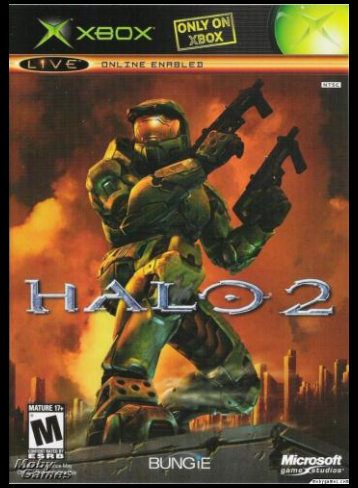
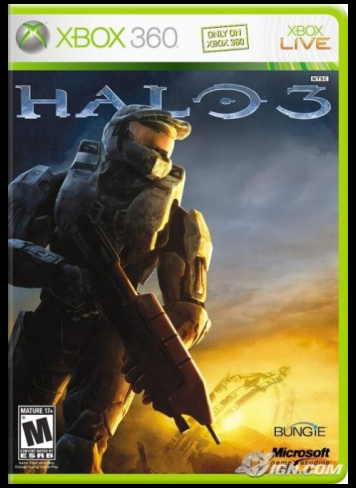
AI POSITIONING AND SPATIAL EVALUATION: A PRIMER

Damián Isla, Co-Founder, The Molasses Flood

Massachusetts Institute of Technology

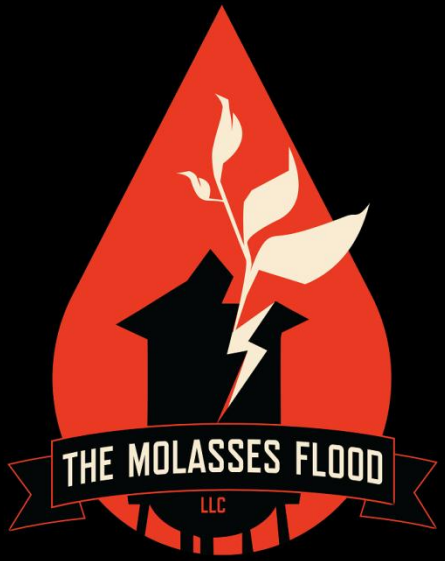
Media Laboratory





BIOSHOCK INFINITE

MOONSHOT GAMES


THE MOLASSES FLOOD
LLC


THE FLAME
IN
THE FLOOD
STEAM
UNREAL
ENGINE

“Spatial Awareness”

(A Theme, not a technique, or technology, or algorithm)

Spatial Abilities in Halo 2

- Static Pathfinding
  - Navigation mesh (ground)
  - Waypoint network (airborne)
  - Raw pathfinding
  - Path-smoothing
  - Hint integration (jumping, hoisting, climbing)
  - Static scenery-based hints
  - Static scenery carved out of environment mesh
- Static feature extraction
  - Ledges and wall-bases
  - Thresholds
  - Corners
  - Local environment classification
- Object features
  - Inherent properties (size, mass)
  - Oriented spatial features
  - Object behaviors (mount-to-uncover, destroy cover)
- Dynamic Pathfinding
  - Perturbation of path by dynamic obstacles
  - "Meta-search" / Thresholds / Error stages
  - Obstacle-traversal behaviors
    - Vaulting, hoisting, leaping, mounting, smashing, destroying
- Path-following
  - Steering on foot (with exotic movement modes)
  - Steering a vehicle (e.g. ghost, warthog, banshee)
- Interaction with behavior
  - What does behavior need to know about the way its requests are being implemented?
  - How can pathfinding impact behavior?
- Body configuration
  - Flying, landing, perching
  - Cornering, bunkering, peeking
- Spatial analysis
  - Firing position selection
  - Destination evaluation based on line-of-sight, range-to-target, etc.
- "Local spatial behaviors"
  - Line-tracing (e.g. for diving off cliffs)
  - Not facing into walls
  - Crouch in front of each other
  - Don't walk into the player's line of fire
  - Curing isolation
  - Detecting blocked shots
- Reference frames
  - The viral nature of the reference frame
- Cognitive model / Object persistence
  - Honest perception
  - Simple partial awareness model
- Search
  - Simple by design
  - Group search
- Spatial conceptualization
  - DESIGNER-PROVIDED
  - Zones, Areas (areas), Firing positions (locations)

The Most Fundamental of Questions

Where do I stand right now?

