GAME AI

with Jeff Wilson, PhD

Perception and Fairness

Fairness

- Game players are easily frustrated with games that they don't perceive as being fair
- Mario Kart with the handicapping via powerups is a classic example
- Many games have enemies that seem to be aware of more than they should, especially in 3D games.
- Two primary aspects to fairness:
  - Game Design/Rules
  - Agent Ability and Perception
  - Perception is arguably the most difficult to adjust

Georgia Tech

Perception Modeling

- In robotics, perception is a fundamental aspect of AI solutions.
- The robot needs to be aware of its environment
- But in video games, the agent has access to all the information it might need in the simulation, often in an already discretized form.

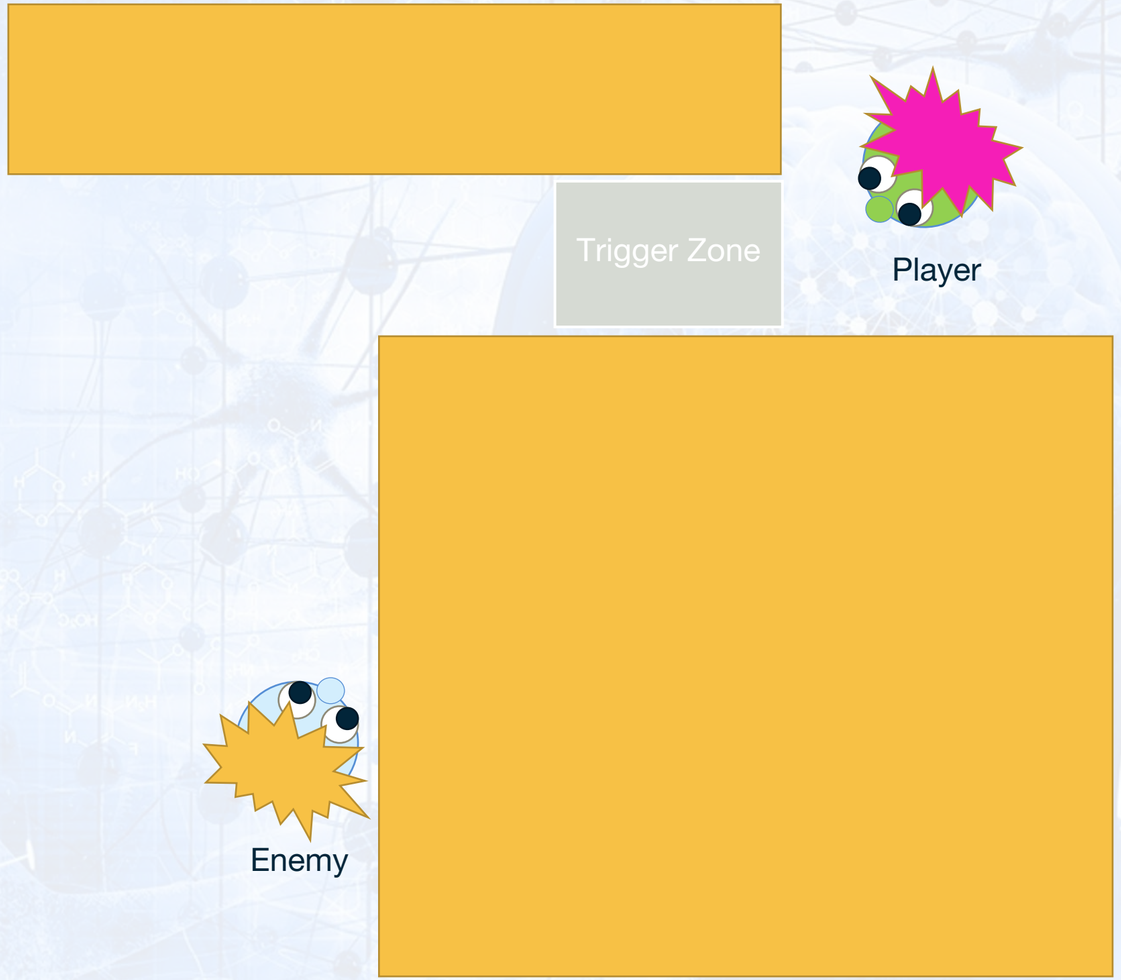
Perception Modeling

- But having access to all game state means that agents can be more informed than the player might expect
- To model agent perception accurately means holding information back from the agent's decision making
- This "holding back" can become computationally expensive and prohibitive in video game development

