



ITT

# Sensor Location Optimization Tool Set (SLOTS)

Chemical Biological Information Systems (CBIS) Conference  
11 January 2007

**Michael J. Smith**, Javad Sedehi, Scott  
Mitchell,  
Mark Henning, Samuel Freund, Stuart Edick,  
Julie Tittler

*Engineered for life*

# Overview

- SLOTS Objective
- Technical Approach
- Automating Rules-based Placement
- Genetic Algorithm Optimization
- Test Case Runs
- Summary

# Sensor Location Optimization Tools Set (SLOTS) Program Objective

***Automate*** the rules-based process and ***optimize*** the location of sensors to detect, identify, and quantify the CB hazard in support of the commander's intent.



# Sensor Location and Optimization Tool Set (SLOTS)

## Technical Approach

### Approach:

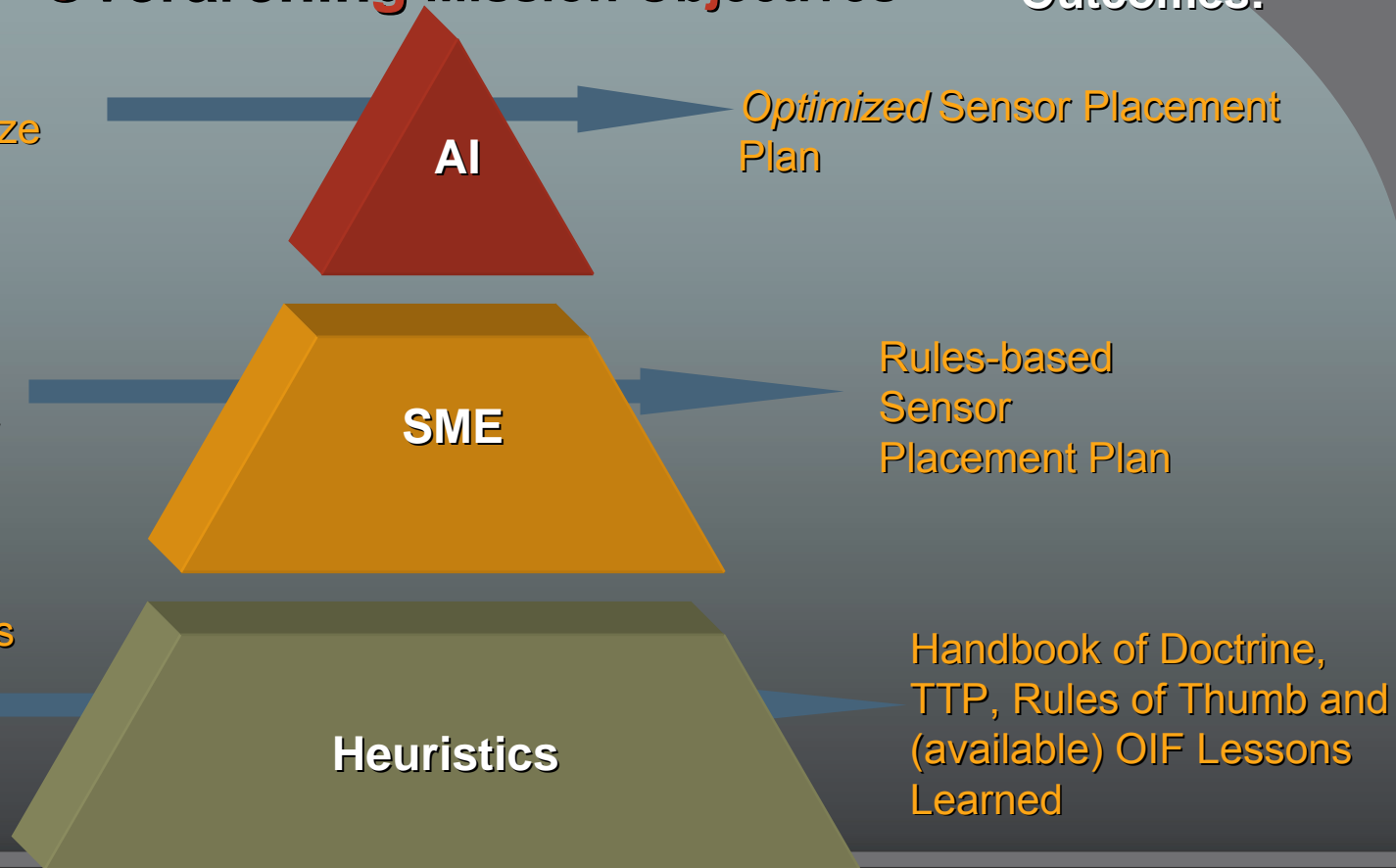
### Overarching Mission Objectives

### Outcomes:

3. Leverage artificial intelligence to optimize the ultimate sensor configuration.

2. Utilize information technologies to automate the sensor placement decision process.

1. Establish a set of rules governing the emplacement of sensors.



# The SLOTS Architecture

## Automated Rules-based Placement

- Provide a graphical representation to visualize the operational implications of sensor placement heuristics. Provide a benchmark placement solution against which the Genetic Algorithm solution can be compared.

## Simulation Cache

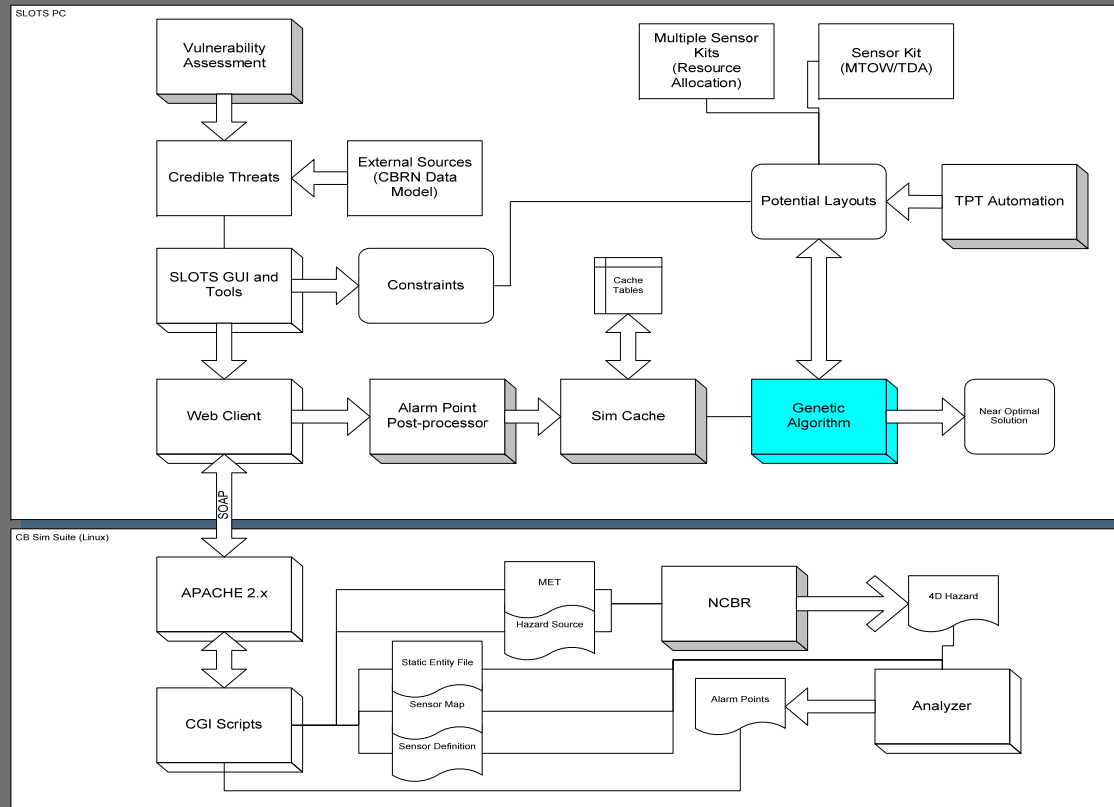
- Provide a means to employ physics based modeling and simulation to generate sensor placement environment and mitigate impact to operational timelines

## Web Services Interface

- Provide an interchangeable interface to modeling and simulation tools, allowing user selectable hazard modeling applications (e.g. NCBR, JEM, etc.)

## Genetic Algorithm

- Provide a global optimization solution for sensor placement.



# Sensor Location Optimization Tools Set (SLOTS) Handbook



Placement Task: Supervise Positioning of the Chemical-Agent Alarm  
Number: 031-503-1020

## 1. Plan the positioning of the chemical-agent alarm.

### Consider the following:

- a. Determine the wind direction and speed from a current CDM or by referring to GTA 3-2-2 for a field expedient method.
- b. Determine how far upwind to place the detectors. This distance is based on the following:
  - (1) The wind speed. The faster the wind speed, the farther upwind the detector should be placed, not to exceed 150 meters.
  - (2) The weather. Rain or snow tend to wash the agent out of the air. Hot, sunny weather will tend to create higher vapor concentrations.
  - (3) The terrain. The more broken the terrain and the more obstacles (trees and buildings), the closer the detectors should be placed.
  - (4) The threat situation.
- c. Determine the detector array to be used by considering all of the concerns above, the number of detectors available, and the position of the unit (Figure 185).

## 2. Brief the emplacement teams on the exact location of the alarm.

## 3. Supervise the positioning of the chemical agent alarm. Check the following:

- a. Ensure that the detectors are emplaced the maximum possible distance from the unit not to exceed 150 meters.
- b. Ensure that the detectors are spaced no more than 300 meters apart.
- c. Ensure that the detectors are not placed where obstructions could alter wind currents.
- d. Ensure that the detectors are connected to alarms with telephone cable (WD/TT). Maximum wiring distance should not exceed 400 meters.

*NOTE: Up to five alarms may be connected to one detector.*

- e. Ensure that the alarms are placed near monitoring positions.

## 4. Determine the warning time. Warning times for different distances and wind speed can be

determined using the following formula:  $\text{Warning Time (min)} = (\text{Distance (m)} \times 60 \text{ (min/hr)}) / (\text{Wind Speed (kmph)} \times 1,000)$   
 $\text{Warning time (sec)} = (\text{Distance (m)} \times 36 \text{ (sec/hr)}) / (\text{Wind Speed (kmph)} \times 10)$

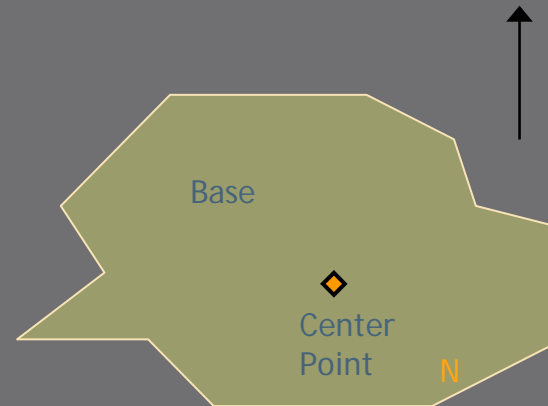
*NOTE: The automatic chemical-agent alarm system can be used only to warn against chemical agents drifting into the unit location. It provides no warning against on-target attacks.*

## 5. Ensure that the detectors are repositioned when the wind direction changes.

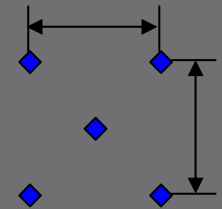
# SLOTS Automated Rules-based Placement (ARP)

## Dice Five Algorithm

- For each base,
  - The user must pick a center,  $UAcenter$ , or it is calculated as the centroid of the user-defined polygon representing the base outline.
  - The coordinates of each base are shifted such that  $UAcenter$  is at the origin, and rotated so that wind direction is parallel to the Y axis.