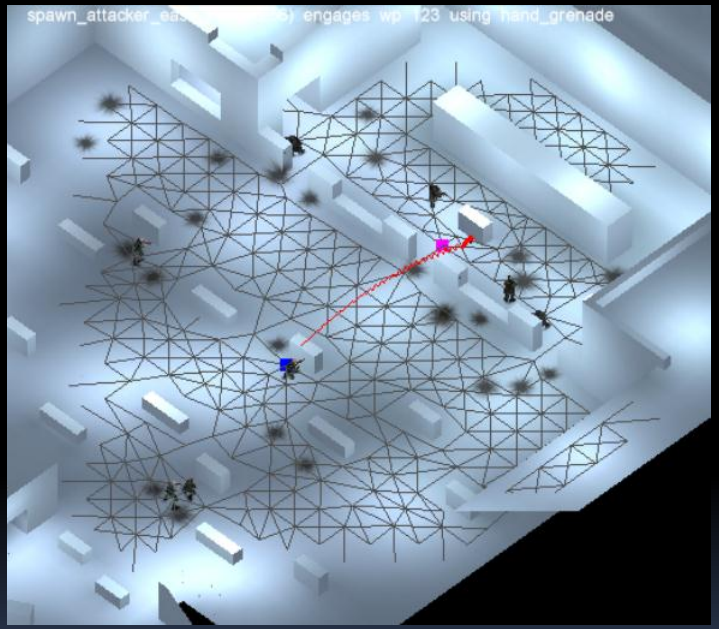
- Depends on a *huge* amount of context.
  - Internal: goals, intentions, behaviors, etc.
  - External: target position, actions, obstacles, etc.
- Extremely player-facing / gameplay relevant
- Should be in the hands of the designers.

Position Selection

1. Gather potential positions
2. Score each [accessible] position with F(x)
3. Choose the best one
4. Go there

Representation

Point cloud (Halo 2) +
Navigation Mesh



Navigation Graph (Killzone)

(Image from *Killzone's AI: Dynamic Procedural Combat Tactics*, by R. Straatman, W. Van Der Sterren, A. Beij, GDC 2005)

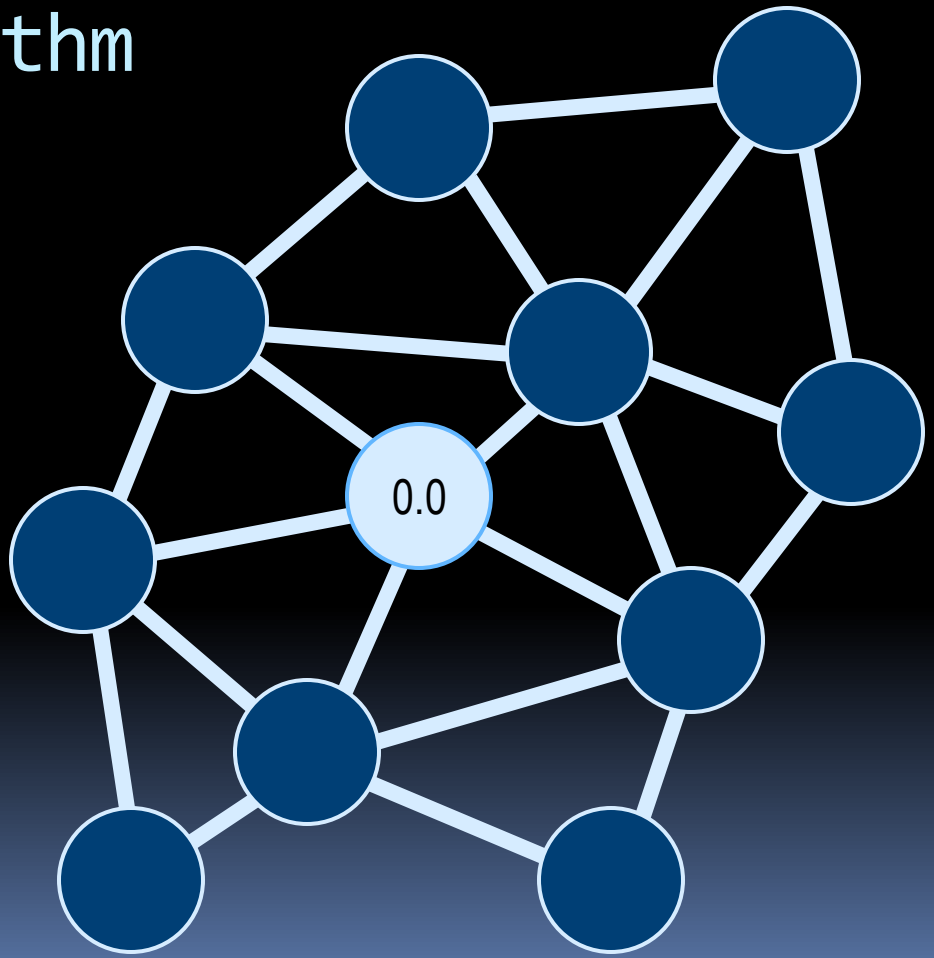
Representation

Regular Grid (Third Eye Crime)

Gather Step

- Point clouds
  - Points assigned by designers (e.g. Halo 2)
  - Spatial query (points within radius or box)
- Nav-mesh & Regular Grid
  - all the above, plus
  - Dijkstra's algorithm to find accessible positions

