CS380/CS580 Final, version 20240725 NAME_____

1	2	3	4	5	Total	Fraction	Grade

(Disclaimer: Some questions might be discarded / not graded, point values might be changed)

Architecture (50pts)

FSM model state better than BTs, while BT better represent plans. Placing BT into FSM creates a hierarchical model FSM becomes the high level reasoner that maintains the state and BST becomes the low level reasoner to execute flexible plans.

(3pt) Why is it preferred to start with an FSM and put BTs inside the states?

intercomponent communication to cache expensive operations such as LoS tests and share data between components

(3pt) What is the key purpose of a blackboard inside a BT? Knowledge management

(2pt) Smart objects are a programming architecture embodiment of what concept popularized by Don Norman? Intelligent Everything Affordance

very hard to fine-tune

(3pt) Why is Planning not used in most games? Planning is very expensive as it has to search for possible actions to achieve the goal of the AI. Has to reprogram every time if world or goal changes.

change options

(3pt) How is Utility better than a BT Selector node in terms of action selection? Selectors follow a predetermined plan of actions. However, the set of actions may have different priorities, which forces duplication of BT branches to reflect this. Utility instead chooses the action by scoring them based on priority and taking the highest score, simplifying the decision.

change based on context

(3pt) What aspect of Utility makes it possible to customize personality? By adding and modifying weights., Heuristic reasoners

(3pt) In Utility, what method can be used to avoid always selecting the highest scoring? Weighted Random

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 50% hit rate against a Slime with 50 hit points: 2.5

$5 \times 0.50 = 2.5$

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 80% hit rate against a Slime with 1 hit point: 4

$5 \times 0.80 = 4$

(24pt) Draw the Utility curves (all except for Patrol) for an NPC with the following requirements:

- NPC that would like to Reload below 50% ammo. Always prefer Reload over Shoot at less than 25% ammo.
- NPC that prefers Reload over Shoot if ammo 25% to 50% and enemy is more than 5m away.
- NPC wants to Shoot enemies up to 9m away.
- NPC will Punch enemies up to 1m away (always preferable to Shoot or Reload).
- If NPC has less than 10 health, NPC must always Run Away.
- Patrol if nothing else is desired.

Movement (18pts)

(3pt) Why is pathfinding not needed for flying NPCs? There are no u-shaped obstacles in the sky.

(3pt) List the 3 flocking rules (one word each): separation, alignment, cohesion

(3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). tether, wandering, obstacle avoidance

(9pt) List the other 9 steering behaviors from Craig Reynolds:

seek and flee, pursue and evade, arrival, constainment and wall following

path following, flow field following, group/crowd path following, leader following, queuing

A* Pathfinding Problem (100pts)

(100pt) Given the following search space, start node (A), and goal node (G), calculate on this sheet of paper a path found using A*. Please show all work using the Open and Closed lists, **marking the order deleted using numbers starting at 1** when deleting a node **off of either list**. Please mark parent pointers and all cost terms. Use the indicated actual cost along each node-to-node connection and use the distance values in the table for the heuristic. **(NOTE: One mistake loses half points, two or more loses all points – don't make even one mistake – double check all math – double check the final path is optimal). The correct solution uses all blank rows. Don't forget to put the final path in the bottom right box.**

Heuristic estimate to Goal node (G) from any given node (heuristic is admissible, but not consistent):

A	B	C	D	E	F	G	H	K
5.4	4.4	5.0	3.4	2.4	1.4	0.0	10.0	1.0

Open List:

	Order Deleted	Node	Parent	g(x)	h(x)	f(x)
1	1	A	null	~~0~~	5.4	5.4
2	2	B	A	1	4.4	5.4
3	4	C	A	1	5.~~0~~	6.~~0~~
4	3	D	B	2	3.4	5.4
5	5	E	D	6	2.4	8.4
6	6-R	K	C	13	1.~~0~~	14.~~0~~
7	7	F	E	7	1.4	8.4
8	8-R	K	E	12	1	13.~~0~~
9	10	H	F	8	10.~~0~~	18
10	9	K	F	11	1	12.~~0~~
11	13-R	G	K	19	~~0~~	19.~~0~~
12	12	K	H	10	1	11.~~0~~
13	14	G	K	18	~~0~~	18.~~0~~

Closed List:

	Order Deleted	Node	Parent	f(x)
1		A	null	5.4
2		B	A	5.4
3		D	B	5.4
4		C	A	6.~~0~~
5		E	D	8.4
6		F	E	8.4
7	11-R	K	E	12.~~0~~
8		H	F	14.~~0~~
9		K	H	11.~~0~~

Final waypoint path from start to goal:

A→B→D→E→F→H→K→G

Pathfinding (65pts)

(30pt) Carefully draw each search space representation (include ***ALL*** details required by a search algorithm):

(2pt) Given all pathfinding search space(s), which ones do not represent the walkable surface area? waypoint, corner graphs

(2pt) Given all pathfinding search space(s), which ones can be used with the Roy-Floyd-Warshall algorithm? grids, waypoints, corner graphs, circle-based waypoints, nav meshes grid

(3pt) If the heuristic cost in A* is absolutely accurate, what does it exactly represent? (<=6 words, be precise) best path without searching unnecessary nodes

(3pt) If a heuristic is admissible in A*, then it equal to or less than the true cost.

(3pt) In a NavMesh, why can't the floor polygons of a level be directly used? impassable obstacles may be overlaid over the floor polygons

(3pt) Why would a Waypoint Graph be preferred over a Grid in an FPS like Call of Duty? waypoints represent a small fraction of points on a grid, making it cheaper in memory for large maps

(3pt) What is the primary run-time CPU advantage of NavMesh over Waypoint Graph? better preprocessing and postprocessing for smoothing and animations pathfinding efficiency

(3pt) What is the optimal A* heuristic in a grid-based environment that contains teleporters? Ø (Dijkstra's)

(3pt) If non-optimal paths are acceptable, when is it bad to over-estimate the heuristic with A*? significant backtracking ~~when the terrain has random obstacles that would cause significant backtracking~~

(4pt) Name 2 primary problems with Goal Bounding 1. can only be used on static maps 2. requires extra memory at runtime for precomputed data (4 values per edge)

(3pt) A flow field could be generated using which search algorithm? Dijkstra (by floodfilling)

(3pt) In order to smooth a waypoint path on a NavMesh, you'll need this algorithm catmull-rom spline

Agent Awareness (20pts)

(6pt) Name 3 unique benefits of pathfinding sounds.

1. players can sense the volume, indicating source distance
2. players can sense pitch, indicating walls are present
3. players can sense location of the source

(10pt) Draw a complete vision model with percentage zones that a game like Splinter Cell would have had (make up reasonable %s):

(4pt) In the vision model, list 2 non-spatial aspects that can't be drawn in the previous question that would decrease the percentages.

1. player is in shadow
2. player is hiding behind cover

camouflage

Learning (50 points)

(5pt) List 5 problems with Neural Networks. 1. overfitting 2. hard to find or fix bugs

3. require tons of training data 4. hard to program 5. network isn't complicated to learn the problem

(3pt) Given the "Draw the Utility" exam question, what broad category of ML using labeled data could replace it? supervised learning
(4pt) Given the "Draw the Utility" exam question, how many features would it have? 3 How many outputs would it have? 5

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

(12pt) Use this sequence to determine the **probability** of the next move being a **Kick** for Bi-Gram, Tri-Gram, and Quad-Gram:
Punch, Kick, Punch, Kick, Punch, Uppercut, Punch, Punch, Kick, Punch, Punch, Kick, Punch, Punch, Kick, Punch

Bi-Gram: 5/10 5/9, Tri-Gram: 1/4 ✓, Quad-Gram: 1/3 1/4

(3pt) In player modeling, UsesSmokeGrenades is stored as what data type? float

(3pt) In player modeling, AttacksOnTwoFronts, UsesSmokeGrenades, and CanDoTrickyJumps are examples of traits.

(5pt) In player modeling, when you make your Excel sheet to evaluate what to implement, what categories/columns should you score?

1. useful to AI (stats) 2. easy to program 3. show to player (fun stats) 4. collection difficulty 5. show useful stats to player

(5pt) Using the moving average equation w/ alpha=0.2, what is the final average: (newest to oldest) 40, 80, 70, 60, 50, 0: Avg = 42.5888

(5pt) Using the moving average history window of length 5, what is the final average of the previous question's sequence: Avg = 53.136

50+60+70+60 = 260
260/5 = 52

Terrain Analysis (5 points)

(5pt) Combine the influence of three maps using the geometric mean (A: [0, 0.5, 0.2], B: [1, 1, 1], C: [0.5, 0.6, 0.5]) [0, 0.669, 0.464]

[~~0.7937~~, 0.6694, 0.~~62~~996]

Randomness (8 points)

(5pt) Write a **ONE LINE** equation to generate Gaussian randomness in the range [-3,3] with roughly a **NORMAL** distribution (which has a standard deviation of 1). Use the random function RandReal() that returns a uniform random number in the range [0,1]:

(RandReal() + RandReal() + RandReal()) * (1.0 / 0x7FFF'FFFF)

(RR()*2-1)+(RR()*2-1)+(RR()*2-1)

(2pt) Explain how to calculate a natural random offset for bullets aimed at the center of a target:

Use uniform randomness to generate a random angle; use gaussian randomness to generate the distance from the center

(1pt) What was my seemingly odd belief about randomness in games? Rand() is not random enough but actually true randomness is not random to humans.

PCG (11 points)

(5pt) Name the five PCG techniques described in class: 1. Constructionist 2. simulation-based

3. grammars 4. optimization-based 5. constrainst-based

(2pt) What is the new name given to modern transformer-based techniques to create content, like images and text? Generative AI

(4pt) What are the two primary problems stopping large game companies from using ChatGPT or Midjourney?

1. Legal risk in generation (copyright) 2. Quality (free from copyrighted, embarrassing, offensive material)

Draw Behavior Trees (60 points)

Directions: You can use any number of Selectors, Sequencers, or Parallel nodes. Parallel nodes Succeed if any children Succeed and Fail if any children Fail. You can use any number of Always Succeed, Always Fail, Rerun until Success, Rerun until Fail, or Invert decorator nodes. For all other nodes, you can only use the nodes given to you in the question.

(10pt) Draw a Behavior Tree that endlessly patrols between three points: A, B, and C. Use multiple "Move To _" Leaf nodes which will move toward the point you give it, will return InProgress if moving toward the point, and will return Success if it reaches the point (it will never Fail).