*SensorSpacing*



- Sensor positions of the Dice 5 layout are then defined by the functions:

$$CenterPoint = (i * SensorDistance, j * SensorDistance)$$

$$UpperLeftSquarePoint = \left( \frac{2i-1}{2} * SensorDistance, \frac{2j-1}{2} * SensorDistance \right)$$

- where  $i$  and  $j$  are integers constrained by the size of the Base. The bounds of  $i$  and  $j$  are calculated by

$$\text{floor}\left(\frac{MaxW}{SensorDistance} - \frac{1}{2}\right) \leq i \leq \text{ceiling}\left(\frac{MaxE}{SensorDistance} + \frac{1}{2}\right)$$

- and

$$\text{floor}\left(\frac{MaxS}{SensorDistance} - \frac{1}{2}\right) \leq j \leq \text{ceiling}\left(\frac{MaxN}{SensorDistance} + \frac{1}{2}\right)$$

- where  $Max^*$  are the maximum extents of the base (relative to  $UAcenter$ ) in each cardinal direction.

- The set of points must be checked against the actual bounds of the base. Sensor Positions outside this area must be discarded.

- If not enough sensors are available for a complete Dice 5 pattern, sensors are placed around the perimeter first, sorted secondarily by how far they are upwind (furthest toward  $MaxN$ , after rotation).

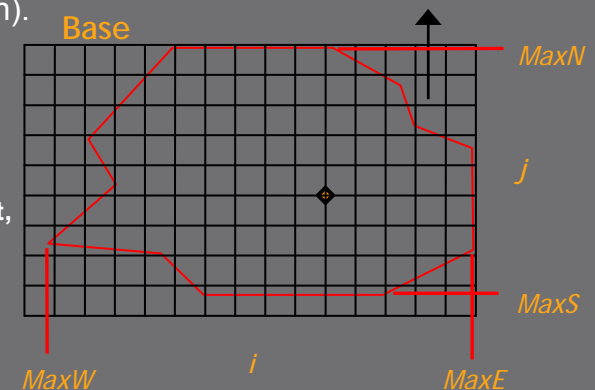


Figure 3 – Centerpoint, rotated grid

# SLOTS Automated Rules-based Placement

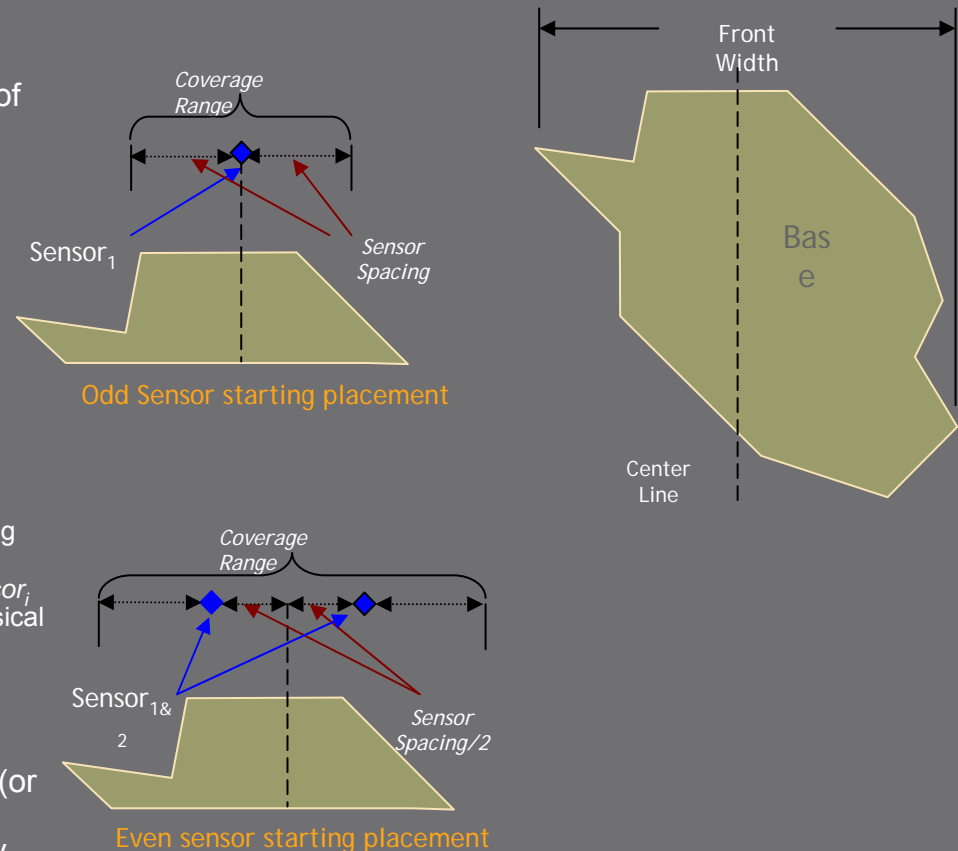
## Picket Fence Placement Algorithm

### Find Central Front-most manned position

1. select  $\overline{M}_i$  with the minimum distance from intersection of ideal x & max( $\overline{y}_i$ )

### Place Sensor

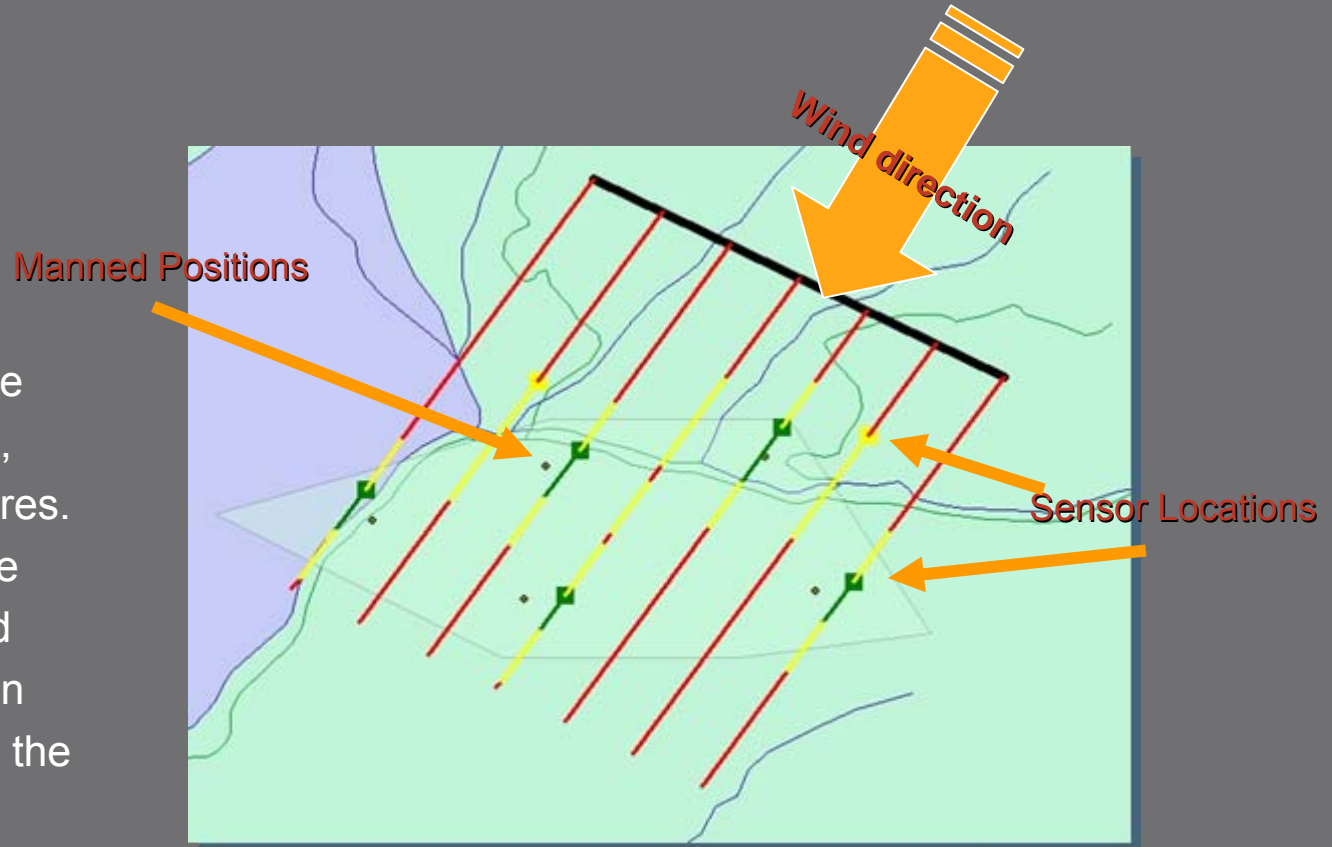
1.  $Sensor_i$  is the intersection of
  - a. Circle  $(x - \overline{M}_{ix})^2 + (y - \overline{M}_{iy})^2 = SensorDistance^2$
  - b. The  $SensorLine \dots \approx Centerline \pm n * SensorDistance$
  - c.  $y = \overline{M}_{iy} \pm \sqrt{SensorDistance^2 - (SensorLine - \overline{M}_{ix})^2}$ 
    - This can result in 0, 1, or 2 points.
    - For 0 (i.e.  $y$  is imaginary), recalculate a, b, and c using the sensor's maximum physical distance in place of  $SensorDistance$ . If there are still 0 points, place  $Sensor_i$  at the maximum plus the sensor's maximum  $\overline{M}_{iy}$  physical distance at  $SensorLine$ .
    - For 1, choose that point.
    - For 2 pick the upwind point.
- e. Repeat a through d for each  $\overline{M}_i$  within  $SensorDistance$  (or physical distance) and pick the point furthest upwind, letting  $SensorDistance$  placements always have priority over physical distance placements.





# SLOTS Automated Rules-based Placement (ARP)

The ARP provides visualization of sensor positions and compliance with doctrine and tactics, techniques and procedures. Also suggests alternative positions and associated risk. It provides decision maker with quick look at the “goodness” of given a sensor placement scheme.