Dijkstra's Algorithm



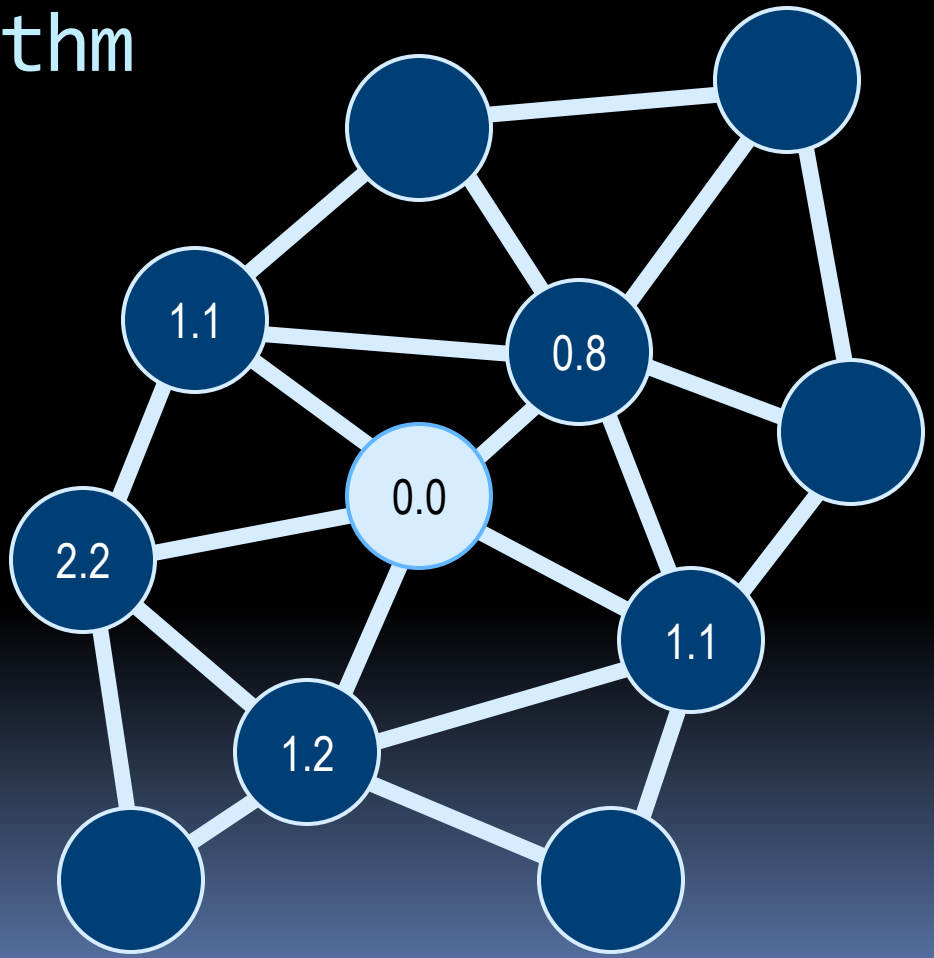
Find

- Accessible points
- Path distances
- Reconstruct paths

Dijkstra's Algorithm

Find

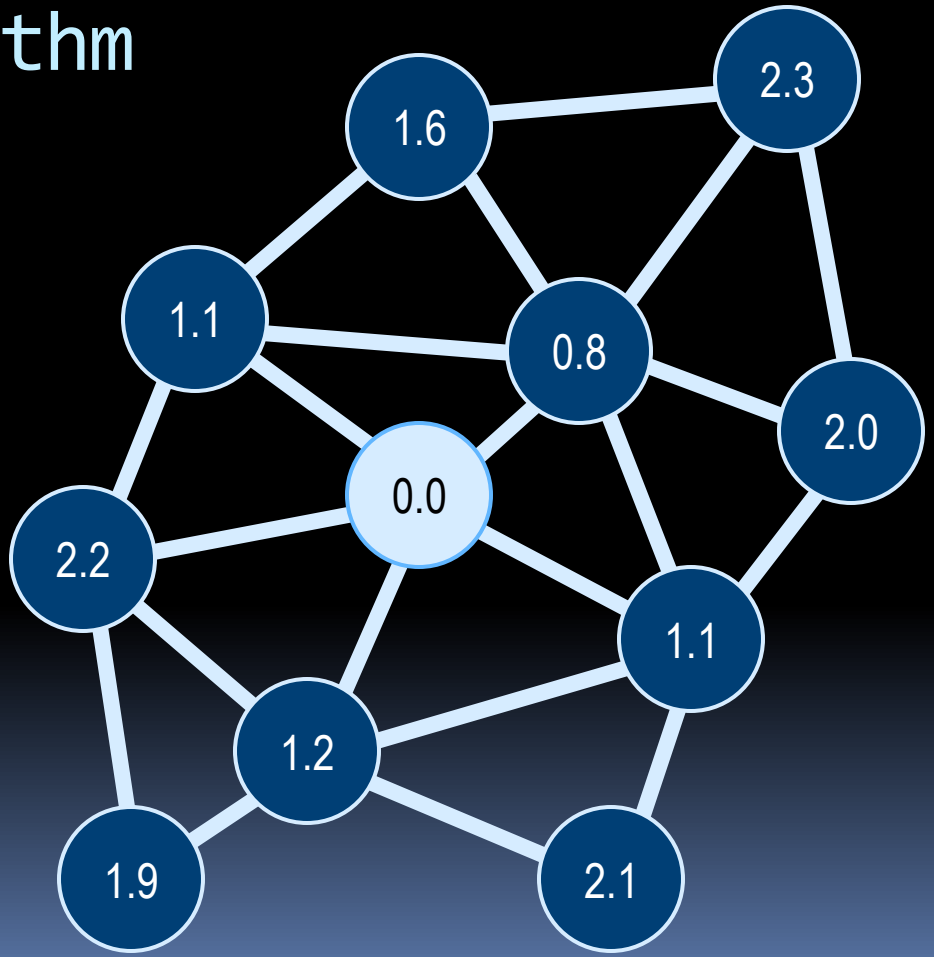
- Accessible points
- Path distances
- Reconstruct paths



