Thinking Like Players:

How Halo Infinite's Bots Make Decisions

GDC 2022	March 24th 2022	Brie Chin-Deyerle

Talk Timeline

Overview
- Intro
- What is a game mode?

Architecture
- Describing game rules to bots
- A framework for how to make decisions

Development
- Tuning and iterating on decision making
- Case studies

Hi!

I'm Brie Chin-Deyerle (she/her)

- Senior Gameplay Engineering Lead on the Halo Infinite Multiplayer team (Academy & Bots)
- Theatre nerd
- Enjoyer of puns
- Has played a lot of Chrono Trigger

Why are we here today?

Halo Infinite was the first Halo game to feature multiplayer bots

Halo has a history of cool AI characters

But AI playing multiplayer was a very different challenge

How did bots decide what to do?

!=

Applying some foundation

Halo Infinite Multiplayer would be Free to Play & Cross platform

- Halo has been around for 20 years

We needed a better onboarding experience

Enter: The Academy

Enter: Bots!

Our goal: Develop bots to be good training partners for new and returning players

- We also used bots to backfill players in some unranked experiences

Team of 1 engineer + 1 designer, grew over time to 3 engineers

Debuted bots in our first technical preview in July 2021

The bot AI in Halo Infinite is honestly the best I've ever seen.

"

There are people who think the grapple mans bot is a real person, I've gotten multiple messages from people telling me to uninstall or saying unsavory things or asking how I got the samurai armor, and it's all from times when I wasn't even online 😂 they look up the name and see there's an account, haha

[redacted] · Nov 23

Replying to @BrieChinDeyerle [redacted]

as someone who cannot play PVP due to my inability to git gud the bots are literally the only reason i can enjoy the MP rn. Ive always wanted bots in Halo and im so happy i can actually the mp now, easily the most important feature for me!

1 2

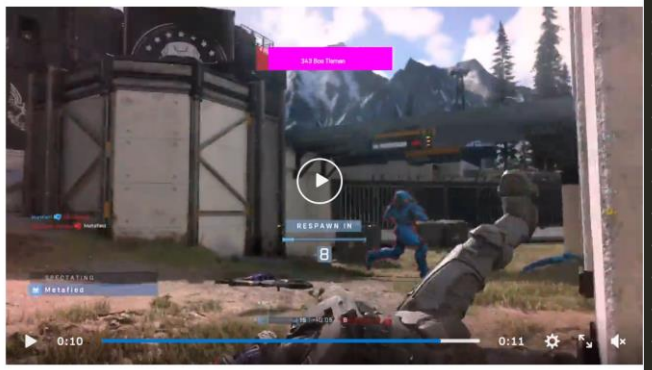
Halo Infinite's Bots Are Impressive (And Making Me Nervous)

The bots seen in the ongoing beta on Xbox and PC are surprisingly good and will only get better

By Zack Zwiezen | 8/01/21 2:30PM | Comments (63) | Alerts

If you had told me that Halo would add bots that could style on me like this, I would have called you a liar

Gameplay



173 Comments Share Save Hide Report 99% Upvoted

Halo Infinite's bots aren't messing about

Combat evolved.

News by Wesley Yin-Poole, Deputy Editorial Director
Updated on 31 July 2021

Talk Timeline

Overview
- Intro
- What is a game mode?

Architecture
- Describing game rules to bots
- A framework for how to make decisions

Development
- Tuning and iterating on decision making
- Case studies

What’s a game mode?

A game mode is the set of rules that describes how to play and how to score points

Arena mode breakdown

For our initial launch, we had 4 arena modes of focus

- Slayer
- Capture the Flag
- Strongholds
- Oddball
- Variants

Acquiring new items & fighting

Interacting with objects

Picking up & delivering objectives

Hiding & hunting

Guarding

Verbs -> Behaviors

These common verbs formed the basis of our initial behaviors

- We added a few more over time as needed

But we needed a mechanism to link behavior with specific things

```
CE(BehaviorIndex::_BotEngageBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotPickupObjectiveBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotDeliverObjectiveBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotInteractBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotHideBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotHuntBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotContestItemBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotAcquireWeaponBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotAcquireEquipmentBehavior, 1, CHECK_ALWAYS),
CE(BehaviorIndex::_BotGuardBehavior, 1, CHECK_ALWAYS),
```

Talk Timeline

Overview
- Intro
- What is a game mode?