# Picket Fence

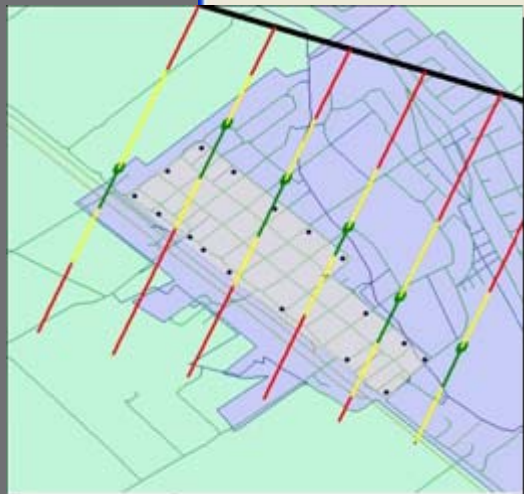
- Sensors evenly spaced along border
- Upwind of base to yield longer warning times
- Automated Implementation
- Rule Violations –Green, Yellow, & Red
- Two Implementations - Sensors placed from:
  - Fixed locations
  - Movable locations within base

The screenshot shows the 'Picket Fence' configuration window with the following settings:

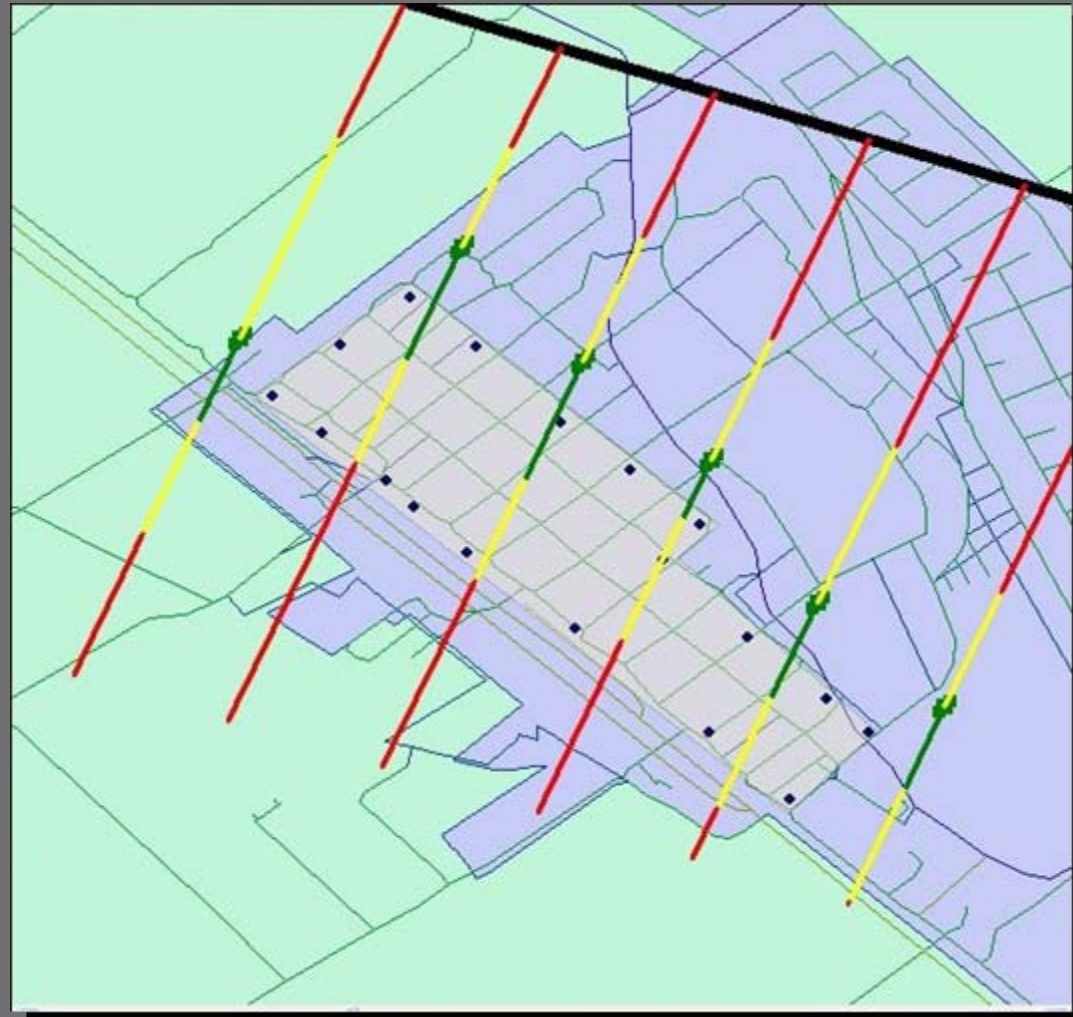
Sensor Attributes		Wind Attributes		Critical Timeline Attributes	
Sensor Spacing	300	Wind Azimuth	0	Event 1	9
Sensor Distance	150	Windspeed (kph)	10	Event 2	30
Sensors Available	500			Event 3	90
Time to detect (s)	30			Event 4	180
<input type="checkbox"/> Sensor Placement Lines				Event 5	240
<input type="button" value="Apply"/>	<input type="button" value="Edit Sensors"/>			Event 6	480
<input type="button" value="OK"/>	<input type="button" value="Cancel"/>				

Timeline Attributes:

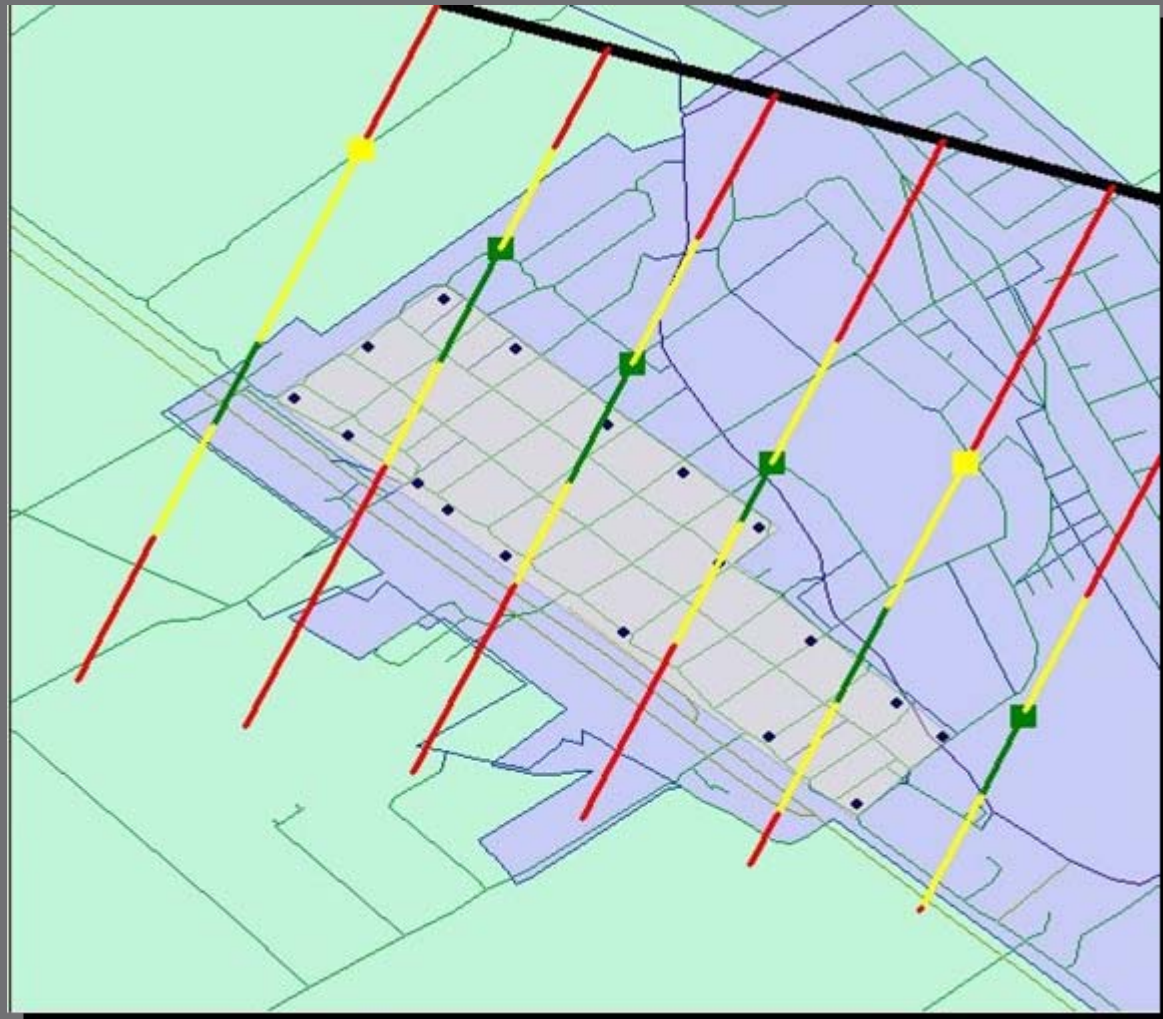
Timeline Attributes	
Start Time	0
End Time	300
Timestep	60
<input type="checkbox"/> Time Lines	
<input type="button" value="Get time on map"/>	



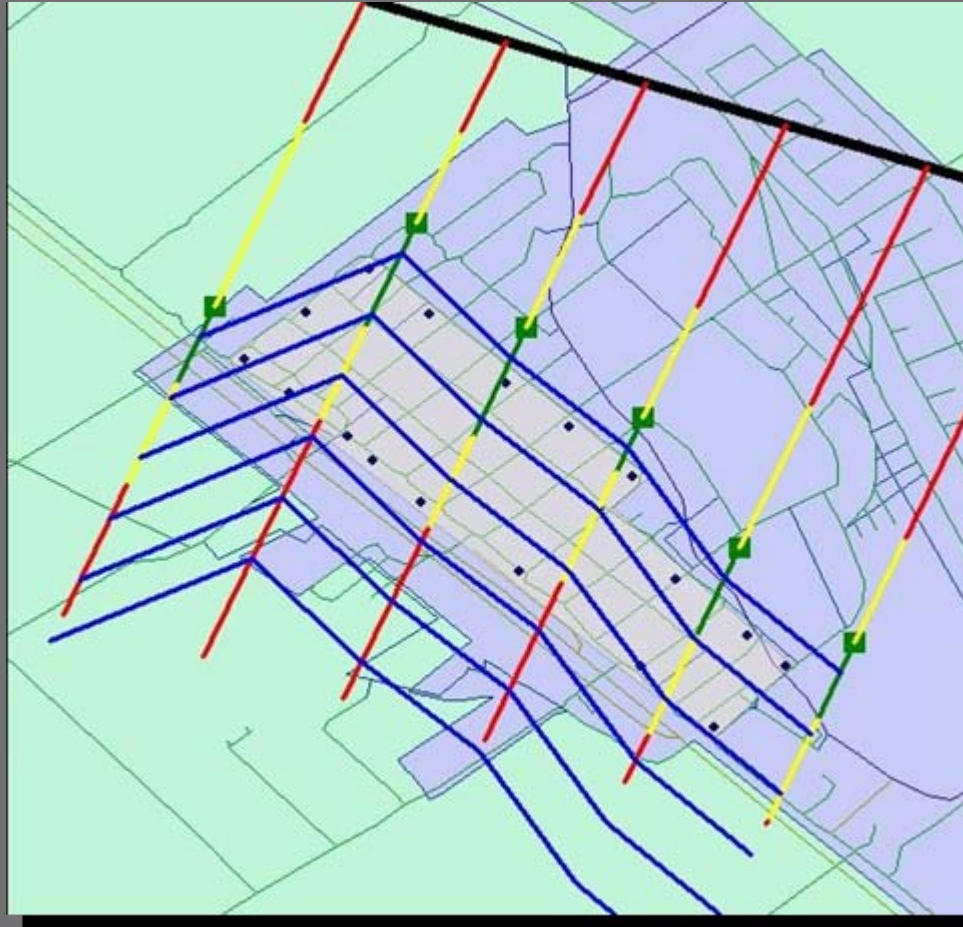
# ARP Module



# ARP Module



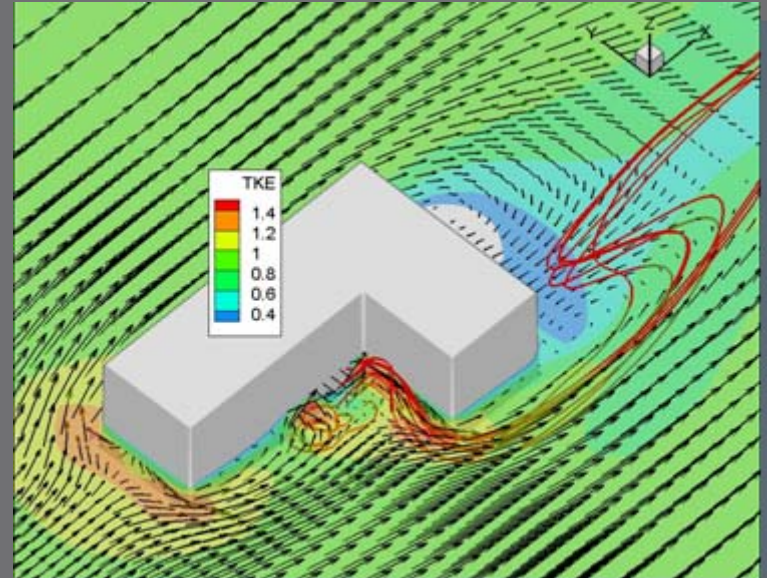
# ARP Module



# SLOTS Automated Rules-based Placement

## Shortcomings of rules-based placement

- Inability to adequately address battlespace environment
- Inability to adequately account for multiple constraints
- Inability to account for a mixed sensor kit

