

UTM Integration of Weather Information

By Daniel Mulfinger

Outline

- Introduction
- Service Architecture
- Case Study – Conformance Buffer
- Case Study – Operation Wind Constraint
- Conclusion

USS – Service Architecture

- (insert service architecture chart)

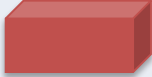
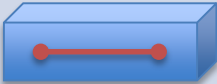
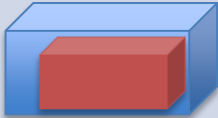
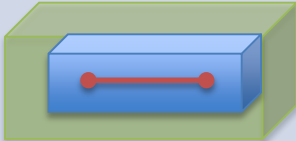
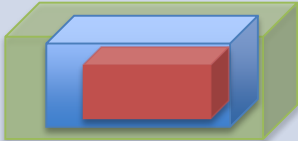
Case Study 1:

Conformance Buffer Calculation Server

Goal: Calculate conformance buffer of operations based on weather and vehicle performance.

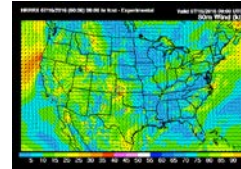
Approach: Create a server to process operation, vehicle performance, and wind data.

Airspace Buffering

UTM Creates	Geography Types		Notes
	Flight Plan	Volume	
Flight Geography	LineString() 	Polygon() 	Submitted by client
Conformance Geography			Used for determining non-conformance of position updates
Parameters	Buffer _{ConfHorizFP} Buffer _{ConfVertFP} Buffer _{ConfTimeFP}	Buffer _{ConfHorizVol} Buffer _{ConfVertVol} Buffer _{ConfTimeVol}	Calculated using wind data
Protected Geography			Used for separation when approving operations
Parameters	Buffer _{ProtHorizFP} Buffer _{ProtVertFP} Buffer _{ProtTimeFP}	Buffer _{ProtHorizVol} Buffer _{ProtVertVol} Buffer _{ProtTimeVol}	Based on GPS error

Conformance Buffer Calculation

- NOAA's HRRR Weather Model:
 - Applied for the Reno-Stead Area
 - Determined maximum wind component (u or v) at 10 meters altitude
 - Determined direction of wind from components
- SJSU Sonar Anemometer Data
 - Provides 3-dimensional wind vector data
 - Used this to determine the proportion of vertical wind to the horizontal wind:



$$n_{w,factor} = \frac{w}{\sqrt{u^2 + v^2 + w^2}} \Big|_{SJSU\ data}$$

$$\vec{V}_{wind} = u_{HRRR} \hat{i} + v_{HRRR} \hat{j} + n_{w,factor} \sqrt{\frac{u_{HRRR}^2 + v_{HRRR}^2}{1 - n_{w,factor}^2}} \hat{k}$$

How Wind Velocity was Used?

- Wind vector was additive to the vehicle velocity:

$$\vec{V} = \vec{V}_{UAV} + \vec{V}_{wind}$$

- A vertical and lateral buffer is calculated:

– Lateral:

$$d_{lateral} = 2R \arctan \left[\frac{\sqrt{\sin^2 \frac{\phi_{new} - \phi_{fp}}{2} + \cos \phi_{fp} \cos \phi_{new} \sin^2 \frac{\theta_{new} - \theta_{fp}}{2}}}{\sqrt{1 - \sin^2 \frac{\phi_{new} - \phi_{fp}}{2} + \cos \phi_{fp} \cos \phi_{new} \sin^2 \frac{\theta_{new} - \theta_{fp}}{2}}} \right]$$

where ϕ_{new} and θ_{new} are a function of $\vec{V}$ and other variables

– Vertical: $h_{vertical} = t_{control} V \sin \gamma_{wind}$

where V is the magnitude of $\vec{V}$, $t_{control}$ is an estimate of UAV recovery time, and γ_{wind} is the flight path angle cause by the wind vector

