AI ARTIFICIAL INTELLIGENCE SUMMIT

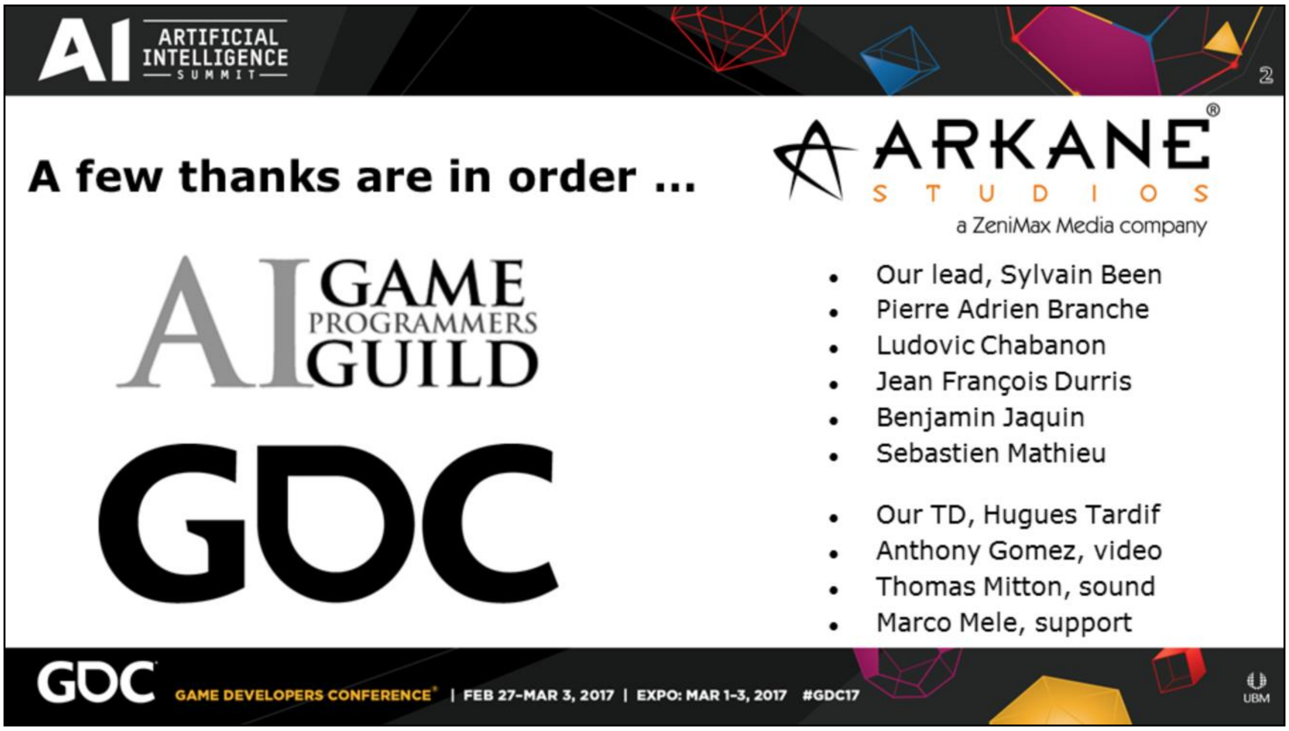
Taking Back What's Ours: The AI of Dishonored 2

Laurent Couvidou & Xavier Sadoulet
Gameplay Programmers
Arkane Studios Lyon, France

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

I am Xavier Sadoulet, and with my colleague Laurent Couvidou we flew all the way from France to talk about the AI of Dishonored 2.



A few thanks are in order …

- Our lead, Sylvain Been
- Pierre Adrien Branche
- Ludovic Chabanon
- Jean François Durris
- Benjamin Jaquin
- Sebastien Mathieu

- Our TD, Hugues Tardif
- Anthony Gomez, video
- Thomas Mitton, sound
- Marco Mele, support

AI ARTIFICIAL INTELLIGENCE SUMMIT
2
ARKANE STUDIOS
a ZeniMax Media company
AI GAME PROGRAMMERS GUILD
GDC
GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
UBM

Before getting started, we think a few thanks are in order.