(50pt total) Draw a Behavior Tree that performs the previous endless patrolling behavior with "Move To _" nodes (10pt), but if it sees an enemy at any time during its movement, it immediately moves to point X (20pt) and stays there forever (20pt). Use the Leaf node "See Enemy?" that returns Success if it sees an enemy and returns Fail if it does not.

CS380/CS580 Final, version 20240725 NAME_____

1	2	3	4	5	Total	Fraction	Grade

(Disclaimer: Some questions might be discarded / not graded, point values might be changed)

Architecture (50pts)

(3pt) Why is it preferred to start with an FSM and put BTs inside the states? _____

(3pt) What is the key purpose of a blackboard inside a BT? _____

(2pt) Smart objects are a programming architecture embodiment of what concept popularized by Don Norman? _____

(3pt) Why is Planning not used in most games? _____

(3pt) How is Utility better than a BT Selector node in terms of action selection? _____

(3pt) What aspect of Utility makes it possible to customize personality? _____

(3pt) In Utility, what method can be used to avoid always selecting the highest scoring? _____

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 50% hit rate against a Slime with 50 hit points: ____

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 80% hit rate against a Slime with 1 hit point: _____

(24pt) Draw the Utility curves (all except for Patrol) for an NPC with the following requirements:

- NPC that would like to Reload below 50% ammo. Always prefer Reload over Shoot at less than 25% ammo.
- NPC that prefers Reload over Shoot if ammo 25% to 50% and enemy is more than 5m away.
- NPC wants to Shoot enemies up to 9m away.
- NPC will Punch enemies up to 1m away (always preferable to Shoot or Reload).
- If NPC has less than 10 health, NPC must always Run Away.
- Patrol if nothing else is desired.

Movement (18pts)

(3pt) Why is pathfinding not needed for flying NPCs? _____

(3pt) List the 3 flocking rules (one word each): _____ _____ _____

(3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). _____ _____ _____

(9pt) List the other 9 steering behaviors from Craig Reynolds:

_____ _____ _____ _____

_____ _____ _____ _____ _____

CS380/CS580 Final, version 20240725 NAME_____

1	2	3	4	5	Total	Fraction	Grade

(Disclaimer: Some questions might be discarded / not graded, point values might be changed)

Architecture (50pts)

(3pt) Why is it preferred to start with an FSM and put BTs inside the states? FSM model state better than BTs, while BT better represent plans. Placing BT into FSM creates a hierarchical model FSM becomes the high level reasoner that maintains the state and BST becomes the low level reasoner to execute flexible plans.

(3pt) What is the key purpose of a blackboard inside a BT? intercomponent communication to cache expensive operations such as LoS tests and share data between components

(2pt) Smart objects are a programming architecture embodiment of what concept popularized by Don Norman? Intelligent Everything

(3pt) Why is Planning not used in most games? Planning is very expensive as it has to search for possible actions to achieve the goal of the AI. Has to reprogram every time if world or goal changes.

(3pt) How is Utility better than a BT Selector node in terms of action selection? Selectors follow a predetermined plan of actions. However, the set of actions may have different priorities, which forces duplication of BT branches to reflect this. Utility instead chooses the action by scoring them based on priority and taking the highest score, simplifying the decision.

(3pt) What aspect of Utility makes it possible to customize personality? By adding and modifying weights.

(3pt) In Utility, what method can be used to avoid always selecting the highest scoring? Weighted Random

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 50% hit rate against a Slime with 50 hit points: 2.5

$5 \times 0.50 = 2.5$

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 80% hit rate against a Slime with 1 hit point: 4

$5 \times 0.80 =$

(24pt) Draw the Utility curves (all except for Patrol) for an NPC with the following requirements:

- NPC that would like to Reload below 50% ammo. Always prefer Reload over Shoot at less than 25% ammo.
- NPC that prefers Reload over Shoot if ammo 25% to 50% and enemy is more than 5m away.
- NPC wants to Shoot enemies up to 9m away.
- NPC will Punch enemies up to 1m away (always preferable to Shoot or Reload).
- If NPC has less than 10 health, NPC must always Run Away.
- Patrol if nothing else is desired.

Movement (18pts)

(3pt) Why is pathfinding not needed for flying NPCs? There are no u-shaped obstacles in the sky.

(3pt) List the 3 flocking rules (one word each): separation alignment cohesion

(3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). tether wandering obstacle avoidance

(9pt) List the other 9 steering behaviors from Craig Reynolds:

seek and flee pursue and evade arrival containment and wall following

path following flow field following group/crowd path following leader following queuing

(3pt) Why is it preferred to start with an FSM and put BTs inside the states? _

- FSM is very good at modeling state
  - ↳ let FSM take care of "state" → high level reasoner
- BT good at representing fixed plans
  - ↳ let BT take care of the details → lower level reasoner that chooses actions
- BT bad when needing to figure out what state it's in
- hierarchical by allowing FSM to go state and make cyclical more obvious

Behavior Tree

1 2 3

Procedural architecture

Meta-architecture

- Nodes can hold any architecture

Modular

Represents a fixed plan

Cyclical but not obvious

Poor at interruption returning

Poor at state-based / modes

Finite-State Machine

Models **cyclic behavior** extremely well

Excellent for simple behavior

Fewer states is better

- **Brittle**, hard to modify

Game AI Pro 3
Chapter 11

Hierarchical Finite-State Machine

1 2 3

HFSM is **modular** and scales better

State machines are really good at representing state

Behavior trees are not

sometimes people put a tree of a few states, and inside the state is a behavior tree.

also try to keep the parts per tree 3.

In state, the FSM had a planning in it.

Behavior Tree

Start at the root, then trickle down to a leaf

only leaf is choice; the in-betweeners are reasoners

(3pt) Why is it preferred to start with an FSM and put BTs inside the states? _

One common way to address this challenge—an approach that has been applied to nearly every architecture, both in games and in academia—is to break the decision making up hierarchically. That is, have a high-level reasoner that makes the big, overarching decisions, and then one or more lower-level reasoners that handle implementation of the higher-level reasoners' decisions. For example, a high-level reasoner might decide whether to execute a daily schedule of ambient tasks (e.g., get up, get breakfast, go to work, etc.), start a conversation with the player, go into combat, and so forth. Each of those options would contain another reasoner, which decides how to go about accomplishing the goal.

The advantage here is that the complexity of AI configuration scales worse than linearly on the number of options *in a particular reasoner*. To give a sense of the relevance, imagine that the cost of configuring the AI is $O(n^2)$ on the number of options (as it is for FSMs). If we have 25 options, then the cost of configuring the AI is on the order of $25^2 = 625$. On the other hand, if we have five reasoners, each with five options, then the cost of configuring the AI is only $5 \times (5^2) = 125$. Conceptually, this makes sense. When we add a new option, we only need to consider how it relates to other options within the same reasoner—which is much simpler than comparing it to every other option anywhere in the AI.

Examples of this approach abound, from hierarchical FSMs [Millington et al. 09b] to HTN planners [Gorniak et al. 07] and GOAP implementations [Cerpa et al. 08] to strategy game AIs that break the opposing player AI into command hierarchies [Pittman 08]. Behavior trees are perhaps the archetypical example—a BT is really nothing more than a hierarchical infrastructure in which you can place whatever sorts of reasoning architectures best encapsulate the decisions to be made.

Game AI Pro Chapter 4

(3pt) Why is it preferred to start with an FSM and put BTs inside the states? _

FSM are better at modelling state than BTs, whereas BTs are better at representing plans. BT are also very difficult to observe their cyclic behaviour, compared to FSM

Placing BT into FSM allows FSM to be the higher level reasoner that maintains the AI's overarching state. The BT inside the FSM then becomes the lower level reasoner that executes a fixed plan for the state which is more flexible

Hierarchical

Scalability is a huge problem

Hierarchical reasoning is the fix

Stateless trees: problem at returning back to its old state

Since behaviors themselves are stateless, care must be taken when creating behaviors that appear to apply memory. For example, imagine a citizen running away from a battle. Once well away from the area, the "run away" behavior may stop executing, and the highest-priority behavior that takes over could take the citizen back into the combat area, making the citizen continually loop between two behaviors. While steps can be taken to prevent this sort of problem, traditional planners can tend to deal with the situation more easily.

(3pt) What is the key purpose of a blackboard inside a BT? __

- intercomponent communication + coordination
  - ↳ cache expensive operations such as LoS on global blackboard to minimize computations
  - ↳ store output from one reasoner for the input of another/same reasoner in the character blackboard

Conclusions

The main source of the power of the blackboard is the fact that the architecture places such strict constraints on intercomponent communication. It is the demand that KSs communicate only through the blackboard—and that all communication take more or less the same superficial form—that results in behavior that is opportunistic, cooperative, coordinated, and extensible.

7.1 AI Game Wisdom

suspend another, without changing the overall plan of attack. We can use a *blackboard* to share information and ideas between AI components or between characters. We can

Game AI Pro

Chapter 5

5.6.1 Blackboards

In the academic AI community, blackboard architectures typically refer to a specific approach in which multiple reasoners propose potential solutions (or partial solutions) to a problem, and then share that information on a *blackboard* [Wikipedia 12, Isla et al. 02]. Within the game community, however, the term is often used simply to refer to a shared memory space which various AI components can use to store knowledge that may be of use to more than one of them, or may be needed multiple times. In our architecture, for example, every character has access to two blackboards. The *character blackboard* stores information specific to the character, and is only accessible by that character's AI. The *global blackboard* is accessible by all characters, and is used to store general information.

There are many types of information that can be stored on a blackboard. One common use is to store expensive checks on the blackboard, so as to avoid the cost of running them more than once. Line of sight (LOS) checks are a common example. Quite often, more than one component in the AI (or more than one character's AI) will want to check for visibility between the same two objects. These checks can be extremely expensive. To lessen the impact of this problem, we can run the check once and then cache it on the blackboard. Path-planning checks are similar.

Another common use is to store information used to coordinate AI components. This could include partial solutions such as those found in classic blackboard systems, but it could also simply be information used to coordinate between multiple characters or between different AI components within a character. For example, if you want your characters to focus their attacks on a single enemy or to ensure that every enemy is attacked, you can store target assignments on the blackboard. If you want to coordinate spatial motion—perhaps flanking, or placing the tanks in front and the DPS and healers in back, then you can store movement plans on the blackboard. If you want to ensure that two characters don't try to use the same cover spot, then you can have them reserve it on the blackboard. If you have one reasoner whose output is the input for another—for example, the emotional reasoner that we discussed earlier in this paper—then that output can be placed on the blackboard.

There is an interview with Damián Isla on AIGameDev.com, which gives an excellent introduction to blackboard architectures as they are commonly used in games [Isla 10].