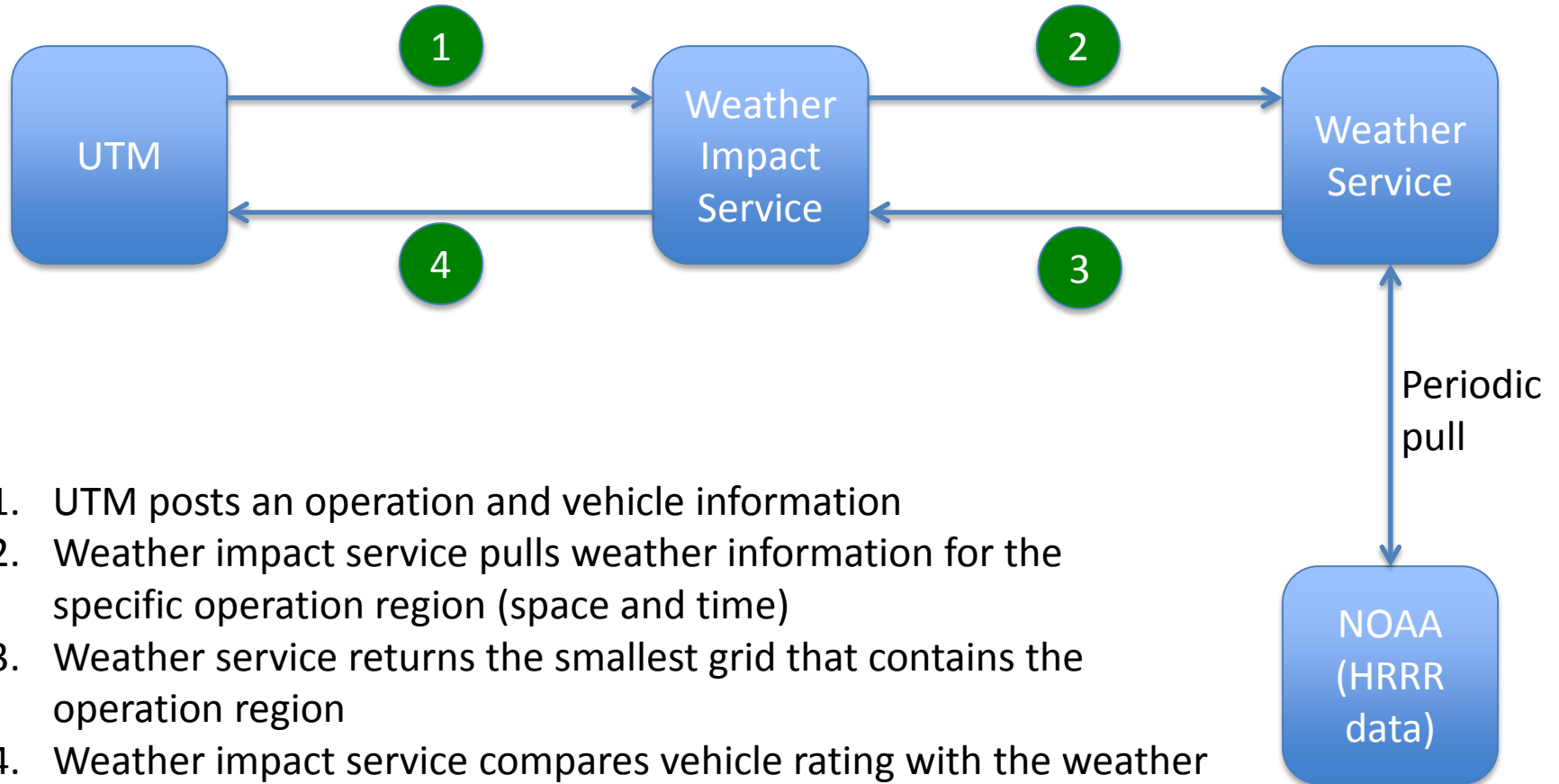
Case Study 2:

HRRR Wind Impact Server

Goal: Evaluate the impact of forecasted winds on UTM operations

Approach: Create a wind data provider and wind impact server.