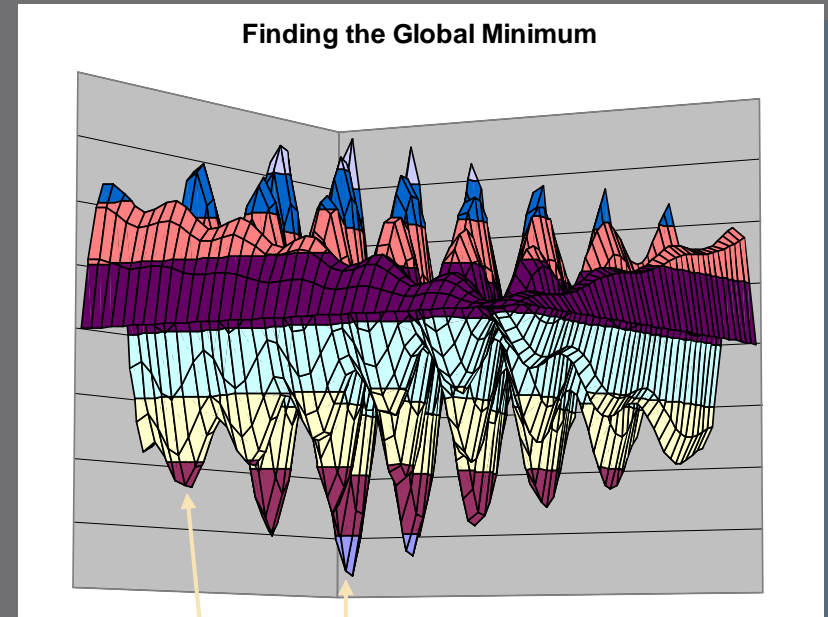
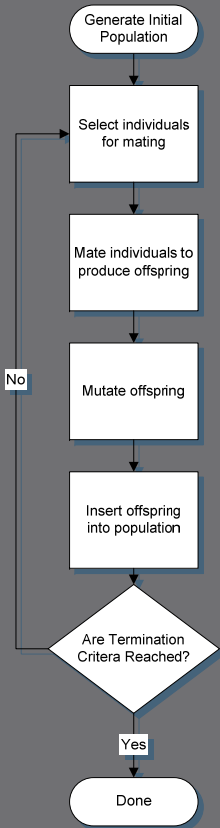


# Sensor Placement Optimization

## Genetic Algorithm

### Advantage of Genetic Algorithms

- Take into account multifaceted battlespace environmental effects
- Optimize placement based on operational mission objectives (performance criteria) with consideration for battlespace limitations (constraints)
- Enable multi-objective optimization



Global Minimum

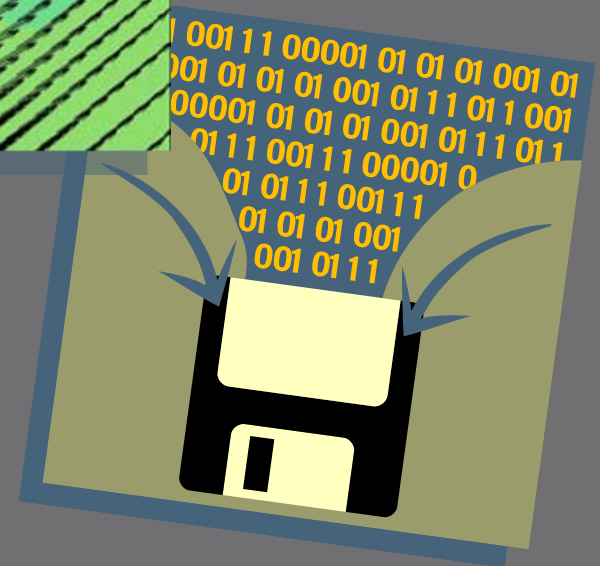
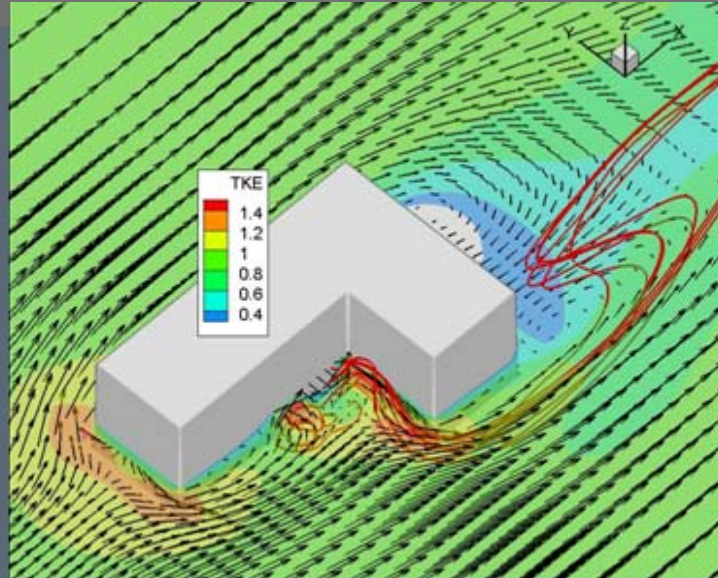
Local Minimum

# Sensor Placement Optimization

## Genetic Algorithm

### Challenges

- very large sets of data
- time required to generate this data





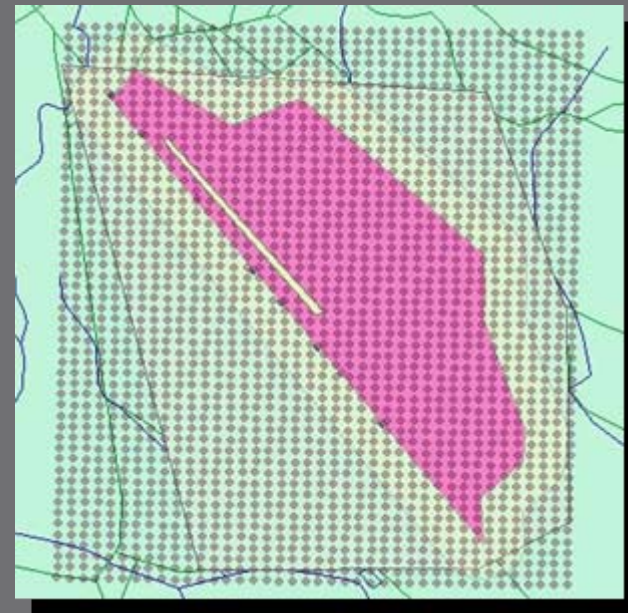
# Simulation & Sensor Modeling

- NCBR II
  - Simulates multiple CB simultaneously in real time
  - Validated physics-based models for hazard propagation - DTRA's SCIPUFF
  - 4D met
  - 3D terrain
- DAS-A (Dial-A-Sensor – Analyzer)
  - Simulation tool for representing any CB particle and vapor sensors
  - Point and stand-off
  - Active and passive systems
  - Capability to “dial” parameters to set performance characteristics for each detector family



# Sensor Grids









- The bulk of the optimization time is spent in the simulation and sensor modeling phase.
- To minimize this, we
  - Separate the hazard from the sensor modeling
  - Separate the sensor modeling from the optimization
- By creating many possible sensor placements and analyze the results for all of them.
  - Sensor placements are defined on a grid.
  - Uniform 2D grids have been used thus far.
- The GA then finds the optimal within the grid.

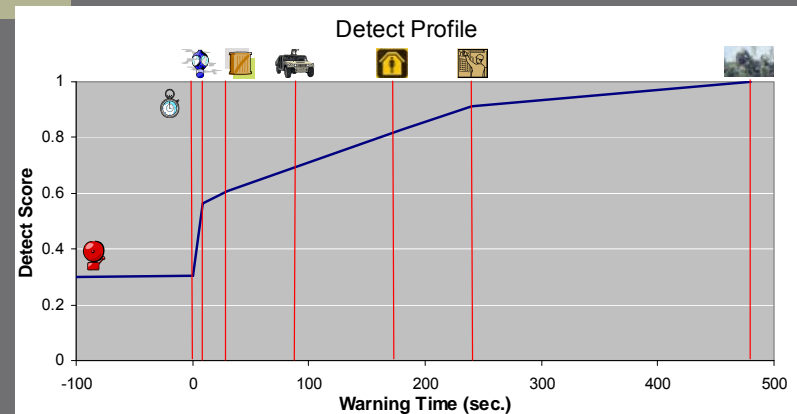


# Scoring the Results

$$fitness = \sum_{threat} \left[ w_{threat} \cdot \sum_{CA} (w_{CA} \cdot f(sensors, alarms_{threat}, alarms_{CA})) \right] \cdot Constraint(sensors)$$

- Combines
  - Threats
    - Agent
    - Delivery
    - Attack placement
    - MET
  - Critical Asset
- Weighting values:
  - Attack Threat
    - Agent vulnerability
    - Agent Likelihood
    - MET Probability
  - Critical Asset importance
- Scoring function determines how complete preparations should be at any given time.
- Determine applicable preventative measures
  - Importance of action
  - Time required to enact

Activity	Required Time (sec)	Relative Importance
 Detect	N/A	5
 Zero Warning	0	1
 Mask	9	6
 Shelter Critical Supplies	30	1
 Shelter Critical Equipment	90	2
 Personnel move to shelter	180	3
 Shutdown Building HVAC	240	2
 Suit up	480	2



# Constraints & TTP

- Hard (Fatal) Constraints
  - Areas where a sensor cannot be placed
  - Solutions edited or removed before continuing
  - Ex. Facilities, Lake, Roadways
- Soft Constraints
  - Areas where we don't want to put the sensor
  - But could if it were a good solution
  - Score penalized
  - Ex. Marshland, unprotected area

## Site Selection

Lt. yellow – CA

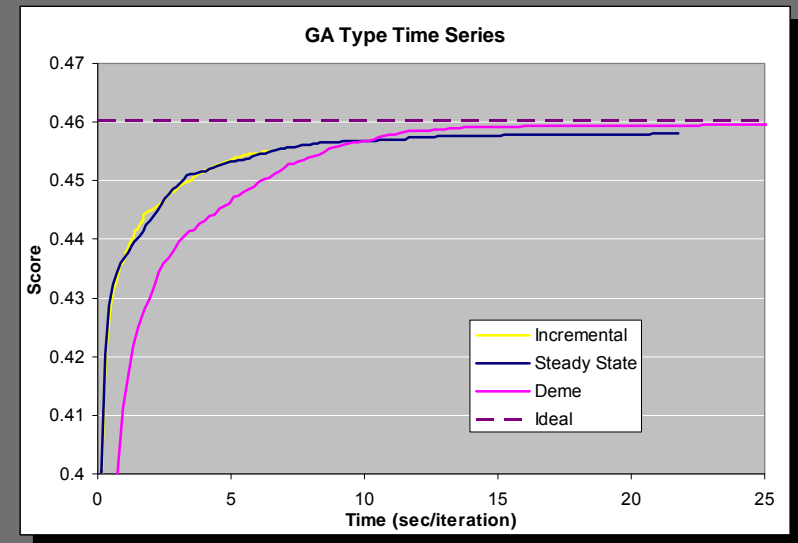
Pink – Perimeter

Lt. Green – Constraint  
(Max. Sensor Distance)



