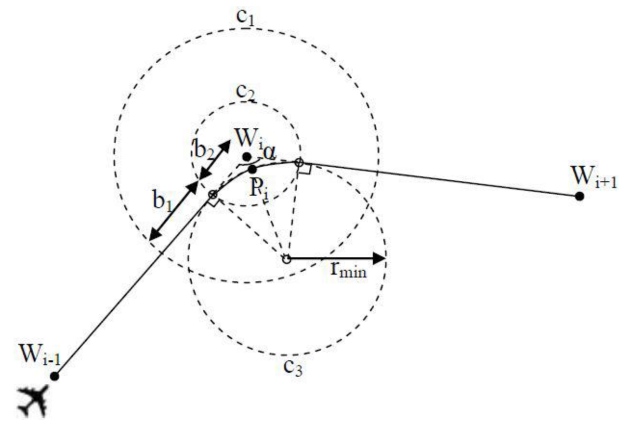


Figure 4.7: Minimum radius $r_{\min}$ which allows a fixed-wing UAV to change its direction without any collision.

For the fixed-wing UAVsystem properties of $v_{\min} = 0.5m/s$, $v_{\max} = 2m/s$, $a_{\max} = 0.5m/s$,

$$e_{t2} = b_1 + b_2 = 3.75 + 0.29 = 4.04m.$$

Therefore, the final buffer size will be,

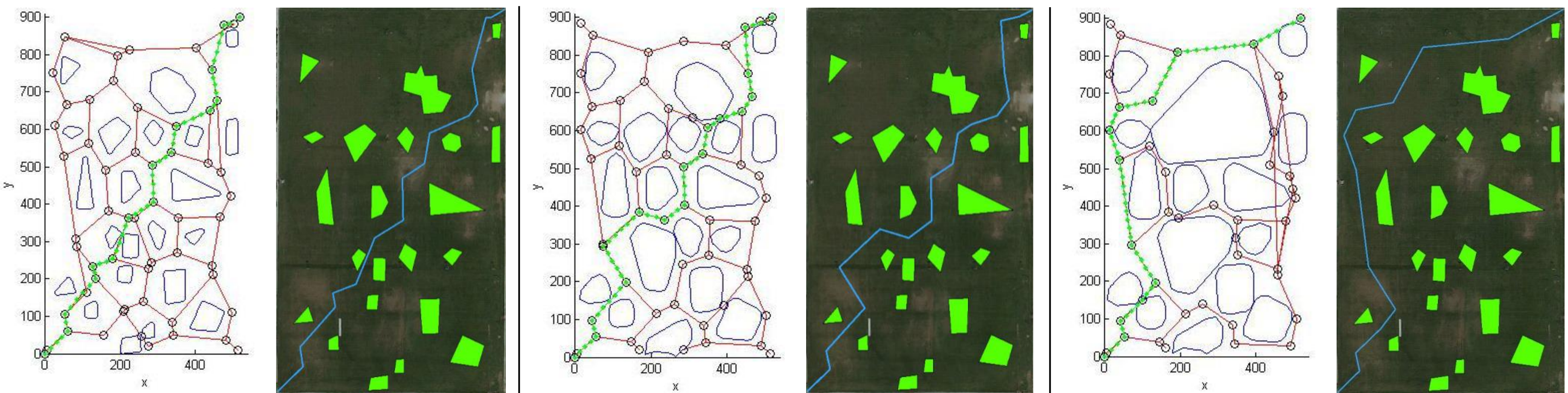
$$e_t = e_{t1} + e_{t2} = 13.8 + 4.04 = 17.84m.$$

Algorithmic Robustness Analysis

•All simulations are done with a fixed-wing UAV flying from the starting location, [0,0], to the goal location, [472,808].

•The UAV has system properties of;

$$v_{\min} = 0.5m/s,\ v_{\max} = 2m/s,\ v_{initial} = [0,0,0]m/s,\ a_{\max} = 0.5m/s,$$
$$Altitude_{\min} = 3m,\ Altitude_{\max} = 50m,\ Altitude_{normal} = 30m,$$
$$T = 0.01s,\ \tau_x = 0.25s,\ \tau_v = 0.5s.$$



(a) UAV trajectories with a buffer size of e_{t2} (4.04m).

(b) UAV trajectories with a buffer size of e_{t1} (13.8m).

(b) UAV trajectories with a buffer size of e_{t1} and e_{t2} (17.84m).