Georgia Tech

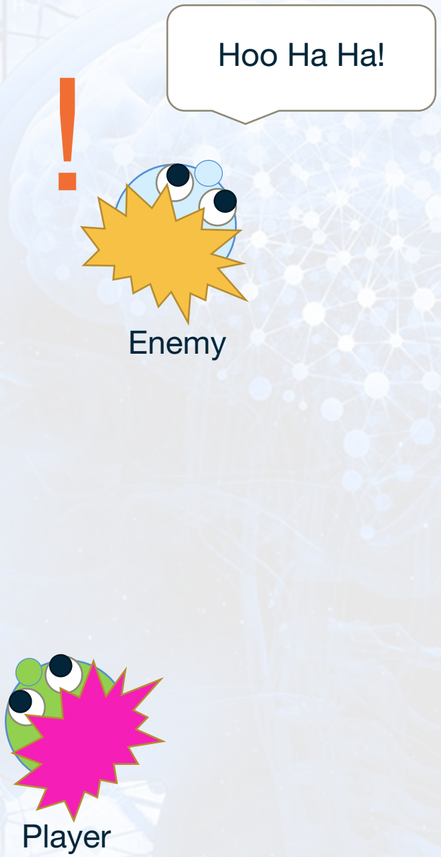
Perception

- Simple agents might always acknowledge the player and immediately act
- To avoid overwhelming the player, enemy agents get activated or spawned as the player progresses in the game



Perception

- But players are frustrated if enemy agents can detect them if it is expected to be impossible
- Examples:
  - Player is very far away
  - Agent has eyes in the front of their head and they are looking away from player