(3pt) What is the key purpose of a blackboard inside a BT? __

ge Management

↳ what do you do with things you know

a <key, value>

like taking down notes

oards

in behaviour tree it's taking notes for other nodes

agents can also use higher level blackboards

↳ instead of broadcast messages, it's a persistent place to store knowledge to be shared (although cleaning up is its own problem)

edge Everywhere

Knowledge Management: Blackboards

Store data in communal places

Store data in the terrain or shared data structures

Store expensive checks (LOS or pathing)

Coordinate AI characters

(3pt) What is the key purpose of a blackboard inside a BT? __

Blackboard Cooperation

Complex problem is posted on a shared communication space

- Agents propose solutions
- Solutions scored and selected
- Continues until problem is solved

Alternatively, use concept to facilitate communication and cooperation

(2pt) Smart objects are a programming architecture embodiment of what concept popularized by Don Norman?

Intelligent Everything

Knowledge Management: Intelligent Everything

smart objects

Store intelligence in objects and the world

Popularized by The Sims

- **Smart objects**
- **Smart terrain**

Game AI Pro Chapter 5

5.6.2 Intelligent Everything

When we think about the AI for a character, it seems intuitive to put all of the knowledge needed in the character. This can result in a monolithic, difficult to extend AI, however. It can also result in considerable duplication between similar (but not identical) characters.

One trick is to put the intelligence in the world, rather than in the character. This technique was popularized by *The Sims*, though earlier examples exist. In *The Sims* (and its sequels), objects in the world not only advertise the benefits that they offer (for example, a TV might advertise that it's entertaining, or a bed might advertise that you can rest there), they also contain information about how to go about performing the associated actions [Forbus et al. 01].

Wikipedia

The Design of Everyday Things is a best-selling[1] book by cognitive scientist and usability engineer Donald Norman. Originally published in 1988 with the title ***The Psychology of Everyday Things***, it is often referred to by the initialisms ***POET*** and ***DOET***. A new preface was added in 2002 and a revised and expanded edition was published in 2013.[2]

The book's premise is that design serves as the communication between object and user, and discusses how to optimize that conduit of communication in order to make the experience of using the object pleasurable. It argues that although people are often keen to blame themselves when objects appear to malfunction, it is not the fault of the user but rather the lack of intuitive guidance that should be present in the design.

(3pt) Why is Planning not used in most games? _

Planning

the most declarative of the declarative structures

Declarative architecture

Two common types

- **GOAP** (Goal-Oriented Action Planning)
- **HTN** (Hierarchical Task Network)

useful for multicores

Planning search algorithm is the **reasoner**

Actions or tasks are the **options**

↳ find a sequence of actions that accomplish a goal

Planning

Goal: Enemy Dead

- shoot them
- lure them to a trap
- etc

reasoner will choose the cheapest option

Search for a sequence of actions

enemy ☐ alive

- Transform the "Current State of the World" into a "Goal State of the World"

↳ enemy is dead

Replan if the goal changes

- If near death, then replan with "Don't Die" goal

Planning

Goal: Rabin is a billionaire

a really weird way to program AI but very powerful

Search for a sequence of actions

- Transform the "Current State of the World" into a "Goal State of the World"

Replan if the goal changes

the alignment problem

- If charged with fraud, then replan with "Don't Go to Jail" goal

(3pt) How is Utility better than a BT Selector node in terms of action selection? _

Utility-Based

Declarative architecture

Each **option** gets scored by a heuristic **reasoner**

- **Score** is a float
- Also known as a **weight**, **priority**, or **utility**

FPS Example

- Flee: 0.30 utility
- Reload: 0.10 utility
- Attack: 0.60 utility

→ all the reasons why I would do that then score

Utility-Based

FPS Example

- Flee: 0.30 utility
- Reload: 0.10 utility
- Attack: 0.60 utility

deterministic and can prioritize on the fly
the heuristic reasoner can rescore these based on situation

↓ statement if reload already

lookup table ↓ how much do I want to reload

Choose an option

- Highest score → you become predictable
- **Weighted random**

↳ ...what if our gun has full bullets, then reload will be zeroed out

try all children until success

Selector Code

```
C_Selector::C_Selector() : currentIndex(0) {}

void C_Selector::on_enter()
{
    currentIndex = 0;
    BehaviorNode::on_enter();
}

void C_Selector::on_update(float dt)
{
    // if any child succeeds, node succeeds
    // if all children fail, node fails
    BehaviorNode *currentNode =
                children[currentIndex];
    currentNode->tick(dt);
```

```
    if (currentNode->succeeded() == true)
    {
        on_success();
    }
    else if (currentNode->failed() == true)
    {
        // move to the next node
        ++currentIndex;

        // if we hit size, that means all nodes failed
        if (currentIndex == children.size())
        {
            on_failure();
        }
    }
}
```

on first success, we succeeded!

In a standard behavior tree, priority is static. It is baked right into the tree. The simplicity is welcome, but in practice it can be frustratingly limiting. The same behavior may require different relative priorities, depending on the context. Ensuring our Monster Hunter's primary weapon has a full clip should always be a consideration, even if we're casually patrolling the jungle. But if we're engaged with a savage monster, it's absolutely necessary that we continue to deal damage. Behavior tree authors often deal with this conundrum by duplicating sections of the tree at different branches, with different conditions and/or priorities. Even with slick sub-tree instancing or referencing, this still becomes inefficient, verbose, and potentially fragile.

Even more troublesome cases surface when a simple yes versus no determination isn't easily established. If our *Combat* selector is evaluating its options, should it choose to have us seek a rendezvous with our medic and his space-age healing tech, or should we put everything we have into quickly dropping the giant alien beast threatening to eat us all? This sort of decision is best made only after considering a potentially broad combination of inputs.

Decisions are rarely binary, and many behaviors simply do not have priorities we can comfortably establish offline. Let's start with a simple example behavior tree (Figure 10.1). Having no ability to shoot is a precondition for the *Seek Medic* behavior, forcing us to duplicate the behavior, as seen in Figure 10.2. We could start by giving *Seek Medic* stricter conditions and prioritizing it over *Shoot*, but this will likely create the opposite problem where the Monster Hunter immediately takes the *Seek Medic* action the instant conditions pass. This is the sort of fundamental problem we want to address with the integration of utility.

(3pt) What aspect of Utility makes it possible to customize personality? _

Many aspects of decision making aren't quite as tidy, however. There are numerous questions that can be asked where a "yes or no" answer is not appropriate. For example, we may want to consider how far away the enemy is, how many bullets I have left, how hungry I am, how wounded I am, or any number of continuous values. Correspondingly, these continuous values can be mapped over into *how much* I want to take an action rather than simply whether to take the action or not. A utility-based system measures, weighs, combines, rates, ranks, and sorts out many considerations in order to decide the *preferability* of potential actions. Using the above example as a guide, we could assess how *strongly* we want (or need!) to attack, reload, hide, etc.

While utility techniques can be used to supplement the transition logic of other architectures, it is very possible to build an entire decision engine based on utility. In fact, there are times when building a utility-based AI is far preferable to other methods. These might include games where there are many possible actions, and either there isn't a single "right" answer or the selection of a preferable action might be based on a large number of competing inputs. In these cases, we are going beyond simply using utility to measure or rate something. Instead, we are using it to drive the actual decision mechanism as well. Another way of stating it is that, rather than saying "This is the one action you will do," the utility-based system *suggests*, "Here are some possible options that you might want to do."

One well-documented example of this is the use of utility in *The Sims*. In these games, the agents (i.e., the actual "Sims") take information from their environment and combine it with their own internal state to arrive at a preferability score for each potential action. For example, the fact that I am "very hungry" combined with the availability of "poor food" would certainly be more attractive than if I was only "a little hungry." Additionally, the proximity of "spectacular" food might still make for a high priority even if I was only "a little hungry." Note that the descriptors "spectacular," "rather," "poor," and "a little" would actually be numbers between some set minimum and a maximum. (A typical method to use is a floating point number between 0 and 1.)

When it is time to select a new action (either because the current one is finished or through some sort of interrupt system), some method is used to select from among the candidates. For example, the scores for the potential actions could be sorted so that we can simply select the "most appropriate" action—that is, the one with the highest score. An alternate way is to use the scores to seed a weighted random selection. By casting a random number against these weighted probabilities, the most preferable actions have a higher chance of being selected. As an action's suitability goes up, its score goes up, as does its chance of being selected.

Game AI Pro Chapter 4

9.4 Decision Factors

It is rare that any given decision will only rely on a single piece of data. A decision factor can be thought of as a single point of consideration for a decision. For example, when deciding which website to purchase the cat toy from, we don't usually just consider price, but also consider brand loyalty, shipping times, customer reviews, etc. Each of these data points are factors that we weigh into the calculation of the final utility score for whether or not to buy from that website. Factors can also be further modified by weights that determine how much the AI cares about that particular factor, which emulates personality.

One way to achieve this result is to apply the expected utility calculation in Equation 9.1 to each decision factor to come up with the utility of that factor. Assuming those scores are all normalized, you can average them together to come up with a final utility score for that decision. This lets us define a decision as nothing more than some combination of decision factors.

Game AI Pro

Chapter 9

(3pt) In Utility, what method can be used to avoid always selecting the highest scoring? _

Utility-Based

deterministic and can prioritize on the fly
the heuristic reasoner can rescore these based on situation

FPS Example

- Flee: 0.30 utility
- Reload: 0.10 utility
- Attack: 0.60 utility

Choose an option

- Highest score
- **Weighted random**

↳ you became predictable

↳ what if our gun has full bullets, then reload will be zeroed out

Weighted random

also bucketing, but probably less important

9.6 Picking an Action

Game AI Pro Chapter 9

Once the utility has been calculated for each action, the next step is to choose one of those actions. There are a number of ways you can do this. The simplest is to just choose the highest scoring option. For some games, this may be exactly what you want. A chess AI should definitely choose the highest scoring move. A strategy game might do the same.

For some games (like *The Sims*), choosing the absolute best action can feel very robotic due to the likelihood that the action will *always* be selected in that situation. Another solution is to use the utility scores as weight, and randomly choose one of the actions based on the weights. This can be accomplished by dividing each score with the sum of all scores to get the percentage chance that the action will be chosen. Then you generate a random number and select the action that number corresponds to. This tends to have the opposite problem, however. Your AI agents will behave reasonably well most of the time, but every now and then, they'll choose something utterly stupid.

