CS4386 AI Game Programming

Lecture 10
Tactical AI

Semester B, 2020-2021
Department of Computer Science
City University of Hong Kong

Tactical Locations (1)

- In games tactical locations can be characterized by various attributes:

  Defensive locations/cover points

  – In a static area of the game, designer will typically mark locations behind barrels or protruding walls as good cover points so that a character can move to the nearest cover point to provide itself with shelters when it engages the enemy

  Sniper locations

  – The level designer marks locations as being suitable for snipers, and then characters with long-range weapons can head there to find both cover and access to the enemy

  Shadow points

  – In stealth games, characters that also move secretly need to be given a set of locations where there are intense shadows

Tactical Locations (2)

Fire points
- A large arc of fire can be achieved here

Power-up points
- Power-ups are likely to respawn here

Reconnaissance points
- A large area can be viewed easily here

Quick-exit points
- Characters can hide here with many escape options if they are found

- Tactical locations can also be locations to avoid:

  Ambush hotspots

  Exposed areas

  Sinking sand

Tactical Analyses

- Tactical analyses of all kinds are sometimes known as influence maps
- Influence mapping is a technique pioneered and widely applied in real-time strategy games, where the AI keeps track of the areas of military influence for both sides
- Similar techniques have also made inroads into squad-based shooters and massively multi-player games

Representing the Game Level

- For tactical analysis we need to split the game level into chunks
- The areas contained in each chunk should have roughly the same properties for any tactics we are interested in
- Because of the ancestry of tactical analysis in RTS games, the overwhelming majority of implementations are based on a tile-based grid
- For a non-tile-based level, we can impose a grid over the geometry and use the grid for tactical analysis

Simple Influence Maps

- An influence map keeps track of the current balance of military influence at each location in the level
- There are many factors that might affect military influence:
  - the proximity of a military unit
  - the proximity of a well-defended base
  - the duration since a unit last occupied a location
  - the surrounding terrain
  - the current financial state of each military power
  - the weather
  - etc.
- Most games make influence mapping easier by applying a simplifying assumption: military influence is primarily a factor of the proximity of enemy units and bases and their relative military power

Influence

- Example on influence:
  - If 4 infantry soldiers in a fire team are camped out in a field, then the field is certainly under their influence, but probably not very strongly
  - If there is a helicopter gunship hovering over the same corner, then the field is considerably more under their control
- Influence is taken to drop off with distance and we can model it mathematically such as

  $I_d = \frac{I_0}{\sqrt{1+d}}$ or $I_d = \frac{I_0}{(1+d)^2}$ where I_d is the influence at distance d
- Each unit in the game needs to be assigned a single military influence value usually set by game designers with some tuning to get the balance right
- It is often useful to be able to visualize the influence map, as a graphical overlay into the game, to make sure that areas clearly under a unit's influence are being picked up by the tactical analysis

Calculating the Influence Map

- Given the drop-off formula for the influence at a distance and the intrinsic power of each unit, we can work out the influence of each side on each location in the game: who has control there and by how much
- The influence for a whole side is found by simply summing the influence of each unit belonging to that side
- The side with the greatest influence on a location can be considered to have control over it, and the degree of control is the difference between its winning influence value and the influence of the second placed side
- If this difference is very large, then the location is said to be secure
- The final result is an influence map: a set of values showing both the controlling side and the degree of influence (and optionally the degree of security) for each location in the game

Simple Example

- Unit A has the influence model
ed by $I_d = \frac{I_0}{(1+d)^2}$ with I_0=50

  If the location of Unit A is as shown on the left map, then the influence map can be calculated as shown on the right

	A		

13	50	13	6
9	13	9	5

- Unit B has the influence modeled by $I_d = \frac{I_0}{\sqrt{1+d}}$ with I_0=20

  If the location of Unit B is as shown on the left map, then the influence map can be calculated as shown on the right

