intelligent NPCs

a crash course on NPC AI

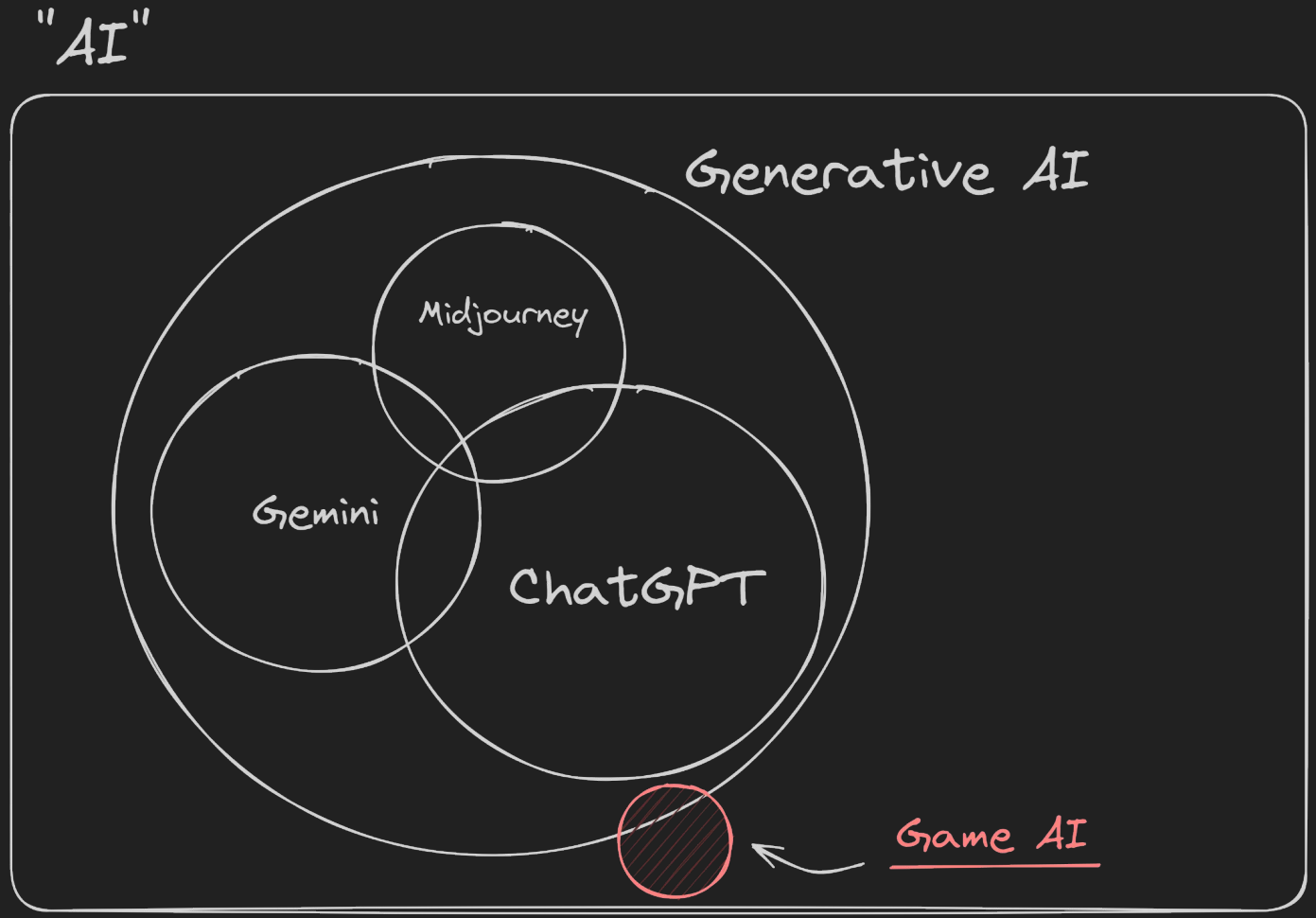
GJ Tiquia

Game Developer & Code Architect
@ 9Cat Studio

GitHub: @gjtiquia

Twitter/X: @gjtiquia

https://github.com/gjtiquia/IntelligentNPCs-Talk

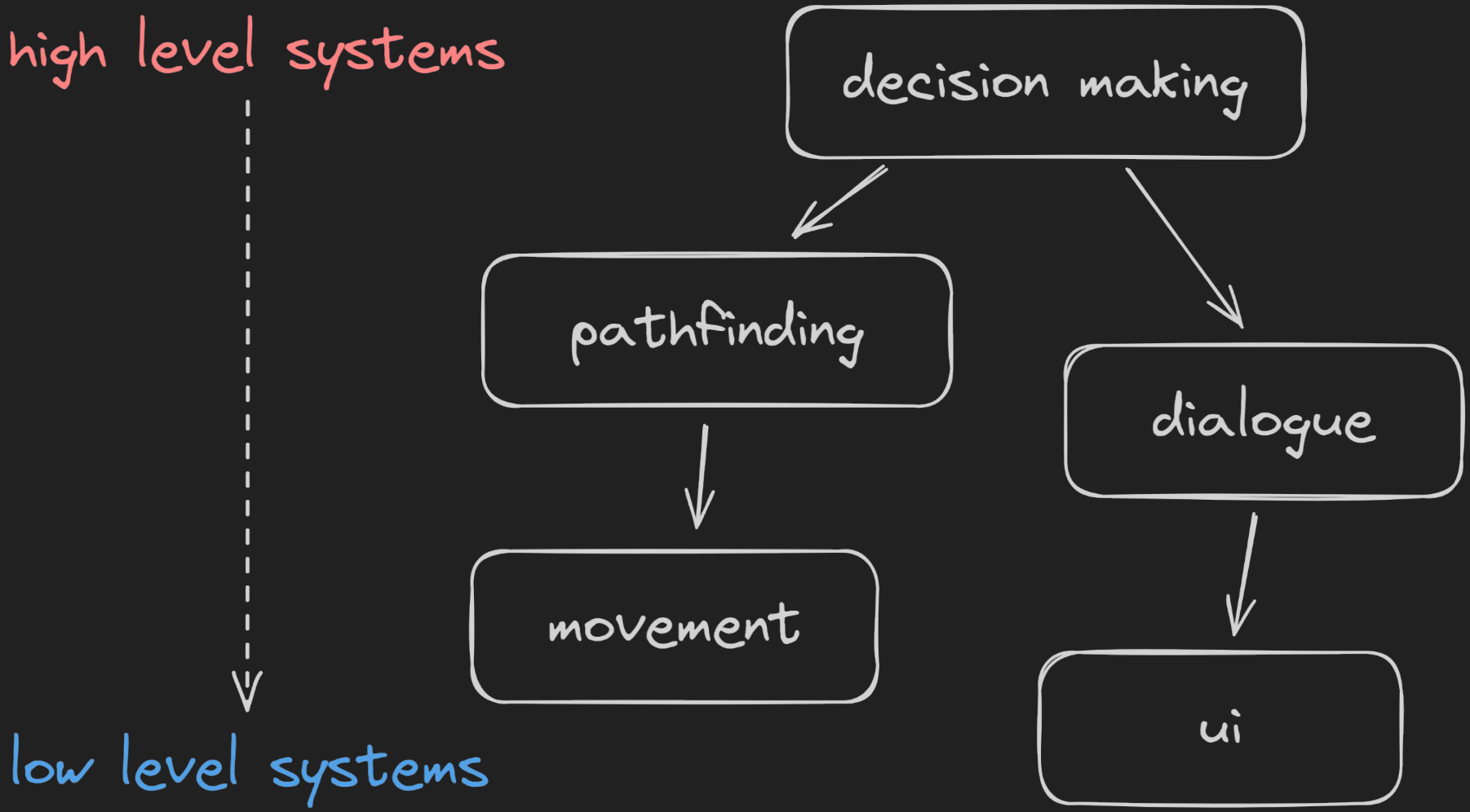


(a totally inaccurate sort-of-venn-diagram i guess)

brief introduction

Game AI

- “intelligent” vs “creative control”
- “10/10 perfect but boring” vs “dumb but cute”


high level systems
low level systems
decision making
pathfinding
movement
dialogue
ui

contents

decision making

- unstructured ai
- finite state machines (FSM)
- behavior trees (BT)
- goal-oriented action planning (GOAP)

pathfinding

- physics approach
- algorthmic approach

dialogue

“it depends”