Architecture
- Describing game rules to bots
- A framework for how to make decisions

Development
- Tuning and iterating on decision making
- Case studies

Slipspace script-space

Halo Infinite's game mode logic lives in lua

- The authority for game rules and state

Our bots largely act autonomously, but need a way to share some information

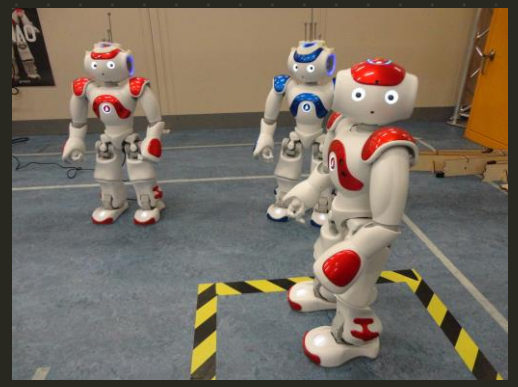
The BotManager is effectively a blackboard that lua & bots read/write to

- Stores ambitions and some additional bot state

Write

Read

Write



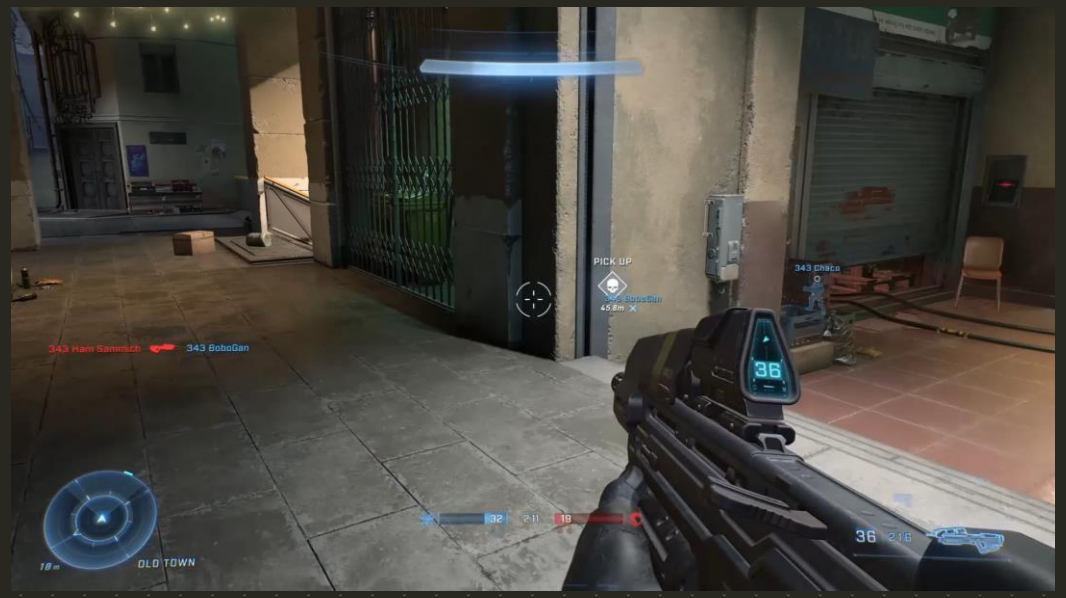
Ambitions

An **ambition** is an object or area in the level that is relevant to score points in a game mode

- A zone in strongholds
- A ball in oddball

Ambitions have a bit of metadata associated with them.

- What type of ambition it is
- What object the ambition is related to
- What team(s)/bots are allowed to utilize the ambition



Applied ambitions

```
function Flag.OnDeliverableSpawned(deliverObjectParcel:table, argDeliverObject:object, argSpawnLocationObject:object, isInitialSpawn:boolean):void
    self.UpdateFlagState(self.CONSTANTS.THSValues.reason_FlagObjectSpawned);

    self:TriggerEvent(self.EVENTS.onFlagSpawned, argSpawnLocationObject, isInitialSpawn, self);

    --if the flag is neutral, add pickup ambition for bots of all teams. If owned, add pickup ambition for bots on the opposing team
    if (self.owningTeam == GetNeutralMPTeam()) then
        Bot_AddAmbitionObject(self.currentFlagObject, BOT_AMBITION.Pickup, 1.0, MP_TEAM_DESIGNATOR.Neutral);
    else
        Bot_AddAmbitionObject(self.currentFlagObject, BOT_AMBITION.Pickup, 1.0, Team_GetOpposingTeamDesignator(self.owningTeam));
    end
end
```

Talk Timeline

Overview
- Intro
- What is a game mode?

Architecture
- Describing game rules to bots
- A framework for how to make decisions

Development
- Tuning and iterating on decision making
- Case studies

What do I *do* with this flag?