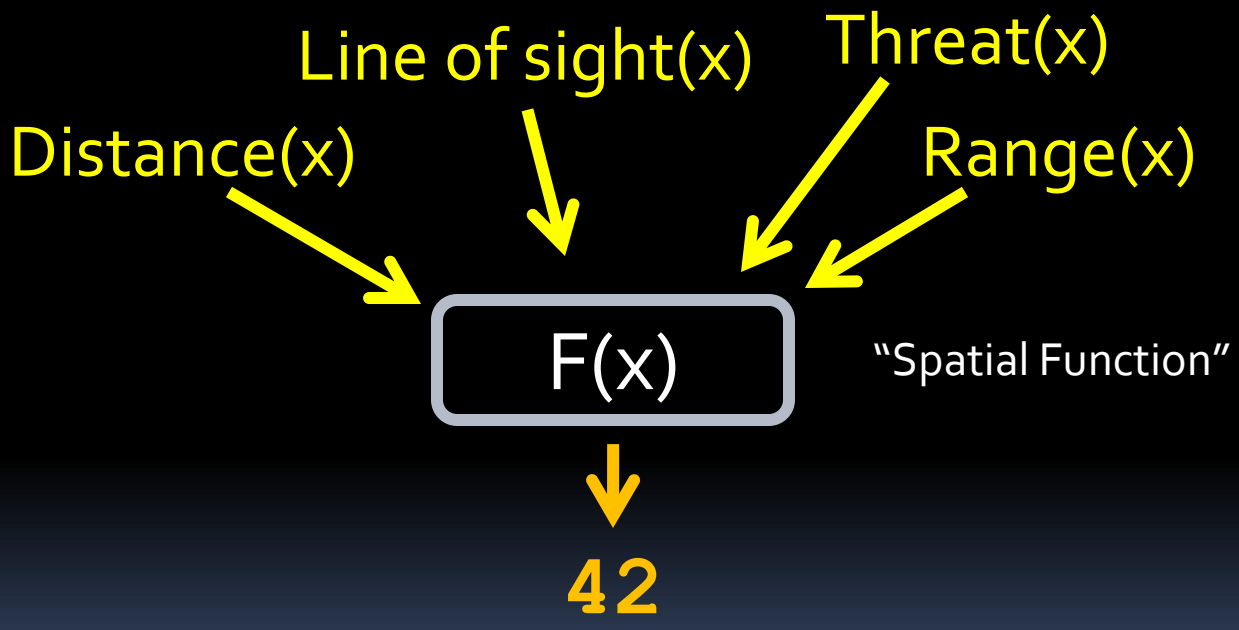
Dijkstra's Algorithm

Find

- Accessible points
- Path distances
- Reconstruct paths



Position Scoring



The Apples-to-Oranges problem.

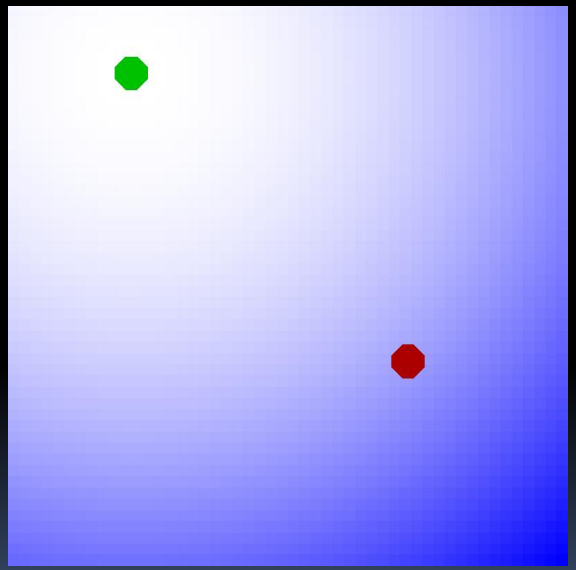
Spatial Function

Inputs

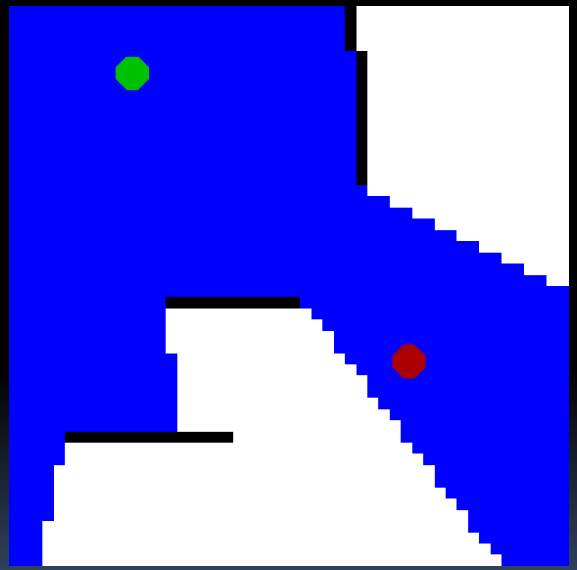
- A(x) = range from x to target
- B(x) = path distance to x
- C(x) = line of sight from x to target
  (1.0 = 100% clear)
- D(x) = distance to occupied space
- …