B			

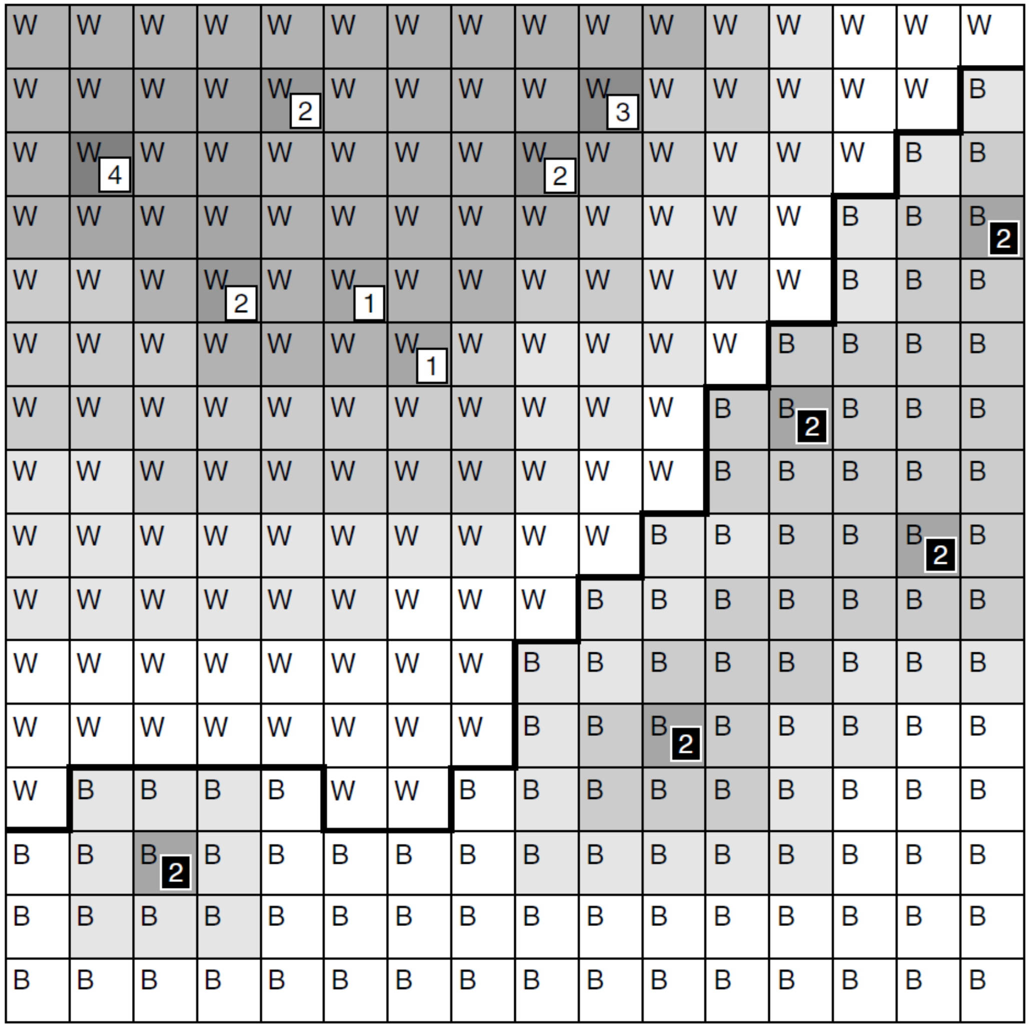
14	13	11	10
20	14	12	10

- If Unit A and Unit B are located on the same map as shown on the left, then the influence map can be calculated as shown on the right

	A		
B			

27	63	24	16
29	27	21	15

Example Influence Map



W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
W	W	W	W	W 2	W	W	W	W	W 3	W	W	W	W	W	B
W	W 4	W	W	W	W	W	W	W 2	W	W	W	W	W	B	B
W	W	W	W	W	W	W	W	W	W	W	W	W	B	B	B 2
W	W	W	W 2	W	W 1	W	W	W	W	W	W	W	B	B	B
W	W	W	W	W	W	W 1	W	W	W	W	W	B	B	B	B
W	W	W	W	W	W	W	W	W	W	W	B	B 2	B	B	B
W	W	W	W	W	W	W	W	W	W	W	B	B	B	B	B
W	W	W	W	W	W	W	W	W	W	B	B	B	B	B 2	B
W	W	W	W	W	W	W	W	W	B	B	B	B	B	B	B
W	W	W	W	W	W	W	W	B	B	B	B	B	B	B	B
W	W	W	W	W	W	W	W	B	B	B 2	B	B	B	B	B
W	B	B	B	B	W	W	B	B	B	B	B	B	B	B	B
B	B	B 2	B	B	B	B	B	B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B

Calculating the Influence Map (cont.)

- To calculate the map we need to consider each unit in the game for each location in the level
- With a thousand units and a million locations (well within the range of current RTS games), a billion calculations would be needed
- Execution time is $O(nm)$, and memory is $O(m)$, where m is the number of locations in the level, and n is the number of units
- There are three approaches we can use to improve matters:
  1. Limited radius of effect
  2. Map flooding
  3. Convolution filters

Limited Radius of Effect

- Along with a basic influence, each unit has a maximum radius. Beyond this radius the unit cannot exert influence, no matter how weak
- The maximum radius might be manually set for each unit, or we could use a threshold
- This approach allows us to pass through each unit in the game, adding its contribution to only those locations within its radius
  - We end up with $O(nr)$ in time and $O(m)$ in memory, where r is the number of locations within the average radius of a unit
  - Because r is going to be much smaller than m (the number of locations in the level), this is a significant reduction in execution time
- The disadvantage of this approach is that small influences do not add up over large distances

Limited Radius of Effect Simple Example

- Unit A has the influence modeled by $I_d = \frac{I_0}{(1+d)^2}$ with I_0=50, and $d \le 1$

  If the location of Unit A is as shown on the left map, then the influence map can be calculated as shown on the right

	A		

13	50	13	
	13		

- Unit B has the influence modeled by $I_d = \frac{I_0}{\sqrt{1+d}}$ with I_0=20 , and $d \le 2$

  If the location of Unit B is as shown on the left map, then the influence map can be calculated as shown on the right

B			

14	13		
20	14	12	

Still has no influence on this location with this approach!