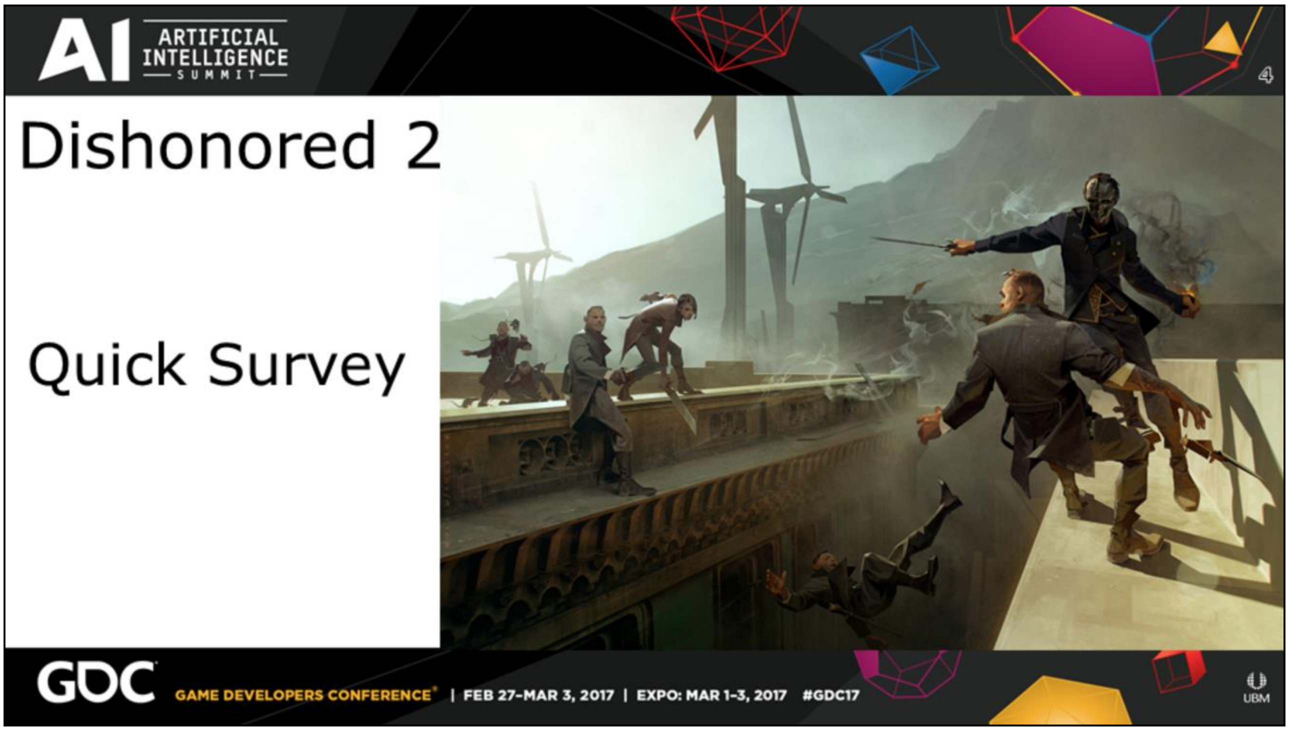
First we'd like to thank this community: be it the members of the guild, or people presenting over the years at places like GDC.

What we will discuss today was inspired by what you guys shared, so we're happy to give some back today.

Then of course we want to mention all the guys that worked on the AI for Dishonored 2, and supported us for the preparation of this talk.

And lastly, a big shout out to all the team in Lyon, it is these people work we're representing today so they deserve proper credits.

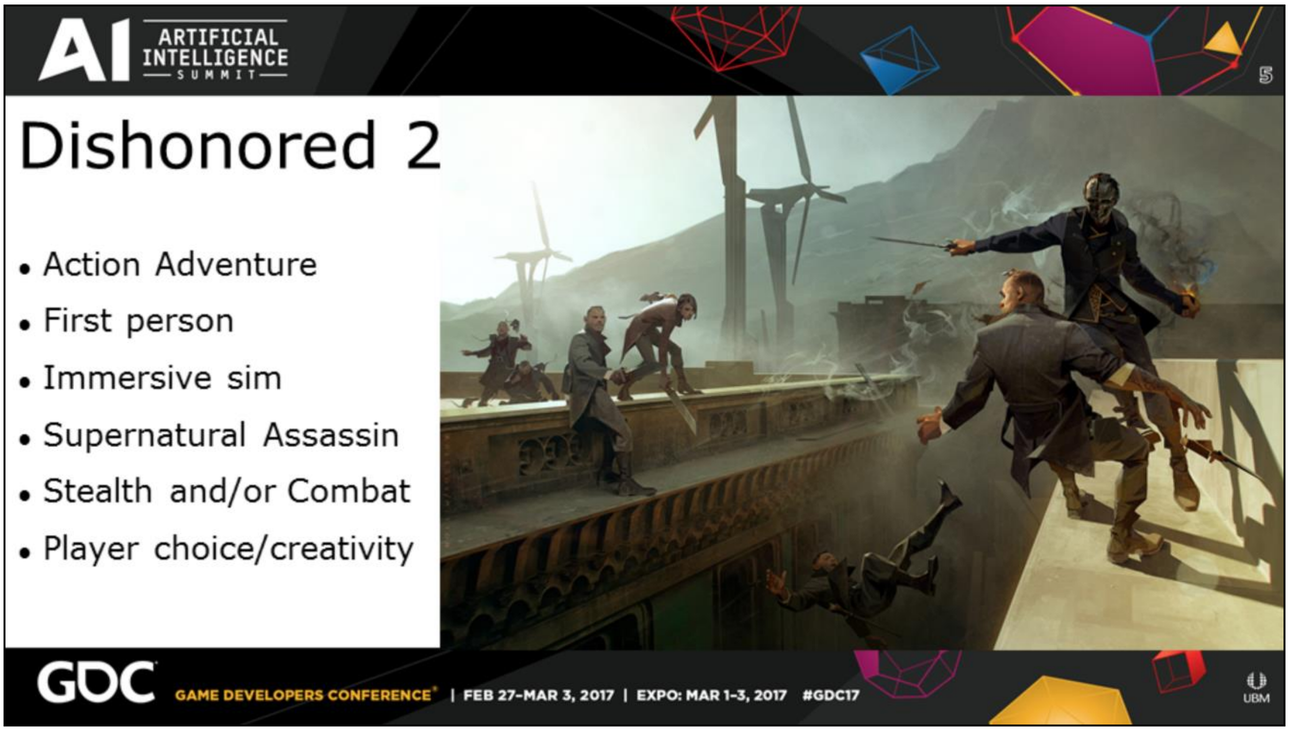
There, we're done being emotional, let's get to it.


AI ARTIFICIAL INTELLIGENCE SUMMIT
4
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Dishonored 2

Quick Survey

Who already played the first Dishonored?
Dishonored 2?