- every game is unique
- 👍 have a HUGEEE toolbox
- 👎 use EVERY TOOL YOU HAVE
- simplicity > complexity

decision-making 101

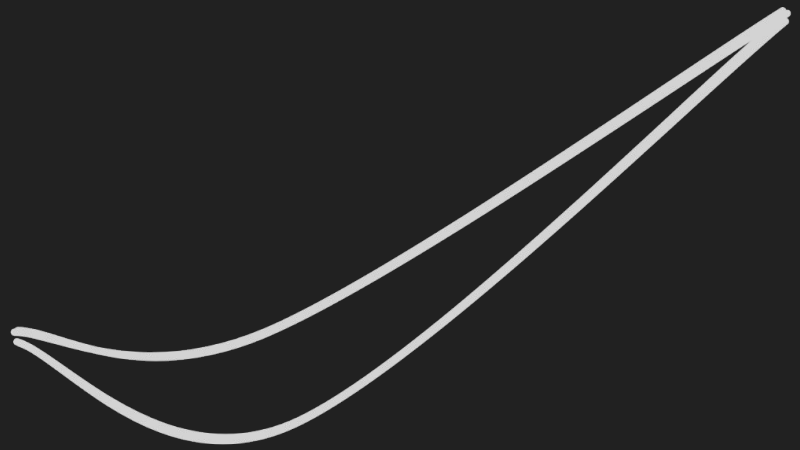
decision-making 101

enemy v1

- pursue player
- if in range, attack

unstructured ai

decision-making 101 - unstructured ai



you got this bro

just trust me bro

decision-making 101

enemy v1

- pursue player
- if in range, attack

movement script

- every frame
  - move to player

attack script

- every frame
  - if player in range
    - attack

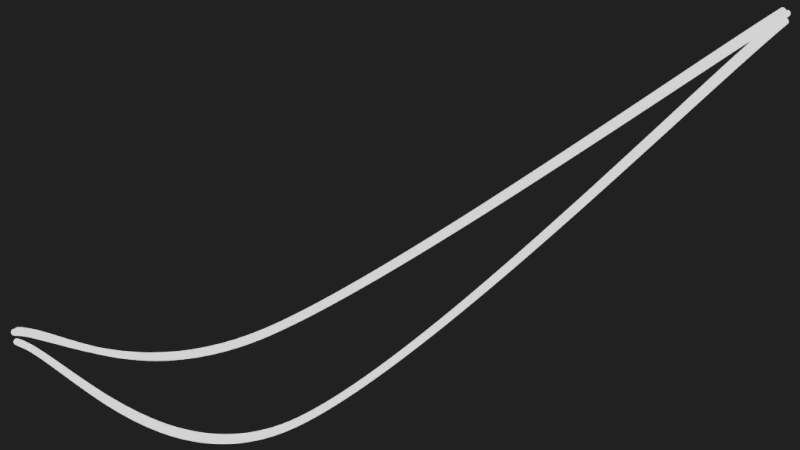
decision-making 101

enemy v2

- pursue player
- if in range, attack

- ++ if stunned,
  - stop moving
  - stop attacking

decision-making 101 - unstructured ai



you got this bro

just trust me bro

decision-making 101

enemy v2

- ++ if stunned,
  - stop moving
  - stop attacking

movement script

- ++ if *isStunned* - do nothing
- every frame
  - move to player

++ stun script

- if received damage, *isStunned* = true
- *isStunned* = false after stunDuration

attack script

- ++ if *isStunned* - do nothing
- every frame
  - if player in range
    - attack

decision-making 101

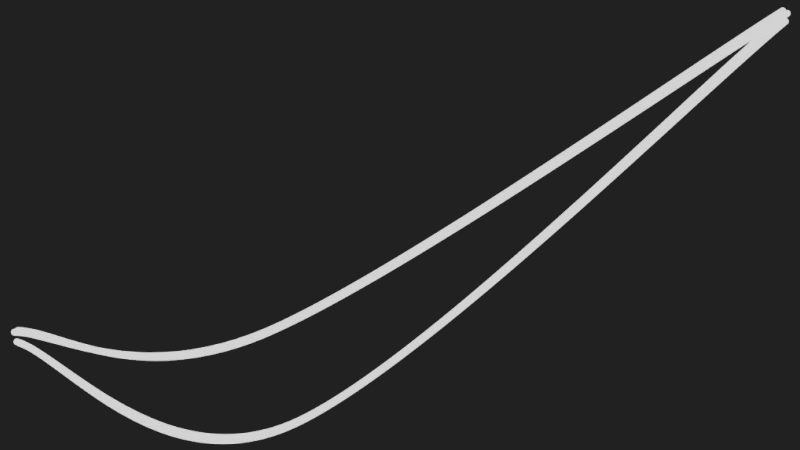
enemy v3

- ++ if no player in proximity
  - ++ wander
- ++ if have player in proximity
  - ++ pursue

- if in range,
  - attack

- if stunned,
  - stop moving
  - stop attacking

decision-making 101 - unstructured ai



you got this bro

just trust me bro

decision-making 101

enemy v3 ++ wander

++ wander script

- if in proximity
  - *isWandering* = true
- *isWandering* = false

stun script

- *isStunned* = true/false

movement script

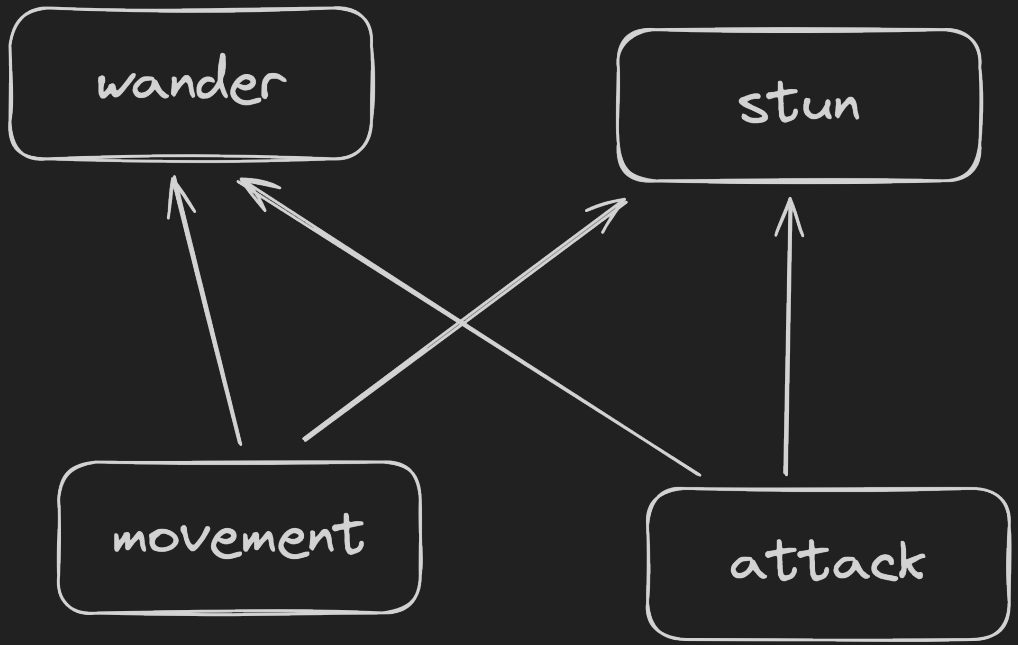
- if *isStunned* - do nothing
- ++ if *isWandering* - wander
- else
  - every frame - move to player

attack script

- if *isStunned* - do nothing
- ++ if *isWandering* - do nothing
- every frame
  - if player in range
    - attack