- If Unit A and Unit B are located on the same map as shown on the left, then the influence map can be calculated as shown on the right

	A		
B			

27	63	13	
20	27	12	

	A×10		
B×10			

270	630	130	
200	270	120	

Map Flooding

- Map flooding uses a more dramatic simplifying assumption: the influence of each location is equal to the largest influence contributed by any unit
- It is easy to implement and is fast in operation
- This approach may lead to some errors, as the AI assumes that a huge number of weak troops can be overpowered by a single strong unit (a very dangerous assumption)
- In this assumption if a tank is covering a street, then the influence on that street is the same even if 20 solders arrive to also cover the street

Map flooding Simple Example

- Unit A has the influence modeled by $I_d = \frac{I_0}{(1+d)^2}$ with I_0=50, and $d \leq 1$

  If the location of Unit A is as shown on the left map, then the influence map can be calculated as shown on the right

	A		

13	50	13	
	13		

- Unit B has the influence modeled by $I_d = \frac{I_0}{\sqrt{1+d}}$ with I_0=20 , and $d \leq 2$

  If the location of Unit B is as shown on the left map, then the influence map can be calculated as shown on the right

B			

14	13		
20	14	12	

No change in the influence map even when more units are added!

- If Unit A and Unit B are located on the same map as shown on the left, then the influence map can be calculated as shown on the right

	A		
B			

14	50	13	
20	14	12	

	A×10		
B×10			

14	50	13	
20	14	12	

Convolution Filters – Overview

- We start with the influence map where the only values marked are those where the units are actually located
- Then the algorithm works through each location and changes its value so it incorporates not only its own value but also the values of its neighbors
- This has the effect of blurring out the initial spots so that they form gradients reaching out. Higher initial values get blurred out further
- This approach uses a filter: a rule that says how a location's value is affected by its neighbors
- Depending on the filter, we can get different kinds of blurring
- The most common filter is called a Gaussian, and it is useful because it has mathematical properties that make it even easier to calculate
- To perform filtering, each location in the map needs to be updated using this rule
- To make sure the influence spreads to the limits of the map, we need to then repeat the whole update several times again

Convolution Filters – Filter Matrix

- All convolution filters have the same basic structure: we define an update matrix to tell us how the value of one location in the map gets updated based on its own value and that of its neighbors
- For a square tile-based level, we might have a matrix that looks like the following

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- The convolution filter matrix normally has an odd number of rows and columns

Convolution Filters – Location Matrix

- The locations can also be represented as another matrix

$$L = \begin{bmatrix} 5 & 6 & 2 \\ 1 & 4 & 2 \\ 6 & 3 & 3 \end{bmatrix}$$

- The above example assumes that the map can be represented by 3×3 locations and in general its dimension can be $W \times H$ where W is the number of columns and H is the number of rows

Convolution Filters – Filtering

- For each location in the location matrix, the center of the convolution filter matrix is aligned with this location
- A new value from this location is calculated by multiplying each value in the map by the corresponding value in the matrix and summing the results
- For example,

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 5 & 6 & 2 \\ 1 & 4 & 2 \\ 6 & 3 & 3 \end{bmatrix}$$

We need to derive the new location matrix

$$L' = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

The value e in the new location matrix can be calculated by

$$e = \frac{1}{16}(1\times 5 + 2\times 6 + 1\times 2 + 2\times 1 + 4\times 4 + 2\times 2 + 1\times 6 + 2\times 3 + 1\times 3)$$
$$= \frac{56}{16} = \frac{7}{2} = 3.5$$

Convolution Filters – Boundaries

- We need to consider what happens at the edges of the map
- Here we are no longer able to apply the matrix because some of the neighbors for the edge locations do not exist

$$M = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad L = \begin{bmatrix}5 & 6 & 2\\ 1 & 4 & 2\\ 6 & 3 & 3\end{bmatrix} \qquad L' = \begin{bmatrix}a & b & c\\ d & e & f\\ g & h & i\end{bmatrix}$$

- There are two approaches to this problem
    1. modify the matrix
    2. modify the map

Modifying the Matrix (1)

$$M = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad L = \begin{bmatrix}5 & 6 & 2\\ 1 & 4 & 2\\ 6 & 3 & 3\end{bmatrix} \qquad L' = \begin{bmatrix}a & b & c\\ d & e & f\\ g & h & i\end{bmatrix}$$

$$M_a = \frac{1}{9}\begin{bmatrix}0 & 0 & 0\\ 0 & 4 & 2\\ 0 & 2 & 1\end{bmatrix} \qquad a = \frac{1}{9}(4 \times 5 + 2 \times 6 + 2 \times 1 + 1 \times 4) = \frac{38}{9} = 4.22$$

$$M_c = \frac{1}{9}\begin{bmatrix}0 & 0 & 0\\ 2 & 4 & 0\\ 1 & 2 & 0\end{bmatrix} \qquad c = \frac{1}{9}(2 \times 6 + 4 \times 2 + 1 \times 4 + 2 \times 2) = \frac{28}{9} = 3.11$$

$$M_g = \frac{1}{9}\begin{bmatrix}0 & 2 & 1\\ 0 & 4 & 2\\ 0 & 0 & 0\end{bmatrix} \qquad g = \frac{1}{9}(2 \times 1 + 1 \times 4 + 4 \times 6 + 2 \times 3) = \frac{36}{9} = 4$$

$$M_i = \frac{1}{9}\begin{bmatrix}1 & 2 & 0\\ 2 & 4 & 0\\ 0 & 0 & 0\end{bmatrix} \qquad i = \frac{1}{9}(1 \times 4 + 2 \times 2 + 2 \times 3 + 4 \times 3) = \frac{26}{9} = 2.89$$