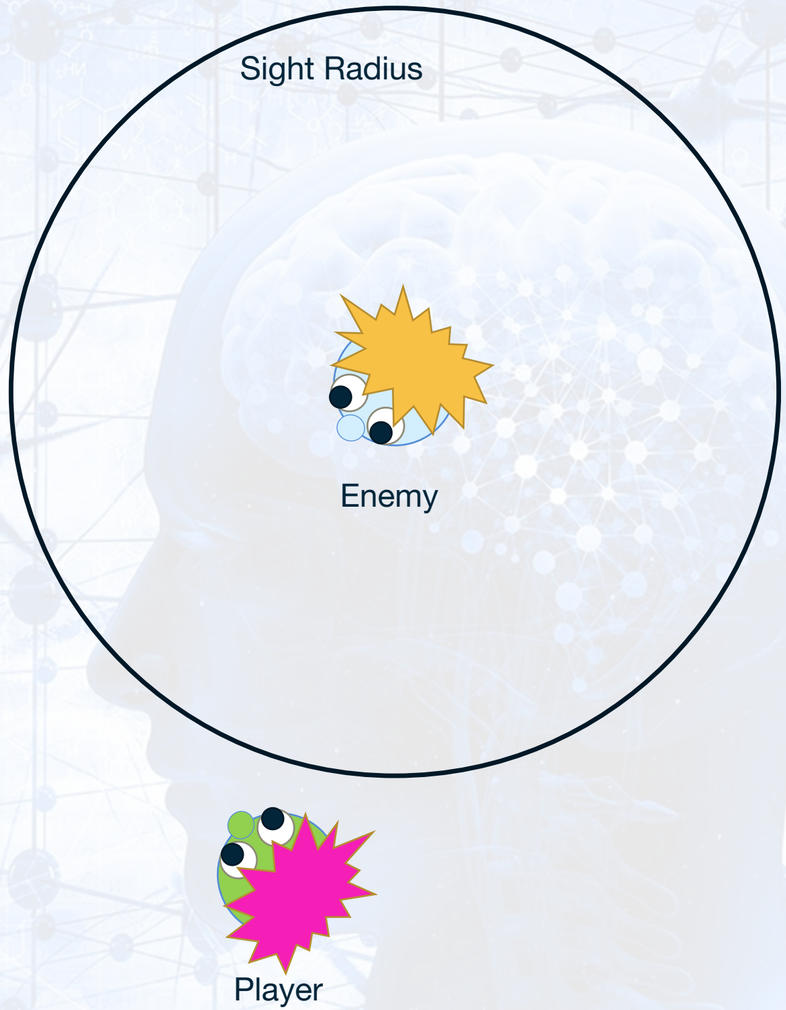
Perception Problem

- From a development perspective, it is easiest to just access any simulation state you desire.
- If the enemy agent needs to know where the player is then you can simply access it
- However, this does not reflect the limitations imposed on the agent’s (expected) perceptual ability