You can get the best of both worlds by taking a subset of the highest scoring actions and choosing one of those with a weighted random. This can either be a tuned value, such as choosing from among the top five scoring actions, or it can be percentile based where you take the highest score and also consider things that scored within, say, 10% of it.

This will generally solve the problem at hand, but there could also be times when some set of actions are just completely inappropriate. You may not even want to score them. For example, say you're making an FPS and have a guard AI. You might have some set of actions for him to consider, like getting some coffee, chatting with his fellow guard, checking for lint, etc. If the player shoots at him, he shouldn't even consider any of those actions and only try to score actions that involve combat. In a similar example from *The Sims*, if a Sim is starving to death, she shouldn't bother scoring actions that result in her satisfying her *Fun* motive.

The most straightforward way to solve this is with *bucketing*, also known as *dual utility AI* [Dill 11]. All actions are categorized into buckets and each bucket is given a weight. The higher priority buckets are always processed first. If there are any valid actions in those higher priority buckets, they are always selected before actions in lower priority buckets. In the FPS example above, all combat actions would be in a higher priority bucket than the idle actions. If there are any valid combat actions to take (i.e., they score higher than 0), the guard will always choose one of them and won't consider any of the idle actions. Only when none of the combat actions are valid will the guard choose an idle action.

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 50% hit rate against a Slime with 50 hit points: ____

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 80% hit rate against a Slime with 1 hit point: _____

Expected Utility

Utility Scores are Normalized: [0, 1]

- Makes scores comparable

in real life, nothing is a 100%

Expected Utility

chance of it succeeding

- **Utility** * **Probability**

chance of it happening

Maximum Expected Utility

- Choose one with highest Expected Utility

Expected Utility

Attack with dagger

- Utility: 0.4
- Probability of hit: 95%
- Attack with dagger: 0.4 x 0.95 = **0.38**

Attack with two handed sword

- Utility: 0.9
- Probability of hit: 60%
- Attack with sword: 0.9 x 0.60 = **0.54**

a better decision

For example, an enemy AI in an RPG attacking the player has two possible outcomes—either the AI hits the player or it misses. If the AI has an 85% chance to hit the player, and successfully hitting the player has a calculated utility score of 0.6, the adjusted utility would be $0.85 \times 0.6 = 0.51$. (Note that, in this case, missing the player has a utility of zero, so there's no need to factor it in.) Taking this further, if this attack were to be compared to attacking with a different weapon, for example, with a 60% chance of hitting but a utility score of 0.9 if successful, the adjusted utility would be $0.60 \times 0.9 = 0.54$. Despite having a lesser chance of hitting, the second option provides a greater overall *expected utility*.

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 50% hit rate against a Slime with 50 hit points: ____

(3pt) Compute Expected Utility for a sword that does 5 hit points damage with a 80% hit rate against a Slime with 1 hit point: _____

Game AI Pro Chapter 9

The most common technique is to multiply the utility score by the probability of each possible outcome and sum up these weighted scores. This will give you the *expected utility* of the action. This can be expressed mathematically with Equation 9.1.

$$EU = \sum_{i=1}^{n} D_i P_i \tag{9.1}$$

In this case, D is the desire for that outcome (i.e., the utility), and P is the probability that the outcome will occur. This probability is normalized so that the sum of all the probabilities is 1. This is applied to every possible action that can be chosen, and the action with the highest expected utility is chosen. This is called the principle of *maximum expected utility* [Russell et al. 09].

Desire is the sword's damage
Probability is hit rate. not enough info to normalize

a). $5 \times 0.5 = 2.5$

b). $5 \times 0.8 = 4.0$

(24pt) Draw the Utility curves (all except for Patrol) for an NPC with the following requirements:

- NPC that would like to Reload below 50% ammo. Always prefer Reload over Shoot at less than 25% ammo.
- NPC that prefers Reload over Shoot if ammo 25% to 50% and enemy is more than 5m away.
- NPC wants to Shoot enemies up to 9m away.
- NPC will Punch enemies up to 1m away (always preferable to Shoot or Reload).
- If NPC has less than 10 health, NPC must always Run Away.
- Patrol if nothing else is desired.

Draw a Utility Curve

Draw the Utility curves (all except for Patrol) for an NPC with the following requirements:

- NPC that would like to Reload below 50% ammo. Always prefer Reload instead of Shoot at less than 25% ammo.
- NPC that prefers Reload instead of Shoot if ammo 25% to 50% and enemy is more than 5m away.
- NPC wants to Shoot enemies up to 9m away.
- NPC will Punch enemies up to 1m away (always preferable to Shoot or Reload).
- If NPC has less than 10 health, NPC must always Run Away.
- Patrol if nothing else is desired.

Movement (18pts)

a). (3pt) Why is pathfinding not needed for flying NPCs? _____

(3pt) List the 3 flocking rules (one word each): _____

(3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). _____

(9pt) List the other 9 steering behaviors from Craig Reynolds:

a).

THIS IS THE REASON THAT FLYING AGENTS TYPICALLY DO NOT NEED TO PATHFIND

what is are U-shape obstacles?

Planned Movement

unfortunately U-obstacles exist
the more mazelike it gets, the more AI get stuck

flying creatures usually never pathfind — they don't need to — the sky's empty

Movement (18pts)

(3pt) Why is pathfinding not needed for flying NPCs? _____

b). (3pt) List the 3 flocking rules (one word each): _____

c). (3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). _____

(9pt) List the other 9 steering behaviors from Craig Reynolds:

b).

Flocking: 3 Simple Rules

Separation

we consider the boids close to us only

Alignment

higher cohesion, tighter flock

Cohesion

c).

Flocking: 3 Other Rules

1. Obstacle avoidance → to avoid the side of the "pond" they may be in. however if they are in the pond, they will stick to the borders
2. Wandering → need this, otherwise they will eventually look like fighter jets this will make 'em wiggle and break up the fighter jets
3. Tether → we don't have a pond, we don't want them to go together.

unofficial

Also, each boid should have random properties (within a small range)

- Speed
- Acceleration

Movement (18pts)

(3pt) Why is pathfinding not needed for flying NPCs? _____

(3pt) List the 3 flocking rules (one word each): _____

(3pt) Name 3 other rules/steering behaviors to enhance flocking (randomness is not an answer). (<=2 words each). _____

d). (9pt) List the other 9 steering behaviors from Craig Reynolds:

- Simple behaviors for individuals and pairs:
  - *Seek* and *Flee* 1
  - *Pursue* and *Evade* 2
  - *Wander* - Flocking enhance
  - *Arrival* 3
  - *Obstacle Avoidance* - Flocking enhancement
  - *Containment* } 4
  - *Wall Following* }
  - *Path Following* 5
  - ~~*Flow Field Following*~~ 6
- Combined behaviors and groups:
  - Group ~~*Crowd Path Following*~~ 7 } make these count as one
  - *Leader Following* 8
  - *Unaligned Collision Avoidance* 9
  - *Queuing* (at a doorway) 10
  - *Flocking* (combining: *separation, alignment, cohesion*)

A* Pathfinding Problem (100pts)

(100pt) Given the following search space, start node (A), and goal node (G), calculate on this sheet of paper a path found using A*. Please show all work using the Open and Closed lists, **marking the order deleted using numbers starting at 1** when deleting a node **off of either list**. Please mark parent pointers and all cost terms. Use the indicated actual cost along each node-to-node connection and use the distance values in the table for the heuristic. **(NOTE: One mistake loses half points, two or more loses all points – don't make even one mistake – double check all math – double check the final path is optimal). The correct solution uses all blank rows. Don't forget to put the final path in the bottom right box.**

Heuristic estimate to Goal node (G) from any given node (heuristic is admissible, but not consistent):

A	B	C	D	E	F	G	H	K
5.4	4.4	5.0	3.4	2.4	1.4	0.0	10.0	1.0

Open List:

Order Deleted	Node	Parent	g(x)	h(x)	f(x)

Closed List:

Order Deleted	Node	Parent	f(x)

Final waypoint path from start to goal:

A* Pathfinding Problem (100pts)

(100pt) Given the following search space, start node (A), and goal node (G), calculate on this sheet of paper a path found using A*. Please show all work using the Open and Closed lists, **marking the order deleted using numbers starting at 1** when deleting a node **off of either list**. Please mark parent pointers and all cost terms. Use the indicated actual cost along each node-to-node connection and use the distance values in the table for the heuristic. **(NOTE: One mistake loses half points, two or more loses all points – don't make even one mistake – double check all math – double check the final path is optimal). The correct solution uses all blank rows. Don't forget to put the final path in the bottom right box.**

Heuristic estimate to Goal node (G) from any given node (heuristic is admissible, but not consistent):

A	B	C	D	E	F	G	H	K
5.4	4.4	5.0	3.4	2.4	1.4	0.0	10.0	1.0

Open List:

Order Deleted	Node	Parent	g(x)	h(x)	f(x)
1	A	null	~~0~~	5.4	5.4
2	B	A	1	4.4	5.4
4	C	A	1	5.~~0~~	6.~~0~~
3	D	B	2	3.4	5.4
5	E	D	6	2.4	8.4
6-R	K	C	13	1.~~0~~	14.~~0~~
7	F	E	7	1.4	8.4
8-R	K	E	12	1	13.~~0~~
10	H	F	8	10.~~0~~	18
9	K	F	11	1	12.~~0~~
13-R	G	K	19	~~0~~	19.~~0~~
12	K	H	10	1	11.~~0~~
R	G	K	18	~~0~~	18.~~0~~

Closed List:

Order Deleted	Node	Parent	f(x)
	A	null	5.4
	B	A	5.4
	D	B	5.4
	C	A	6.~~0~~
	E	D	8.4
	F	E	8.4
11-R	K	F	12.~~0~~
	H	F	18.~~0~~
	K	H	11.~~0~~

Final waypoint path from start to goal:

A→B→D→E→F→H→K→G

Pathfinding (65pts)

(30pt) Carefully draw each search space representation (include ***ALL*** details required by a search algorithm):

Grid

Corner Graph

Nav Mesh

(2pt) Given all pathfinding search space(s), which ones do not represent the walkable surface area? _____

(2pt) Given all pathfinding search space(s), which ones can be used with the Roy-Floyd-Warshall algorithm? _____

(3pt) If the heuristic cost in A* is absolutely accurate,
what does it exactly represent? (<=6 words, be precise) _____

(3pt) If a heuristic is admissible in A*, then it _____ the true cost.