ARTIFICIAL INTELLIGENCE SUMMIT

Dishonored 2

- Action Adventure
- First person
- Immersive sim
- Supernatural Assassin
- Stealth and/or Combat
- Player choice/creativity

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

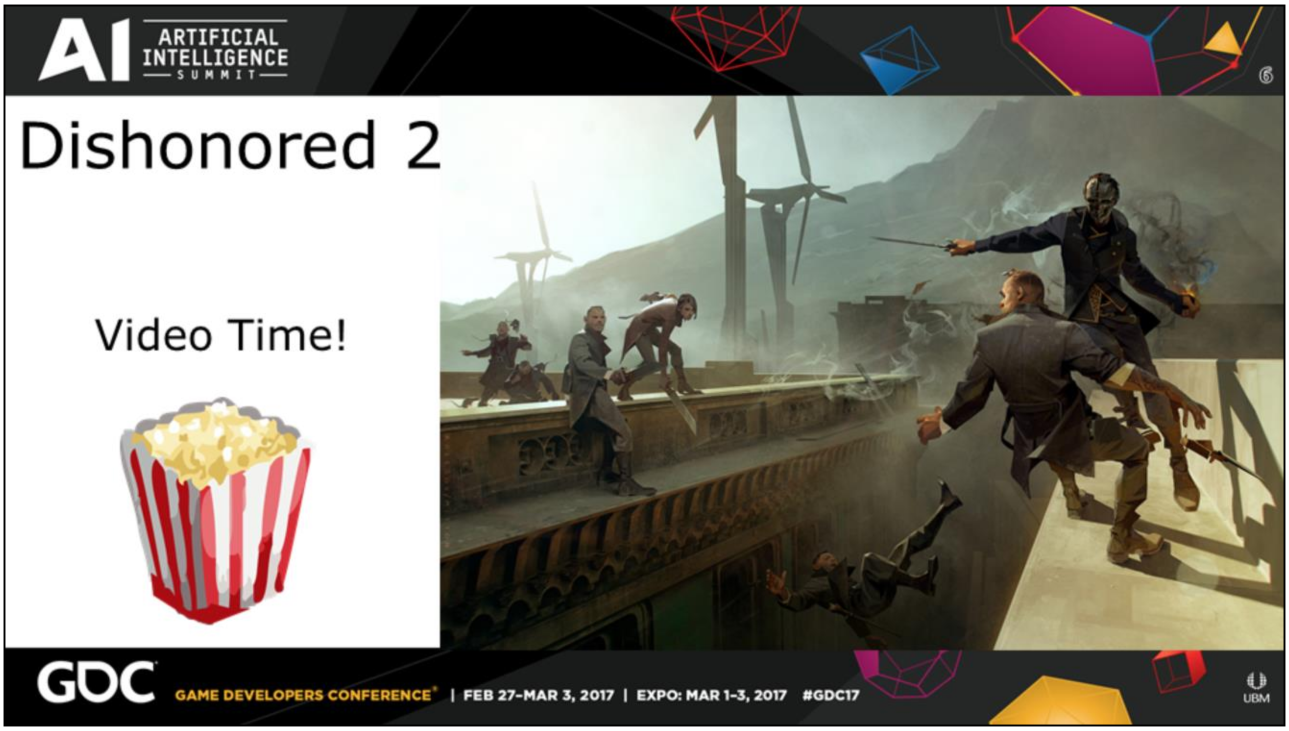
Then let me talk about the game

Dishonored 2 is an action adventure game, played in first person.

Like all Arkane games, it belongs to the immersive sim genre, among titles like System shock, Deus Ex or the Thief series.

In this game you play an assassin that can use either stealth or combat approach, and is given supernatural powers as tools to achieve his goals.

But essentially the game is really about how the player chooses to approach each mission, each situation.



Dishonored 2

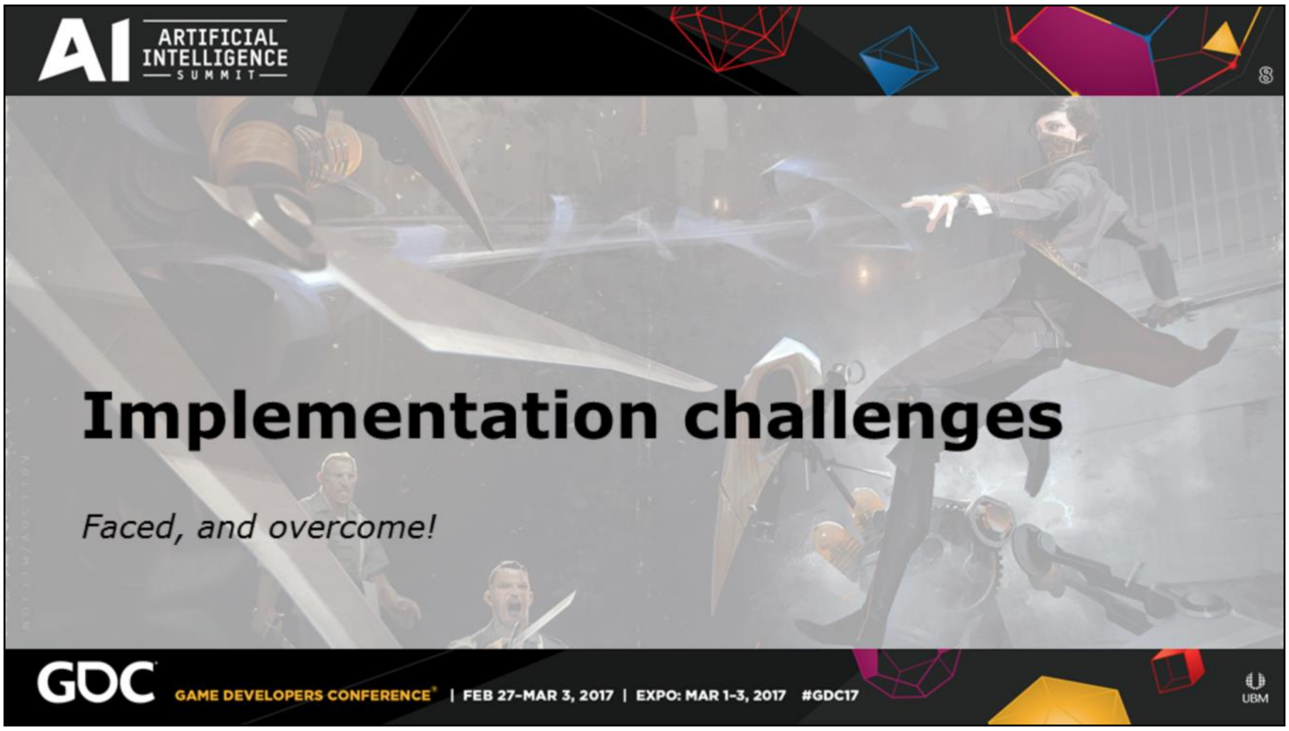
Video Time!