Imagine that you have the flag

You have to decide between:

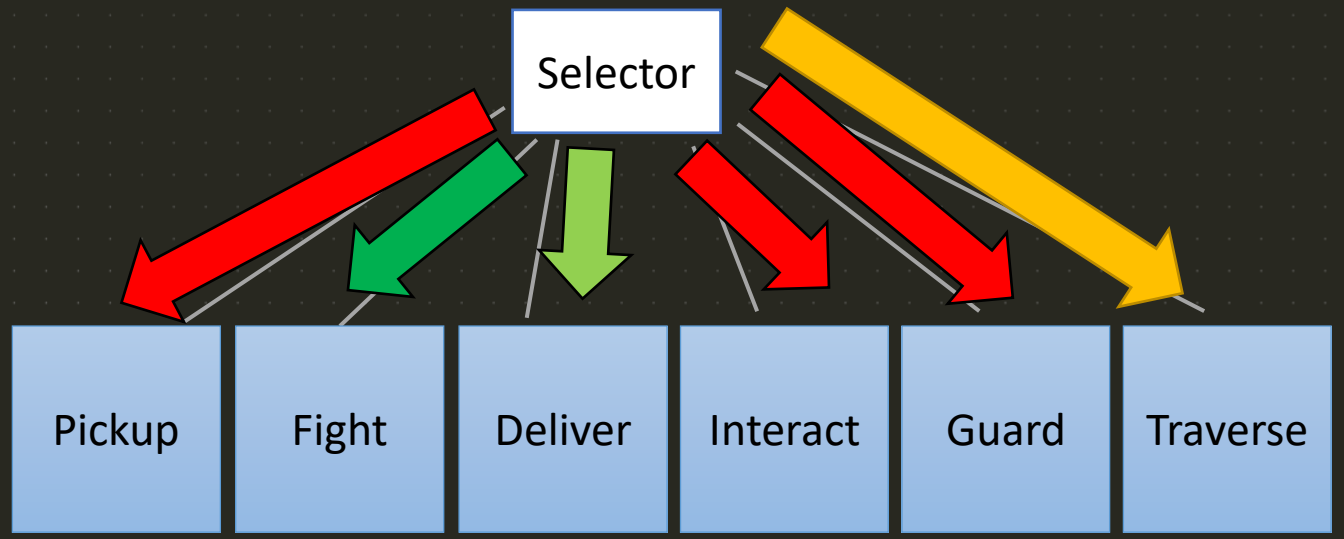
- try to fight that enemy
- ignore the enemy and keep trying to deliver the flag

The “right” answer is really hard to define

- Depends on a lot of game state

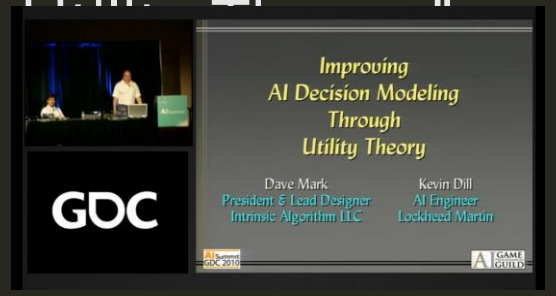
Reusing existing architecture…?

Our behavior tree evaluates nodes in order and finds a behavior that is valid to execute