(3pt) In a NavMesh, why can't the floor polygons of a level be directly used? _____

(3pt) Why would a Waypoint Graph be preferred over a Grid in an FPS like Call of Duty? _____

(3pt) What is the primary run-time CPU advantage of NavMesh over Waypoint Graph? _____

(3pt) What is the optimal A* heuristic in a grid-based environment that contains teleporters? _____

(3pt) If non-optimal paths are acceptable, when is it bad to over-estimate the heuristic with A*? _____

(4pt) Name 2 primary problems with Goal Bounding 1. _____ 2. _____

(3pt) A flow field could be generated using which search algorithm? _____

(3pt) In order to smooth a waypoint path on a NavMesh, you'll need this algorithm _____

Agent Awareness (20pts)

(6pt) Name 3 unique benefits of pathfinding sounds. 1. _____

2. _____ 3. _____

(10pt) Draw a complete vision model with percentage zones that a game like Splinter Cell would have had (make up reasonable %s):

(4pt) In the vision model, list 2 non-spatial aspects that can't be drawn in the previous question that would decrease the percentages.

1. _____

2. _____

Pathfinding (65pts)

(30pt) Carefully draw each search space representation (include ***ALL*** details required by a search algorithm):

(2pt) Given all pathfinding search space(s), which ones do not represent the walkable surface area? waypoint, corner graphs

(2pt) Given all pathfinding search space(s), which ones can be used with the Roy-Floyd-Warshall algorithm? grids, waypoints, corner graphs, circle-based waypoints, nav meshes

(3pt) If the heuristic cost in A* is absolutely accurate, what does it exactly represent? (<=6 words, be precise) best path without searching unnecessary nodes

(3pt) If a heuristic is admissible in A*, then it equal to or less than the true cost.

(3pt) In a NavMesh, why can't the floor polygons of a level be directly used? impassable obstacles may be overlaid over the floor polygons

(3pt) Why would a Waypoint Graph be preferred over a Grid in an FPS like Call of Duty? waypoints represent a small fraction of points on a grid, making it cheaper in memory for large maps

(3pt) What is the primary run-time CPU advantage of NavMesh over Waypoint Graph? better preprocessing and postprocessing for smoothing and animations

(3pt) What is the optimal A* heuristic in a grid-based environment that contains teleporters? Ø (Dijkstra's)

(3pt) If non-optimal paths are acceptable, when is it bad to over-estimate the heuristic with A*? when the terrain has random obstacles that would cause significant backtracking

(4pt) Name 2 primary problems with Goal Bounding 1. can only be used on static maps 2. requires extra memory at runtime for precomputed data (4 values per edge)

(3pt) A flow field could be generated using which search algorithm? Dijkstra (by floodfilling)

(3pt) In order to smooth a waypoint path on a NavMesh, you'll need this algorithm catmull-rom spline

Agent Awareness (20pts)

(6pt) Name 3 unique benefits of pathfinding sounds.

1. players can sense the volume, indicating source distance
2. players can sense pitch, indicating walls are present
3. players can sense location of the source

(10pt) Draw a complete vision model with percentage zones that a game like Splinter Cell would have had (make up reasonable %s):

(4pt) In the vision model, list 2 non-spatial aspects that can't be drawn in the previous question that would decrease the percentages.

1. player is in shadow
2. player is hiding behind cover

Pathfinding (65pts)

(30pt) Carefully draw each search space representation (include ***ALL*** details required by a search algorithm):

Grid

Corner Graph

Nav Mesh

Grid

Corner Graph

Nav Mesh

Pathfinding (65pts)

(30pt) Carefully draw each search space representation (include ***ALL*** details required by a search algorithm):

Grid

Corner Graph

Nav Mesh

Grids

looks weird so we can move diagonally

smooth out the corners

rubberband, make it more direct

Corner Graphs (Points of Visibility)

put it as far out as the agent radius
becomes a little problem if agents have different radii
might need different graphs

industrial strength

waypoint is easy

Navigation Meshes (Nav Mesh)

but these can!

any n-size polygon as long as they are convex

node interfaces

I need an algo for here so here! it's a funnel algorithm

back off a little from the walls

some artist made the floor
you could pay them to do it...

(2pt) Given all pathfinding search space(s), which ones do not represent the walkable surface area? _____

(2pt) Given all pathfinding search space(s), which ones can be used with the Roy-Floyd-Warshall algorithm? _____

Grids

walkable

Navigation Meshes (Nav Mesh)

industrial strength

waypoint is easy

but these can!

any n-size polygon as long as they are convex

some artist made the floor you could pay them to do it...

Waypoint Graphs

i want to go to Y

THESE TWO PATHFINDING SEARCH SPACE REPRESENTATIONS DO NOT REPRESENT THE WALKABLE SURFACE AREA

circle was area

what is a waypoint graph? corner graph?

Corner Graphs (Points of Visibility)

put it as far out as the agent radius becomes a little problem if agents have different radii & might need different graphs

Circle-Based Waypoint Graph

exceels for outdoor games, and very good indoors

they can't perfectly represent where we can walk...

(2pt) Given all pathfinding search space(s), which ones do not represent the walkable surface area? _____

(2pt) Given all pathfinding search space(s), which ones can be used with the Roy-Floyd-Warshall algorithm? _____

Roy-Floyd-Warshall
All-Pairs Shortest Path

spacecraft

	A	B	C	D	E
A	A	B	C	B	B
B	D	B	D	D	D
C	A	A	C	A	A
D	E	B	E	D	E
E	C	C	C	C	E

alternatively areas

Speed of search?

Cost to compute?

What's the downside during runtime?

Which search space representations does it work on?

(3pt) If the heuristic cost in A* is absolutely accurate, what does it exactly represent? (<=6 words, be precise) best path without searching unnecessary nodes

(3pt) If a heuristic is admissible in A*, then it _____ the true cost.

A*

Completeness

Heuristic Search

Admissible heuristic = Optimal path

This is a challenge. The closer you get to the true cost, the more efficient A* will be. But if you go over, A*will no longer guarantee an optimal path.

Factoid: An admissibl heuristic does nothin for Greedy Best First.

Factoid: A heuristic guess of zero is admissible! A* will just become Dijkstra if h(x) = 0.

A*

will guarantee the best path, but your guess must be admissable

in a world of teleporters, you would always need to guess ∅ not to overguess
→ basically dijkstra's

can be less or equal

this would just be a straight line. because you won't overguess

closer you get to the truth A* becomes more efficient

if it's over, then A* will not guarantee best path

Combines Dijkstra and Greedy Best-First

- **Given cost** and **Heuristic cost** usually straight line distance

Open List sorted by:

- Final cost = Given cost + Heuristic cost
- f(x) = g(x) + h(x)
- Final cost = Given cost + (Heuristic cost * **weight**)
- f(x) = g(x) + (h(x) * **weight**)

Recap of Search Algorithms

Guarantee a path is found if it exists (**completeness**)

- Breadth-First, Depth-First, Dijkstra, A* and all variants

Guarantee an **optimal** path

- Dijkstra, A* and all variants (if **admissible** heuristic)

Heuristic search algorithms (smart, head toward goal)

- Greedy Best-First, A* and all variants

Pathfinding Search Algorithms

looks like a search ellipse

cost can also be influenced by inward's vision

1 5 1

the combo of Dijkstra and greedy best-first

doesn't use much memory

- Random Bounce — not complete
- Robust Trace — complete but not optimal

A*

Jump Point Search (JPS+)

Roy-Floyd-Warshall

tracks progress with open and closed lists

- Breadth-First — "ply by ply" — doesn't really use cost
- Depth-First — may return either path, time both-ways
- Dijkstra — takes in cost, guarantees optimal — blindly searches
- Greedy Best-First — uses the goal to inform its path — complete, but may not be best optimal path because of the pressure applied from the goal like "gravity" → heuristic → the h-cost to the goal (the distance I am from the goal)

look-up table?

(3pt) In a NavMesh, why can't the floor polygons of a level be directly used? _

no reference. just threads

still a good chapter

Game AI Pro Chapter 27

explains the use of navmeshes

(3pt) Why would a Waypoint Graph be preferred over a Grid in an FPS like Call of Duty? _

(3pt) What is the primary run-time CPU advantage of NavMesh over Waypoint Graph? ___

18.4 Waypoint Graphs

Waypoint graphs represent the world as an abstract graph. Importantly, waypoint graphs do not have an explicit mapping between nodes in the graph and walkable space. Waypoint graphs were widely used before the popularity of navigation meshes grew. While they have been criticized for their shortcomings [Tozour 08], they have also been praised for their strengths [Champandard 10].

The pros are:

- Waypoint graphs are relatively easy to implement.
- Waypoint graphs are easy to modify if the changes are known ahead of time. For instance, if a door in the world closes and is locked, it is easy for the developer to mark the edges in the graph that cross the opening of the door and block them when the door is shut.
- Waypoint graphs represent only a small fraction of the points found in a grid. This sparse representation of walkable space is both cheap to store and leads to inexpensive path planning requests.

The cons are:

- Path quality can suffer if there are not enough walkable edges in the graph, but too many walkable edges will impact storage and planning complexity.
- Waypoint graphs may require manual placement of nodes to get good path quality.
- Localization on waypoint graphs requires mapping between game space and the graph. If a character is knocked off of the graph, it may be unclear where the character should actually be within the waypoint graph.
- Because there is no explicit representation of the underlying state space, smoothing off the waypoint graph can result in characters getting stuck on physics or other objects.
- Dynamic changes are difficult when they aren't known ahead of time. If a character can create an unexpected hole in a wall, new connections on the waypoint graph are needed. However, it can be expensive to check all nearby connections to verify if they have become passable due to the changes in the map.

Game AI Pro, Chapter 18

(3pt) Why would a Waypoint Graph be preferred over a Grid in an FPS like Call of Duty? _

(3pt) What is the primary run-time CPU advantage of NavMesh over Waypoint Graph? ___

18.5 Navigation Meshes

Navigation meshes represent the world using convex polygons [Tozour 04]. A special case of navigation meshes are constrained Delaunay triangulations [Chen 09], for which the world is only represented by triangles. Note that grids can also be seen as a special case of navigation meshes, as both representations use convex polygons, but their usage is significantly different in practice.

The pros are:

- Polygons can represent worlds more accurately than grids, as they can represent non-grid-aligned worlds.
- With the accurate representation of a polygon it is easier to correctly perform smoothing both before and during movement. This accuracy can also be used for tighter animation constraints.
- Path planning on navigation meshes is usually fast, as the representation of the world is fairly coarse. But, this does not impact path quality, as characters are free to walk at any angle.
- Navigation meshes are not as memory-intensive as grids as they can represent large spaces with just a few polygons.