The data flow



1. UTM posts an operation and vehicle information
2. Weather impact service pulls weather information for the specific operation region (space and time)
3. Weather service returns the smallest grid that contains the operation region
4. Weather impact service compares vehicle rating with the weather grid and returns any concerns to UTM

Input/Output

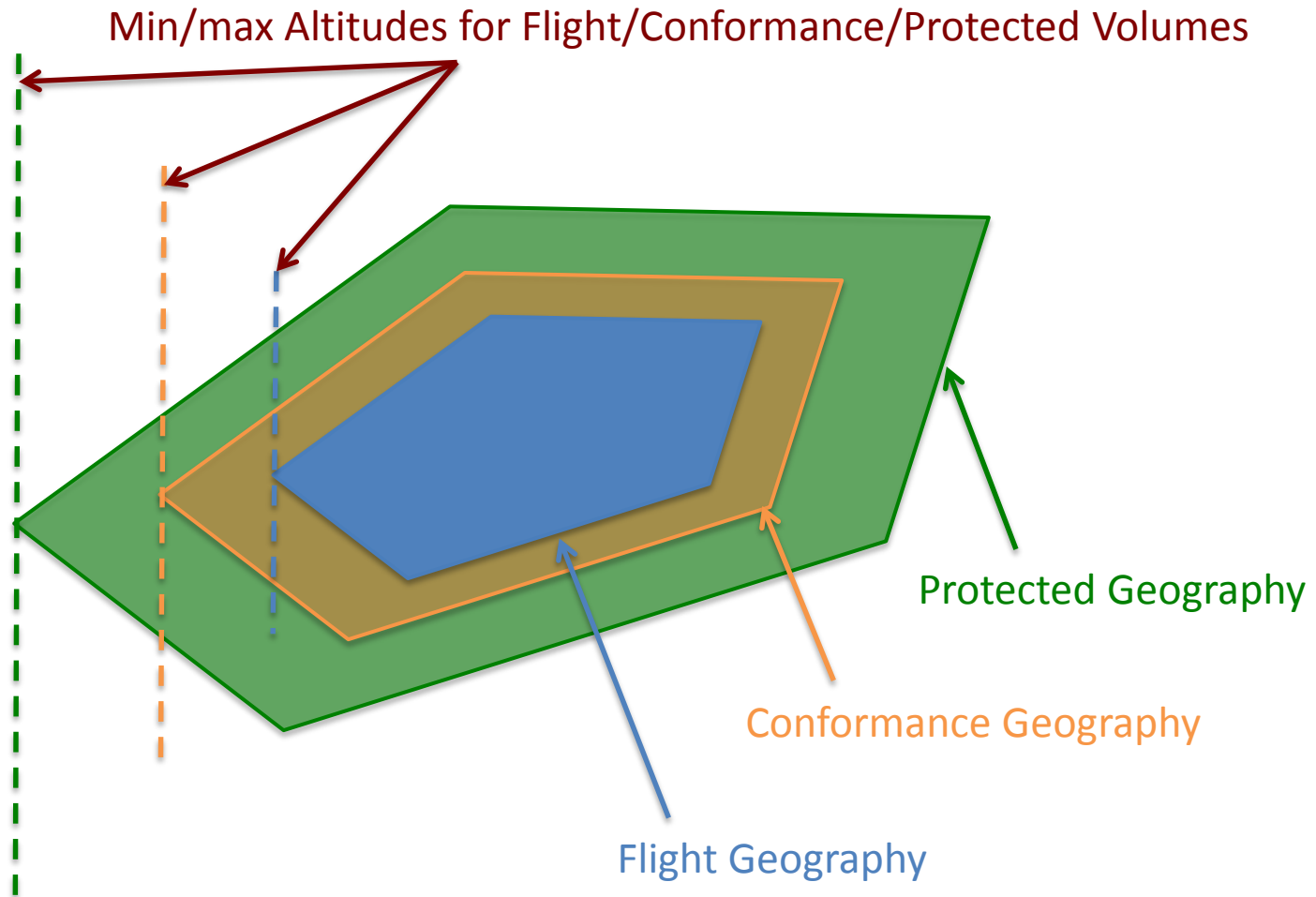
- Input:
 - Operation plan
 - Vehicle performance
 - HRRR data
- Output:
 - Weather-checking result:
 - **ACCEPT**, **WARNING**,
REJECT
 - Explanation