RE-use Dijkstra's from gather phase (Nav-mesh or grid)

Spatial Function Inputs



range



LOS

Spatial Function

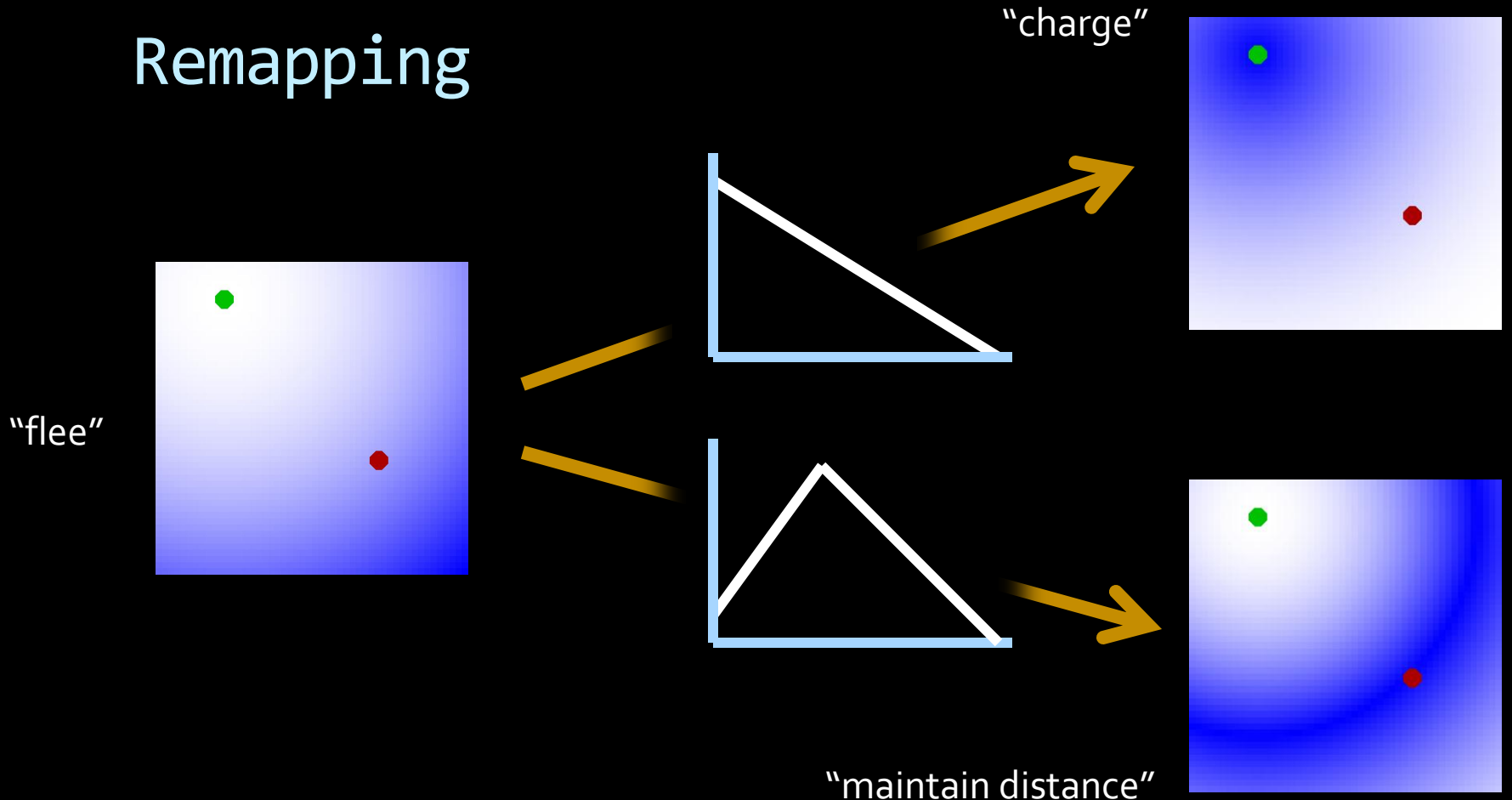
- Simplest form

  $F(x) = k_1A(x) + k_2B(x) + k_3C(x) + \ldots$

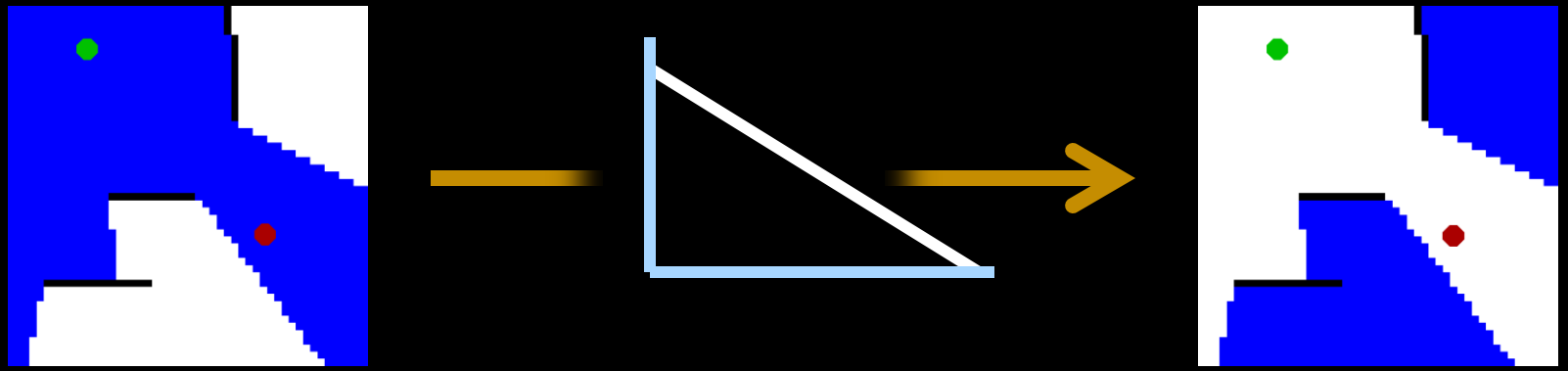
- With remapping:

  $F(x) = f_1(A(x)) + f_2(B(x)) + f_3(C(x)) + \ldots$

Remapping

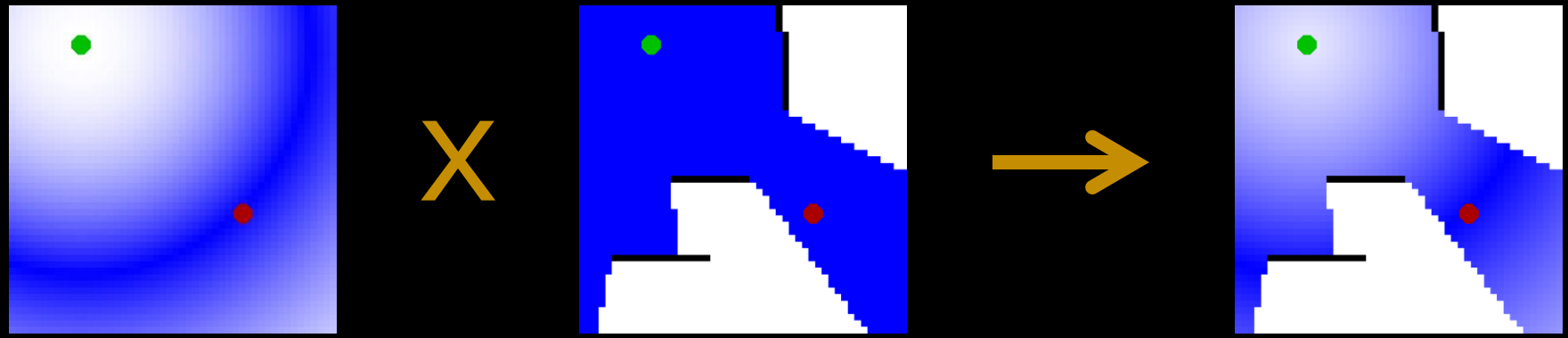


Remapping



“find”

“cover”



Spatial Function

- Simplest form

  $F(x) = k_1A(x) + k_2B(x) + k_3C(x) + \ldots$

- With remapping:

  $F(x) = f_1(A) + f_2(B) + f_3(C) + \ldots$

- Ideally, use a flexible syntax:

  $F(x) = k(f_1(A) - f_2(B)) / (f_3(C)*f_4(C)) \ldots$

- Our own idiosyncratic form:

  $F(x) = (((f_1(A) + f_2(B)) + f_3(C)) * f_4(D)) + f_5(E) \ldots$

"Layer"

Implementation

Layers

- Input source
  - range
  - los
  - path-distance
  - etc.
- Combination method
  - Additive
  - Multiplicative
- Remapping Function
  - output = F(input)
- Global modifications
  - Blur factor
  - Normalization

DEMO

Position Selection + Pathfinding

The criteria for choosing points is not the same as the criteria for getting there

e.g. “choose a spot with clear LOS but try and stay covered while you travel there”

Observation #1

Input functions are expensive

- LOS, path-distance, obstacle-distance, etc.

BUT remapping / combining/sharing is relatively cheap

Therefore:

Once we've computed the input layers, we can likely afford to run multiple spatial functions

Observation #2

Advantage of Spatial Reps w/ Connectivity:

SINCE we probably have expensive spatial input already computed on grid cells / navgraph vertices

And SINCE Dijkstra/A* can accommodate penalty functions

We can use a SEPARATE spatial function to specify a Dijkstra/A* penalty function

- specify both where to go, and how to get there

However...