decision-making 101

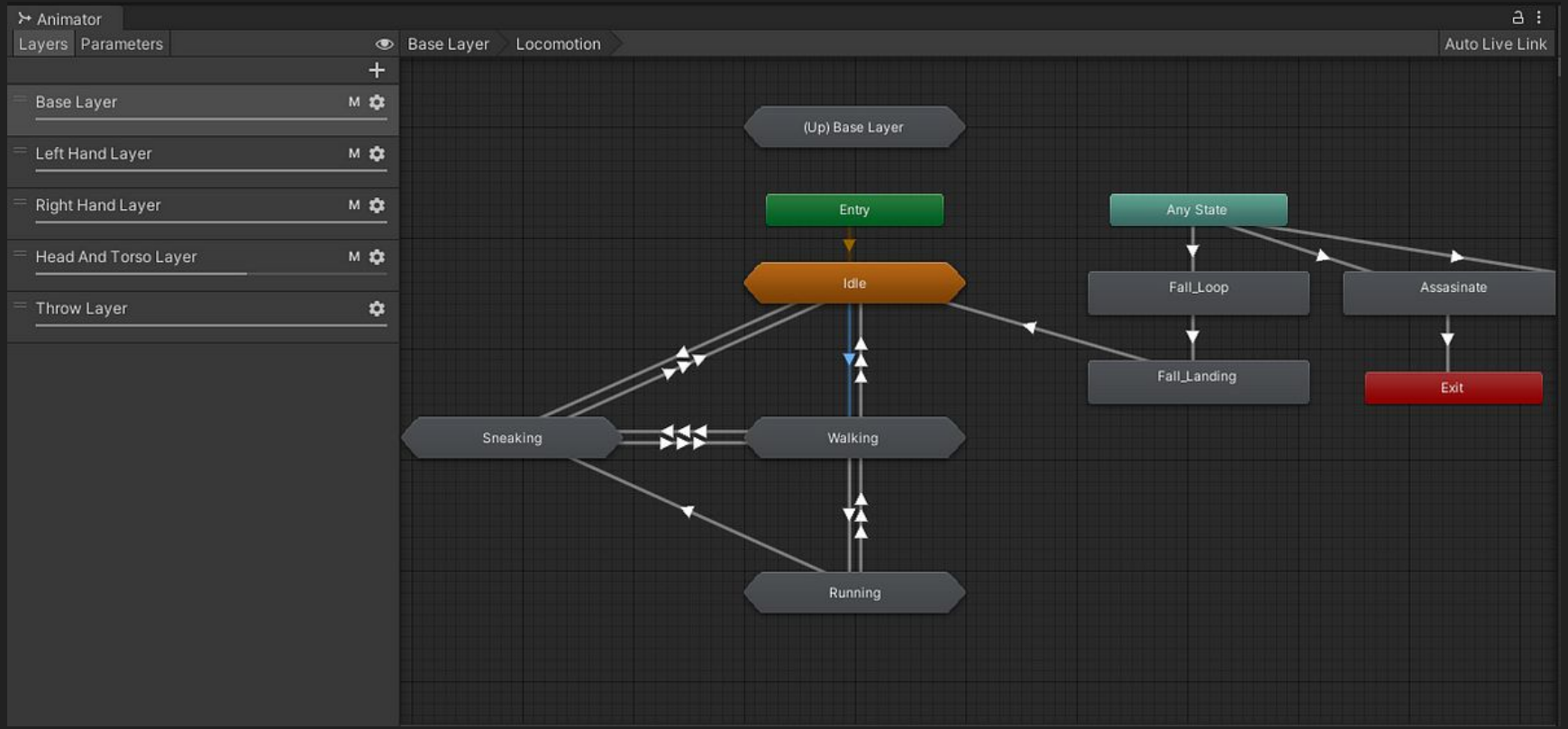
decision-making 101



(enemy v3 dependency graph)

finite state machines (FSM)

decision-making 101 - finite state machines (FSM)



decision-making 101 - finite state machines (FSM)

STATES

entry

switch up

off

on

switch down

TRANSITIONS

EVENTS

decision-making 101 - finite state machines (FSM)

OnEnter: light off

OnExit: (do nothing)

OnEnter: light up

OnExit: (do nothing)

decision-making 101

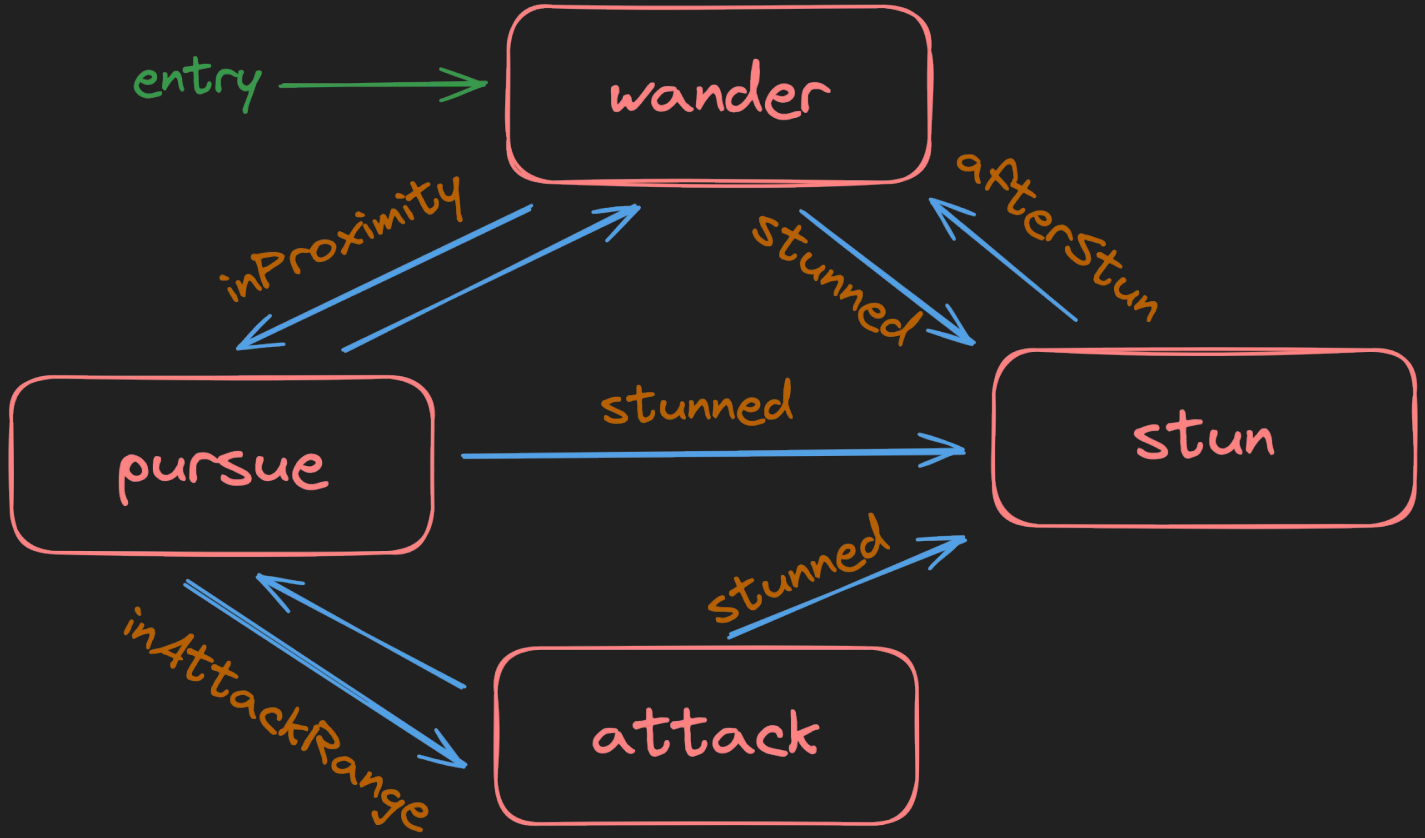
enemy v3

- ++ if no player in proximity
  - ++ wander
- ++ if have player in proximity
  - ++ pursue

- if in range,
  - attack

- if stunned,
  - stop moving
  - stop attacking

decision-making 101 - finite state machines (FSM)



decision-making 101 - finite state machines (FSM)

OnEnter: move randomly

OnExit: stop moving

OnEnter: move towards player

OnExit: stop moving

OnEnter: begin attack cycle

OnExit: stop attack cycle

OnEnter: play stun animation

OnExit: (do nothing)

decision-making 101 - finite state machines (FSM)

tools for popular game engines

as far as i know,
there isn't any popular tool for FSM Game AI for Unity/Unreal/Godot

often suggested to write your own from scratch, "its not that hard"

(debug-ability / analysis / iteration velocity says hi)

Official Unity Tutorial: https://learn.unity.com/project/finite-state-machines-1

⭐⭐⭐ Game Programming Patterns - State

https://gameprogrammingpatterns.com/state.html

(btw Concurrent/Hierarchical State Machines are a thing)

decision-making 101

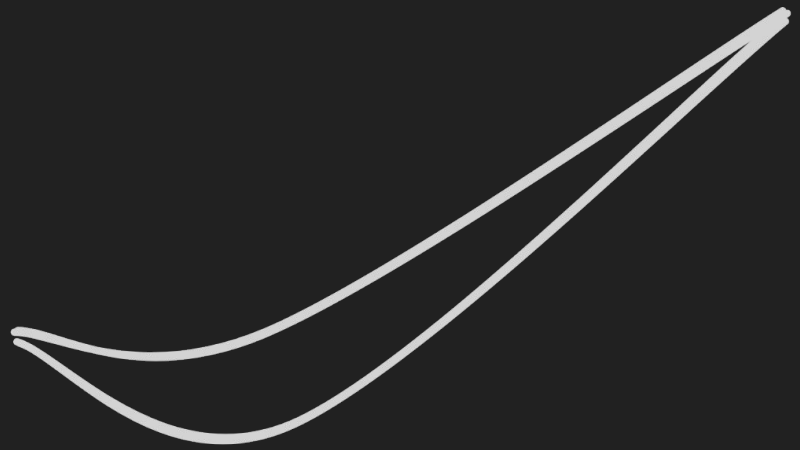
enemy v4

- if no player in proximity
  - wander
- if have player in proximity
  - pursue

- ++ if in ranged attack range,
  - ++ ranged attack
- ++ if in melee range,
  - ++ melee attack