Our solution: A utility system

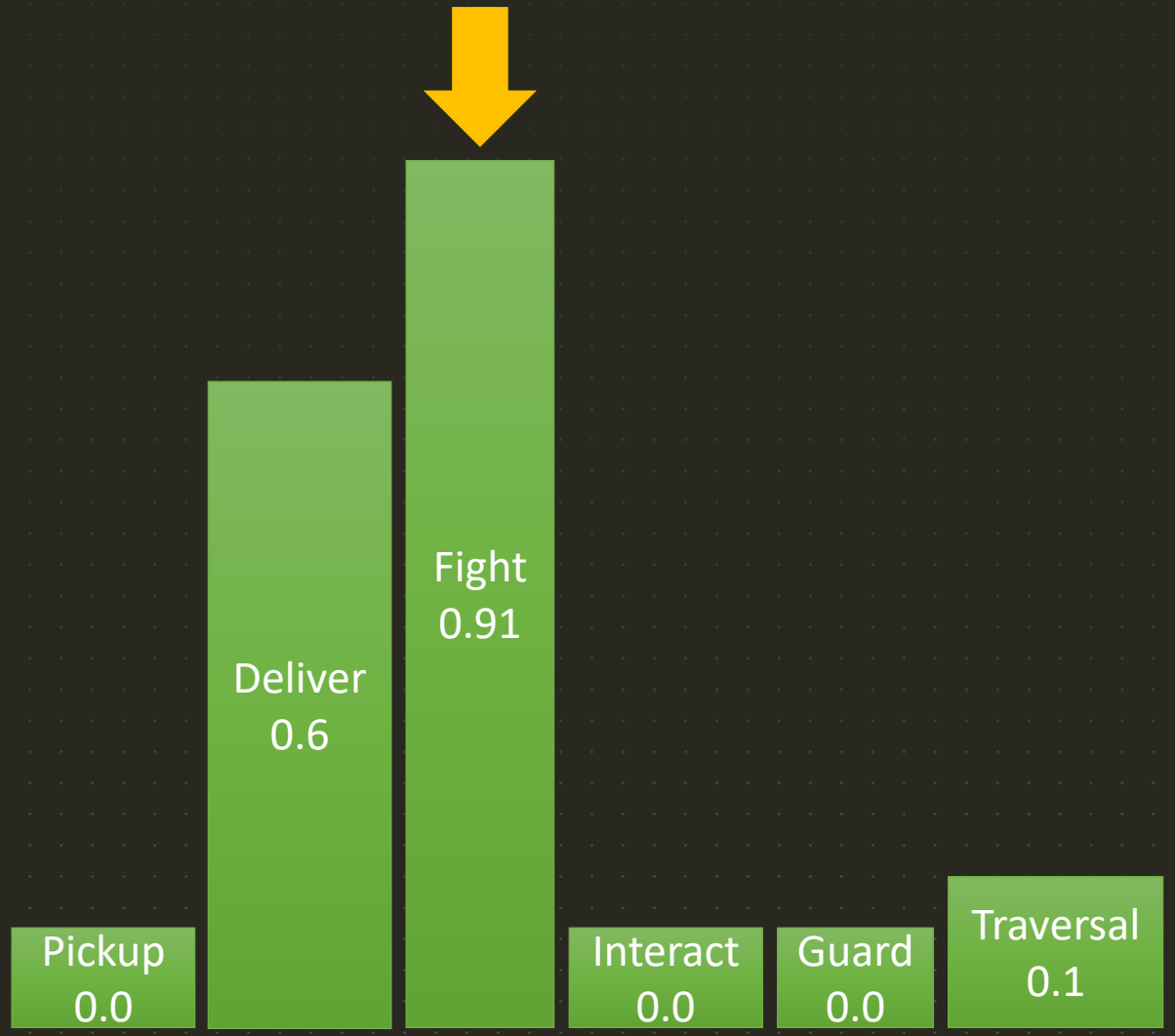
Inspired by Dave Mark & Kevin Dill's 2010 GDC AI summit talk "Improving AI Decision Modeling Through



A "utility system" is where an agent ranks each behavior they could possibly do with a numerical value

- Big number == "this is important"

The agent can then just pick the most important behavior

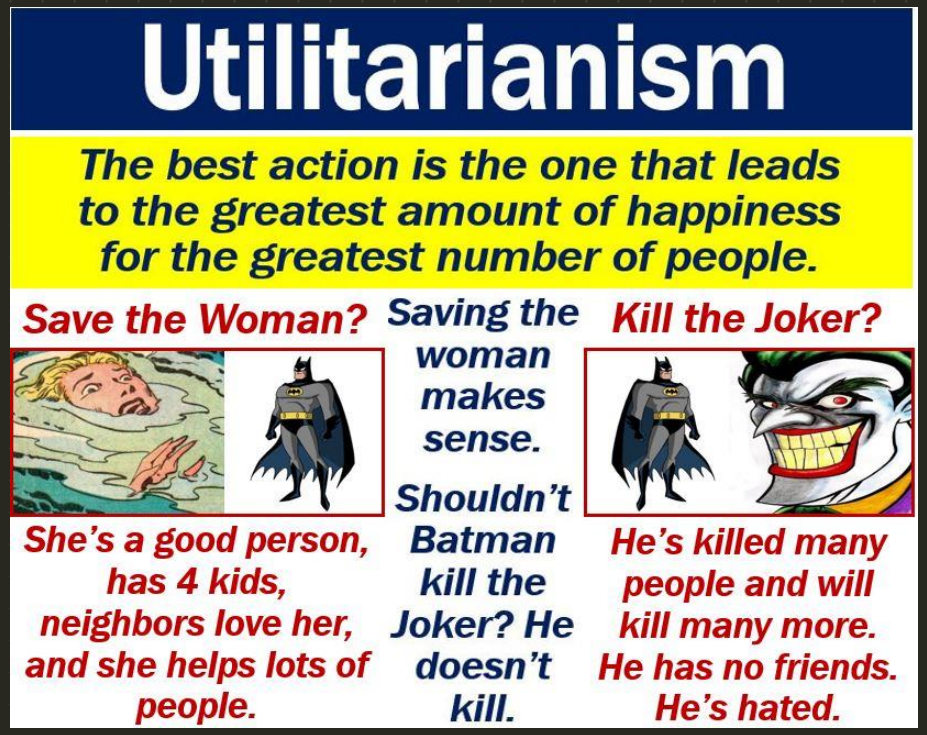


The challenges of utilitarianism

We now had systems to define the game mode and make decisions

Balancing the weights of things that are difficult to compare is hard

- The utility to pickup a flag is very different to pickup the oddball



Save Woman | 0.748367?