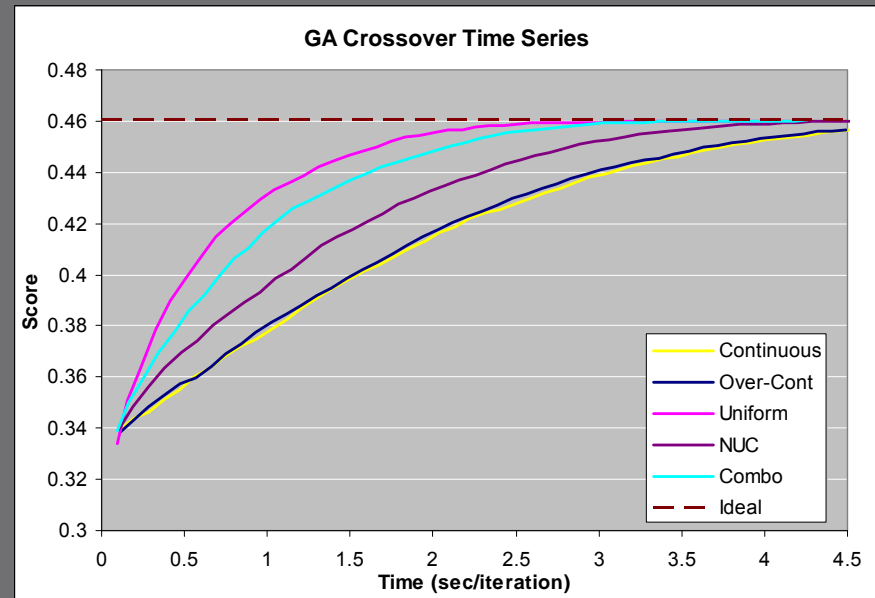
# Genetic Algorithm

- Investigated performance of different GA implementations.
  - The **Steady State** implementation is the standard one.
  - The **Incremental** generates a pair of solutions for each generation instead of creating a nearly-new population.
  - The **Deme GA** is similar to the Steady State but uses several independent sub-populations and allows 'migration' of good solutions between these.
- **Performance:** The Deme takes longer to converge, but for the test problem it consistently produces the ideal solution, which is not true of the other two.



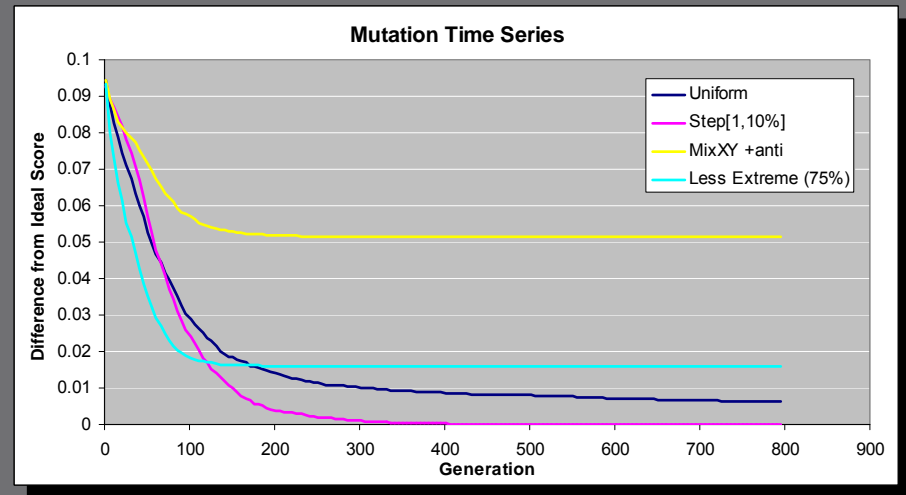
# GA Crossovers

- Tested Several Crossovers: Uniform, Continuous, Over-Continuous, Non-Uniform Continuous, and combinations.
- **Performance:** In the test model, Uniform and Non-Uniform Continuous converged quicker than the Continuous & Over-Continuous crossovers.
- A combination of Uniform and Continuous crossovers randomly chosen has been the crossover of choice.



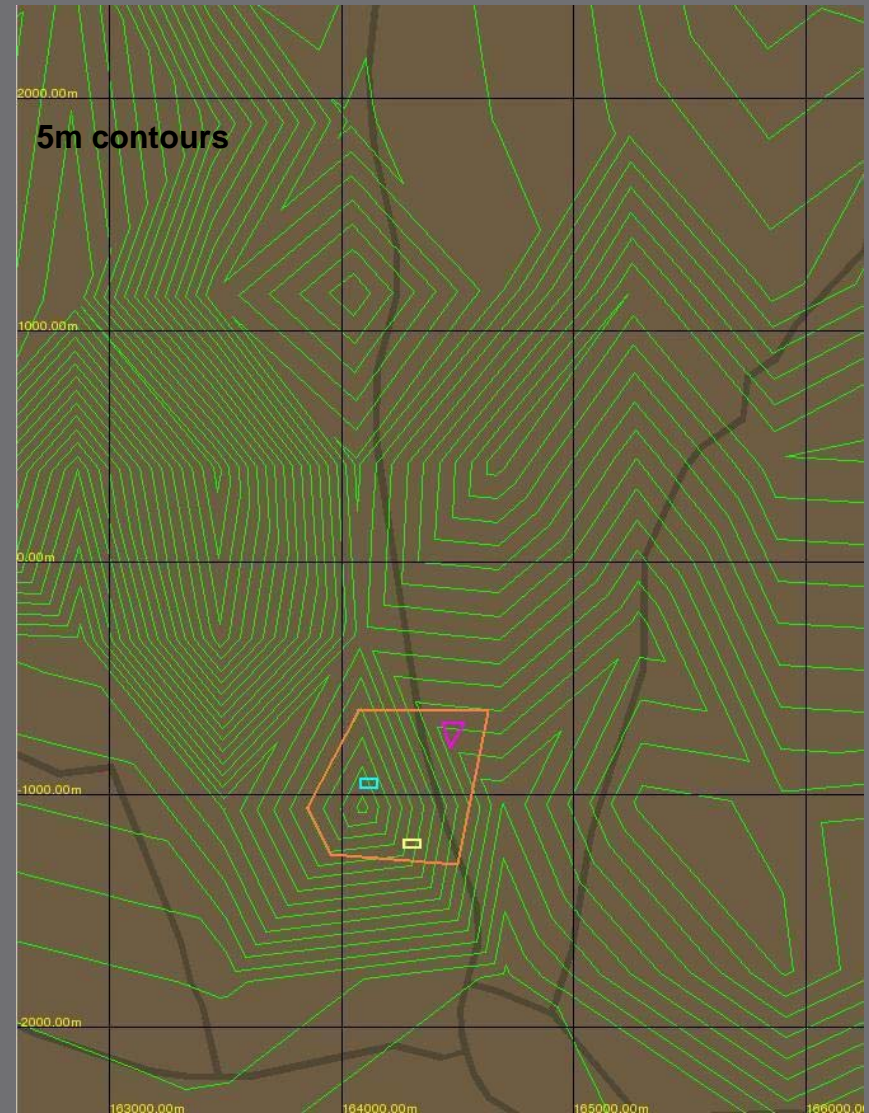
# GA Mutations

- Tested many Mutations: Uniform, n-Steps in a Cardinal Direction, Picking a gene from a good solution, moving to (or toward) an extreme, and combinations.
- **Performance:** A rule of thumb for the ideal mutation rate is  $(2 \cdot \text{Sensors})^{-1}$ . Testing shows this to be fairly accurate.
  - Convergence with the Uniform mutator is slow.
  - Pick a gene speeded convergence early, but hindered convergence to the global optimum.
  - The effectiveness of the Step mutator varies by the step size.
  - The extreme mutator can be helpful, but is solution dependant.
  - A mixture of methods helped.



# SLOTS GA Test

- Fort Hunter Liggett
- Rolling Hills
- Multiple Critical Assets
- Single Agent: GB
- Delivery
  - Scud (500kg)
  - 122mm Artillery Volley
  - 100kg Bomb
  - Line Spray from nearby roads
- Attack Placement
  - Several per Delivery & MET
- Using historic MET
  - Two wind directions (N, NNE)
  - Wind speeds at average + 1 standard deviation
  - Average Temperature
- 34 Simulations Total





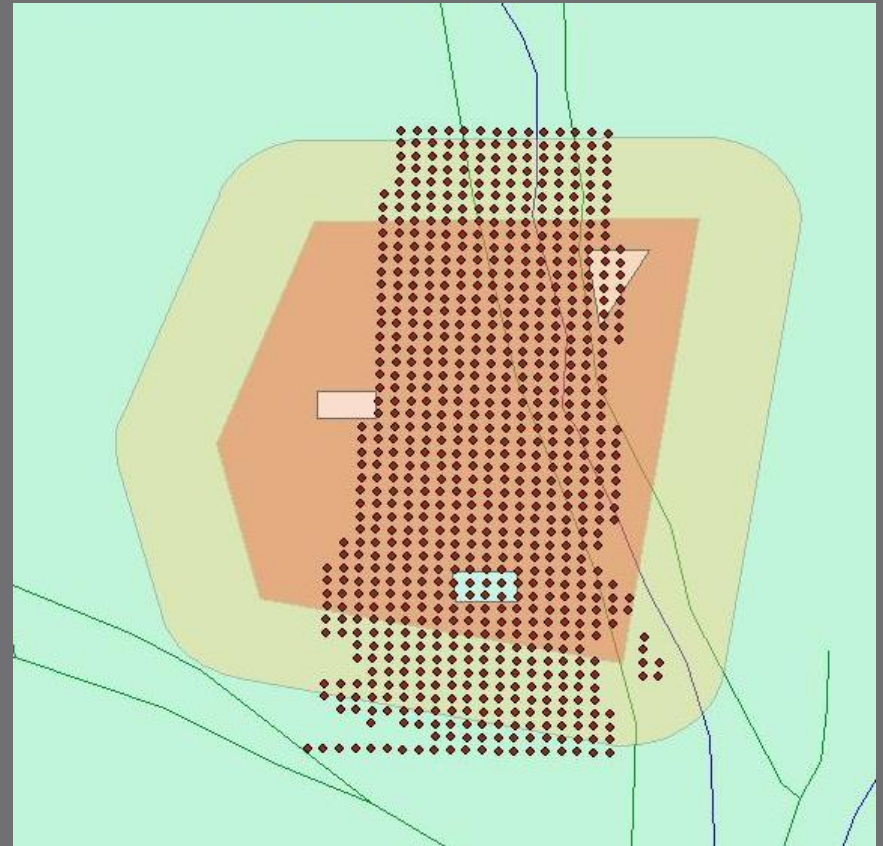
# Optimization

- Layout
  - Grid (25m x 25m)