ARTIFICIAL INTELLIGENCE SUMMIT
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Here is a short video, showing you a glimpse of what the game looks like.

DISHONORED 2

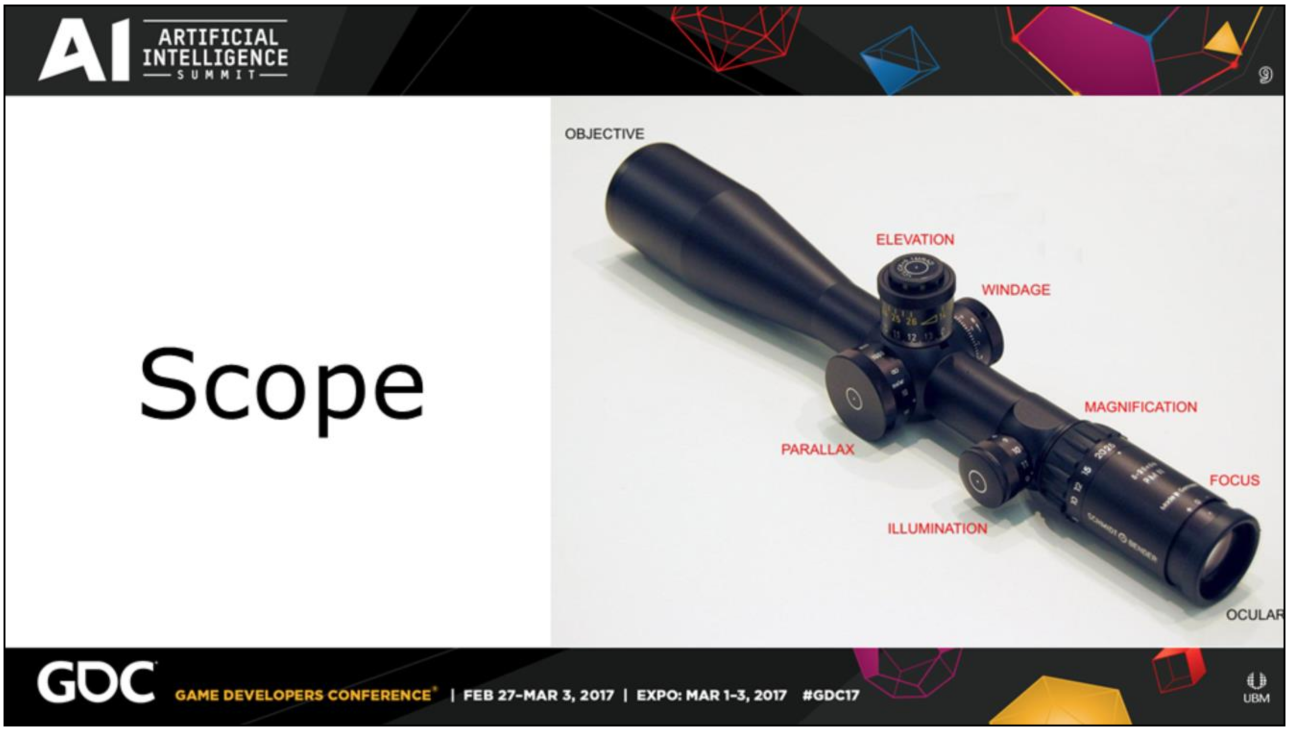
<intro video>



Implementation challenges

Faced, and overcome!

Now that you saw what the game is about, let’s talk about some challenges we had to tackle during the production.