- if stunned,
  - stop moving
  - stop attacking

decision-making 101 - unstructured ai



you got this bro

just trust me bro

decision-making 101

enemy v3 ++ wander

++ wander script

- if in proximity
  - *isWandering* = true
- *isWandering* = false

stun script

- *isStunned* = true/false

movement script

- if *isStunned* - do nothing
- ++ if *isWandering* - wander
- else
  - every frame - move to player

attack script

- if *isStunned* - do nothing
- ++ if *isWandering* - do nothing
- every frame
  - if player in range
    - attack

decision-making 101 - unstructured ai

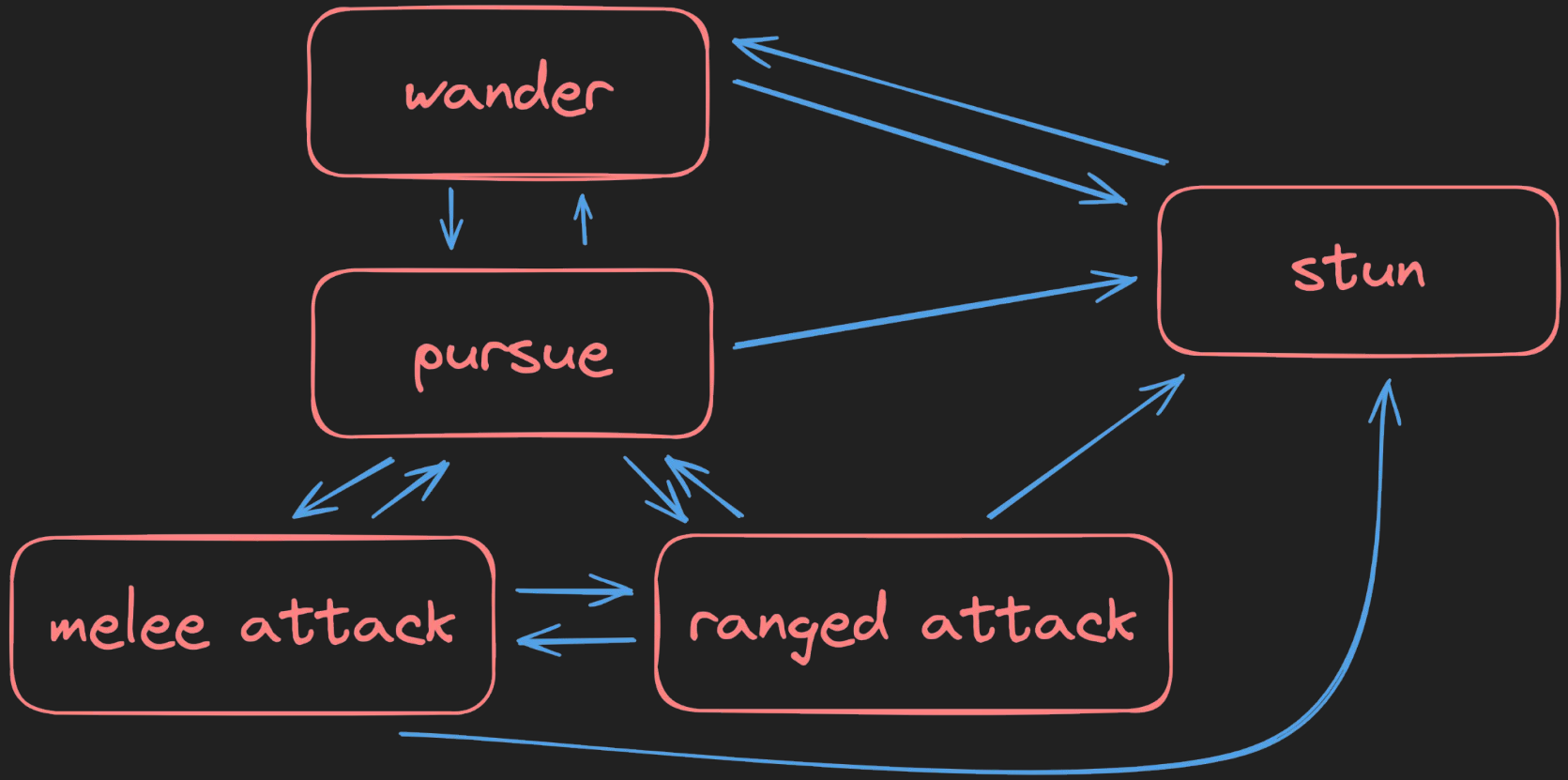
enemy v4

- ++ if in ranged attack range,
  - ++ ranged attack
- ++ if in melee range,
  - ++ melee attack

attack script

- if *isStunned* - do nothing
- if *isWandering* - do nothing
- every frame
  - ++ if in ranged attack range,
    - ++ ranged attack
  - ++ if in melee range,
    - ++ melee attack

decision-making 101 - finite state machines (FSM)



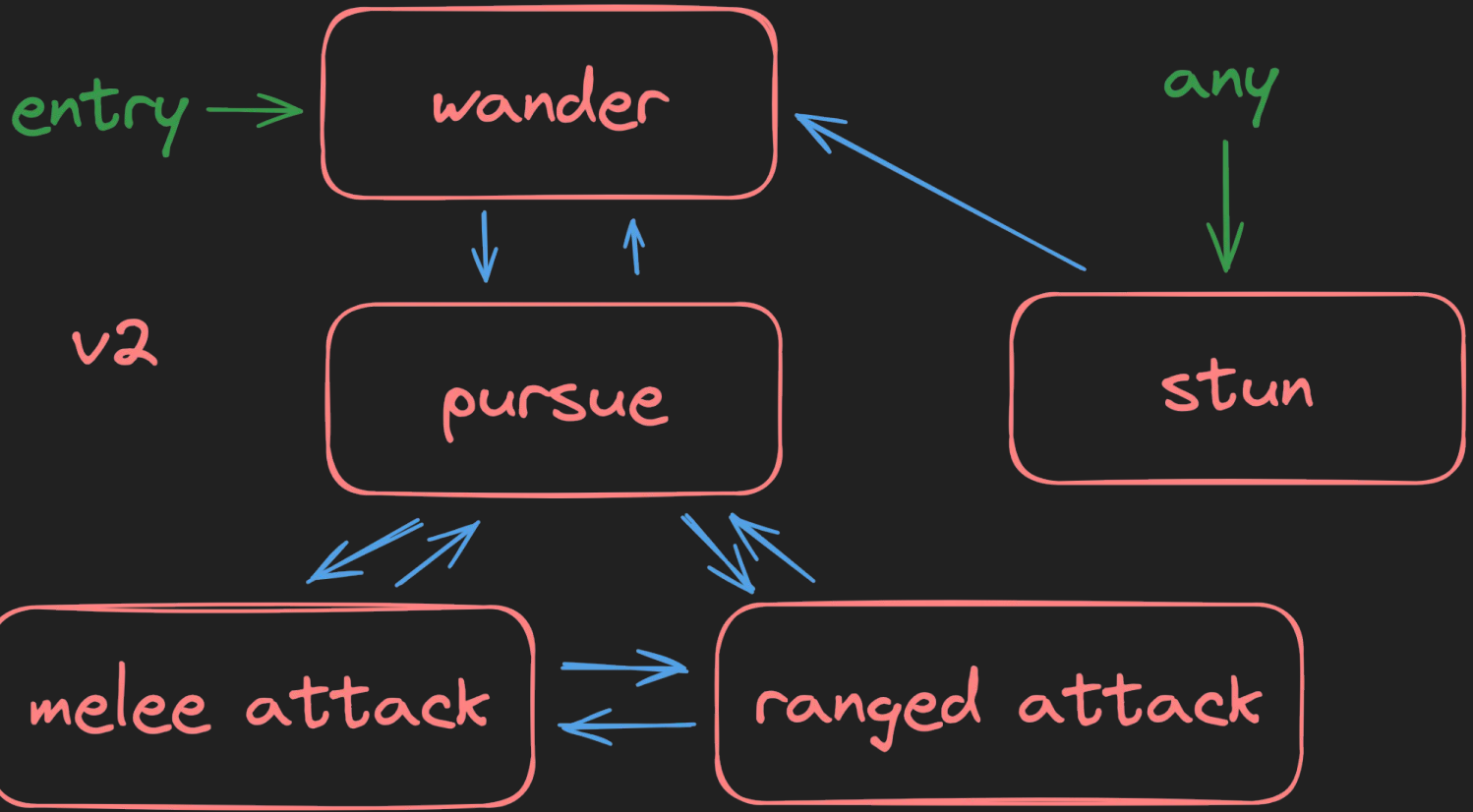
decision-making 101 - finite state machines (FSM)

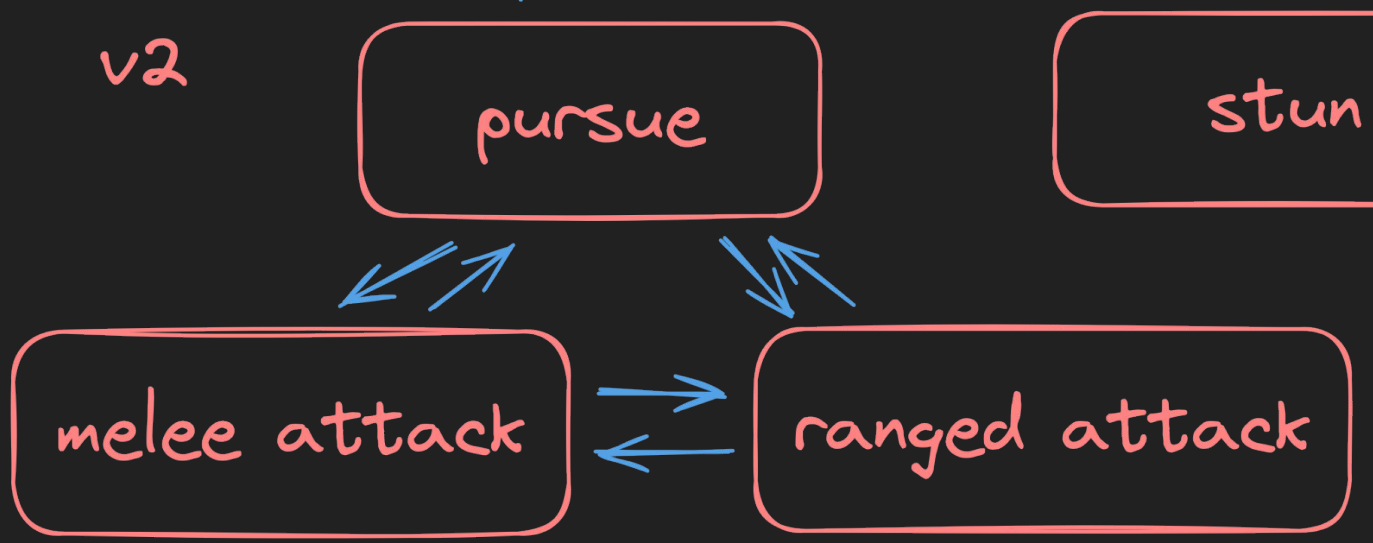
state explosion

reactivity

> go to every state from any state
> O(n^2) complexity

decision-making 101 - finite state machines (FSM)