Modifying the Matrix (2)

$$M = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad L = \begin{bmatrix}5 & 6 & 2\\ 1 & 4 & 2\\ 6 & 3 & 3\end{bmatrix} \qquad L' = \begin{bmatrix}a & b & c\\ d & e & f\\ g & h & i\end{bmatrix}$$

$$M_b = \frac{1}{12}\begin{bmatrix}0 & 0 & 0\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad b = \frac{1}{12}(2\times 5 + 4\times 6 + 2\times 2 + 1\times 1 + 2\times 4 + 1\times 2) = \frac{49}{12} = 4.08$$

$$M_d = \frac{1}{12}\begin{bmatrix}0 & 2 & 1\\ 0 & 4 & 2\\ 0 & 2 & 1\end{bmatrix} \qquad d = \frac{1}{12}(2\times 5 + 1\times 6 + 4\times 1 + 2\times 4 + 2\times 6 + 1\times 3) = \frac{43}{12} = 3.58$$

$$M_f = \frac{1}{12}\begin{bmatrix}1 & 2 & 0\\ 2 & 4 & 0\\ 1 & 2 & 0\end{bmatrix} \qquad f = \frac{1}{12}(1\times 6 + 2\times 2 + 2\times 4 + 4\times 2 + 1\times 3 + 2\times 3) = \frac{35}{12} = 2.92$$

$$M_h = \frac{1}{12}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 0 & 0 & 0\end{bmatrix} \qquad h = \frac{1}{12}(1\times 1 + 2\times 4 + 1\times 2 + 2\times 6 + 4\times 3 + 2\times 3) = \frac{41}{12} = 3.42$$

Modifying the Matrix (3)

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 5 & 6 & 2 \\ 1 & 4 & 2 \\ 6 & 3 & 3 \end{bmatrix}$$

- After applying the convolution filter

$$L' = \begin{bmatrix} 4.22 & 4.08 & 3.11 \\ 3.58 & 3.5 & 2.92 \\ 4 & 3.42 & 2.89 \end{bmatrix}$$

- It involves working with 9 different filter matrices and switching between them at the correct time
- If we need to keep switching matrices, we cannot optimize the algorithm by taking advantage of single instruction, multiple data (SIMD) with the processing of several locations at the same time, so we lose a good deal of the speed

Modifying the Map (1)

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 5 & 6 & 2 \\ 1 & 4 & 2 \\ 6 & 3 & 3 \end{bmatrix} \qquad L' = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- Expand the original map by adding a border around the game locations and clamping their values (i.e., they are never processed during the convolution algorithm; therefore, they will never change their value)
- We can choose some arbitrary value (say zero) for the added border values
- The number of added border rows/columns depends on the size of the convolution filter matrix

$$L_E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 6 & 2 & 0 \\ 0 & 1 & 4 & 2 & 0 \\ 0 & 6 & 3 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad {L_E}' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & d & e & f & 0 \\ 0 & g & h & i & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Modifying the Map (2)

$$M = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad L_E = \begin{bmatrix}0 & 0 & 0 & 0 & 0\\ 0 & 5 & 6 & 2 & 0\\ 0 & 1 & 4 & 2 & 0\\ 0 & 6 & 3 & 3 & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix} \qquad L_E{}' = \begin{bmatrix}0 & 0 & 0 & 0 & 0\\ 0 & a & b & c & 0\\ 0 & d & e & f & 0\\ 0 & g & h & i & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}$$

$$a = \frac{1}{16}(4 \times 5 + 2 \times 6 + 2 \times 1 + 1 \times 4) = \frac{38}{16} = 2.38$$

$$b = \frac{1}{16}(2 \times 5 + 4 \times 6 + 2 \times 2 + 1 \times 1 + 2 \times 4 + 1 \times 2) = \frac{49}{16} = 3.06$$

$$c = \frac{1}{16}(2 \times 6 + 4 \times 2 + 1 \times 4 + 2 \times 2) = \frac{28}{16} = 1.75$$

$$d = \frac{1}{16}(2 \times 5 + 1 \times 6 + 4 \times 1 + 2 \times 4 + 2 \times 6 + 1 \times 3) = \frac{43}{16} = 2.69$$

$$e = \frac{1}{16}(1 \times 5 + 2 \times 6 + 1 \times 2 + 2 \times 1 + 4 \times 4 + 2 \times 2 + 1 \times 6 + 2 \times 3 + 1 \times 3)$$
$$= \frac{56}{16} = \frac{7}{2} = 3.5$$

Modifying the Map (3)

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad L_E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 6 & 2 & 0 \\ 0 & 1 & 4 & 2 & 0 \\ 0 & 6 & 3 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad L_E{}' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 \\ 0 & d & e & f & 0 \\ 0 & g & h & i & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$f = \frac{1}{16}(1 \times 6 + 2 \times 2 + 2 \times 4 + 4 \times 2 + 1 \times 3 + 2 \times 3) = \frac{35}{16} = 2.19$$

$$g = \frac{1}{16}(2 \times 1 + 1 \times 4 + 4 \times 6 + 2 \times 3) = \frac{36}{16} = 2.25$$

$$h = \frac{1}{16}(1 \times 1 + 2 \times 4 + 1 \times 2 + 2 \times 6 + 4 \times 3 + 2 \times 3) = \frac{41}{16} = 2.56$$

$$i = \frac{1}{16}(1 \times 4 + 2 \times 2 + 2 \times 3 + 4 \times 3) = \frac{26}{16} = 1.63$$