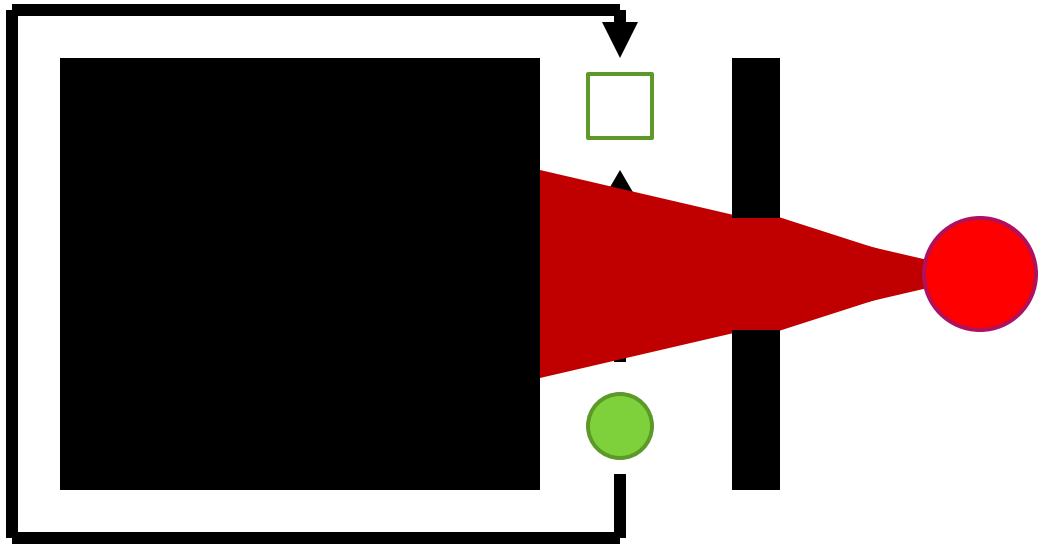
All paths were built into the gather-phase Dijkstra

Demo Solution: Use Dijkstra for gather but NOT for final path creation

- Once position selected, run A* from scratch to that destination using penalty function
- Expensive…

… And still wrong!

- The path-distance input was provided by Dijkstra.
- Not accurate if penalty function is distorting path



Where to Stand vs. How to Get There

Flame in the Flood Solution: Use separate spatial functions for A* penalty (pass 1) and position scoring (pass 2)

Result of penalty function feeds into Dijkstra gather phase of pass 2

- Note that this probably impacts any path-smoothing that you do
- avoid smoothing through masked-out areas

Where to Stand vs. How to Get There

ALSO means two distinct gather phases

- Gather #1: all X within bounding box
  - assume no expensive inputs used
  - or if they are, those input are shareable with pass 2 (e.g. los)
- Gather #2: Dijkstra
  - using penalty values computed in pass 1