- what if….
  - in melee range AND ranged range?
  - in two states at the same time?

behavior trees (BT)

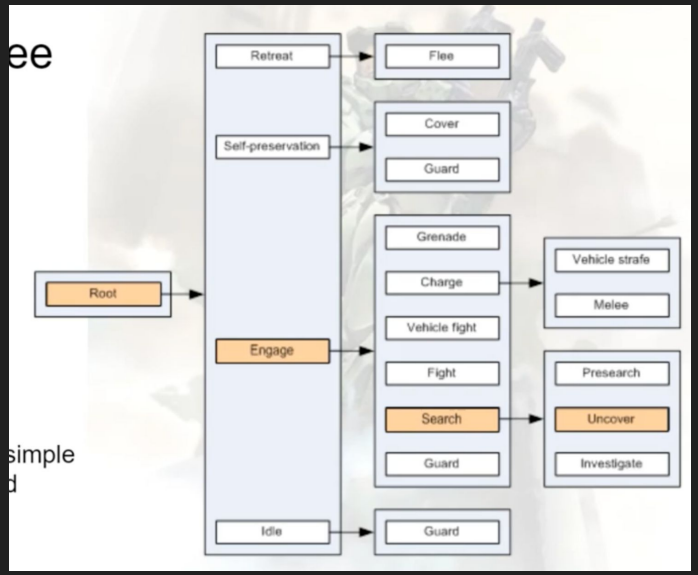
decision-making 101 - behavior trees (BT)

Managing Scalability in the Halo2 AI

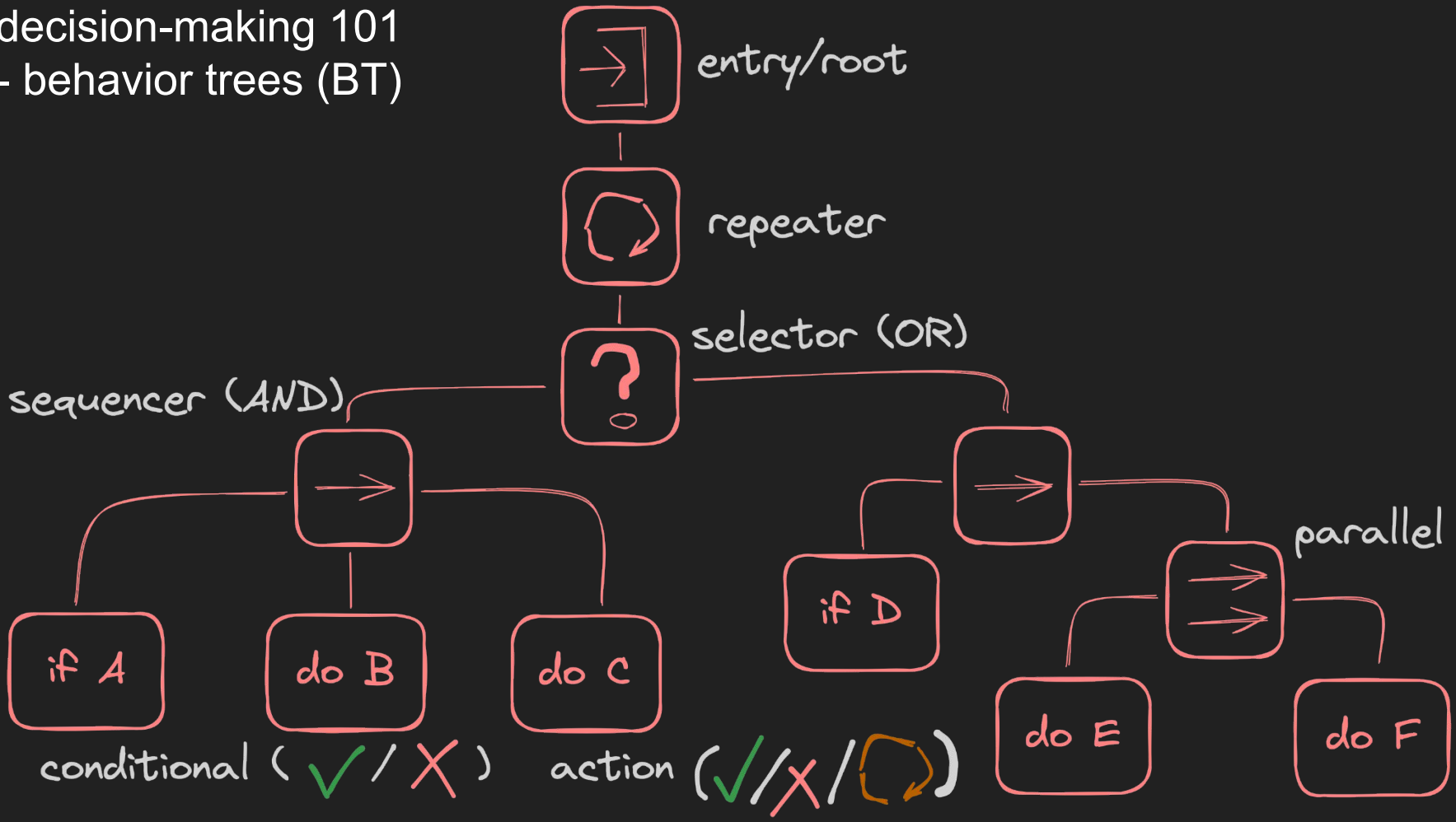
Damián Isla
Bungie Studios

GDC 2005

https://www.youtube.com/watch?v=m9W-hpxuApk&ab_channel=BungieHaloArchive



decision-making 101
- behavior trees (BT)