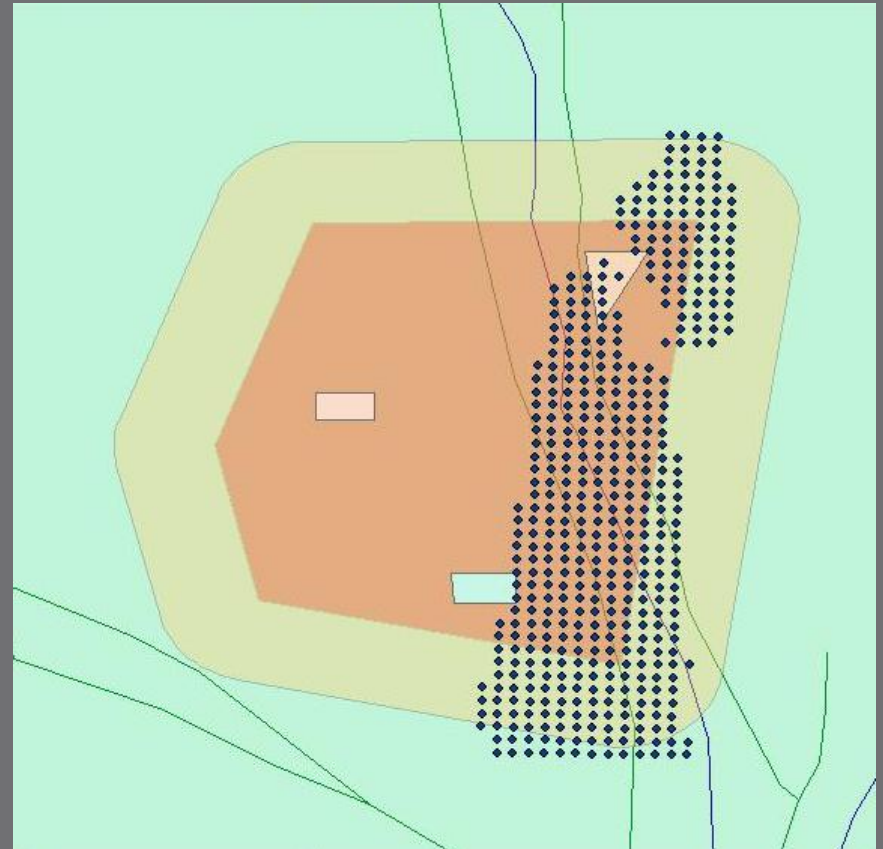
# Optimization

- Layout
  - Grid (25m x 25m)
- Simulations
  - North Winds
    - Artillery @ 1100m



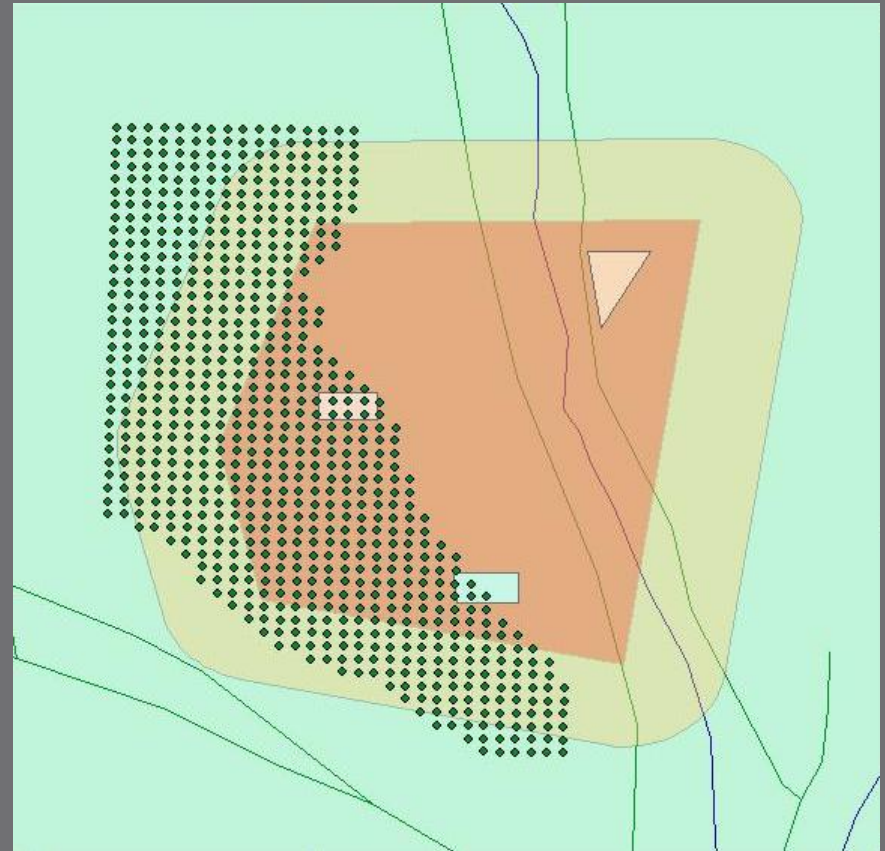
# Optimization

- Layout
  - Grid (25m x 25m)
- Simulations
  - North Winds
    - Artillery @ 1100m
    - Line Spray @ 250-1500m



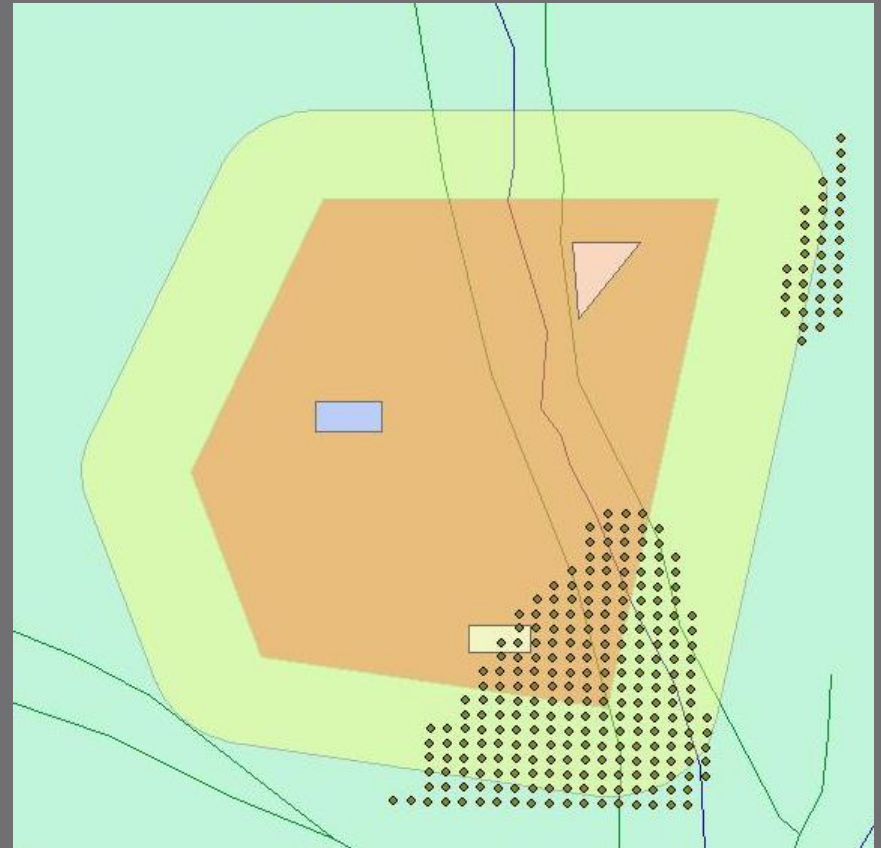
# Optimization

- Layout
  - Grid (25m x 25m)
- Simulations
  - North Winds
    - Artillery @ 1100m
    - Line Spray @ 250-1500m
  - NNE Winds
    - Scud @ 2000m



# Optimization

- Layout
  - Grid (25m x 25m)
- Simulations
  - North Winds
    - Artillery @ 1100m
    - Line Spray @ 250-1500m
  - NNE Winds
    - Scud @ 2000m
    - 100kg Bomb @ 500m



# Optimization

- Layout
  - Grid (25m x 25m)
- Simulations
  - North Winds
    - Artillery @ 1100m
    - Line Spray @ 250-1500m
  - NNE Winds
    - Scud @ 2000m
    - 100kg Bomb @ 500m
- Optimal



# Benchmark and Optimal



**Benchmark**

Score = 0.3814767



**Optimal**

Score = 0.4719485

## Why is the SLOTS generated Optimal solution better?

	Benchmark	Optimal
1 <sup>st</sup> Sensor Detects	71	90
2 <sup>nd</sup> Sensor Detects	30	58
3 <sup>rd</sup> Sensor Detects	6	33
4 <sup>th</sup> Sensor Detects	none	10
Undetected Asset Contaminations	17 assets on 9 attacks	0
# of Advance Warnings	58	64
Average Warning (sec)	134.2	125.9



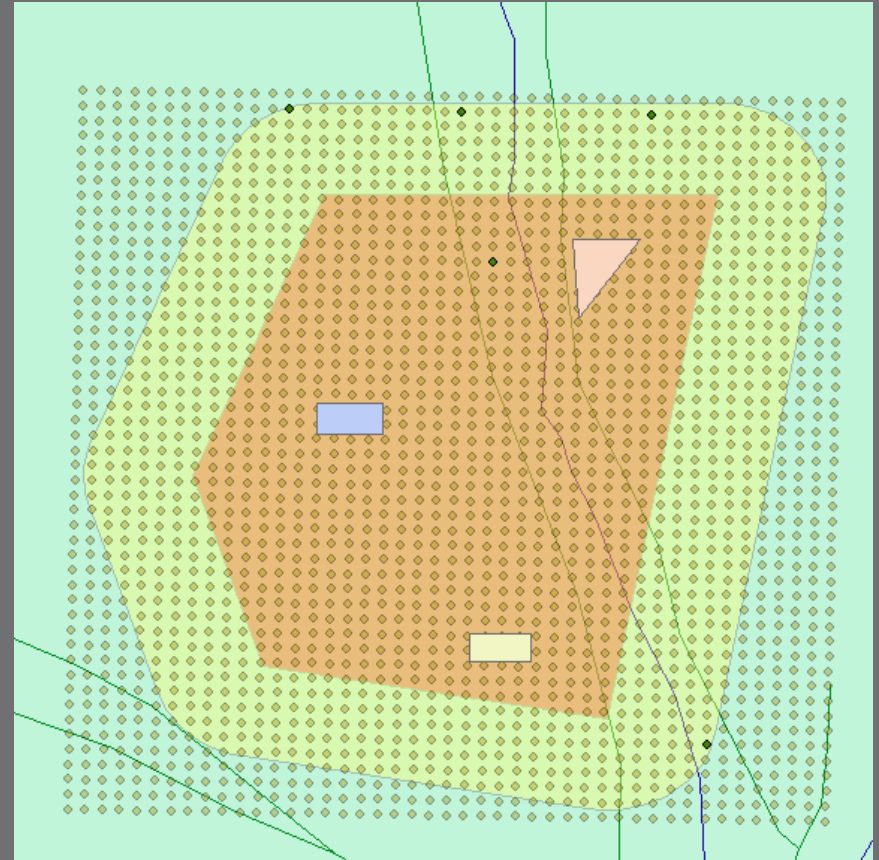
# Why is that Sensor there?

- Sensors 1 & 6 are critical, they catch attacks with no other detection.
- Sensor 5 detects line spray attacks before the other sensors.
- Sensor 4 has the fewest detections, but almost all are 1<sup>st</sup> detects.
- Sensor 6 catches 6 of the 8 attacks the Benchmark misses.