Figure 4.8: Robustness verification by changing the buffer size of buildings (Squirrel park at Purdue University (lat: 40.422108, lon: -86.932187)).

Gaming—Strategy

Collaborators:

Rajdeep Singh and Michael Hulton
(Lockheed Martin)

Securing Physical Facilities

- Get security that can be quantified as in cryptography

- Approximate the security produced by the ‘marketplace’

- Incorporate some descriptions of both the security system and the attackers
  - Number of defense layers
  - Number of attackers
  - Practically instant communications within teams

- Eliminate dependence on personnel reliability

Problem Formulation

- M-attacker team (A) vs N-layered defense (D)
- The state of each layer X_b is a binomial r.v in [0,1]—controlled by player or system 'D' w/ q_b
- Trial or set of attempts, one for each layer with estimates for each layer Z_b by attackers. (sequential or parallel)
- For experiment k at layer b, define

$$I_{X_b^k, Z_b^k} = \begin{cases} 1 \text{ if } X_b^k = Z_b^k \\ 0 \text{ otherwise} \end{cases}, \ \mathrm{C}\left(Z_b^k\right) = \begin{cases} 1 \text{ a detection at layer b for D} \\ 0 \text{ a breach at layer b for A} \end{cases}$$

Detection Rates and False Positives

- Each layer has its false positives and missed detection rates, defined as follows:

$$\alpha_b \doteq p[C(Z_b^K) = 1 | \mathcal{I}(X_b^K, Z_b^K) = 1]$$

$$\beta_b \doteq p[C(Z_b^K) = 0 | \mathcal{I}(X_b^k, Z_b^K) = 0]$$

$$\bar{\alpha}_b \doteq 1 - \alpha_b = p[C(Z_b^K) = 0 | \mathcal{I}(X_b^K, Z_b^K) = 1]$$

$$\bar{\beta}_b \doteq 1 - \beta_b = p[C(Z_b^K) = 1 | \mathcal{I}(X_b^K, Z_b^K) = 0]$$

- Control options for player D $[q_b, \alpha_b, \beta_b]$
- The goal of D (A) is to maximize (minimize) the frequency of detection and minimize (maximize) the breach occurrence given the detector's false alarm rates.

Assumptions

- Instantaneous information sharing
- False positive detection for an attacker is counted as true detection/false positive detection for a good guy is counted as not counted as true detection.
- Independence of global estimates for each layer
- Ignore running or terminal costs for A and simplistic cost function for D except for the feedback part

Basic Calculations

The likelihood for different states of $\mathcal{I}(X_b^K, Z_b^K)$, for any trial K at some layer b, are given by

$$P(\mathcal{I}_{X,Z} = 1) \doteq \sum_{i \in \mathcal{X}} \{P(X = i)P(Z = i) \doteq \tilde{q}^T q$$

$$P(\mathcal{I}_{X,Z} = 0) \doteq \sum_{i \in \mathcal{X}} \{P(X = i)P(Z = k) \doteq \tilde{q}^T q^c$$

where $i \neq k$ and $q^c \sim (n, 1 - \mu_q)$, i.e, $q_0^c = q_1$ and $q_1^c = q_0$.

Probability of detection and breach:

$$p_d = \alpha \tilde{q}^T q + \bar{\beta} \tilde{q}^T q^c \qquad p_b = \bar{\alpha} \tilde{q}^T q + \beta \tilde{q}^T q^c$$