The cons are:

- The time required to implement a navigation mesh is significant, although good open-source implementations are available [Mononen 11].
- Navigation meshes often require geometric algorithms, which may fail in special cases such as parallel lines, meaning that implementation is much more difficult [Chen 09].
- Changes to navigation meshes can be difficult or expensive to implement, especially when contrasted with changes to grid worlds.
- Localization on navigation meshes can be expensive if poorly implemented. Good implementations will use additional data structures like grids to speed up the process [Demyen 06].

Game AI Pro, Chapter 18

(3pt) What is the optimal A* heuristic in a grid-based environment that contains teleporters? _

A*

will guarantee the best path, but your guess must be admissable

can be less or equal | if it's over, then A* will not guarantee the path

in a world of teleporters, you would always need to guess ∅ not to overguess

→ basically dijkstra's

this would just be straight line. because you won't overguess

closer you get to the truth A* becomes more efficient

Combines Dijkstra and Greedy Best-First

usually straight line distance

- **Given cost** and **Heuristic cost**

Open List sorted by:

- Final cost = Given cost + Heuristic cost
- f(x) = g(x) + h(x)
- Final cost = Given cost + (Heuristic cost * **weight**)
- f(x) = g(x) + (h(x) * **weight**)

(3pt) If non-optimal paths are acceptable, when is it bad to over-estimate the heuristic with A*? __

Equation 17.2 shows the addition of a weight on the heuristic portion of the formula.

$$f(x) = g(x) + \left(h(x) \times weight\right) \tag{17.2}$$

By altering the weight, we can tune how A* behaves. If the weight is zero, then the formula reduces down to just $g(x)$, which is identical to the Dijkstra search algorithm. This approach is guaranteed to find an optimal path, but is not a "smart" search because it explores uniformly outward in all directions. If the weight is 1.0, then the equation is the classic A* formula, guaranteed to expand the minimal number of nodes needed to find an optimal path given the current heuristic estimate, modulo tie-breaking. If the weight is larger than 1.0, then we are tilting the algorithm toward the behavior of Greedy Best-First search, which is not optimal but focuses the search on finding the goal as quickly as possible.

Thus, we can tune A* with the weight to lean it toward Dijkstra or Greedy Best-First. By using weights in the neighborhood of 1.1 to 1.5 or higher, we can progressively force the search to more aggressively push toward the goal node, at the increasing expense of a possible sub-optimal path. When the terrain is filled with random obstacles that resemble columns or trees, then a larger weight makes a lot of sense and the path is not noticeably suboptimal. However, if significant backtracking away from the goal is required for the final path, then a lower weight is advisable.

(4pt) Name 2 primary problems with Goal Bounding 1. __

The most important constraint is that your game maps must be static. That is to say that the nodes and edges in your search space must not change during gameplay, and the cost to go from one node to another must also not change. The reason is that connection data must be precomputed. Changing a single node, edge, or cost would invalidate all of the precomputed data. Because the precomputed data takes so long to create, it is not feasible to dynamically rerun the computation if the map changes.

The other primary constraint is that goal bounding requires extra memory at runtime for the precomputed data. For every node, there has to be four values per edge leading to another node. On a grid, each node has eight edges (or connections) to other nodes, so the necessary data for a grid search space are 32 values per node. On a navmesh, typically each poly node has three edges, so the necessary data for a navmesh search space are 12 values per poly node. Typically, these values will need to be 2 bytes each, but for smaller maps, 1 byte each might suffice (e.g., a grid map with a height and width less than 256).

Lastly, a minor constraint is that each map must be precomputed offline, which takes time. The precomputation algorithm is $O(n^2)$ in the number of total nodes on the map. For example, a very large 1000×1000 grid map would have 1 million nodes, requiring $1{,}000{,}000^2$ or 1 trillion operations during precomputation for that map. Depending on the map size, this can take between 5 minutes and several hours per map. It is computationally demanding enough that you could not precompute the data at runtime. However, there are optimizations to Dijkstra for uniform cost grids, such as Canonical Dijkstra that can make this computation much faster (Sturtevant and Rabin 2017).

Game AI Pro 3 Chapter 22

(no source, but could also be A*)

(3pt) A flow field could be generated using which search algorithm? Dijkstra's

(3pt) In order to smooth a waypoint path on a NavMesh, you'll need this algorithm _

Post-Processing: Smoothed Paths

Physics-based

- At movement time – problems?
- On waypoints before movement (demo)

Splines

- Bezier Spline
- Catmull-Rom Spline

Agent Awareness (20pts)

(6pt) Name 3 unique benefits of pathfinding sounds.

1. _____

2. _____

3. _____

Human Hearing

Can recognize sounds
- Knows what emits each sound

Can sense volume
- Indicates distance of sound

Can sense pitch
- Sounds muffled through walls have more bass

Can sense location
- Where sound is coming from

Modeling Hearing

sound falls off distance

modelling waves would be insanely expensive

the walls bent waves

How should an agent know about sounds?

Reflect off every surface?

How do you model hearing efficiently?

pathfinding & find

(10pt) Draw a complete vision model with percentage zones that a game like Splinter Cell would have had (make up reasonable %s):

(4pt) In the vision model, list 2 non-spatial aspects that can't be drawn in the previous question that would decrease the percentages.

1. whether the player is in shadow
2. player hiding behind cover

Spec-ops: oval

• how much you can identify ie is

let's say I'm injured at 30%
↳ maybe i'll go investigate, since i don't know what it is

if something is in shadow, then we can get deduction in the percentages

more: you get a 20% boost to people detecting you

Game AI Pro 2 Chapter 28

The point is that AI developers try to combine the concepts of seeing something and being aware of what we see into a single concept modeled essentially using vision shapes and a timer. On *Blacklist*, when the player is inside of a vision shape of the NPC, unobstructed and lit, a timer kicks off, which is scaled based on the distance to the player, lightness of the player, NPC state, etc., and when that timer reaches 0 (or the progress bar is full), the NPC immediately does two things: he *perceives* the player and becomes *aware* of him as a threat. This abstraction forces us to compromise: we need to define the area where the NPC should perceive *and* become aware of the player within some reasonable time.

Getting back to the vision cone, the reason a cone doesn't model vision at a distance well is that a cone expands as it moves away from the NPC, which implies that as things get further away from the NPC in the direction he is looking, he tends to perceive, and become aware of, things that are laterally farther away. Up to a certain distance that is plausible, but past that distance, it doesn't provide reasonable behavior. If the player is far enough away and off to one side, we don't want the NPCs to become aware of them at all, even though it seems they should still be perceiving him within their field of view. This insight caused us to first replace cones with boxes for vision at a distance; this is the solution we used on *Splinter Cell Conviction*. On *Blacklist*, we refined it further and replaced our standard boxes with coffin-shaped boxes (Figure 28.3) that expand up to a point like a cone and then start to contract as they continue to move further away from the NPC, which gives us the effect we want.

Figure 28.3

This debug drawing represents the detection shapes in *Blacklist*. The coffin-shaped box can be seen, as well as the standard vision cone that defines the area directly in front of the NPC where the player should get detected quickly. The most inclusive shape the player is in defines the range of the detection timer.

Learning (50 points)

(5pt) List 5 problems with Neural Networks. 1. _____ 2. _____

3. _____ 4. _____ 5. _____

(3pt) Given the "Draw the Utility" exam question, what broad category of ML using labeled data could replace it? _____
(4pt) Given the "Draw the Utility" exam question, how many features would it have? ____ How many outputs would it have? _____

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

(12pt) Use this sequence to determine the **probability** of the next move being a **Kick** for Bi-Gram, Tri-Gram, and Quad-Gram:
Punch, Kick, Punch, Kick, Punch, Uppercut, Punch, Punch, Kick, Punch, Punch, Kick, Punch, Punch, Kick, Punch

Bi-Gram: _____, Tri-Gram: _____, Quad-Gram: _____

(3pt) In player modeling, UsesSmokeGrenades is stored as what data type? _____

(3pt) In player modeling, AttacksOnTwoFronts, UsesSmokeGrenades, and CanDoTrickyJumps are examples of _____.

(5pt) In player modeling, when you make your Excel sheet to evaluate what to implement, what categories/columns should you score?

1. _____ 2. _____ 3. _____ 4. _____ 5. _____

(5pt) Using the moving average equation w/ alpha=0.2, what is the final average: (newest to oldest) 40, 80, 70, 60, 50, 0: Avg = ____

(5pt) Using the moving average history window of length 5, what is the final average of the previous question's sequence: Avg = ____

Terrain Analysis (5 points)

(5pt) Combine the influence of three maps using the geometric mean (A: [0, 0.5, 0.2], B: [1, 1, 1], C: [0.5, 0.6, 0.5]) _____

Randomness (8 points)

(5pt) Write a **ONE LINE** equation to generate Gaussian randomness in the range [-3,3] with roughly a **NORMAL** distribution (which has a standard deviation of 1). Use the random function RandReal() that returns a uniform random number in the range [0,1]:

_____

(2pt) Explain how to calculate a natural random offset for bullets aimed at the center of a target:

_____

(1pt) What was my seemingly odd belief about randomness in games? Rand() is _____

PCG (11 points)

(5pt) Name the five PCG techniques described in class: 1. _____ 2. _____

3. _____ 4. _____ 5. _____

(2pt) What is the new name given to modern transformer-based techniques to create content, like images and text? _____

(4pt) What are the two primary problems stopping large game companies from using ChatGPT or Midjourney?

1. _____ 2. _____

Learning (50 points)

(5pt) List 5 problems with Neural Networks. 1. overfitting 2. hard to find or fix bugs

3. require tons of training data 4. hard to program 5. network isn't complicated to learn the problem

(3pt) Given the "Draw the Utility" exam question, what broad category of ML using labeled data could replace it? supervised learning
(4pt) Given the "Draw the Utility" exam question, how many features would it have? 3 How many outputs would it have? 5

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

(12pt) Use this sequence to determine the **probability** of the next move being a **Kick** for Bi-Gram, Tri-Gram, and Quad-Gram:
Punch, Kick, Punch, Kick, Punch, Uppercut, Punch, Punch, Kick, Punch, Punch, Kick, Punch, Punch, Kick, Punch

Bi-Gram: 5/10, Tri-Gram: 1/4, Quad-Gram: 1/3

(3pt) In player modeling, UsesSmokeGrenades is stored as what data type? float

(3pt) In player modeling, AttacksOnTwoFronts, UsesSmokeGrenades, and CanDoTrickyJumps are examples of traits.

(5pt) In player modeling, when you make your Excel sheet to evaluate what to implement, what categories/columns should you score?