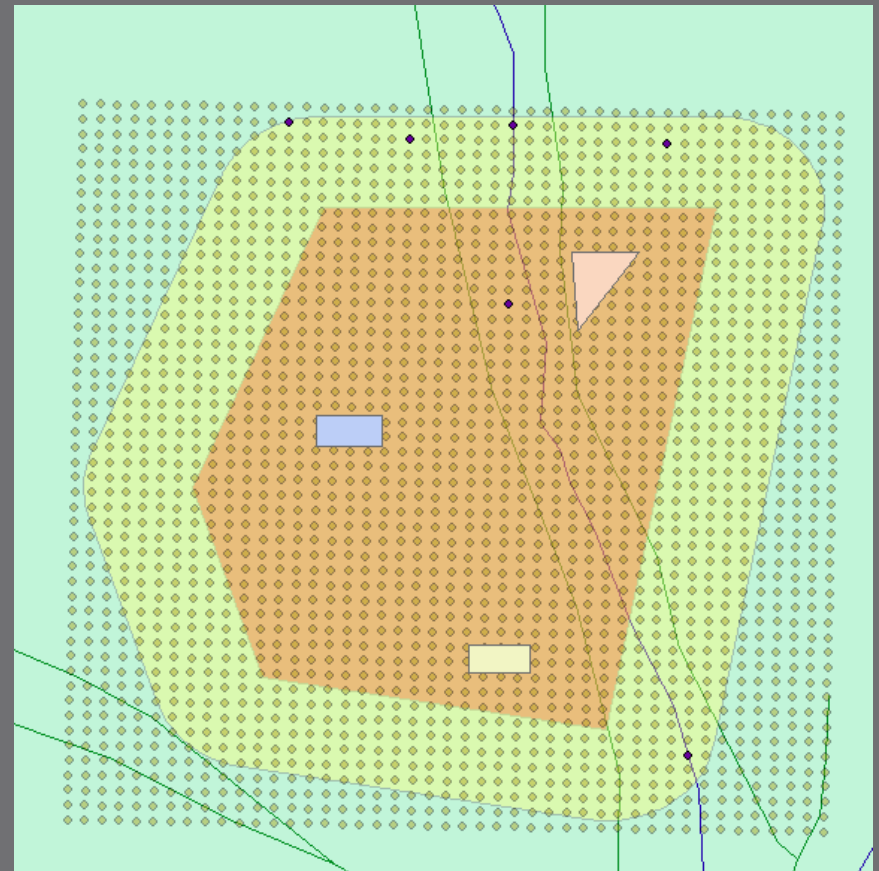
# What happens with a smaller sensor kit?

- The sensor that is lost is from the Picket Fence.
- Interior and SW sensors retained.
- Fitness:
  - 6 sensor = 0.4719485
  - 5 sensor = 0.4509517
  - 4.5% decrease



# Adjacency Constraint

- Added a constraint to penalize sensors for being too close together.
- Exponential Decay function
- Fitness:
  - 6 sensor = 0.4719485
  - Adj Cnstr = 0.4668766
  - 1.0% decrease
  - Min distance > 150m

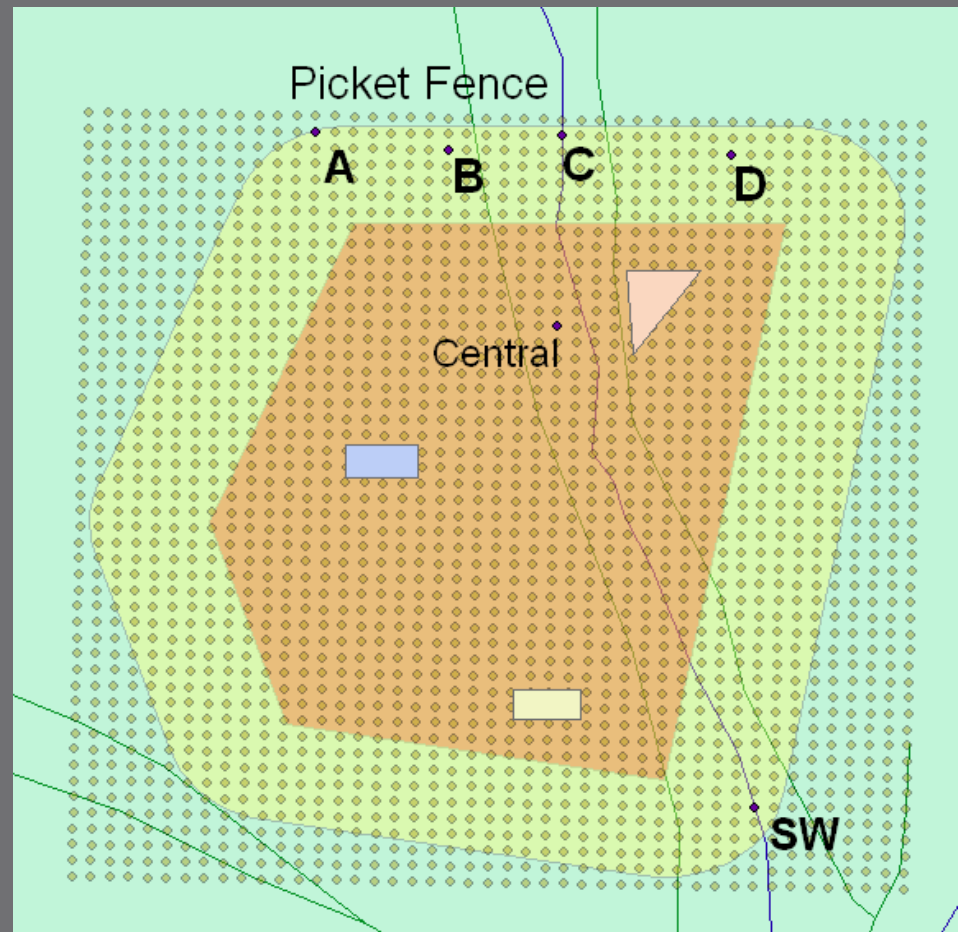


# Why is the Optimal Better?

	Benchmark	Optimal	w/ Adj Constraint
1 <sup>st</sup> Sensor Detects	71	90	90
2 <sup>nd</sup> Sensor Detects	30	58	54
3 <sup>rd</sup> Sensor Detects	6	33	23
4 <sup>th</sup> Sensor Detects	none	10	7
Undetected Asset Contaminations	17 assets on 9 attacks	0	0
# of Advance Warnings	58	64	65
Average Warning (sec)	134.2	125.9	130.2

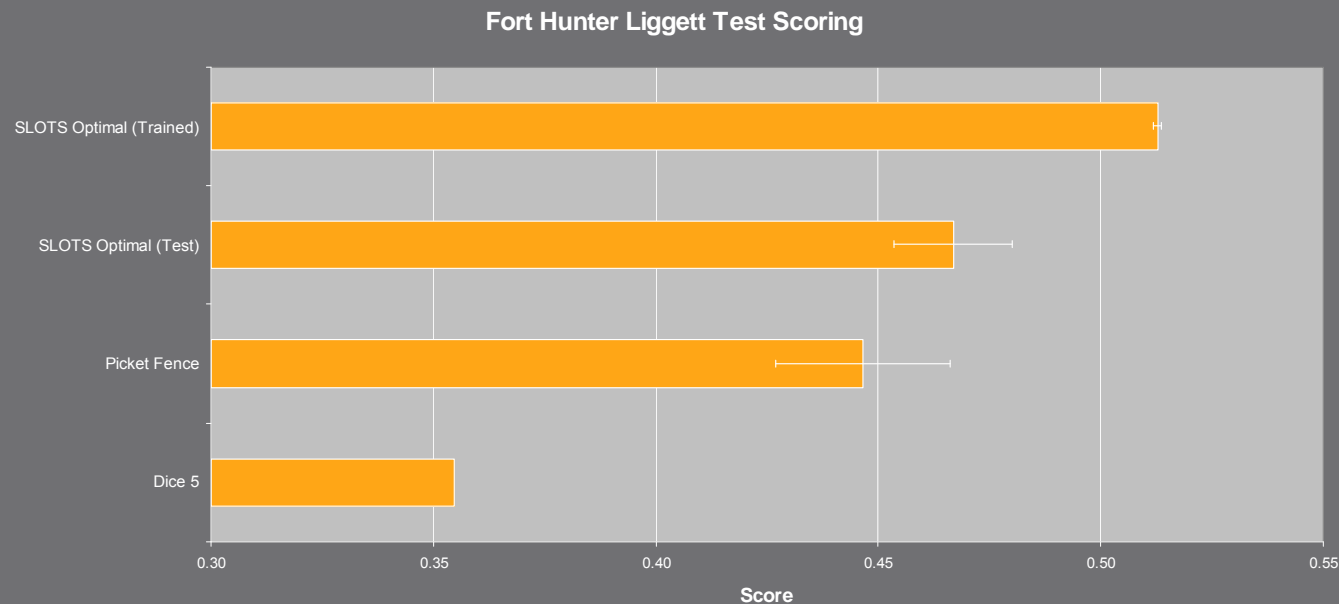
# Why is that Sensor there?

- The SW sensor is critical.
  - Catches 69% of the attacks with no other detection
  - 11 first detects
  - Detects 44% of the attacks
- The Central sensor provides first detects for line spray attacks. And provides detects on 41% of attacks.
- The Picket Fence combines for 21 first detects with each sensor taking a fairly equal share.
- Picket Fence A has 19% of only detects.
- Picket Fence B detects 35% of all attacks

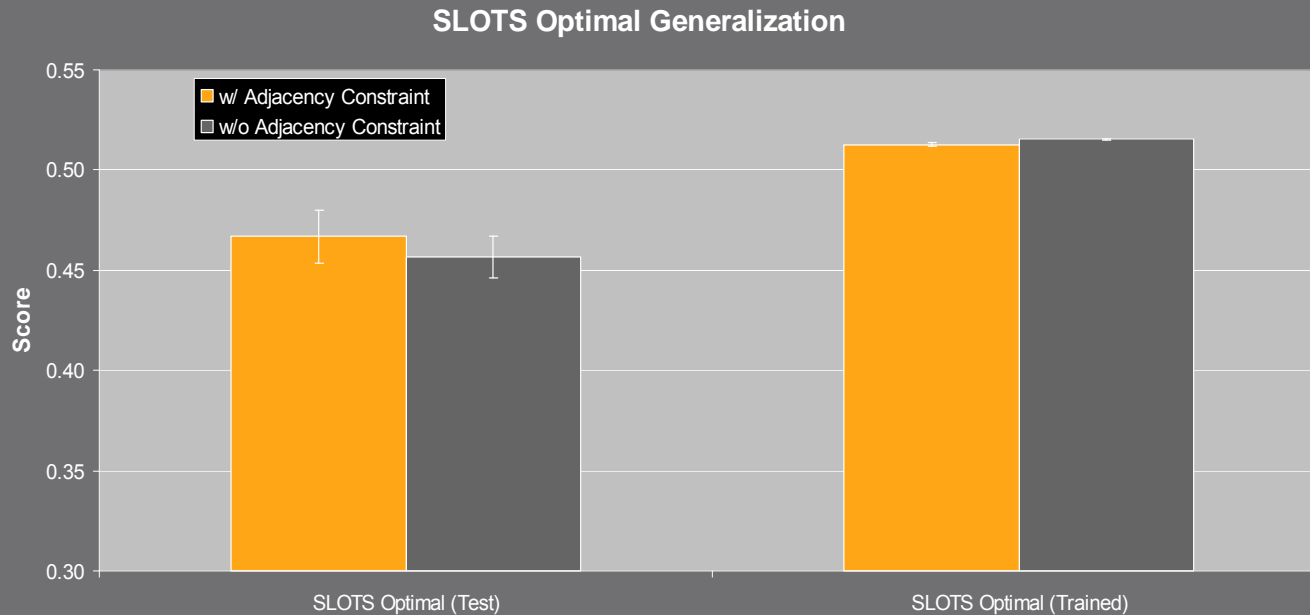


# Does a SLOTS Layout *Generalize*?

- SLOTS optimal layouts are based upon a representative sample of attacks.
- How does it fare against attacks that it has not seen?
- Tested using Leave-one-out Cross-validation.
- Scores better than doctrine Methods