AI ARTIFICIAL INTELLIGENCE SUMMIT
9
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

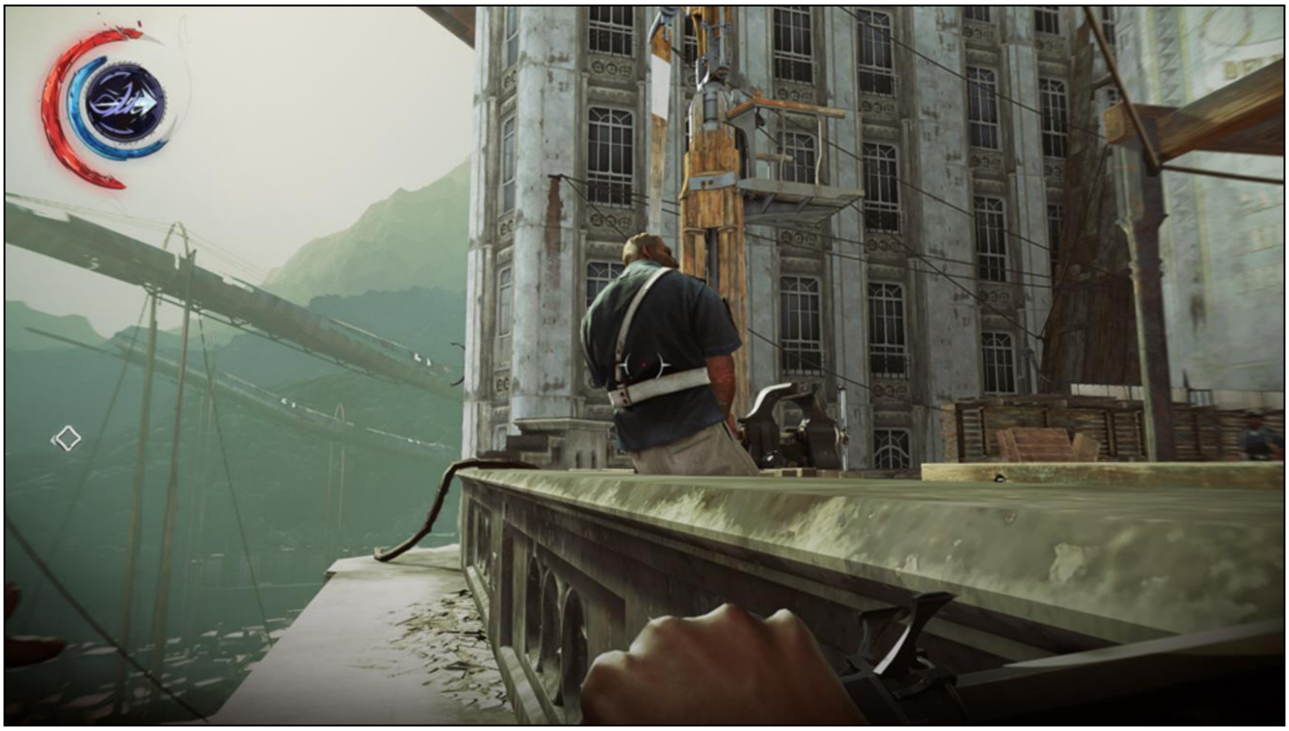
Scope

OBJECTIVE
ELEVATION
WINDAGE
MAGNIFICATION
PARALLAX
FOCUS
ILLUMINATION
OCULAR

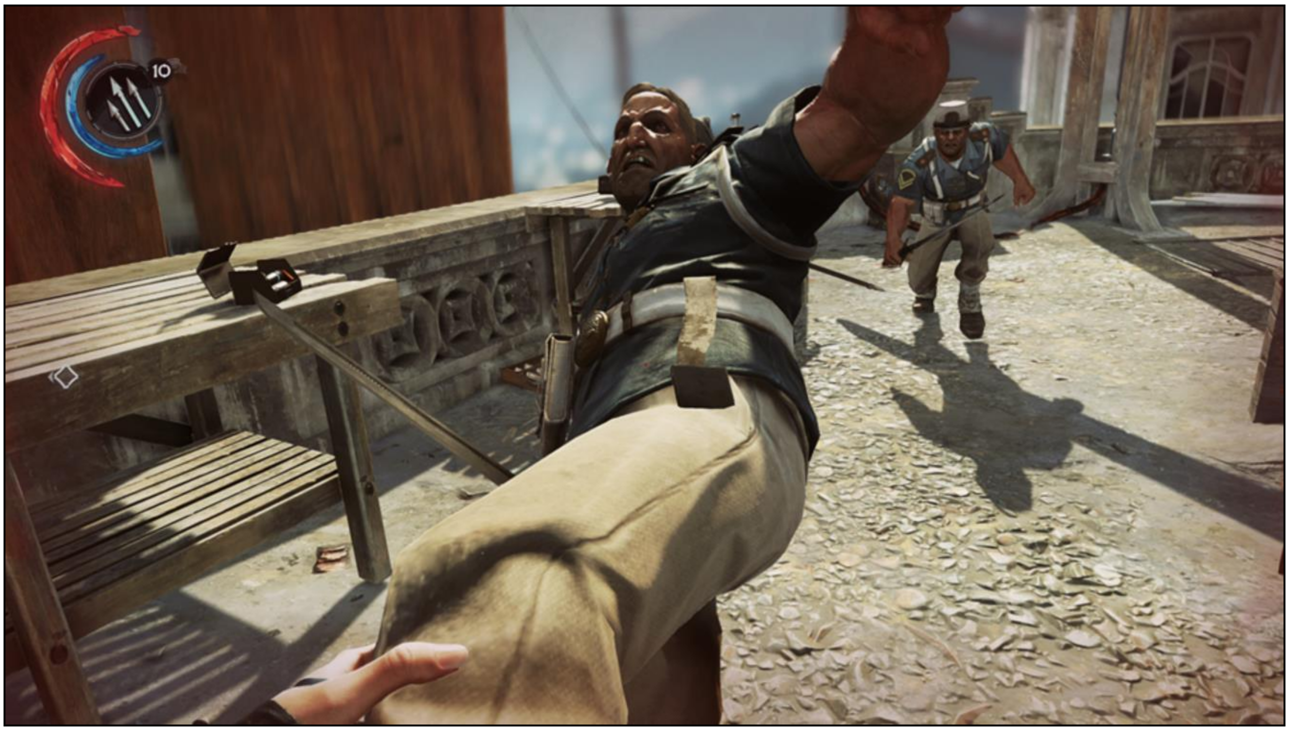
The first one was the scope.