Kill Joker | 0.314159?

Talk Timeline

Overview
- Intro
- What is a game mode?

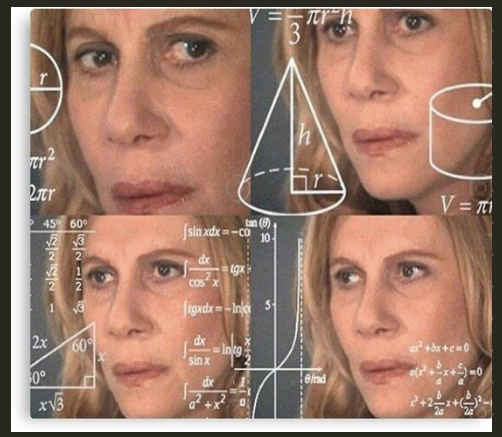
Architecture
- Describing game rules to bots
- A framework for how to make decisions

Case Studies
- Tuning and iterating on decision making
- Anecdotes and examples

Typical utility tuning

Often times, utility systems are tuned with mathematical functions

Sometimes hard to map human thought patterns to clean mathematical formula



Tuning evolved

We already had one tuning variable that we called confidence fully defined in lua

- Used by combat behaviors to determine whether to push or pull back mid combat
- Accounted for personal health, target health, weapons, and weird edge cases

What if we just tuned all the utility values in lua?

This also meant specific game modes could influence tuning more directly

```lua
function CalculateHuntUtility(botParticipant:player):number

    -- If you're holding the objective, your job is not to hunt this day.
    if (TempBotUtilityData.IsHoldingObjective == 1) then
        return 0;
    end

    -- If you're allowed to deliver your own flag and don't know where the flag carrier is, don't need to hunt.
    if(BotUtilityData.CanDeliverFlag[Player_GetMultiplayerTeam(botParticipant)] == true and TempBotUtilityData.DistanceToHuntObject < 0) then
        return 0;
    end

    local utility = -1;
    local maxDistance = 50;
    if(TempBotUtilityData.DistanceToHuntObject >= 0) then
        utility = BotLerpWithCap(BotNormalizeAndInvertValue(TempBotUtilityData.DistanceToHuntObject, maxDistance), 0.5)
    else
        utility = 0.1;
    end

    return utility;
end
```