decision-making 101

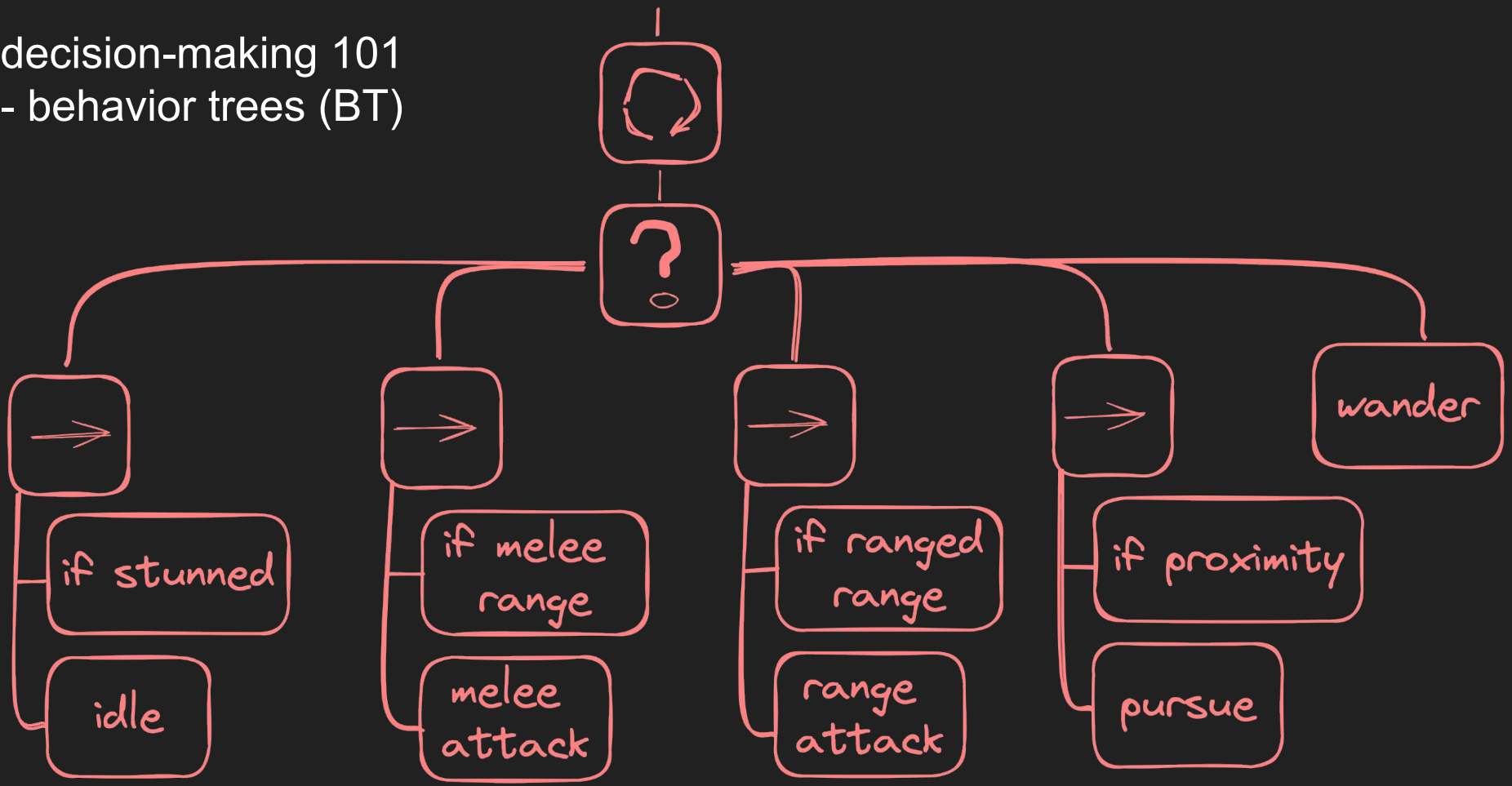
enemy v4

- if no player in proximity
  - wander
- if have player in proximity
  - pursue

- ++ if in ranged attack range,
  - ++ ranged attack
- ++ if in melee range,
  - ++ melee attack

- if stunned,
  - stop moving
  - stop attacking

decision-making 101
- behavior trees (BT)



decision-making 101 - behavior trees (BT)

tools for popular game engines

Unity

- paid assets
  - Behavior Designer (https://opsive.com/support/documentation/behavior-designer/overview/)
  - Arbor 3 (https://assetstore.unity.com/packages/tools/visual-scripting/arbor-3-fsm-bt-graph-editor-112239 _
- open source
  - https://github.com/Qriva/MonoBehaviourTree (206 stars)
  - https://github.com/yoshidan/UniBT (144 stars)
  - https://github.com/BDeshiDev/BTSM-Behavior-Tree-FSM-library-for-Unity (4 stars)

Unreal

- built-in!

Godot

- Beehave (https://github.com/bitbrain/beehave) (1.7k stars)
- LimboAI (https://github.com/limbonaut/limboai) (775 stars)

goal-oriented action planning (GOAP)

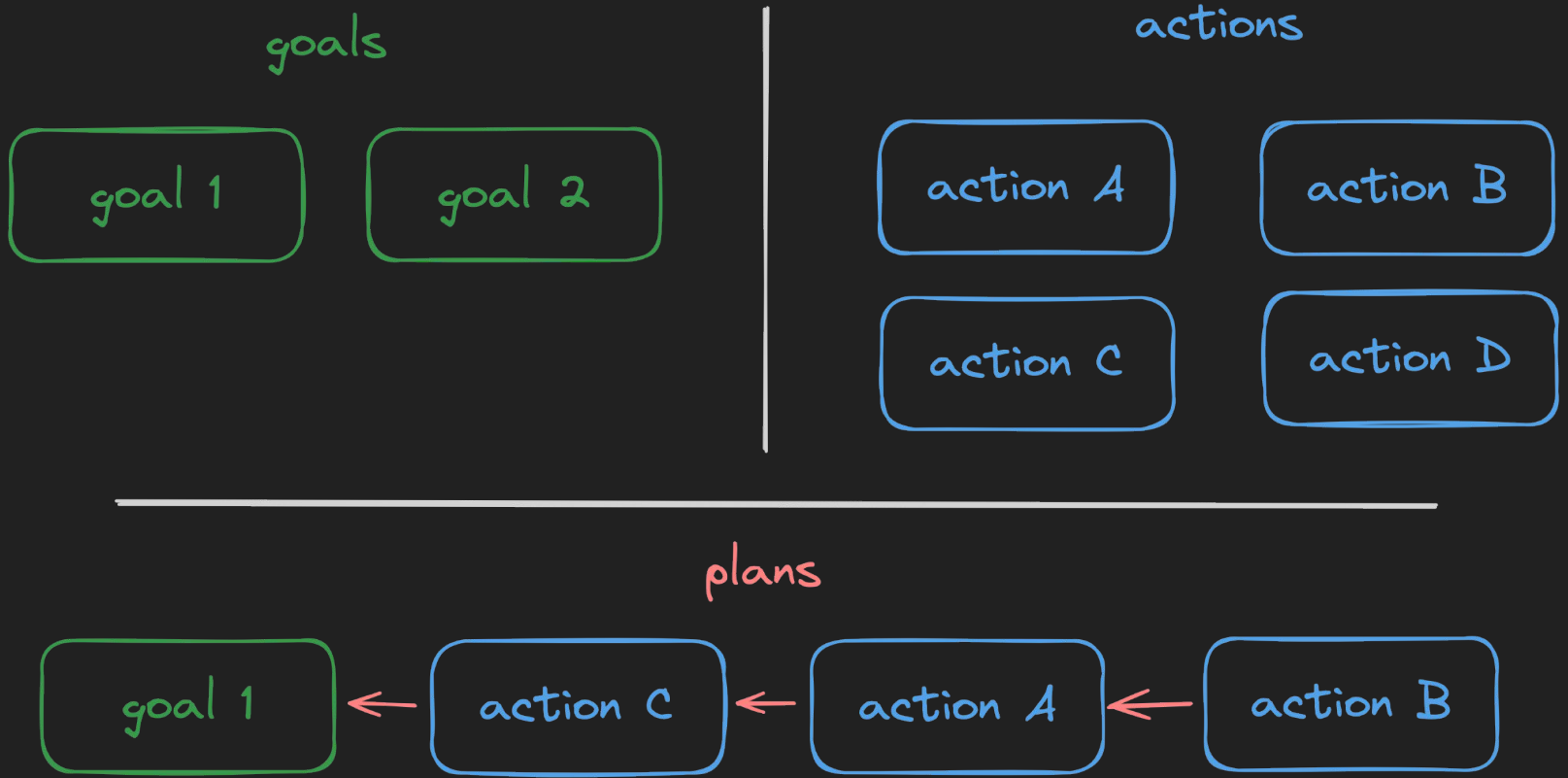
decision-making 101 - goal-oriented action planning (GOAP)

- Developed for F.E.A.R. in 2004
  - Jeff Orkin developed
  - Used in many Monolith titles :
    - F.E.A.R., F.E.A.R. 2
    - Condemned, Condemned 2
    - Middle-earth : Shadow of Mordor

GDC 2015

- https://www.youtube.com/watch?v=gm7K68663rA&ab_channel=GDC

decision-making 101 - goal-oriented action planning (GOAP)