Theorem

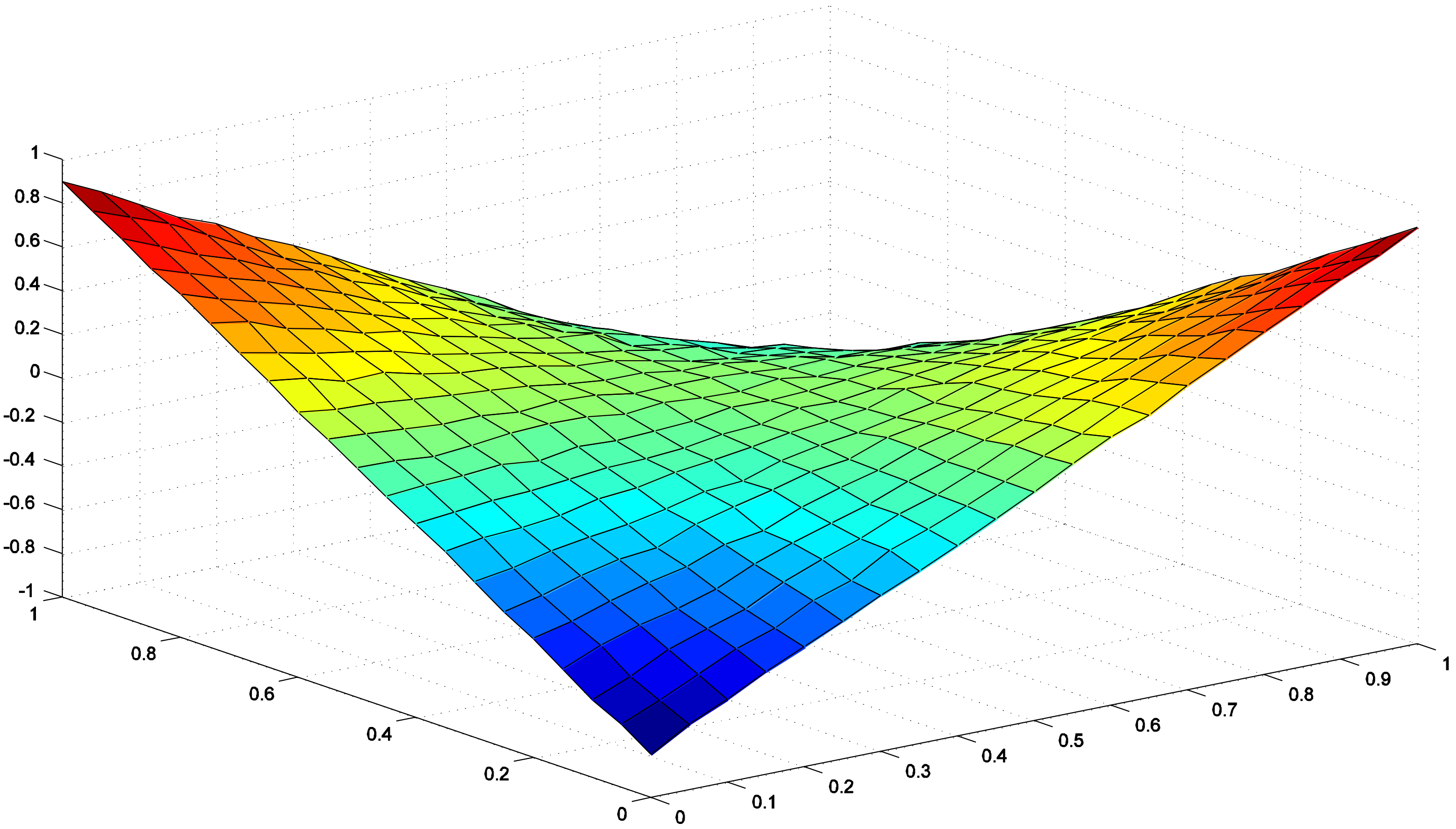
$$
\begin{aligned}
O(\tilde{q}, q) &\doteq p_d(\tilde{q}, q) - p_b(\tilde{q}, q) \\
&= [\alpha \tilde{q}^T q + \bar{\beta} \tilde{q}^T q^c] - [\bar{\alpha} \tilde{q}^T q + \beta \tilde{q}^T q^c] \\
&\doteq \tilde{q}^T [\alpha q + \bar{\beta} q^c - \bar{\alpha} q - \beta q^c]. \\
&= \tilde{q}^T [(2\alpha - 1) q + (1 - 2\beta) q^c] \\
&= \tilde{q}^T [\alpha^* q_0 + \beta^* q_1, \alpha^* q_1 + \beta^* q_0],
\end{aligned}
$$

where $\alpha^* = (2\alpha - 1)$ and $\beta^* = (1 - 2\beta)$.