Ok not like this sniper scope, more like… game scope.
<gesture: small / big>

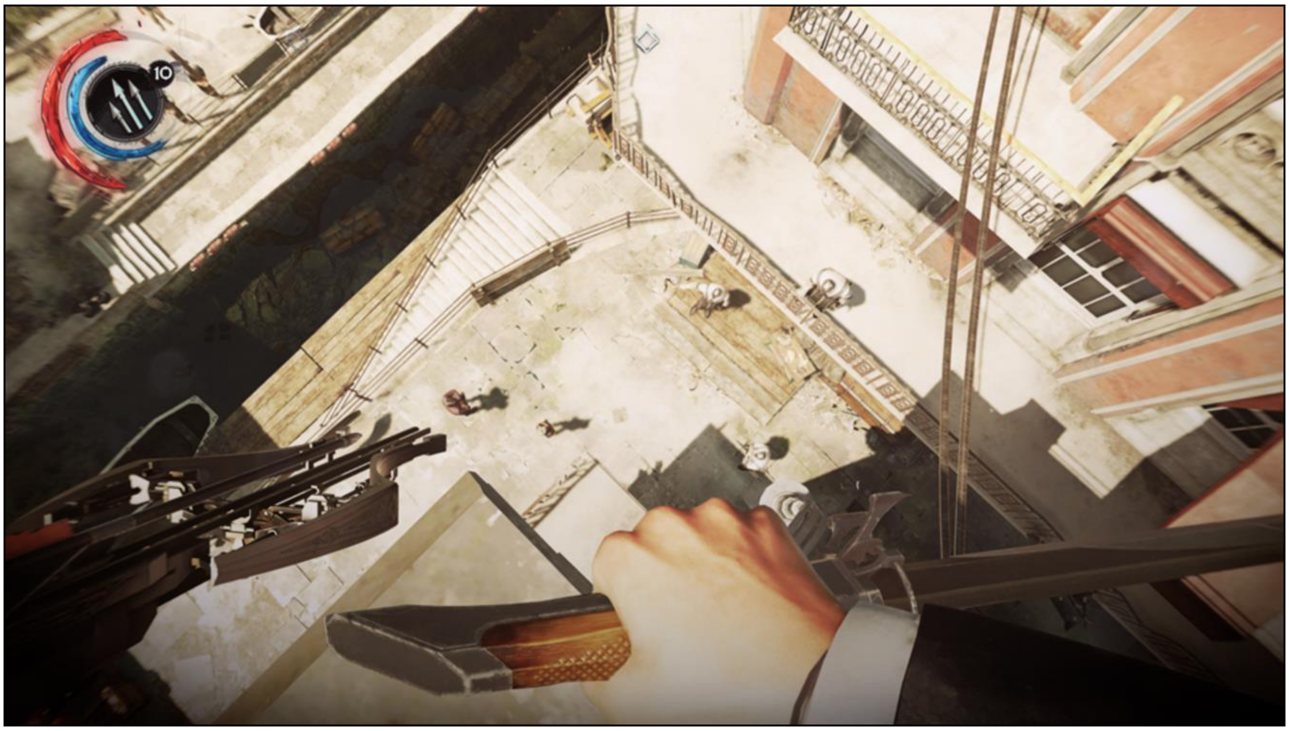
Let’s say the AI had quite a few things to support.