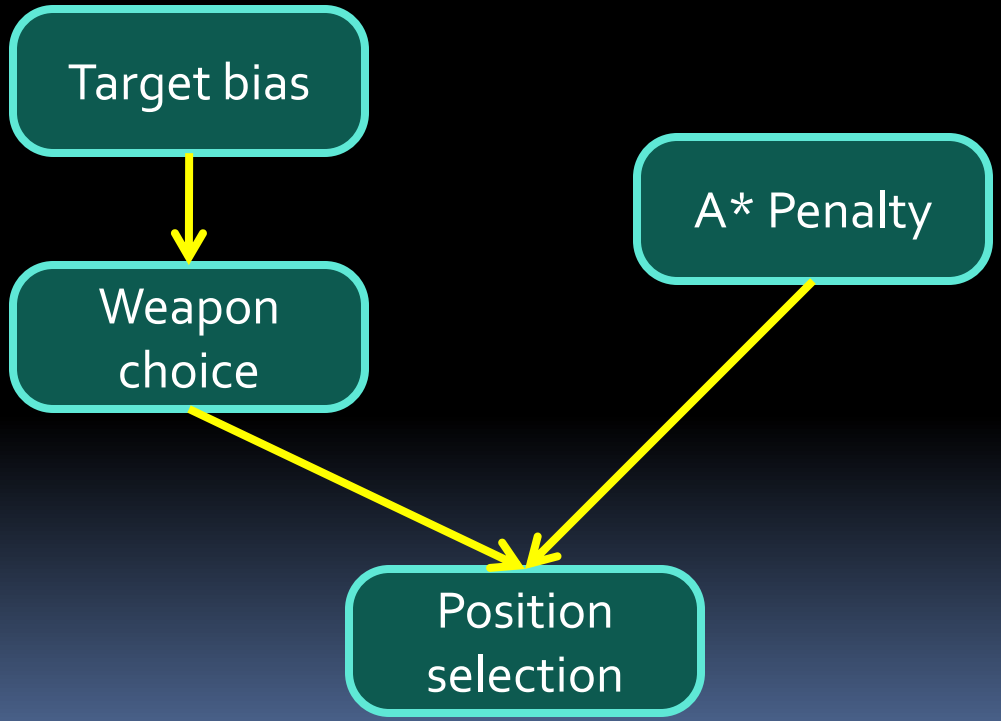
All Behavior is Spatial

Spatial functions can be used for more than just position evaluation

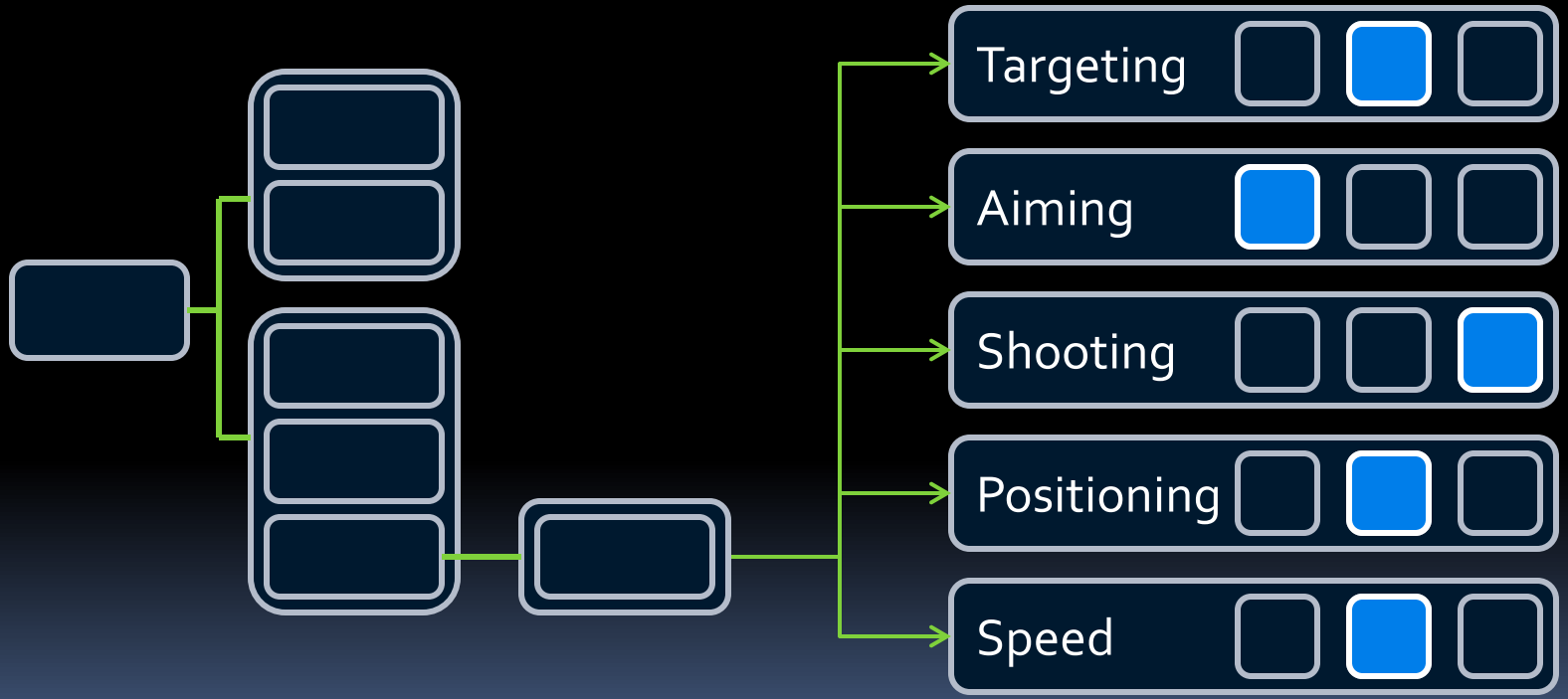
- A* penalty
- path speed
- aim on/off
- target bias
- weapon choice
- …

Remember: Input sources are expensive, but recombining them is cheap
(share inputs across layers, functions and AIs)

Spatial Behavior



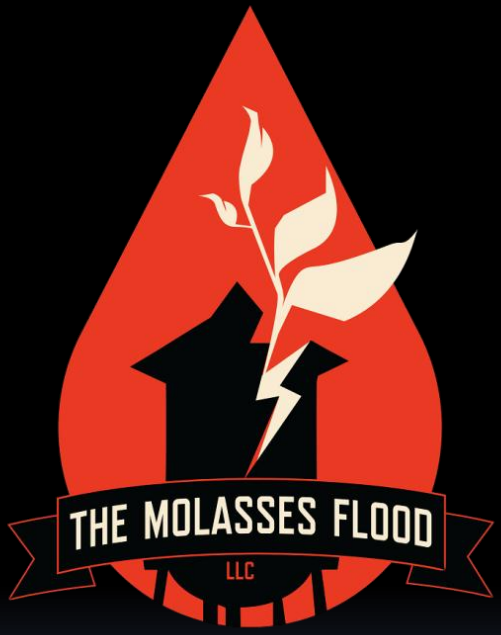
Spatial Behavior



Conclusions

- Apples-to-oranges is defeated through great visualization and iteration tools
- Respect the code/data boundary
- Subtle interaction between position selection and pathfinding
- Spatial functions for many aspects of behavior

Thanks!



Questions?