Operation Plan

- Multiple flights
- One flight = Multiple Segments
- One segment: Geography + min/max altitudes
 1. Flight geography: LINESTRING or POLYGON
 - NOTE: in the example operation received from the software group 100% are POLYGON
 2. Conformance geography: POLYGON
 3. Protected geography: POLYGON

This is based on the sample file provided by the software team

Operation Volume (i.e., Segment)

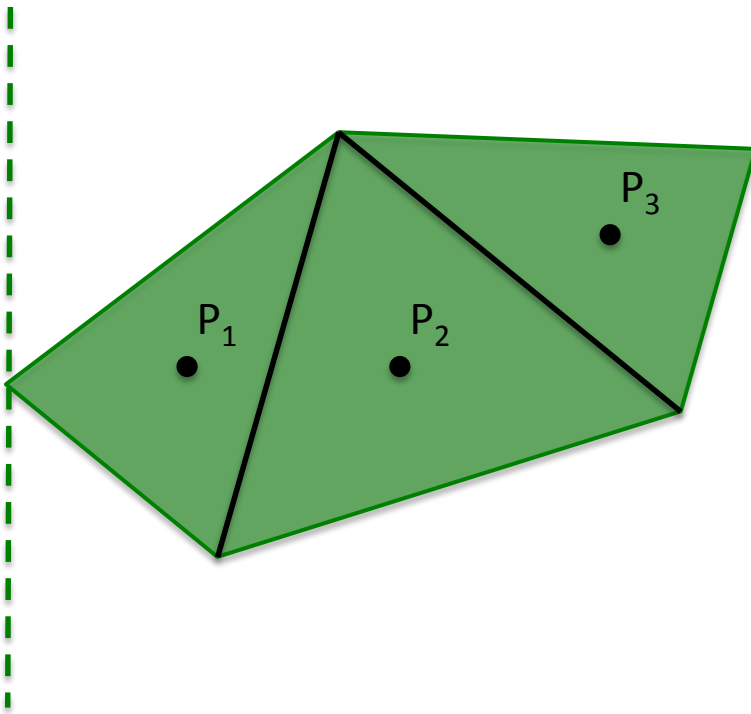


Wind Checking Algorithm

1. Sample points within the **PROTECTED VOLUME**
2. Compute wind velocity at the sample points
3. Check wind-velocity against vehicle performance data
4. Made decision for each segment & flight
 - Then make the decision for each flight

Step 1: Sample 4D Points

max altitude



min altitude

- Time: whole duration
[protected-time-begin, protected-time-end]
- Altitude:
 $(\text{min-alt} + \text{max-alt}) / 2$
- Sample Lat/Lon:
 - Triangulation.
 - Center points of all triangles (P_1, P_2, P_3)

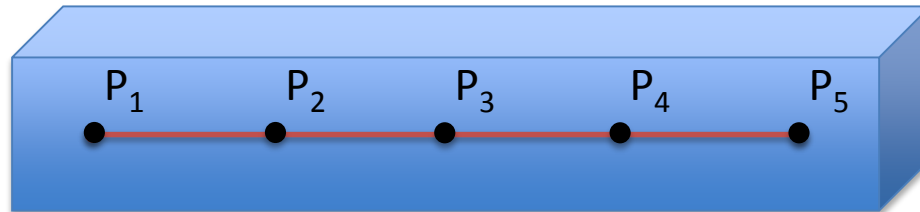
Delaunay Triangulation: https://en.wikipedia.org/wiki/Delaunay_triangulation

Java package used: <https://www.cs.bgu.ac.il/~benmoshe/DT/Delaunay%20Triangulation%20in%20Java.htm>