Georgia Tech

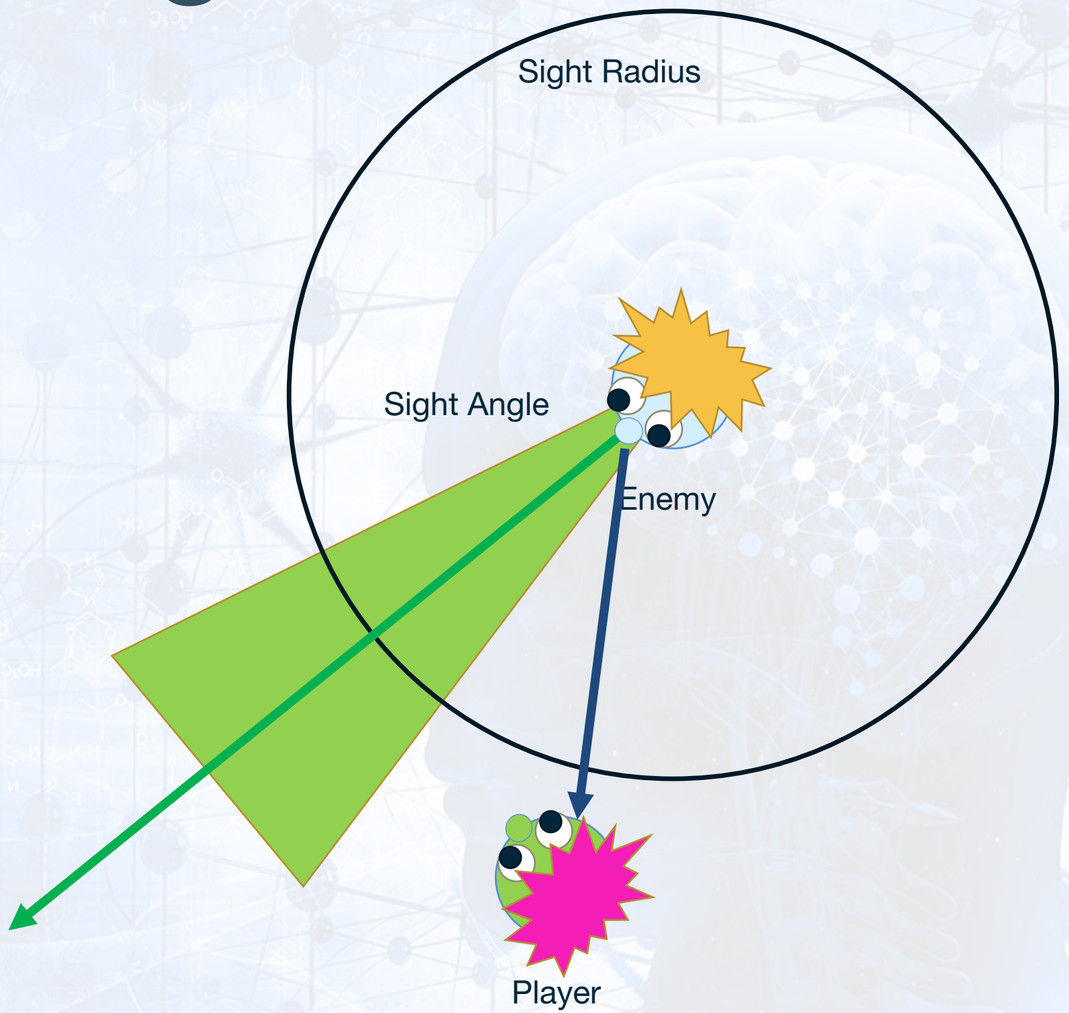
Limited Sight Range

- A simple to implement perceptual limitation is a sight radius.
- If the player's position is beyond the sight radius, then the enemy does not see the player and maintains a neutral behavior



Limited Sight Cone/Angle

- Further realism can be added via a sight angle
- Compare the absolute angle of agent facing angle (or head angle) and player position with a perception angle threshold



Occlusion

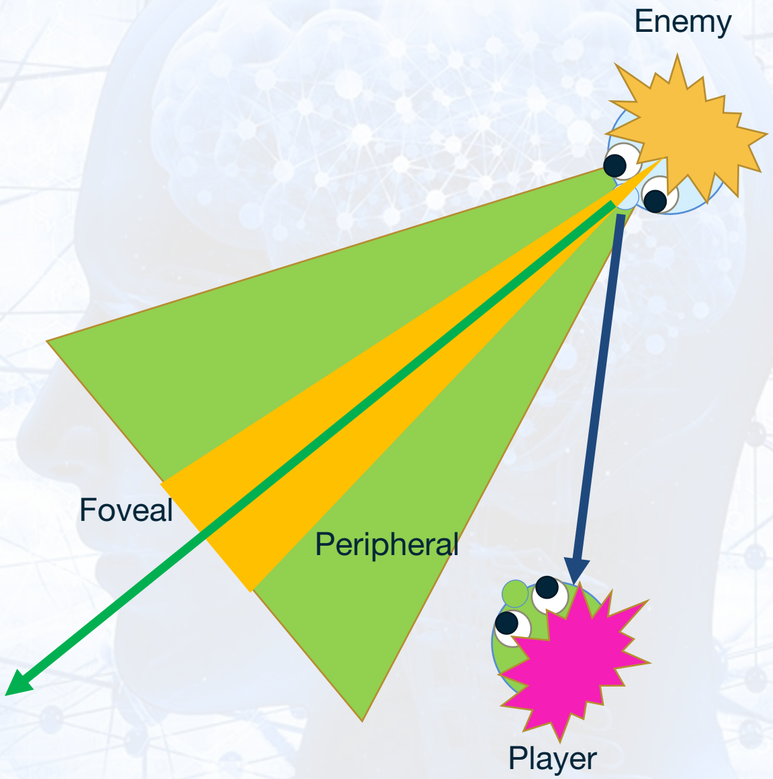
- Player may be hidden by wall or other object
- Raycast is a simple test
- Can add more raycasts more realism
- Render (projection and rasterize with poly fill) to see if color-keyed player is in view

Attention

- It may be desirable to model agent attention level
- Consider a stealth game where the enemy knows that the player is around but doesn't know where
- In this case, detection ability can be increased until the aggression level reduces
- Examples: Enemy can see the player from father away or from a wider angle, hiding spots don't work as well, etc.