Theorem 3.1: Given α and β, an optimal mixed control strategy for either player is

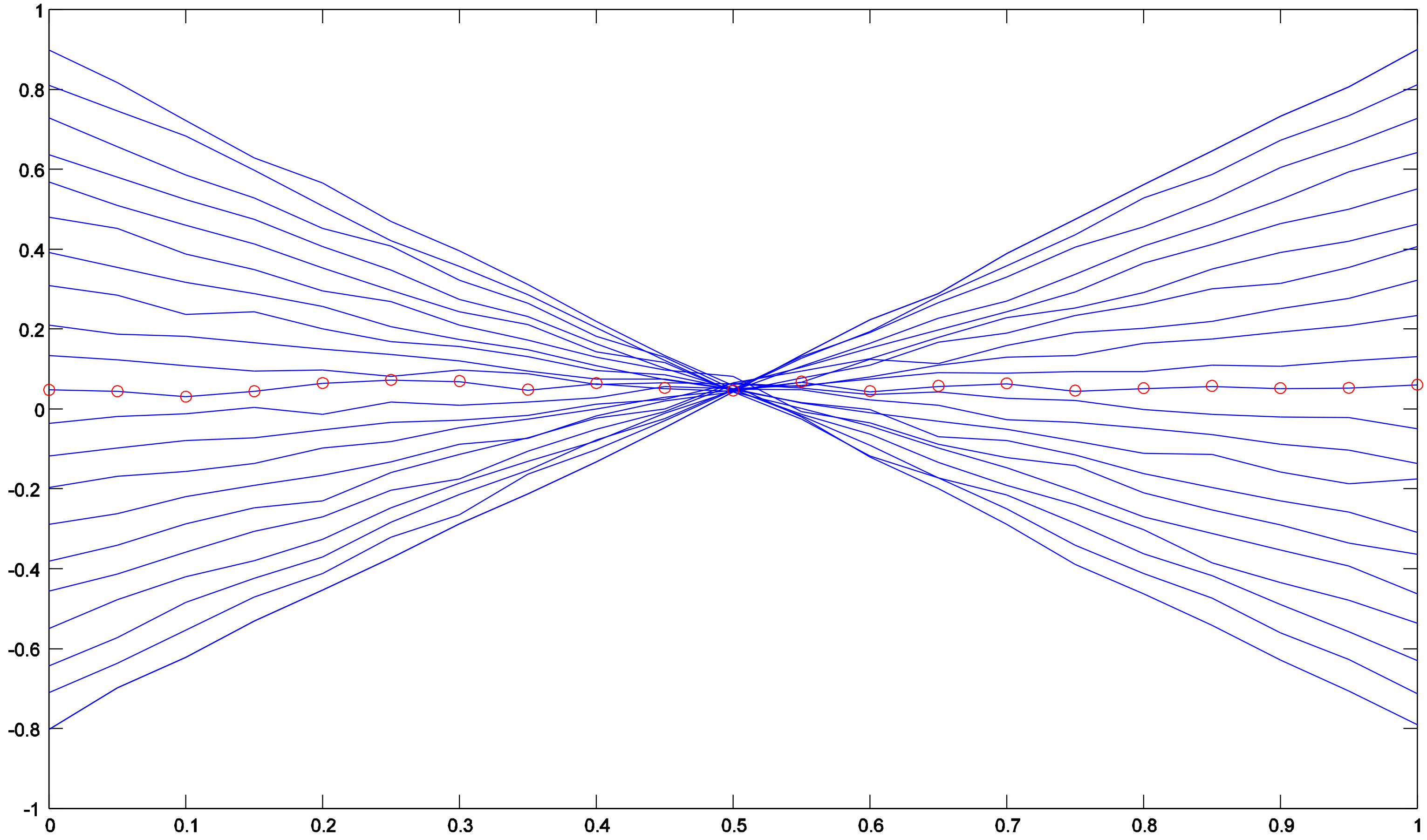
$$q_{OL} = [0.5, 0.5].$$

In fact the equilibrium strategy for 'A' is any binomial distribution (or no control). A saddle point exists for this game with value $O(\tilde{q}_{OL}, q_{OL}) = \alpha - \beta$.



Saddle point exists for one layer guess game (0.1,0.5)

Open Loop Results



Equilibrium strategy is an unbiased distribution, yielding the maximum rate of 0.5

Open Loop summary

Num layers	2	4	6	8
Average num. swipes	0.94	0.89	0.86	0.85
Avg. Num. trials. to breach	4.32	21.08	108.80	421.53
2 x Avg. num. swipes	1.87	1.76	1.70	1.68
Avg. Num. trials to breach	8.20	72.21	735.81	10043

Summary of Computational Challenges

- There is a lot of scope for fast (enough) embedded algorithms
  - This will enable high quality sensing from low quality components
- Systematically handle multiple numerical scales
  - When microscopic phenomena can affect macroscopic behavior
- Algorithms need to be analyzed for propagation of error from processes and measurement noise
  - This is different from numerical error analysis because errors are large.