Tuning knobs

Fixed & functional values

Upper bound cap

Weighted inputs

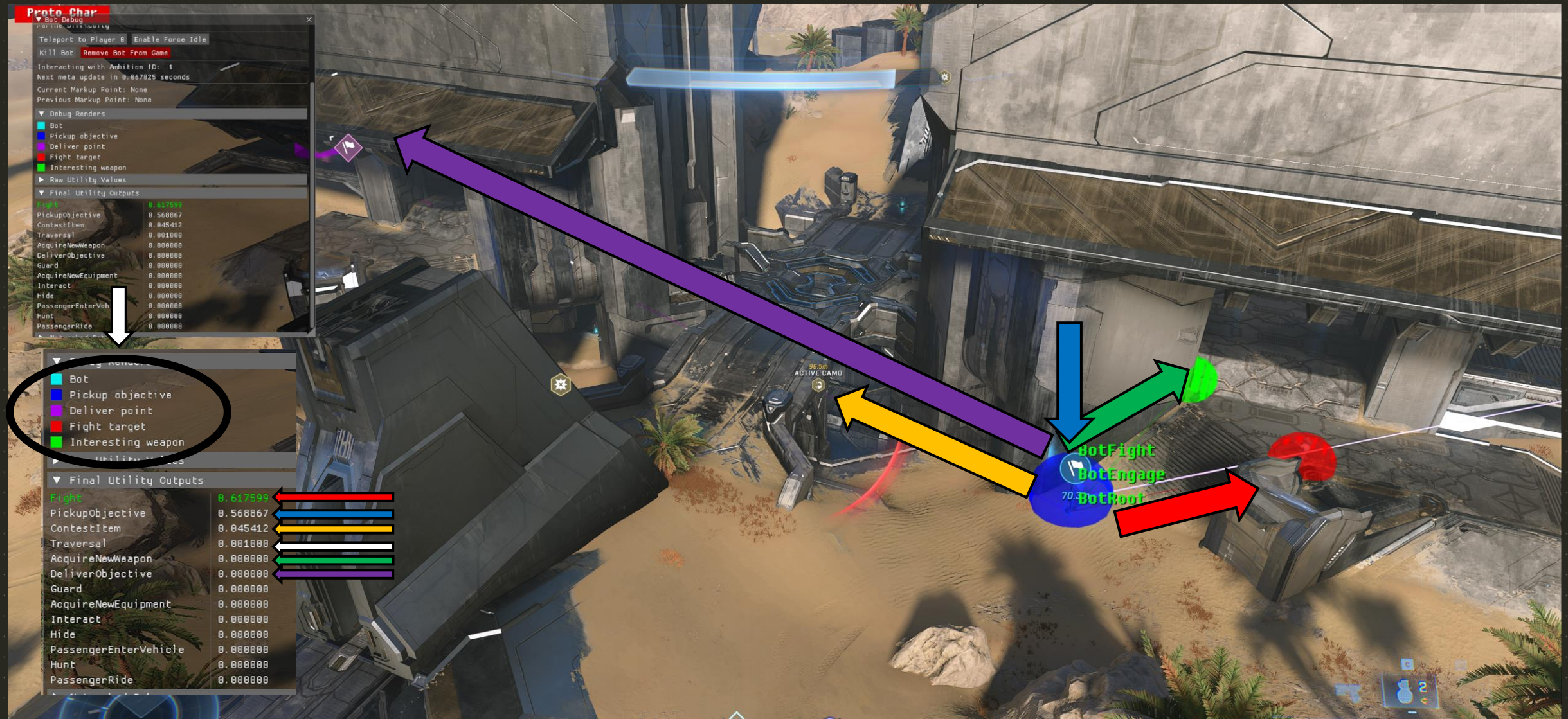
```
local utility = -1;
local maxDistance = 50;
if(TempBotUtilityData.DistanceToHuntObject >= 0) then
    utility = BotLerpWithCap(BotNormalizeAndInvertValue(TempBotUtilityData.DistanceToHuntObject, maxDistance), 0.5)
else
    utility = 0.1;
end

return utility;
```

```
if (BotUtilityData.ObjectiveIsOddball == true) then
    --If we are about to win, we want the oddball
    if (math.abs(Team_GetScore(Player_GetMultiplayerTeam(botParticipant)) - Variant_GetScoreToWinRound()) < OddballScoreDifferentialCloseToWinning) then
        cap = 0.7;
    end
end
```

```
-- If we are close to the flag stand, bump up utility so we pick it up and deliver it
if (BotUtilityData.ObjectiveIsCTF) then
    if (TempBotUtilityData.DistanceToNearestDeliverLocation <= closeDistance and TempBotUtilityData.DistanceToNearestObjective <= closeDistance) then
        distanceWeight = distanceWeight + 4;
        cap = math.max(cap + 0.2, 0.8);
    end
end
```

Applying utility: Fight



Applying utility: Objective

Lua isn't the fastest...

One disadvantage of moving everything into lua is speed

Halo Infinite is a high-performance game and perf demands were tight

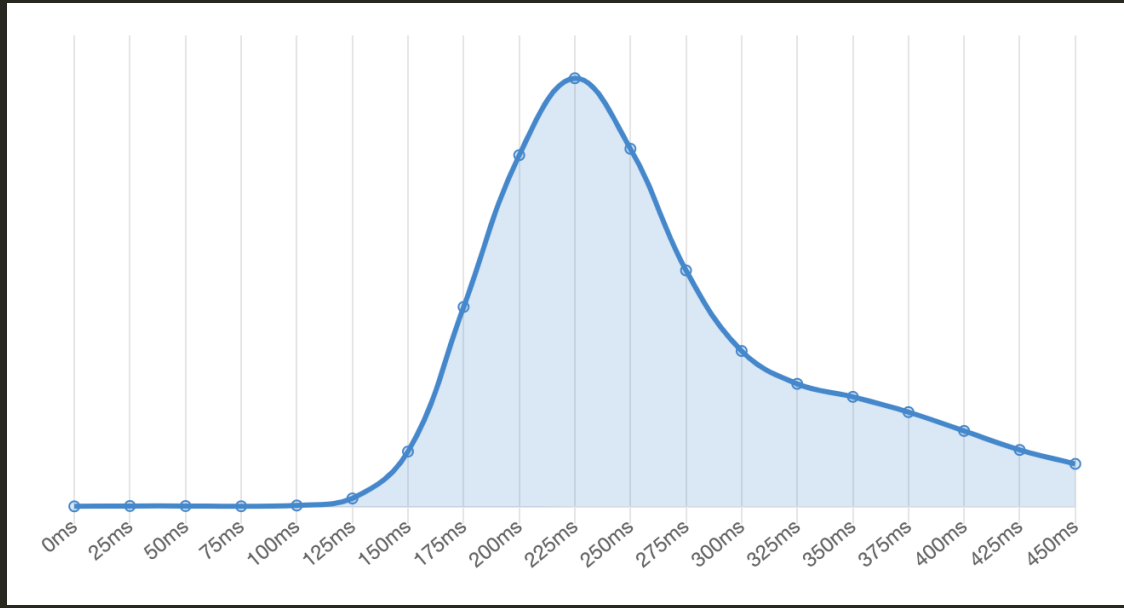
...but human reactions are slower!