Modifying the Map (4)

$$M = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 5 & 6 & 2 \\ 1 & 4 & 2 \\ 6 & 3 & 3 \end{bmatrix}$$

- After applying the convolution filter

$$L_E{}' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2.38 & 3.06 & 1.75 & 0 \\ 0 & 2.69 & 3.15 & 2.19 & 0 \\ 0 & 2.25 & 2.56 & 1.63 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad L' = \begin{bmatrix} 2.38 & 3.06 & 1.75 \\ 2.69 & 3.5 & 2.19 \\ 2.25 & 2.56 & 1.63 \end{bmatrix}$$

- If a high-influence character is next to the border, its influence will be pulled down because the edge locations will be receiving zero-valued contributions from the invisible border

Convolution Filters – Comparison

- Convolution result by modifying the filter matrix:

$$L' = \begin{bmatrix} \frac{38}{9} & \frac{49}{12} & \frac{28}{9} \\ \frac{43}{12} & \frac{56}{16} & \frac{35}{12} \\ \frac{36}{9} & \frac{41}{12} & \frac{26}{9} \end{bmatrix} = \begin{bmatrix} 4.22 & 4.08 & 3.11 \\ 3.58 & 3.5 & 2.92 \\ 4 & 3.42 & 2.89 \end{bmatrix}$$

- Convolution result by modifying the map:

$$L' = \begin{bmatrix} \frac{38}{16} & \frac{49}{16} & \frac{28}{16} \\ \frac{43}{16} & \frac{56}{16} & \frac{35}{16} \\ \frac{36}{16} & \frac{41}{16} & \frac{26}{16} \end{bmatrix} = \begin{bmatrix} 2.38 & 3.06 & 1.75 \\ 2.69 & 3.5 & 2.19 \\ 2.25 & 2.56 & 1.63 \end{bmatrix}$$

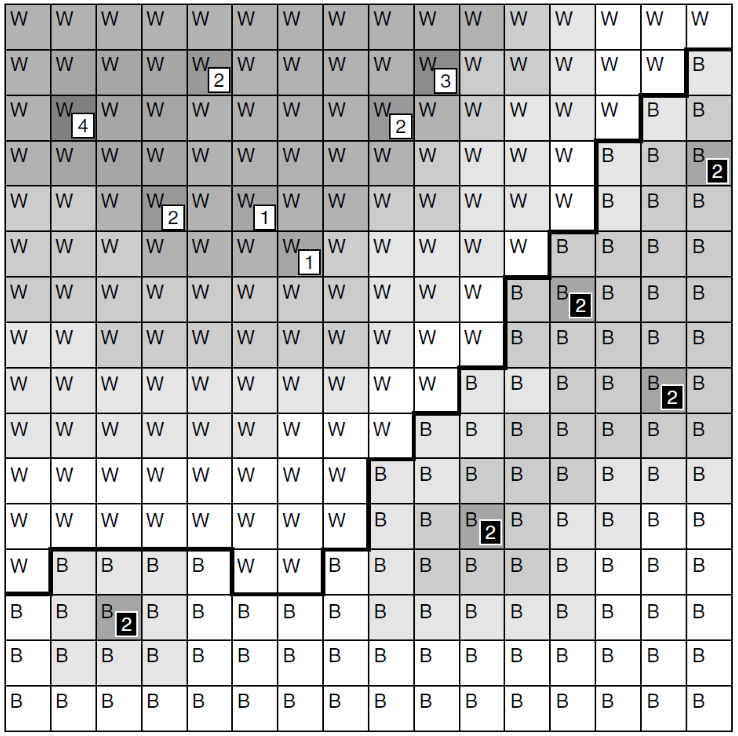
  – If you visualize the influence map as color density, it appears to have a paler color around the edge

Convolution Filters – Performance

- Normally there are significantly fewer units in the game than there are locations in the map, so this approach would be more expensive than even the initial naive algorithm
- Because it is a graphics algorithm, however, it is easy to implement using graphical techniques