Step 1: Sample 4D Points (cont.)

different from what discussed last meeting

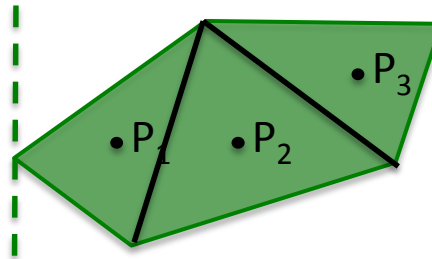
What I thought
most volume are



I believe the reality
is more like this



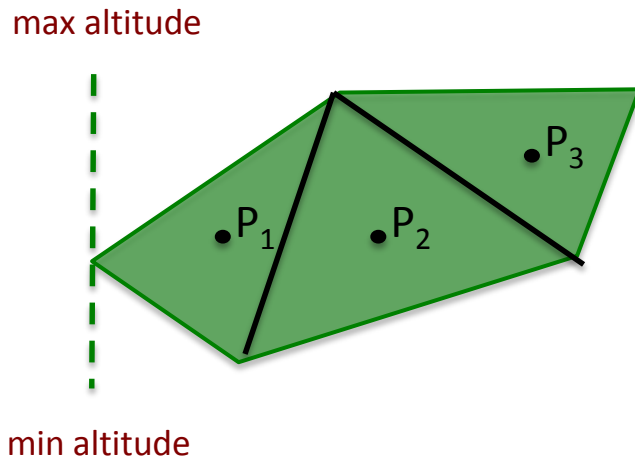
max altitude



min altitude

triangulation is more flexible & works well for any polygon
(compared to trying to divide the volume into cubes)

Step 1: Sample 4D Points -- Timing

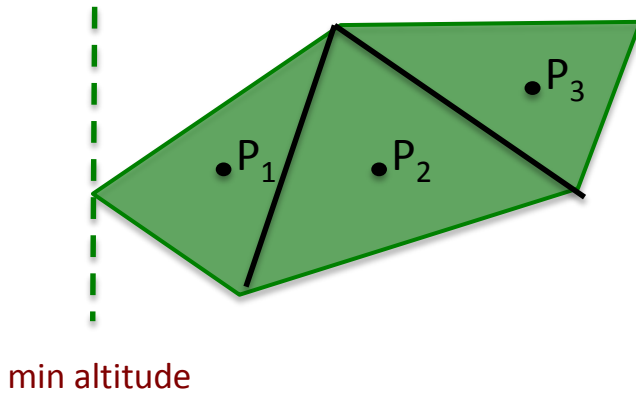


Use **all** relevant HRRR points relevant to the time spending in the protected area

Reason: don't know how UAS spends time within the area/polygon

Step 1: Sample 4D Points -- Example

max altitude



Min-altitude: 10 ft

Max-altitude: 170 ft

Protected time begin: 1:50PM

Protected time end: 2:13PM

Coordinates: $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = (x_3, y_3)$

Sample 4D points to check against HRRR data
(which available every 15') are:

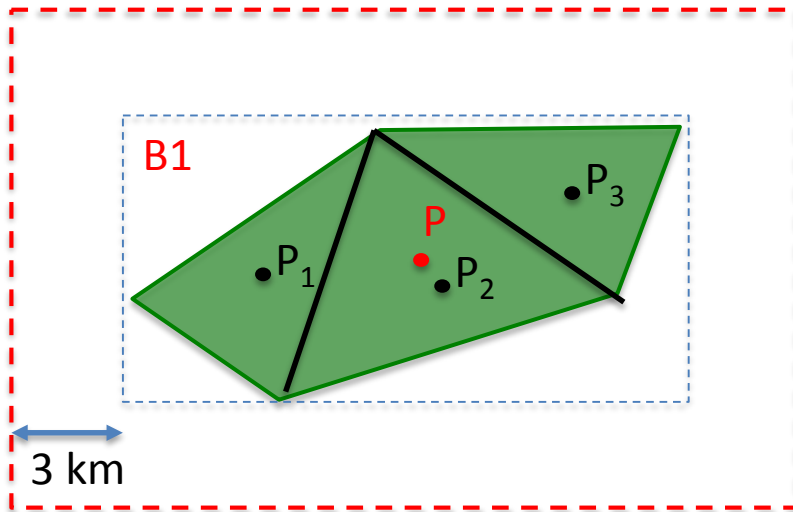
$(x_1, y_1, 90\text{ft}, 1:45\text{PM})$, $(x_1, y_1, 90\text{ft}, 2:00\text{PM})$