Stealth

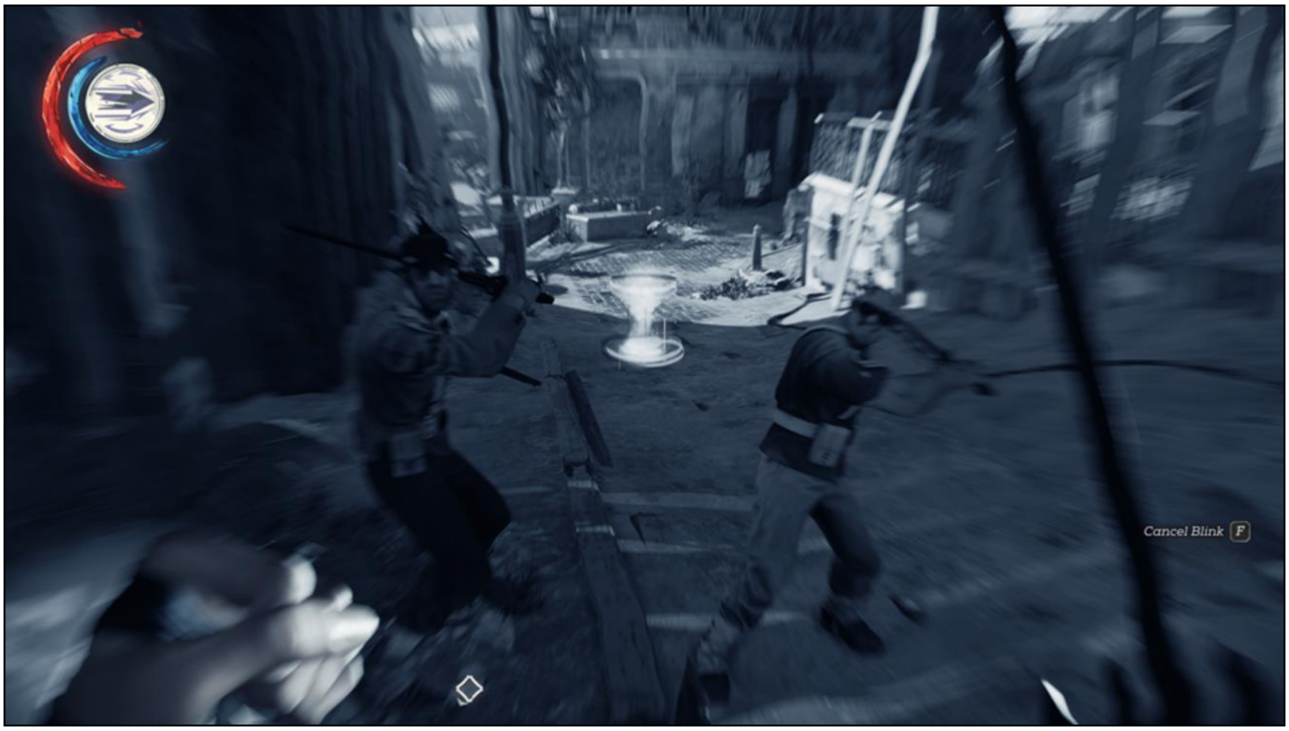


Combat!