Foveal and Peripheral Vision

- Foveal (center vision) and Peripheral Vision (side vision) can be modeled with two cones/angles
- This can be integrated with attention modeling
- Example:
  - The player is wearing camouflage and the enemy is not in an alert state. If the player is still, and is only even seen in the peripheral angle then they aren't detected



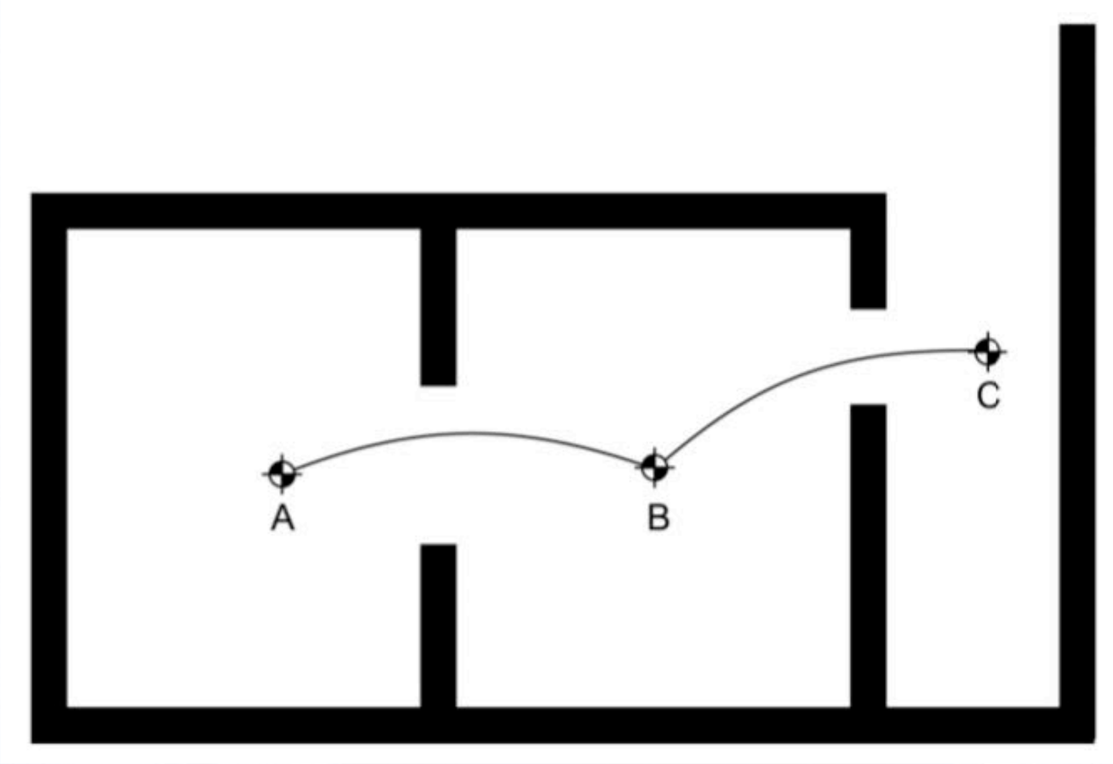
Shadows and Camouflage

- For stealth games, the player often needs a place to hide that may not involve full occlusion
- For shadows, the environment can be discretized (e.g., grid, voxel) and each cell can be tested as if the player is there to see if it is directly illuminated by any static point light sources. This is a preprocessing/baking step. Use the max illumination found to store.
- For camo, often zones are manually assigned by designers (e.g., bushes, grass)

Georgia Tech

Sense Graph

- Finite Element Model (FEM) of game world
- Represent game space as a graph
- Improves scalability of game because events that occur are attenuated down to zero; only agents within non-zero range need to process
- Senses are signals that are passed between graph nodes
- Smells can propagate and linger



Hearing

- Outside of speech communication, human sense of hearing is often used for the purpose of alerting an individual to some event
  - E.g., If you hear a twig snap in the woods you will quickly turn your head to see what it was
- Modeling hearing from actual game audio is overkill
- Instead listen to game events that lead to audio (integrate with sense graph)
- Calculate detectability based on precomputed loudness and sound type and distance from the user
  - Use 3D sound rolloff factors (near and far distances)
  - May also perform Fletcher Munson Curve analysis (pre-process)