$(x_2, y_2, 90\text{ft}, 1:45\text{PM})$, $(x_2, y_2, 90\text{ft}, 2:00\text{PM})$

$(x_3, y_3, 90\text{ft}, 1:45\text{PM})$, $(x_3, y_3, 90\text{ft}, 2:00\text{PM})$

Step 2: Find the relevant HRRR grid data

B2

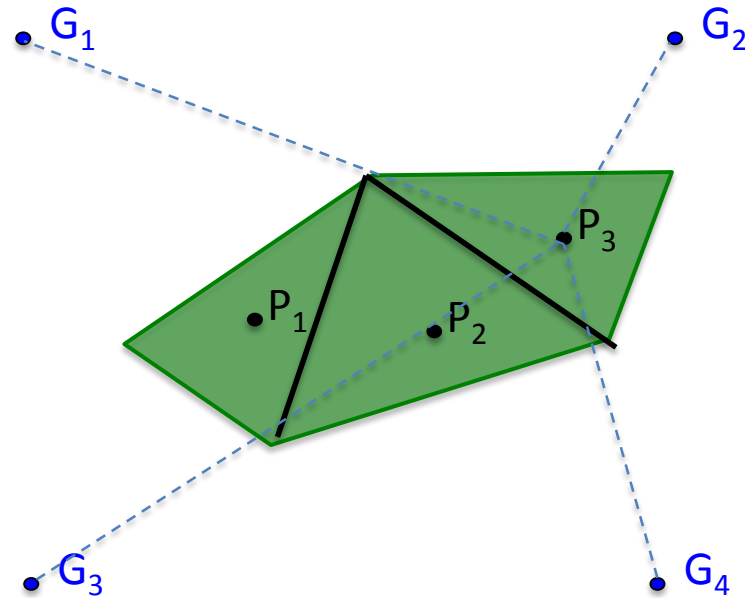


1. Find the bounding box **B1** of the protected polygon
2. Find the center **P** of the bounding box
3. Find a square **B2** with **P** as center and distance of 3km from each side of B1 (HRRR grid is of 3KM resolution)
4. Find all grid points within **B2**:
Abe database query require a box with max/min lat/lon bounds
5. Use wind value at 80 ft altitude (for now)

NOTE: without being able to connect to the HRRR database, current test data are generated with:

- Corners of **B1** act as wind grid points
- Wind-strength at each grid point is randomly generated.

Step 3: Compute wind-strength at each sample points



Wind strength at each P_i point is **interpolated** using wind-strength values at the relevant HRRR wind grid points G_i :

Actual Implementation: **Inverse Distance Interpolation**

(https://en.wikipedia.org/wiki/Inverse_distance_weighting)

Step 4: Weather Recommendation

- Decision Point: wind-strength vs max-air-speed
 - $\geq 3x$: **REJECT**
 - $1x - 3x$: **WARNING**
 - $0 - 1x$: **ACCEPT**
- Segment:
 - One point **REJECT** $\rightarrow$ whole segment **REJECT**
 - Else, one point **WARNING** $\rightarrow$ whole segment **WARNING**
 - Else, **ACCEPT** segment
- Submitted Flight:
 - One segment **REJECT** $\rightarrow$ whole flight **REJECT**
 - Else, one segment **WARNING** $\rightarrow$ whole flight **WARNING**
 - Else, **ACCEPT** flight

Questions?

daniel.g.mulfinger@nasa.gov