1. useful stats to AI 2. easy to program 3. show fun stats to player 4. collection difficulty 5. show useful stats to player

(5pt) Using the moving average equation w/ alpha=0.2, what is the final average: (newest to oldest) 40, 80, 70, 60, 50, 0: Avg = 42.5088

(5pt) Using the moving average history window of length 5, what is the final average of the previous question's sequence: Avg = 53.136

Terrain Analysis (5 points)

(5pt) Combine the influence of three maps using the geometric mean (A: [0, 0.5, 0.2], B: [1, 1, 1], C: [0.5, 0.6, 0.5]) [0.7937, 0.6694, 0.62996]

Randomness (8 points)

(5pt) Write a **ONE LINE** equation to generate Gaussian randomness in the range [-3,3] with roughly a **NORMAL** distribution (which has a standard deviation of 1). Use the random function RandReal() that returns a uniform random number in the range [0,1]:

(RandReal() + RandReal() + RandReal()) * (1.0 / 0x7FFF'FFFF)

(2pt) Explain how to calculate a natural random offset for bullets aimed at the center of a target:

Use uniform randomness to generate a random angle; use gaussian randomness to generate the distance from the center

(1pt) What was my seemingly odd belief about randomness in games? Rand() is not random enough but actually true randomness is not random to humans.

PCG (11 points)

(5pt) Name the five PCG techniques described in class: 1. Constructionist 2. simulation-based

3. grammars 4. optimization-based 5. constrainst-based

(2pt) What is the new name given to modern transformer-based techniques to create content, like images and text? Generative AI

(4pt) What are the two primary problems stopping large game companies from using ChatGPT or Midjourney?

1. Legal risk in generation (copyright) 2. Quality (free from copyrighted, embarrassing, offensive material)

Learning (50 points)

(5pt) List 5 problems with Neural Networks. 1. _____ 2. _____

3. _____ 4. _____ 5. _____

Questions

1. In supervised learning, the algorithm is trying to learn/generalize from labelled training data.
2. In unsupervised learning, the algorithm is trying to find clusters in the data.
3. What do you call supervised learning with a continuous output variable? regression
4. What do you call supervised learning with a discrete output variable? classification
5. What are 5 problems with neural nets?

1. overfitting — train too much on data
2. hard to find/fix bugs
3. require tons of training data
4. hard to program
5. network isn't complicated enough for it to learn your problem

(3pt) Given the "Draw the Utility" exam question, what broad category of ML using labeled data could replace it? _____

(4pt) Given the "Draw the Utility" exam question, how many features would it have? ____ How many outputs would it have? _____

Supervised Learning

→ we have to have it labelled

Learns the **association** between the inputs and outputs

Other tags:

Location
Square Footage
House Age
How Big the Lawn
Number of bedrooms
Number of bathrooms

254

Briefly, feature is input; label is output. This applies to both classification and regression problems.

A feature is one column of the data in your input set. For instance, if you're trying to predict the type of pet someone will choose, your input features might include age, home region, family income, etc. The label is the final choice, such as dog, fish, iguana, rock, etc.

Once you've trained your model, you will give it sets of new input containing those features; it will return the predicted "label" (pet type) for that person.

Share Improve this answer Follow edited Jun 26, 2019 at 16:43 answered Nov 30, 2016 at 22:36 Prune 77.5k • 14 • 62 • 83

1 So [age, home region, family income] would be "3 feature vectors". And in Keras, your NumPy array for your LSTM network would be [samples, time steps, 3] ? – naisanza Aug 20, 2017 at 11:55

3 @naisanza: that's certainly one possibility. I'm not familiar with Keras, but this high-level organization could certainly be the start of a valid implementation. – Prune Aug 22, 2017 at 17:41

is feature and variable the same thing? – Debadatta Apr 30, 2018 at 12:55

1 I would like to state that "label" is also dependent on the context: for example, for training a model, you will used "labelled" data. In this case, the label is the ground truth to which your output data is compared to. – N.Atanasov Jul 29, 2018 at 12:26

1 wow, great answer, thank you this clears up alot of lingering questions on this topic space. – Andrew Ray Jan 28, 2019 at 17:30

Supervised Learning

Continuous output prediction = **Regression**

1 2 3

3 features

5 outputs

Supervised Learning

Discrete output prediction = **Classification**

e.g. trying to tell if something is a dog or a cat

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

Movement

A.

B:

Error	Player	A	B	C	D	E
1	20	20^2	0	0	0	40^2
2	20	20^2	0	20^2	0	40^2
3	0	40^2	40^2	20^2	0	0
4	40	$(-40)^2$	20^2	20^2	40^2	80^2
		$2(400)$	400	$3(400)$	1600	don't even wanna know
		$+2(1600)$	1600			
		X	2000	1200		

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

a filter to penalize big errors

$0+45^2+45^2=4050$ $90^2+45^2+0=10125$ $90^2+0+0=8100$

preference for single strokes because it's easier to see

Example 1 Example 2 Example 3

why do we square the error

big errors get penalized more heavily

Square the individual error, then sum!!!

Player Move

squaring it allows the noises become more apparent if it's bigger

(12pt) Use this sequence to determine the **probability** of the next move being a **Kick** for Bi-Gram, Tri-Gram, and Quad-Gram:
Punch, Kick, Punch, Kick, Punch, Uppercut, Punch, Punch, Kick, Punch, Punch, Kick, Punch, Punch, Kick, Punch

Punch Kick Punch Kick Punch Uppercut Punch
5 4 1

Punch Kick Punch Punch Kick Punch Punch
3 3 2 2 1

Kick Punch
1

Kick: 5
Punch: 4
Uppercut: 1

5/9

Punch Kick Punch Kick Punch Uppercut Punch
1 1

Punch Kick Punch Punch Kick Punch Punch
1

Kick Punch

Kick: 1
Punch: 2
Uppercut: 1

1/4

Punch Kick Punch Kick Punch Uppercut Punch
4 3 1 1

Punch Kick Punch Punch Kick Punch Punch
2 1

Kick Punch

don't use keys

Kick: 1
Punch: 1
Uppercut: 1

1/4

Predict the Next Move

Kick, Punch, Uppercut, Punch, Uppercut, Kick, Punch, Uppercut, Kick, Kick,
Punch, Punch, Kick, Uppercut, Uppercut, Kick, Punch, Punch, Kick, Punch

Uni-gram, Bi-Gram, Tri-Gram, Quad-Gram

- Uni-gram: Punch (7 kick, 8 punch, 5 uppercut)

7/20 8/20 5/20

punch is next

Predict the Next Move

Kick, Punch, Uppercut, Punch, Uppercut, Kick, Punch, Uppercut, Kick, Kick,
Punch, Punch, Kick, Uppercut, Uppercut, Kick, Punch, Punch, Kick, Punch, ?

key

- Bi-Gram: Uppercut (2 kick, 2 punch, 3 uppercut)

2/7 2/7 3/7

uppercut next

Predict the Next Move

Kick, Punch, Uppercut, Punch, Uppercut, Kick, Punch, Uppercut, Kick, Kick,
Punch, Punch, Kick, Uppercut, Uppercut, Kick, Punch, Punch, Kick, Punch, ?

keys

- Tri-Gram: Punch, Uppercut (2 punch, 2 uppercut)

0 kicks

1/7 2/7 2/7

equal chance

(3pt) In player modeling, UsesSmokeGrenades is stored as what data type? _____

(3pt) In player modeling, AttacksOnTwoFronts, UsesSmokeGrenades, and CanDoTrickyJumps are examples of _

Player Modeling

We call the recorded values **traits** of the player

Traits are ideally in the range [0,1]

Brainstorm some RTS Traits

- preferences for how they gather etc
- economy
- the rate they generate stuff and then lose it
- frequency of sending units to fog of war (scouting)

RTS traits

- AvoidsMovingThroughEnemyChokepoints
- AttacksOnTwoFronts
- PrefersGroundForces

Brainstorm some FPS Traits

- favourite place?
- accuracy on shooting
- knife engagement kill
- close/near/far preferment ranges

FPS traits

- UsesSmokeGrenades
- AlwaysRuns
- CanDoTrickyJumps

(5pt) In player modeling, when you make your Excel sheet to evaluate what to implement, what categories/columns should you score?

1. _____ 2. _____ 3. _____ 4. _____ 5. _____

You wanna make an excel

Trait	Useful to AI	Easy to Program	Accuracy	Show to Player	Tuning for Game
Likes to run	4/5	5/5	5/5	2/5	1/5
Fast	0/5	5/5	0/5	5/5	2/5

Tips	Monetization
4/5	4/5
0/5	5/5

You could get NPCs to talk about it.

take traits and plot them

(5pt) Using the moving average equation w/ alpha=0.2, what is the final average: (newest to oldest) 40, 80, 70, 60, 50, 0: Avg = ____

(5pt) Using the moving average history window of length 5, what is the final average of the previous question's sequence: Avg = ____

$$\begin{array}{r} 24 \\ 7 \\ \hline 31 \end{array} \qquad \begin{array}{r} \times \quad 8 \\ \hline 43 \cdot 136 \end{array} \qquad \begin{array}{r} 31 \\ 5.392 \end{array}$$

Moving Average Equation

what neural networks do with huge amounts of data

any one value doesn't change the average too much like outliers and noises don't affect it much — more resilient

$$\text{newAvg} = \alpha \cdot \text{newObservation} + (1 - \alpha) \cdot \text{avg}$$

$$\alpha = 0.01 \qquad 5.05 = 0.01 \cdot 10.0 + (0.99) \cdot 5.0$$

$$\alpha = 0.10 \qquad 5.50 = 0.10 \cdot 10.0 + (0.90) \cdot 5.0$$

$$\alpha = 0.50 \qquad 7.50 = 0.50 \cdot 10.0 + (0.50) \cdot 5.0$$

$\alpha = 0.2 \qquad (1-\alpha) = 0.8$

$$\begin{array}{r} 1\ 31 \\ 5.24 \\ \times \quad 8 \\ \hline 41.92 \end{array} \qquad 48/10 = 4.8 \qquad \begin{array}{r} +6\ 4.8 \\ \times \quad 8 \\ \hline 38.4 \end{array}$$

1 $\quad 40 \quad$ —— $\quad 40$

2 $\quad 80 \quad$ —— $\quad 0.2 \times 80 + 0.8 \times 40 = 16 + 32 = 48$

3 $\quad 70 \quad$ —— $\quad 0.2 \times 70 + (0.8)(48) = 14 + 38.4 = 52.4$

4 $\quad 60 \quad$ —— $\quad 0.2 \times 60 + (0.8)(52.4) = 12 + 41.92 = 53.92$

5 $\quad 50 \quad$ —— $\quad 0.2 \times 50 + (0.8)(53.92) = 10 + 43.136 = 53.136$