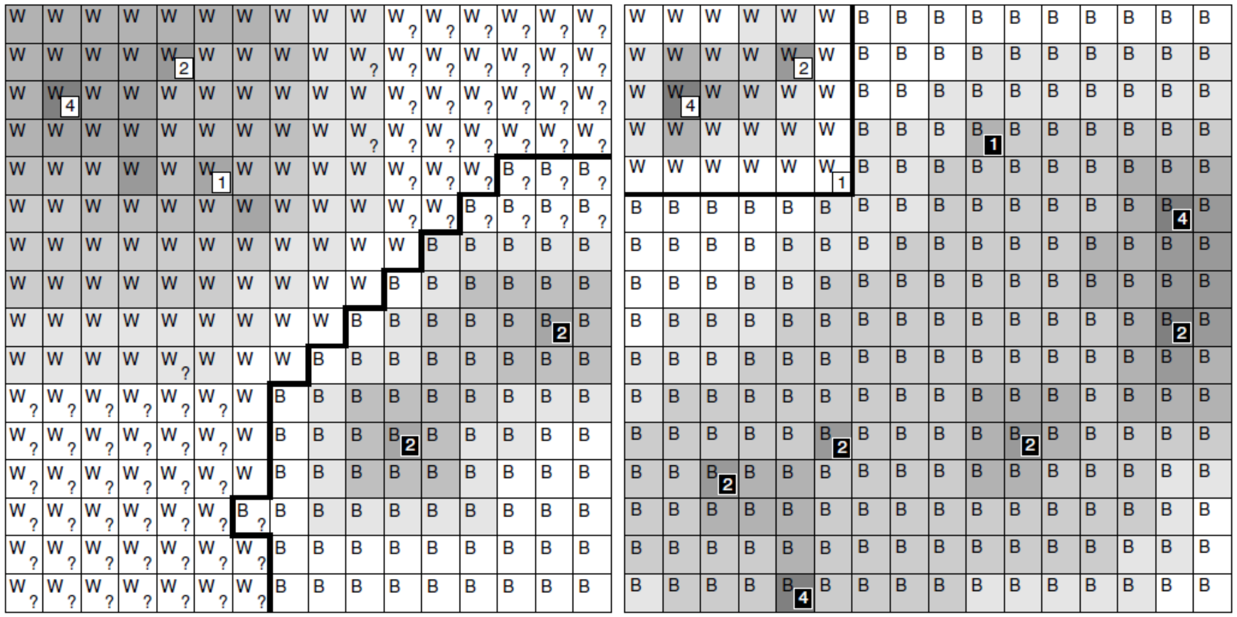
Application

- An influence map allows the AI to see which areas of the game are safe (those that are very secure), which areas to avoid, and where the border between the teams is weakest (i.e., where there is little difference between the influence of the two sides)


W W W W W W W W W W W W W W W W
W W W W W 2 W W W W W 3 W W W W W B
W W 4 W W W W W W W 2 W W W W W B B
W W W W W W W W W W W W W B B B 2
W W W W 2 W W 1 W W W W W W W B B B
W W W W W W W 1 W W W W W B B B B
W W W W W W W W W W W B B 2 B B B
W W W W W W W W W W W B B B B B
W W W W W W W W W W B B B B B 2 B
W W W W W W W W W B B B B B B B
W W W W W W W W B B B B B B B B
W W W W W W W W B B B 2 B B B B B
W B B B B W W B B B B B B B B B
B B B 2 B B B B B B B B B B B B B
B B B B B B B B B B B B B B B B
B B B B B B B B B B B B B B B B

Dealing with Unknowns

- If we do a tactical analysis on the units we can see, then we run the risk of underestimating the enemy forces
- Typically, games do not allow players to see all of the units in the game
- Some games solve this problem by allowing all of the AI players to know everything
  - This allows the AI to build only one influence map, which is accurate and correct for all sides
  - The AI will not underestimate the opponent's military might
  - This is widely viewed as cheating, however, because the AI has access to information that a human player would not have



Tactical Pathfinding

- Tactical pathfinding can provide quite impressive results when characters in the game move, taking account of their tactical surroundings, staying in cover, and avoiding enemy lines of fire and common ambush points

Cost Function (1)

- The cost for moving along a connection in the graph should be based on both distance/time (otherwise, we might embark on exceptionally long routes) and how tactically sensible the maneuver is
- The cost of a connection is given by a formula of the following type

$$C = D + \sum_i w_i T_i$$

where D is the distance of the connection (or time or other non-tactical cost function); w_i is a weighting factor for each tactic i supported in the game; T_i is the tactical quality for the connection for each tactic i

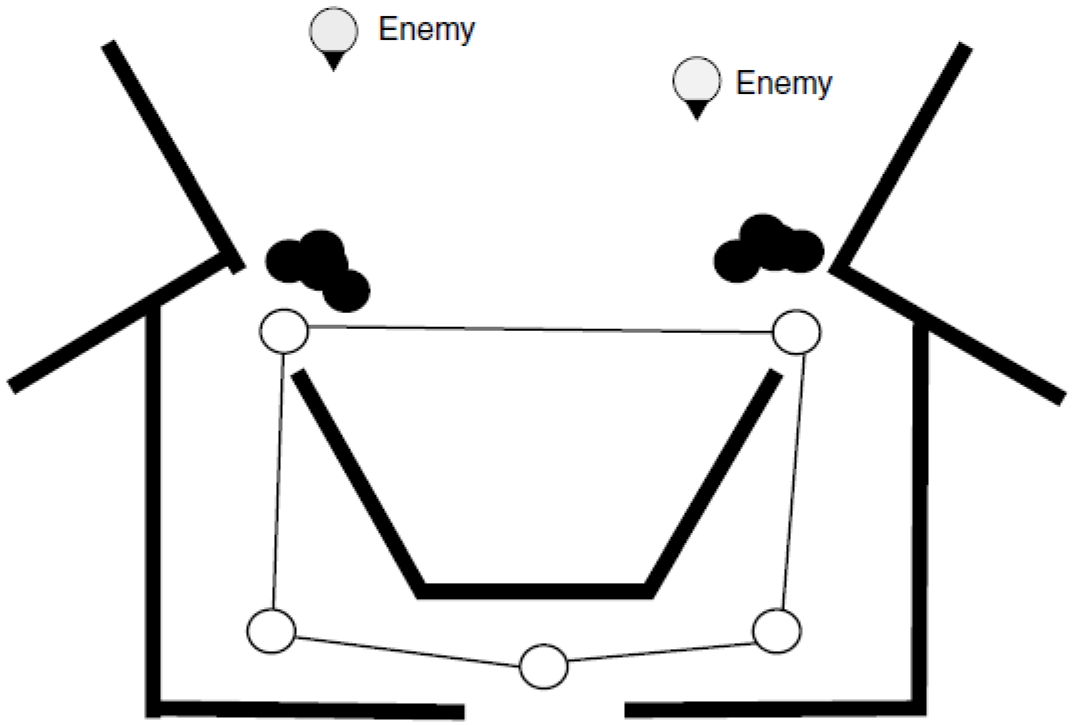
Cost Function (2)