200-250ms to process new information

We were also only targeting 8 bots for our arena experience

- 250ms / 8 bots = 31.25ms per bot

Slowing down bot reactions made them better practice partners



Human Benchmark - Reaction Time Test

Talk Timeline

Overview
- Intro
- What is a game mode?

Architecture
- Describing game rules to bots
- A framework for how to make decisions

Development
- Tuning and iterating on decision making
- Case studies

Case study 1: Too many good options

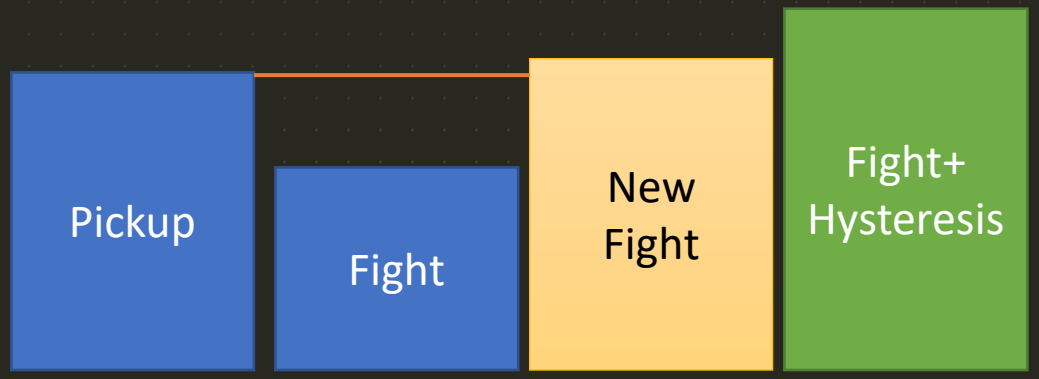
What do you do when two (or more) utilities end up being close?

You thrash, and functionally choose neither

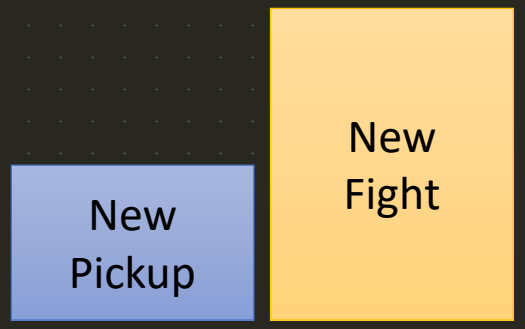
Dealing with thrashing

Hysteresis: Inflate the utility of a new behavior when we switch to it for a while

Add additional logic to smooth things over




```
--If we are in danger, lower the cap so we are more likely to just keep fighting
if (TempBotUtilityData.Danger > 0.3) then
    cap = math.max(cap - 0.2, 0);
end
```