6 $\quad 0 \quad$ —— $\quad 0 + (0.8)(53.136) = 42.5088$

$$\begin{array}{r} 2\ 5.1\ 3\ 1\ 3\ 6 \\ \times \quad 8 \\ \hline 42.5088 \end{array}$$

Terrain Analysis (5 points)

(5pt) Combine the influence of three maps using the geometric mean (A: [0, 0.5, 0.2], B: [1, 1, 1], C: [0.5, 0.6, 0.5]

Geometric Mean

Correct way to combine utility or influence values

$$\sqrt[n]{x_1 x_2 \dots x_n}$$

VALUES	MULTIPLIED	MEAN	GEOMETRIC MEAN
0.9	0.9	0.9	0.9
0.9, 0.9	0.81	0.9	0.9
0.9, 0.9, 0.9, 0.9	0.6561	0.9	0.9
0.1, 0.1, 0.1, 0.9	0.0009	0.3	0.173
0.8, 0.5, 0.2	0.08	0.5	0.43
1.0, 1.0, 1.0	1	1	1
0, 1.0, 1.0	0	0.667	0

doesn't zero it out

	x	y	z
A	0	0.5	0.2
B	1	1	1
C	0.5	0.6	0.5
	0 ~~0.5~~	0.30	~~0.25~~ 1

$\sqrt[3]{x}$

0.7937	0.6694	0.62996

Combine Layers: Combat Desirability Layer

but multiplying is twice bad the more you multiply the number gets smaller

openness layer	1	.8	.6	.4	.2	0
*						
occupancy layer	.6	.8	1	1	1	1
*						
static cover layer	0	1	1	1	1	1
=						
desirability layer	0	.6	.6	.4	.2	0

are high values good? or bad?

Randomness (8 points)

(5pt) Write a **ONE LINE** equation to generate Gaussian randomness in the range [-3,3] with roughly a **NORMAL** distribution (which has a standard deviation of 1). Use the random function RandReal() that returns a uniform random number in the range [0,1]:

(RandReal() + RandReal() + RandReal()) * (1.0 / 0x7FFF'FFFF)

Gaussian Randomness Generator

```
static unsigned long seed = 61829450;

double gaussrand(void)
{
    double sum = 0;

    for(int i=0; i<3; i++)
    {
        // Uses an xorshift PRNG
        unsigned long hold = seed;
        seed^=seed<<13;
        seed^=seed>>17;
        seed^=seed<<5;
        long r = hold+seed;
        sum += (double)r * (1.0/0x7FFFFFFF);
    }

    // Returns [-3.0,3.0] (66.7%-95.8%-100%)
    return sum;
}
```

rand

18995614

$sum += r_1 \times l$

$sum += r_2 \times l$

$sum += r_3 \times l$

$sum = r_1 \times l + r_2 \times l + r_3 \times l$

$= l(r_1 + r_2 + r_3)$

(2pt) Explain how to calculate a natural random offset for bullets aimed at the center of a target:

(1pt) What was my seemingly odd belief about randomness in games? Rand() is _

Is Randomness too Random?

Is the rand() function sufficiently random?
- What does rand() return?
- What other options exist?

Is rand() random enough for your game?
Is rand() too random for your game?

Examples
- 5 enemy spawn points
- Enemy has 10% chance of direct hit on player
- Random idle animation

Appearance of Randomness

True randomness doesn't appear random to humans!
- Supported by psychology researchers at Stanford and University of Massachusetts

Can we filter rand() to get rid of anomalies?
- Does this destroy the randomness?
- What anomalies?

PCG (11 points)

(5pt) Name the five PCG techniques described in class: 1. _____ 2. _____

3. _____ 4. _____ 5. _____

PCG Approaches

1. Constructionist
2. Simulation-Based
3. Grammars
4. Optimization-Based
5. Constraint-Driven

(2pt) What is the new name given to modern transformer-based techniques to create content, like images and text? _____

(4pt) What are the two primary problems stopping large game companies from using ChatGPT or Midjourney?

1. _____ 2. _____

Disruptive technology in games

Problem #1: Legal risk in generation

1. Generated content cannot violate copyright

ARTIFICIAL INTELLIGENCE / TECH / LAW

Getty Images is suing the creators of AI art tool Stable Diffusion for scraping its content

/ Getty Images claims Stability AI 'unlawfully' scraped millions of images from its site. It's a significant escalation in the developing legal battles between generative AI firms and content creators.

By James Vincent, a senior reporter who has covered AI, robotics, and more for The Verge since 2012. | Jan 17, 2023, 2:45 AM PST | 23 Comments / 26 New

An image created by Stable Diffusion showing a recreation of Getty Images' enforcement. Image: The Verge / Stable Diffusion

Problem #2: Quality

1. Is the generated content of sufficient quality?
2. How much human intervention is needed to inspect or improve the content?
3. Can the content be generated on-demand with no human intervention and be acceptable?
   - Copyright free (no violations of logos)
   - Free from embarrassment (no obscene images/language)
   - Free from being offensive (not racist, sexist, homophobic, etc.)

Problem #3: Logistics

1. Is there enough space to store the pre-generated content?
   - In storage
   - In memory
2. Can the pre-generated content be downloaded on-the-fly?
   - What is the cost of hosting and for how long?
   - Can it be delivered sufficiently fast while player is playing?
3. Can a service generate the content on-the-fly?
   - What is the cost of compute? (this could be very high)
   - Can it be delivered sufficiently fast while player is playing?

Problem #4: Cost

1. What is the cost to buy it from another company?
   - Better to buy commodity technology
2. What is the cost to build it ourselves?
   - Better to build competitive advantages if it is not a commodity
   - What is the cost to organize our own data for training?
3. Is the cost worth it to make our games better or faster?

Learning (50 points)

(5pt) List 5 problems with Neural Networks. 1. _____ 2. _____

3. _____ 4. _____ 5. _____

(3pt) Given the "Draw the Utility" exam question, what broad category of ML using labeled data could replace it? _____
(4pt) Given the "Draw the Utility" exam question, how many features would it have? ____ How many outputs would it have? _____

(5pt) In gesture recognition with K-Nearest Neighbors, if the player did the move [20, 20, 0, 40], circle the best matching gesture:

A. [0, 40, 40, 0] B. [20, 20, 40, 20] C. [20, 0, 20, 20] D. [20, 20, 0, 0] E. [-20, -20, 0, -40]

(12pt) Use this sequence to determine the **probability** of the next move being a **Kick** for Bi-Gram, Tri-Gram, and Quad-Gram:
Punch, Kick, Punch, Kick, Punch, Uppercut, Punch, Punch, Kick, Punch, Punch, Kick, Punch, Punch, Kick, Punch

Bi-Gram: _____, Tri-Gram: _____, Quad-Gram: _____

(3pt) In player modeling, UsesSmokeGrenades is stored as what data type? _____

(3pt) In player modeling, AttacksOnTwoFronts, UsesSmokeGrenades, and CanDoTrickyJumps are examples of _____.

(5pt) In player modeling, when you make your Excel sheet to evaluate what to implement, what categories/columns should you score?

1. _____ 2. _____ 3. _____ 4. _____ 5. _____

(5pt) Using the moving average equation w/ alpha=0.2, what is the final average: (newest to oldest) 40, 80, 70, 60, 50, 0: Avg = ____

(5pt) Using the moving average history window of length 5, what is the final average of the previous question's sequence: Avg = ____

Terrain Analysis (5 points)

(5pt) Combine the influence of three maps using the geometric mean (A: [0, 0.5, 0.2], B: [1, 1, 1], C: [0.5, 0.6, 0.5]) _____

Randomness (8 points)

(5pt) Write a **ONE LINE** equation to generate Gaussian randomness in the range [-3,3] with roughly a **NORMAL** distribution (which has a standard deviation of 1). Use the random function RandReal() that returns a uniform random number in the range [0,1]:

_____

(2pt) Explain how to calculate a natural random offset for bullets aimed at the center of a target:

_____

(1pt) What was my seemingly odd belief about randomness in games? Rand() is _____

PCG (11 points)

(5pt) Name the five PCG techniques described in class: 1. _____ 2. _____

3. _____ 4. _____ 5. _____

(2pt) What is the new name given to modern transformer-based techniques to create content, like images and text? _____

(4pt) What are the two primary problems stopping large game companies from using ChatGPT or Midjourney?

1. _____ 2. _____

Draw Behavior Trees (60 points)

Directions: You can use any number of Selectors, Sequencers, or Parallel nodes. Parallel nodes Succeed if any children Succeed and Fail if any children Fail. You can use any number of Always Succeed, Always Fail, Rerun until Success, Rerun until Fail, or Invert decorator nodes. For all other nodes, you can only use the nodes given to you in the question.

(10pt) Draw a Behavior Tree that endlessly patrols between three points: A, B, and C. Use multiple "Move To _" Leaf nodes which will move toward the point you give it, will return InProgress if moving toward the point, and will return Success if it reaches the point (it will never Fail).

(50pt total) Draw a Behavior Tree that performs the previous endless patrolling behavior with "Move To _" nodes (10pt), but if it sees an enemy at any time during its movement, it immediately moves to point X (20pt) and stays there forever (20pt). Use the Leaf node "See Enemy?" that returns Success if it sees an enemy and returns Fail if it does not.

Draw Behavior Trees (60 points)

Directions: You can use any number of Selectors, Sequencers, or Parallel nodes. Parallel nodes Succeed if any children Succeed and Fail if any children Fail. You can use any number of Always Succeed, Always Fail, Rerun until Success, Rerun until Fail, or Invert decorator nodes. For all other nodes, you can only use the nodes given to you in the question.

(10pt) Draw a Behavior Tree that endlessly patrols between three points: A, B, and C. Use multiple "Move To _" Leaf nodes which will move toward the point you give it, will return InProgress if moving toward the point, and will return Success if it reaches the point (it will never Fail).

(50pt total) Draw a Behavior Tree that performs the previous endless patrolling behavior with "Move To _" nodes (10pt), but if it sees an enemy at any time during its movement, it immediately moves to point X (20pt) and stays there forever (20pt). Use the Leaf node "See Enemy?" that returns Success if it sees an enemy and returns Fail if it does not.

Behavior Tree

Procedural architecture

Meta-architecture

- Nodes can hold any architecture

→ you could make your own weird control flow

→ could have a planner! anything!

Modular

Represents a fixed plan

Cyclical but not obvious

Poor at interruption returning

Poor at state-based / modes