decision-making 101

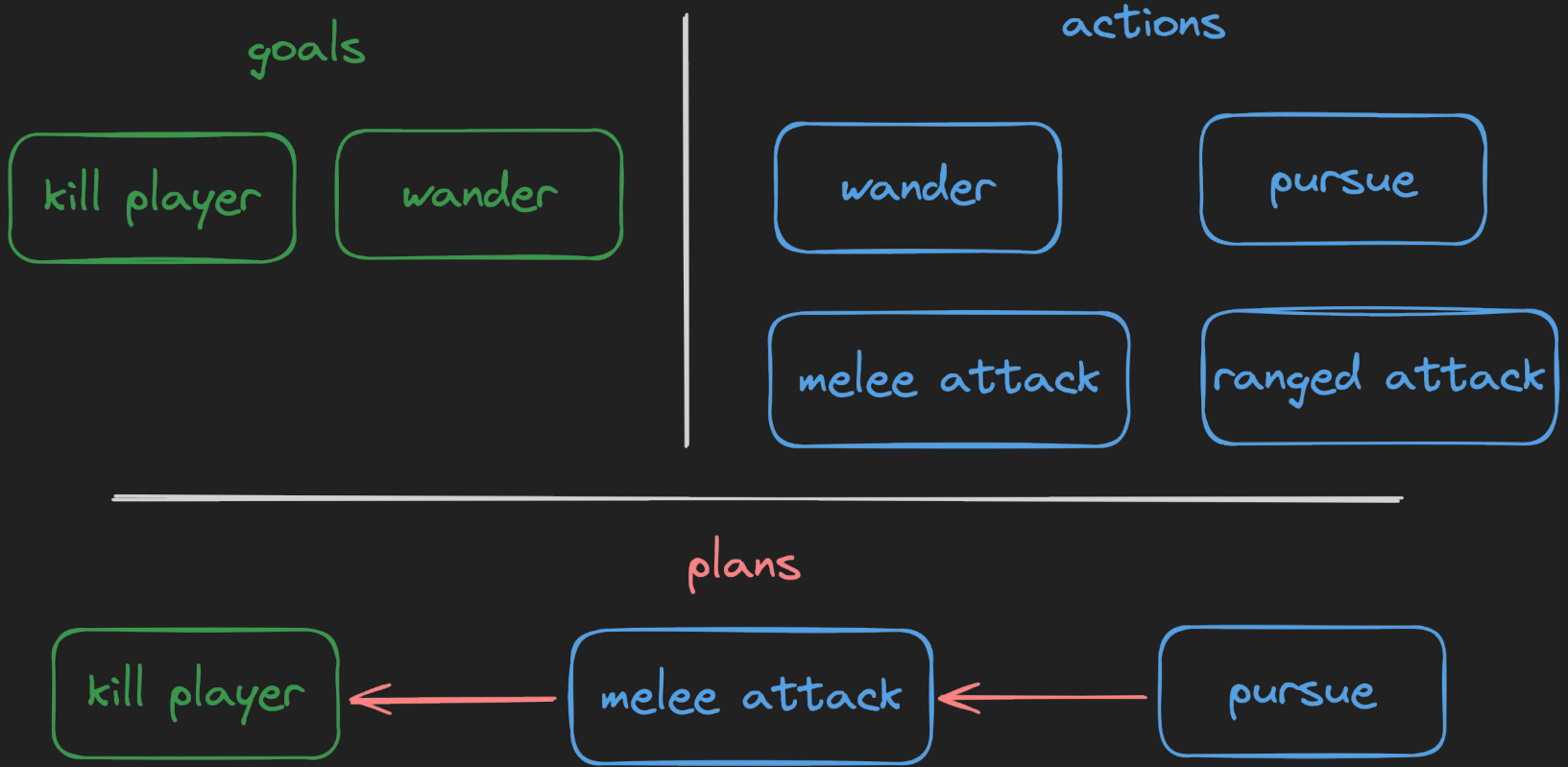
enemy v4

- if no player in proximity
  - wander
- if have player in proximity
  - pursue

- ++ if in ranged attack range,
  - ++ ranged attack
- ++ if in melee range,
  - ++ melee attack

- if stunned,
  - stop moving
  - stop attacking

decision-making 101 - goal-oriented action planning (GOAP)



decision-making 101 - goal-oriented action planning (GOAP)

tools for popular game engines

Unity

- write-your-own
  - official Unity tutorial: https://learn.unity.com/tutorial/the-goap-planner
- open source
  - https://github.com/crashkonijn/GOAP (953 stars)

Unreal

- paid asset
  - GOAP NPC
    https://www.unrealengine.com/marketplace/en-US/product/goap-npc-goal-oriented-action-planning-for-non-player-characters/

Godot

- (cant find…)

(btw Simple Hierarchical Ordered Planner (SHOP))
(btw Hierarchical Task Network Planning (HTN)

decision-making 101 - BTs vs GOAPs

BTs = Encode all possibilities in the tree

GOAPs = Specify the goals and actions, plan builds itself at runtime

use GOAPs if

- you want "smarter" ai
- too many possible action sequences
- you do not know at "design phase" the possible ways a character can act
- especially in games with a lot of user generated content

else, BTs works for 80% of games

https://www.davideaversa.it/blog/choosing-behavior-tree-goap-planning/

decision-making 101 - tips

decision-making 101 - tips

- any single action (eg. attack) is more complicated than u think
  - anticipation time? animation time? can cancel (knockback/stun)?
- plan ahead for friendly/hostile NPCs
  - actions work for both sides? or fundamentally different?
  - friendly NPC is dependent on player? hybrid?
- ⭐⭐⭐ simplicity > complexity
  - always first choose the simple solution
  - always simplify the graph (FSM/BT/GOAP)
  - delegate actions to lower-level systems
- performance concerns
  - BTs and GOAPs cannot handle large number of NPCs, heavy CPU load
  - consider compromising in game design for large number (no obstacles, simple attacks etc)

pathfinding

pathfinding

- a very common mid-low level system used together with decision-making systems

- how to get from A to B

- may/may not include obstacle avoidance
- may/may not consider the "shortest" path

pathfinding - physics approach

pathfinding - physics approach

https://www.youtube.com/watch?v=wC9iu7cuQjI&ab_channel=Challacade

pathfinding - physics approach

https://www.youtube.com/watch?v=wC9iu7cuQjI&ab_channel=Challacade

pathfinding - physics approach

https://www.youtube.com/watch?v=wC9iu7cuQjI&ab_channel=Challacade

pathfinding - physics approach

pros of physics approach

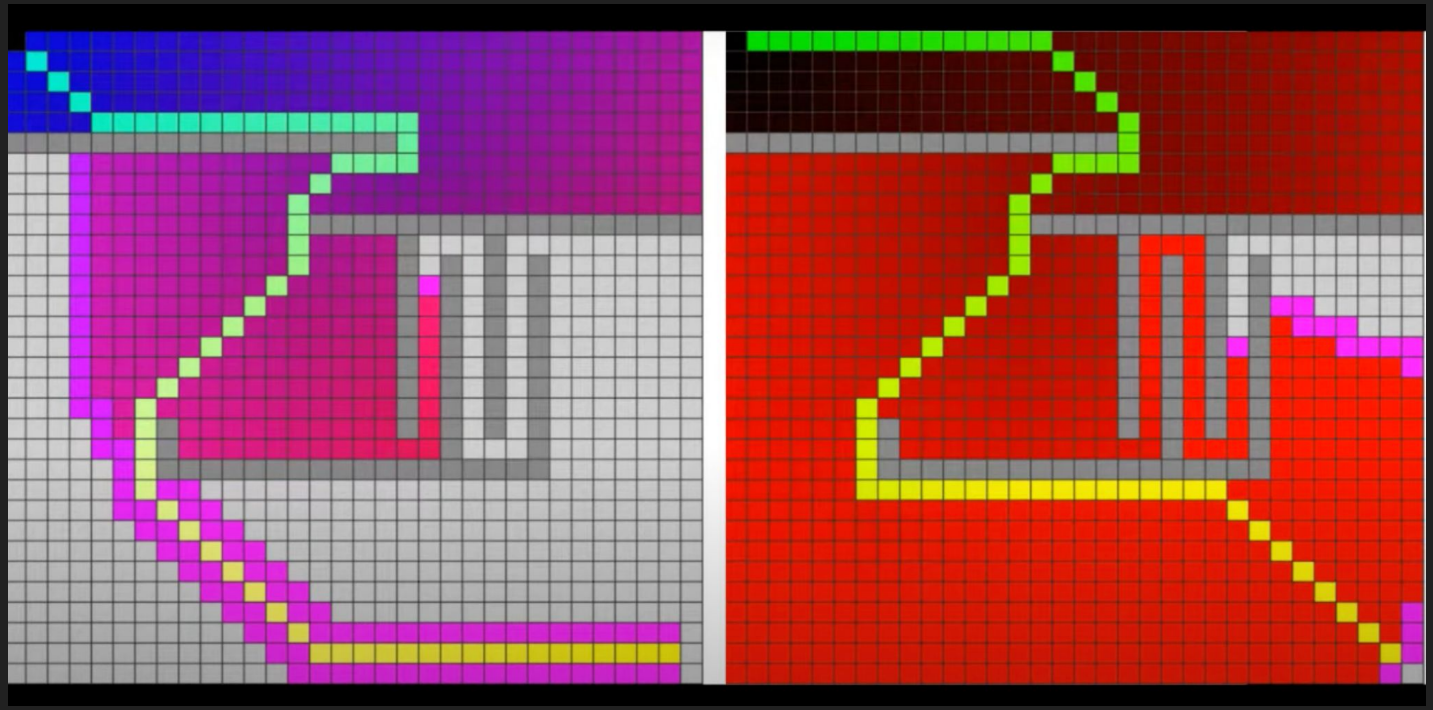
- simple
- great for large number of enemies

cons

- limited "intelligence"
    - obstacle avoidance
    - shortest path
    - tho can be compensated by game design