Georgia Tech

Projectiles and Fairness

- Aiming Error
- Aiming Action
- Reaction Time

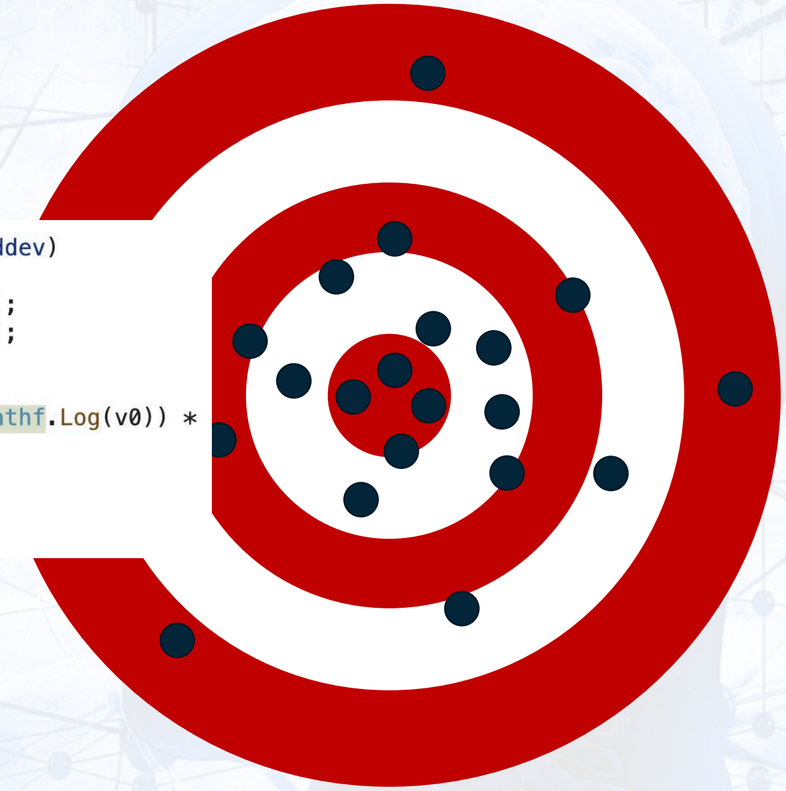
Human-Like Error in Aim

- Gaussian Distribution can be used to introduce error around exact solution
- Use Box-Muller or Zigorat Algorithm
- Can also rand minus rand for a simple random distribution with central tendency
- Cheat detection of aim-bots?

```
float RandGaussian(float mean, float stddev)
{
    float v0 = 1f - Random.Range(0f, 1f);
    float v1 = 1f - Random.Range(0f, 1f);

    // in range [0, 1]
    float randNorm = Mathf.Sqrt(-2f * Mathf.Log(v0)) *
        Mathf.Sin(2f * Mathf.PI * v1);

    return mean + stddev * randNorm;
}
```



```
return mean + stddev * (Random.Range(0f, 1f) - Random.Range(0f, 1f));
```

Georgia Tech

Human-Like Aim Movement

- Fitts's Law
- Time required to rapidly move to a target area is a function of the ratio between the distance to the target and the width of the target
- Can consider pointing a weapon at a target
- $T = Log_2(\frac{2D}{W})$ – Note that there are various formulations
- Perhaps simulate discrete refining aim movements with error as part of a feedback loop; fit to Fitts's predictions
- Also, see GOMS

Georgia Tech

Human-Like Reaction Time

- Around 0.2 s response time to visual stimulus (if expected)
  - A little less for auditory stimulus
- Circular buffer to delay estimates
- Extrapolate according to velocity for purpose of aiming
- Possibly use Kalman Filter
- Probably overkill, but might help avoid aim-bot detection