$$C = D + \sum_i w_i T_i$$

- Tactical information is normally stored on a per-location basis
- To convert location-based information into connection-based costs, we normally average the tactical quality of each of the locations that the connection connects
  - This works on the assumption that the character will spend half of its time in each region and so should benefit or suffer half of the tactical properties of each

Cost Function (3)

- The previous assumption is good enough for most games, although it sometimes produces quite poor results



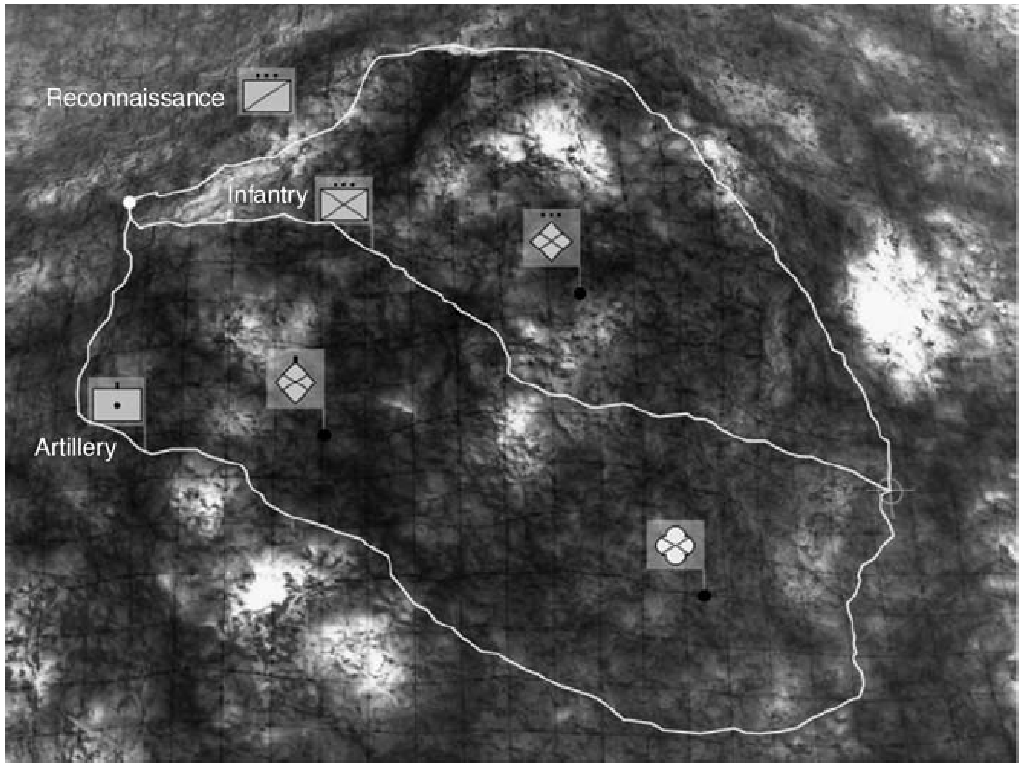
Tactical Weights

$$C = D + \sum_i w_i T_i$$

- If a tactic has a high weight, then locations with that tactical property will be avoided by the character
  - This might be the case for ambush locations or difficult terrain
- Conversely, if the weight is a large negative value, then the character will favor locations with a high value for that property
  - This would be sensible for cover locations or areas under friendly control
- Care needs to be taken to make sure that no possible connection in the graph can have a negative overall weight
  - We may specifically limit the cost value returned so that it is always positive but it adds additional processing time and can also lose lots of tactical information