pathfinding - algorithmic approach

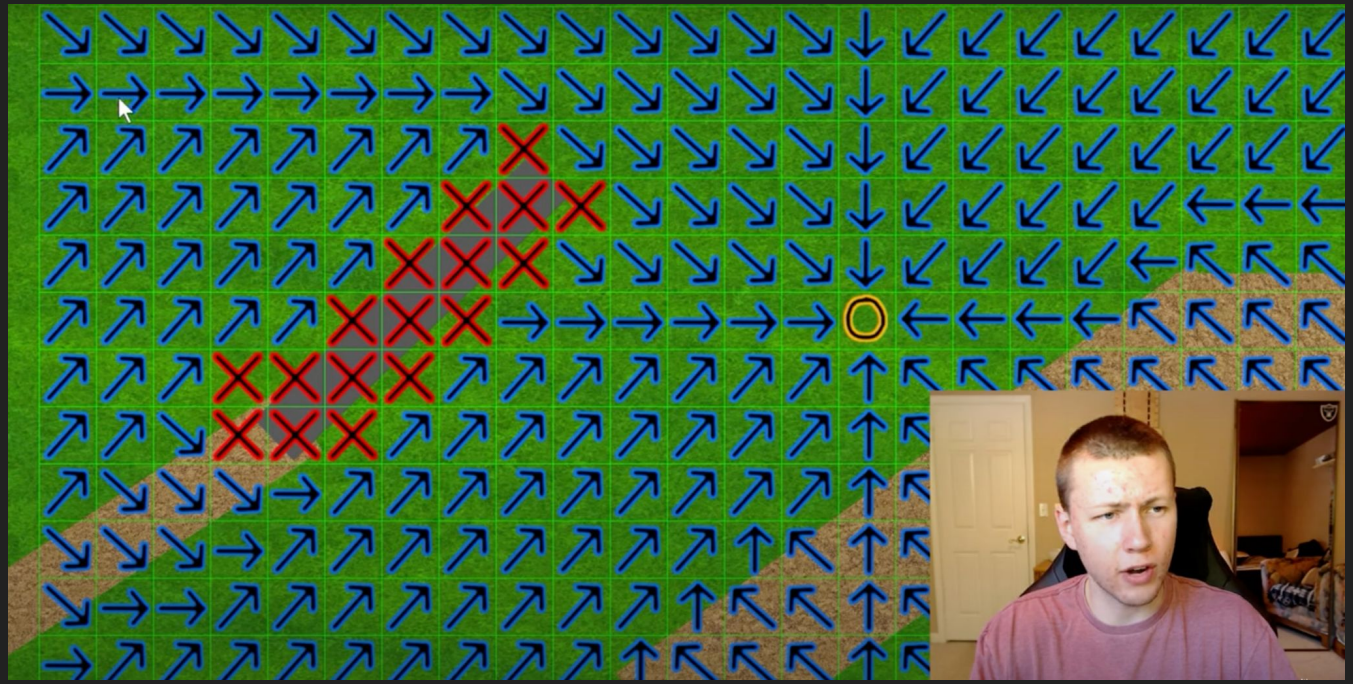
pathfinding - algorithmic approach



A* (A star) vs Dijkstra's algorithm

https://www.youtube.com/watch?v=9REexHx0hDY&ab_channel=Borington

pathfinding - algorithmic approach



Flow Field Pathfinding

https://www.youtube.com/watch?v=zr6ObNVgytk&ab_channel=TurboMakesGames

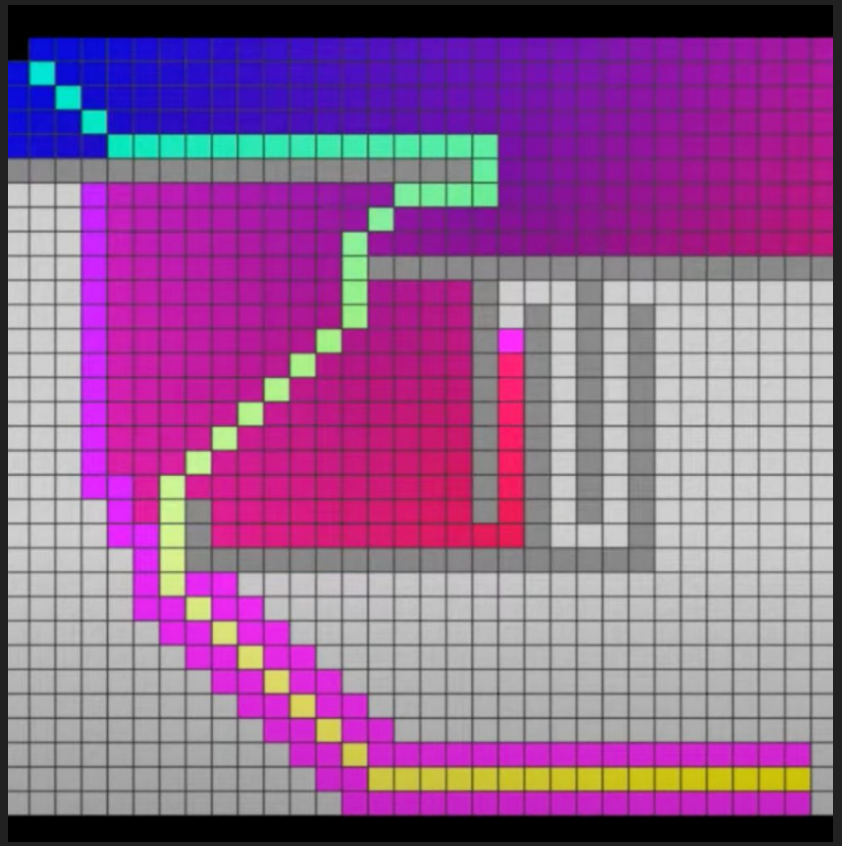
pathfinding - algorithmic approach

pros

- more "intelligence"

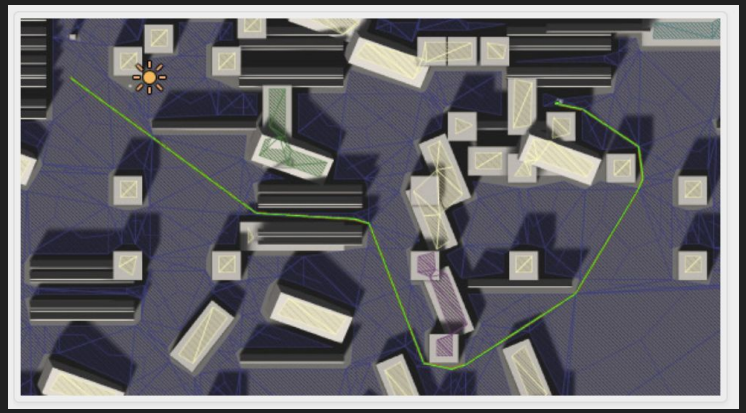
cons

- complexity
- performance
- hard to... customize
    - eg. hopping in AStar



pathfinding - algorithmic approach

tools for popular game engines



Unity

- Navmesh (built-in)
  - https://learn.unity.com/tutorial/unity-navmesh
- AStar Pathfinding Project (Free and Paid)
  - https://arongranberg.com/astar/

Unreal

- Navigation System (built-in)
  - https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/ArtificialIntelligence/NavigationSystem/BasicNavigation/

Godot

- Navigation (built-in)
  - https://docs.godotengine.org/en/stable/tutorials/navigation/index.html

pathfinding - tips

pathfinding - tips

- do you "need" more intelligence?
  - what is your aim?
  - feeling of being chased, perhaps simple ai is good enough?
  - suggest to improve incrementally

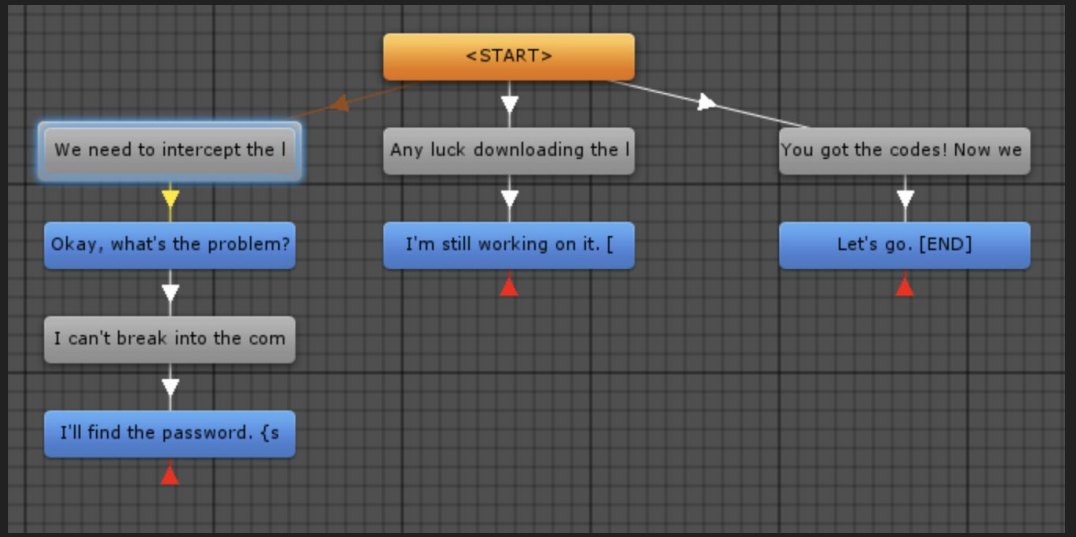
- dun be afraid to try out different solutions and see what works
  - sunk-cost fallacy
  - but also means in the first place don't write sloppy and inflexible code
  - abstract it so its easier to change the implementation without rewriting the entire codebase

dialogue

dialogue

types of dialogues

- barks
- subtitle
- close-up with options



Dialogue System for Unity

https://www.pixelcrushers.com/dialogue_system/manual2x/html/

dialogue

difficulty lies in that, dialogue often depends on game state

- quest system
- inventory system
- surroundings
    - enemies
    - player health
    - actions (ouch!)

dialogue - tips

- localization

- dynamic variables (eg. change according to player name)

- text effects (eg. floating, animated text)

- organization

honorable mentions

honorable mentions

factions

- friendly and hostile
- procedurally generated factions?
- careful about complexity

relationships

- love-hate?
- may affect factions
- may affect dialogue

end