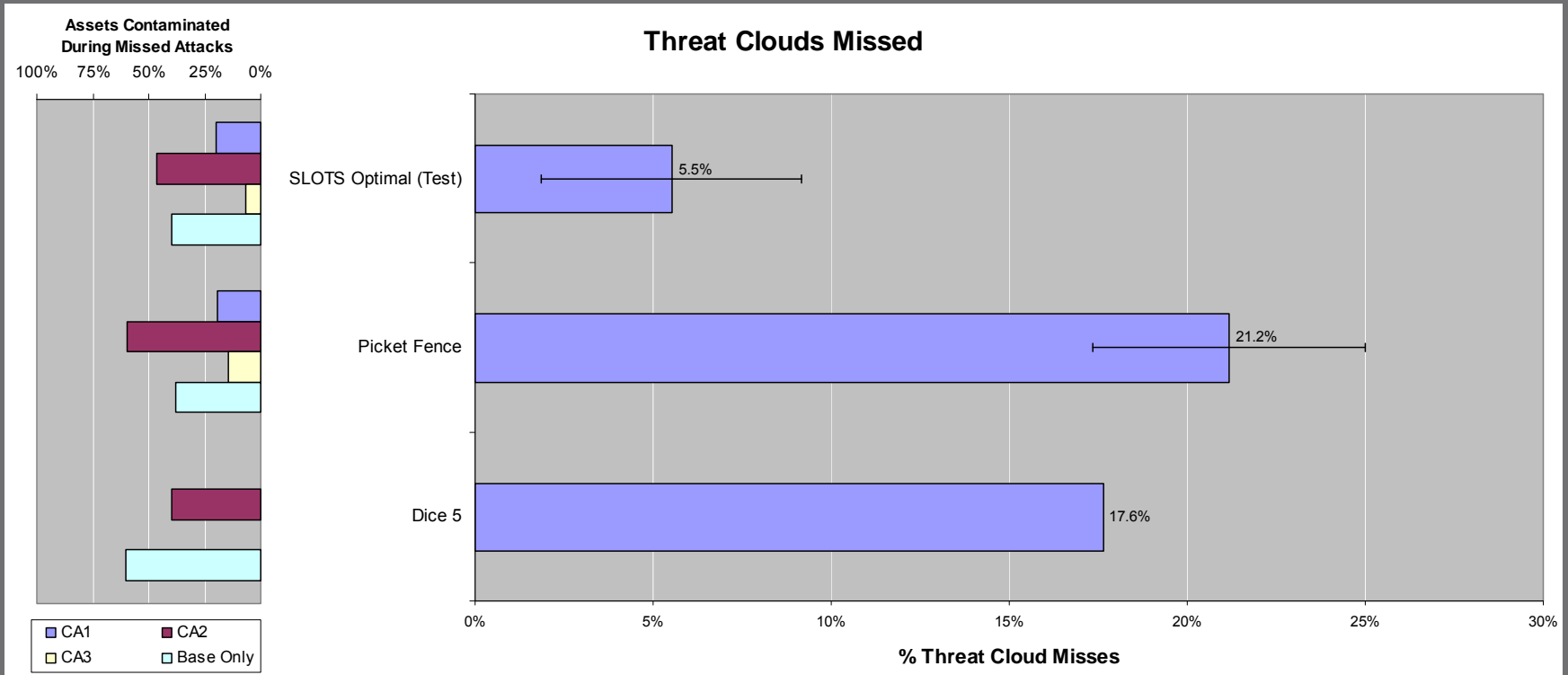


# Generalization and Adjacency



- Adding a common sense ‘Adjacency Constraint’
  - penalizes sensor layouts with sensors near one another
- Results in layouts with slightly poorer scores
- But cross-validated test scores improve by 2.2%

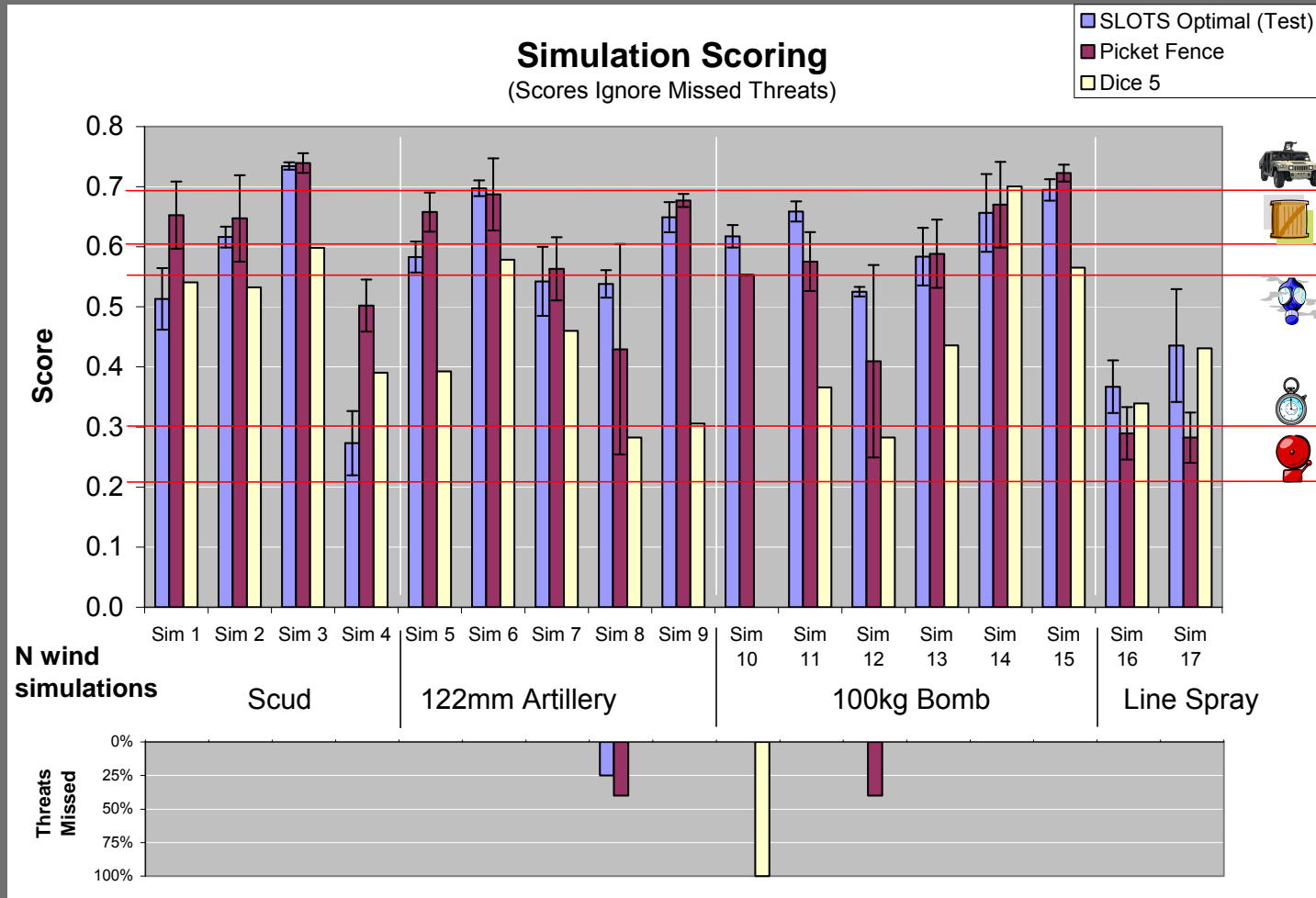
# Threat Cloud Detection



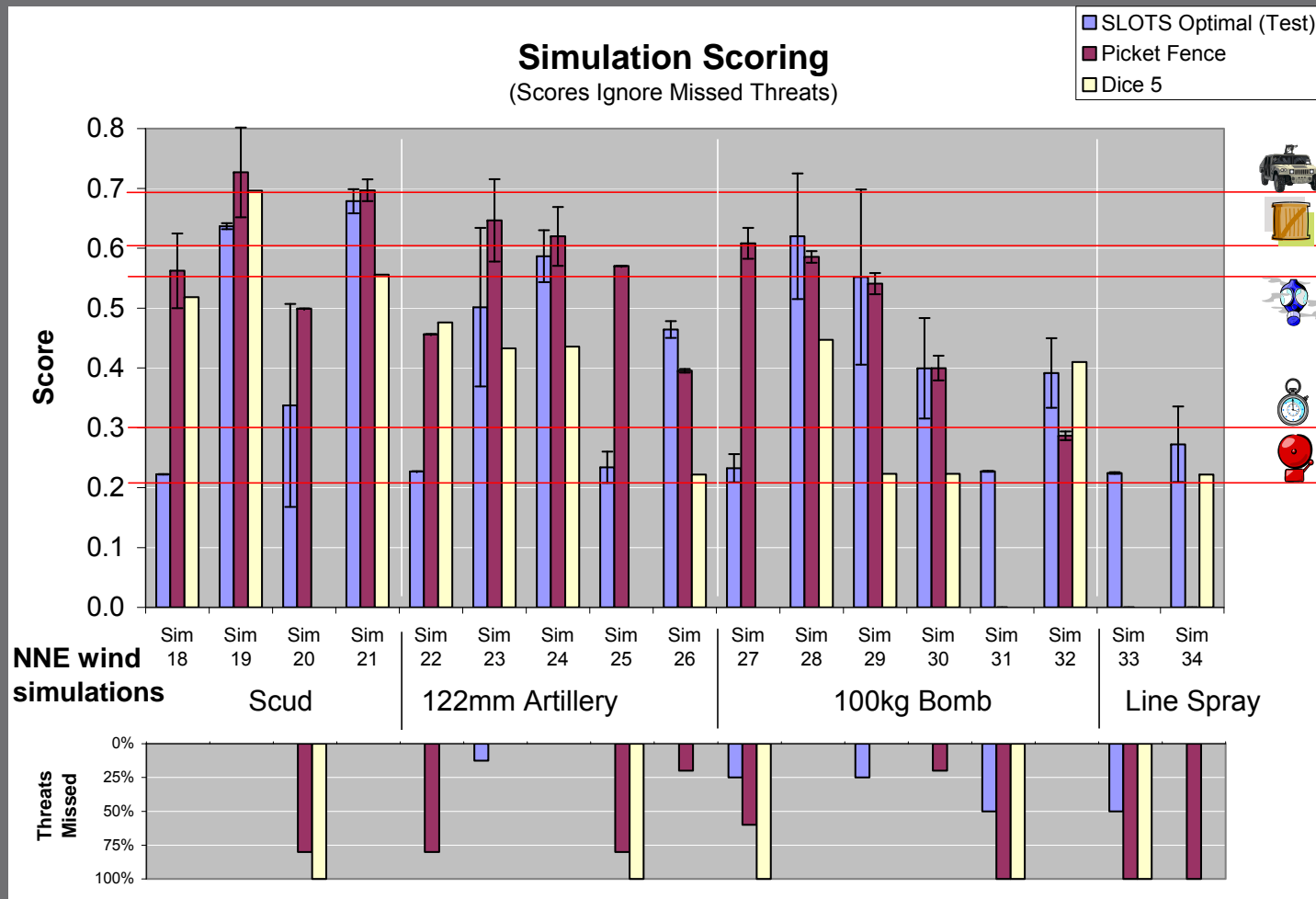
- SLOTS is able to detect threat clouds more reliably than doctrine.



# Scoring Each Simulation



# Scoring Each Simulation



# Summary

- Automated Rules Placement Tool offers insight into the implications of implementation of sensor placement TTP
- Genetic Algorithms offer sufficient capability for optimizing sensor placement given
  - Constraints are defined
  - Performance criteria are understood
- Significant research still remaining