Case study 2: To fight, or not to fight

While in a non-combat behavior, bots were effectively pacifists

This made tuning a real challenge

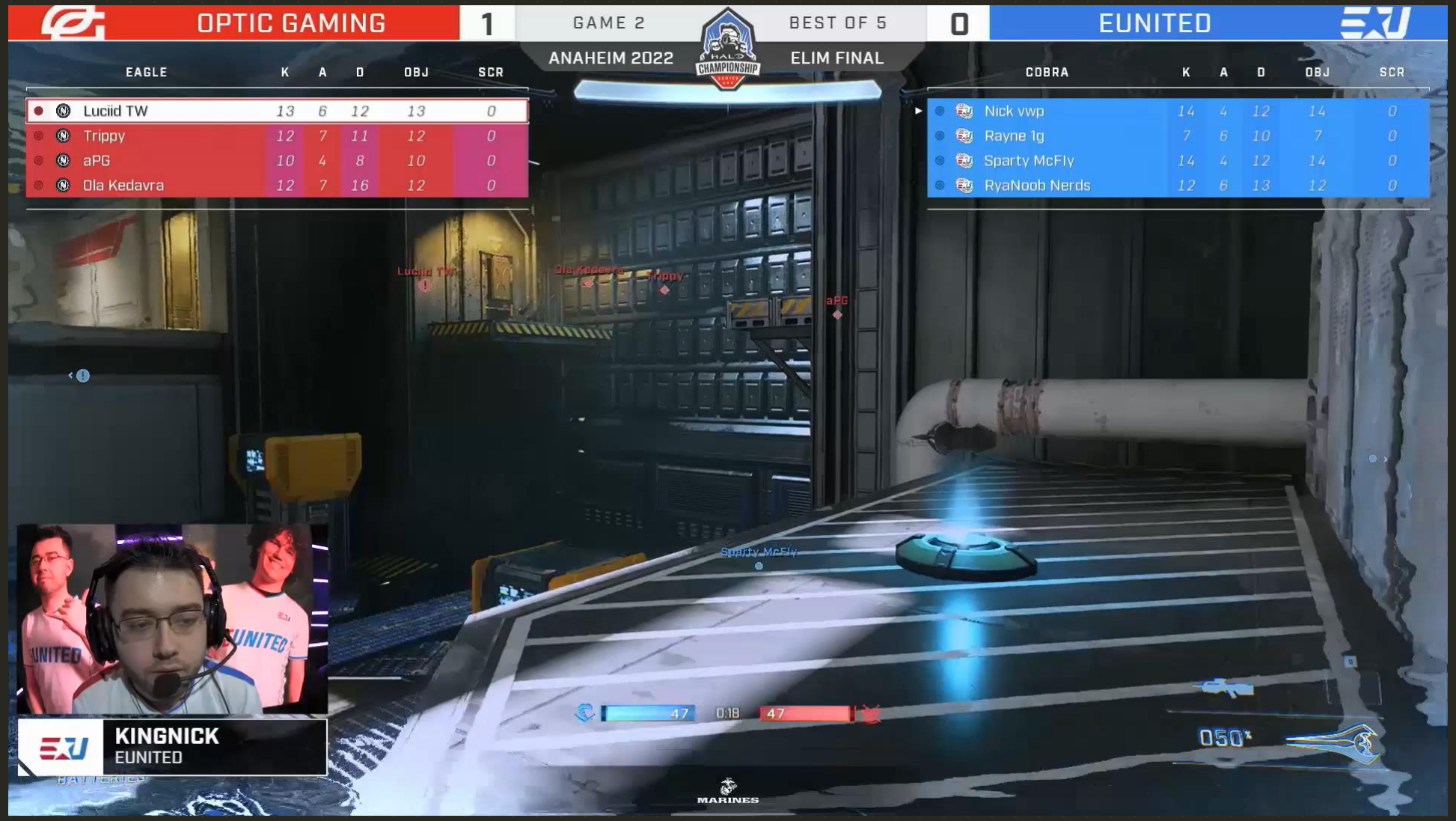
ABS - Always Be Shooting

Case Study 3: Teaching bots to talk

When you start playing Halo competitively, it can often feel like a solo adventure with teammates

When players get much better at the game, they start to communicate

Human communication



Reusing existing systems

Bots have a simple concept of object permanence

We allowed bots of a certain difficulty to share known positions of enemies they saw

Bot (un)awareness

Noticed bots really struggling with objective modes after this change

Observed a ton of time in combat behaviors

Failing to engage with mode objectives

Awareness in practice

Shared awareness in practice

Case Study 4: What not to do

Shared awareness worked well because we had a system we could reuse

What about a system where there was a less straightforward implementation?

Behavioral Complexity


HOUSE

Behavioral simplicity

Case study 5: Quirks of actual ga

Getting to the BOTtom of things

Keep your “why” in mind

You don’t need to do everything perfectly!

Play to your strengths, don’t pursue options that aren’t working for you

Consider how to test and validate changes early, especially subtle ones

Many thanks to 343, partners, and the bots team!

Thank you! (Please fill out your survey <3)

Contact me!

www.briechindeyerle.com

Twitter: @BrieChinDeyerle

LinkedIn: www.linkedin.com/in/briechindeyerle

---

343 Industries talks on the GDC vault:

Technical Artist Summit: Building 'Zeta Halo': Scaling Content Creation for the Largest 'Halo' Ever

Kurt Diegert & Mikael Nellfors

One Frame in 'Halo Infinite'

Daniele Giannetti

Deconstructing the Combat Dance: Designing Multiplayer Bots for 'Halo Infinite'

Sara Stern

---

343 is hiring!

www.343industries.com/careers