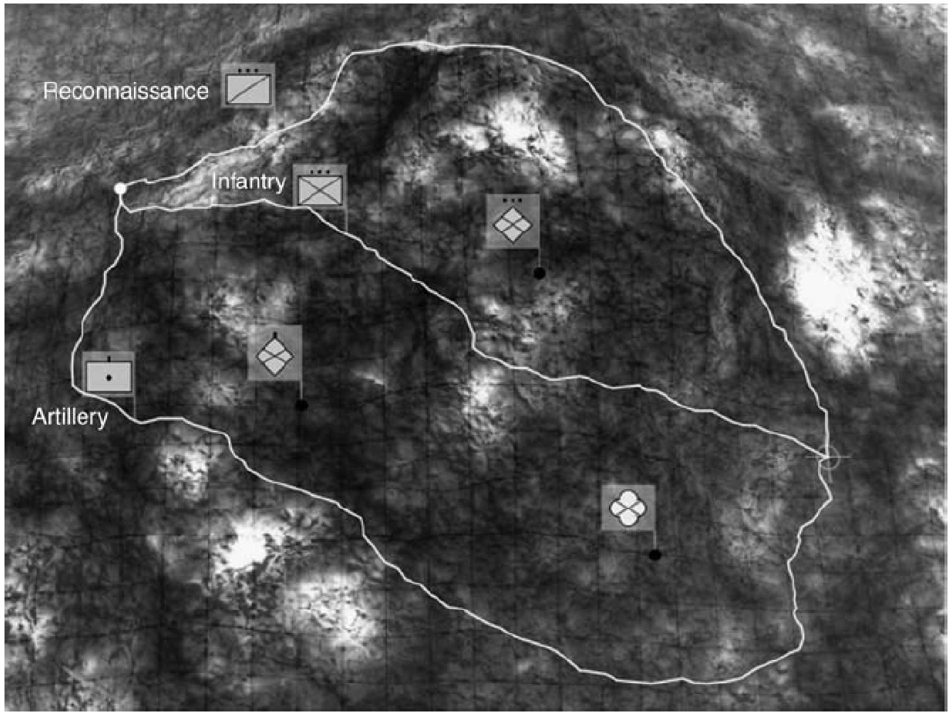
Example – Tactical Pathfinding (1)

- Reconnaissance units
  - can move fairly efficiently over any kind of terrain, so they weight the difficulty of terrain with a small positive weight
  - keen to avoid enemy units, so they weight the proximity of enemy units with a large positive value
  - need to find locations with large visibility, so they weight this with a large negative value



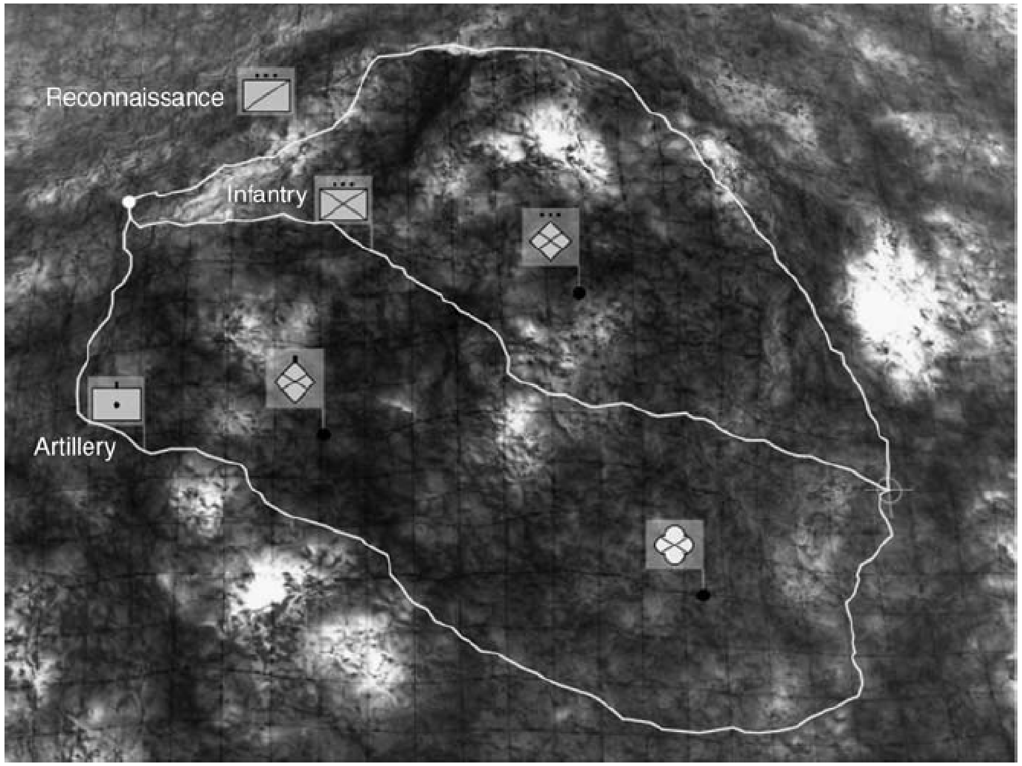
Example – Tactical Pathfinding (2)

- Light infantry units
  - have slightly more difficultly with tough terrain, so their weight is a small positive value, higher than that of the reconnaissance units
  - their purpose is to engage the enemy. However, they would rather avoid unnecessary engagements, so they use a small positive weight for enemy proximity (if they were actively seeking combat, they'd use a negative value here)
  - would rather move without being seen, so they use a small positive weight for visibility



Example – Tactical Pathfinding (3)

- Heavy artillery units
  - cannot cope with tough terrain, so they use a large positive weight for difficult areas of the map
  - are not good in close encounters, so they have large positive weights for enemy proximity
  - when exposed, they are a prime target and should move without being seen (they can attack from behind a hill quite successfully), so they also use a large positive weight for visibility



Reference

- Artificial Intelligence for Games
  - Chapter 6.1, 6.2, 6.3