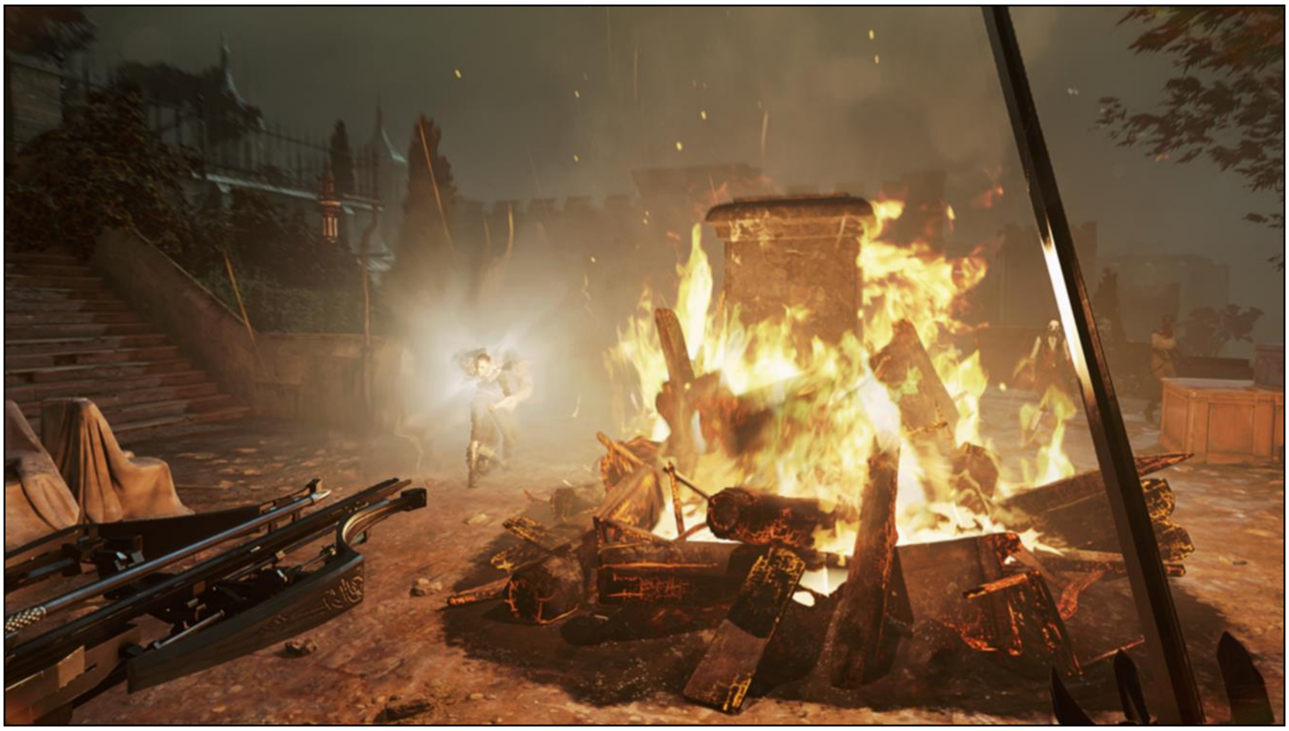


Verticality

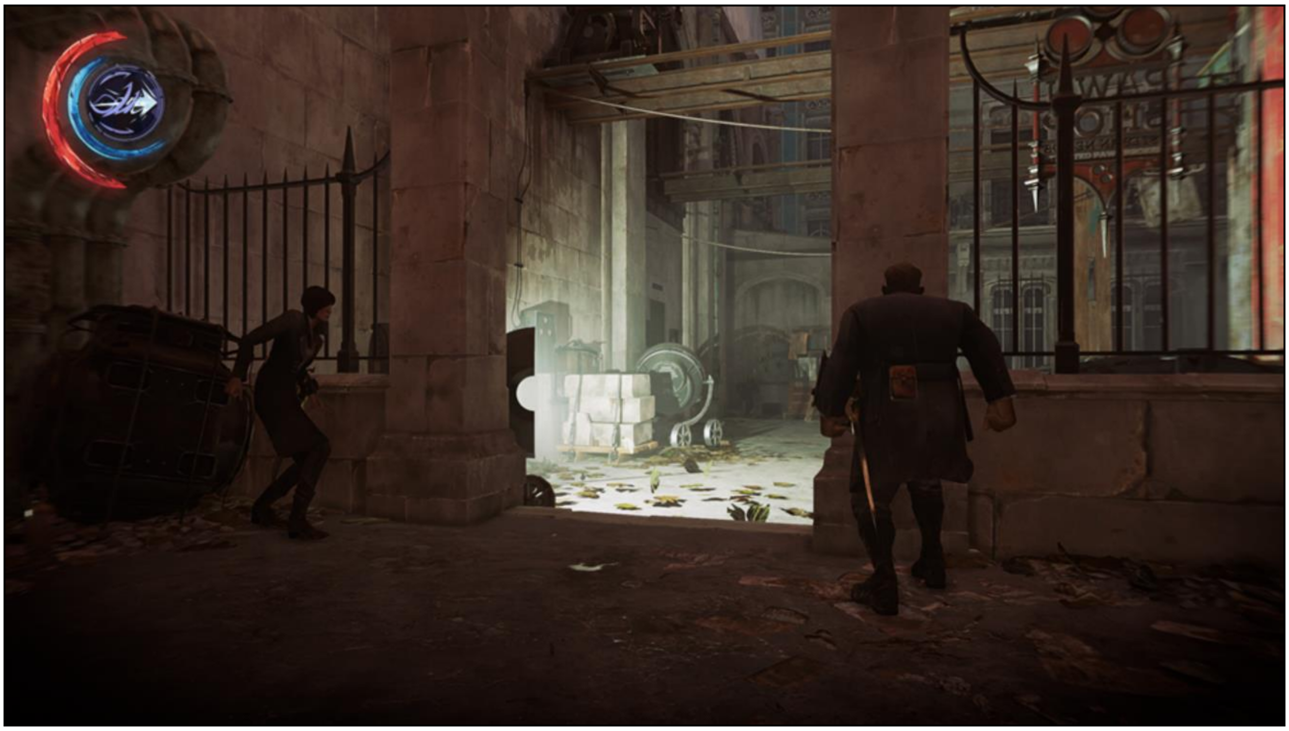
Dynamic relationships



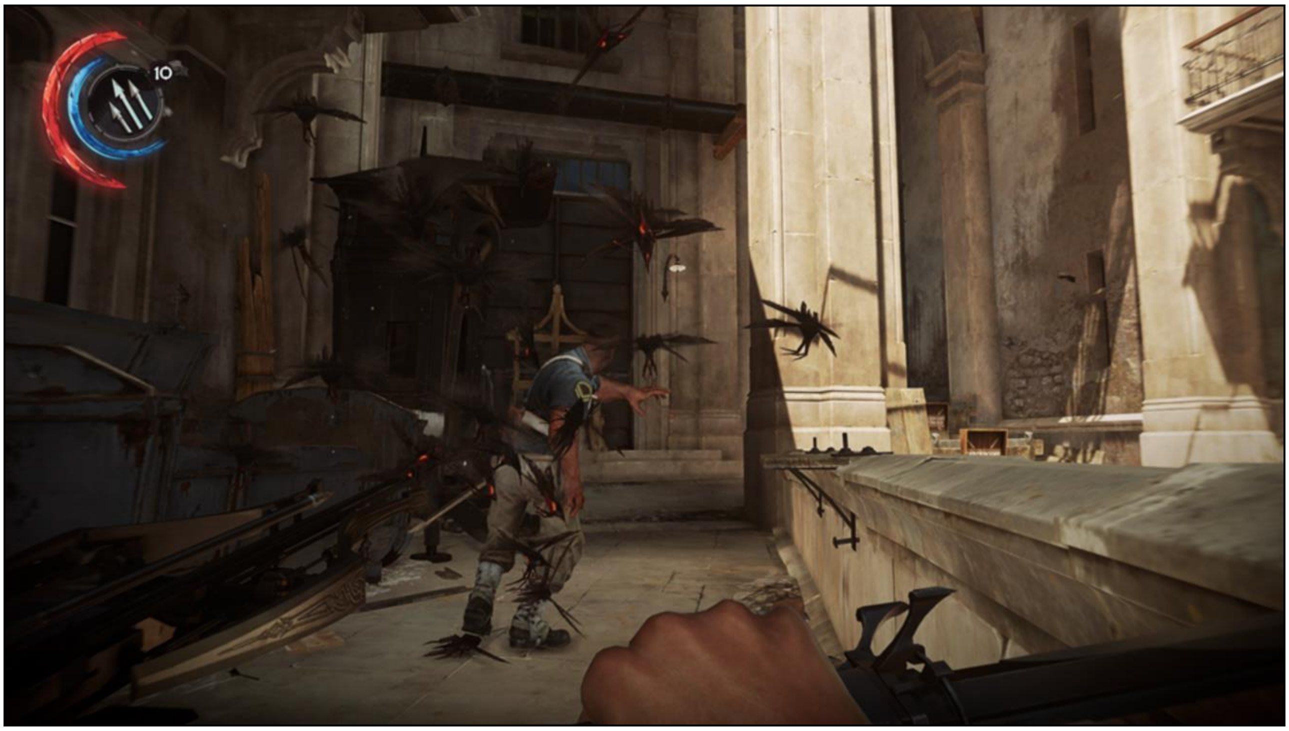
Teleporting player



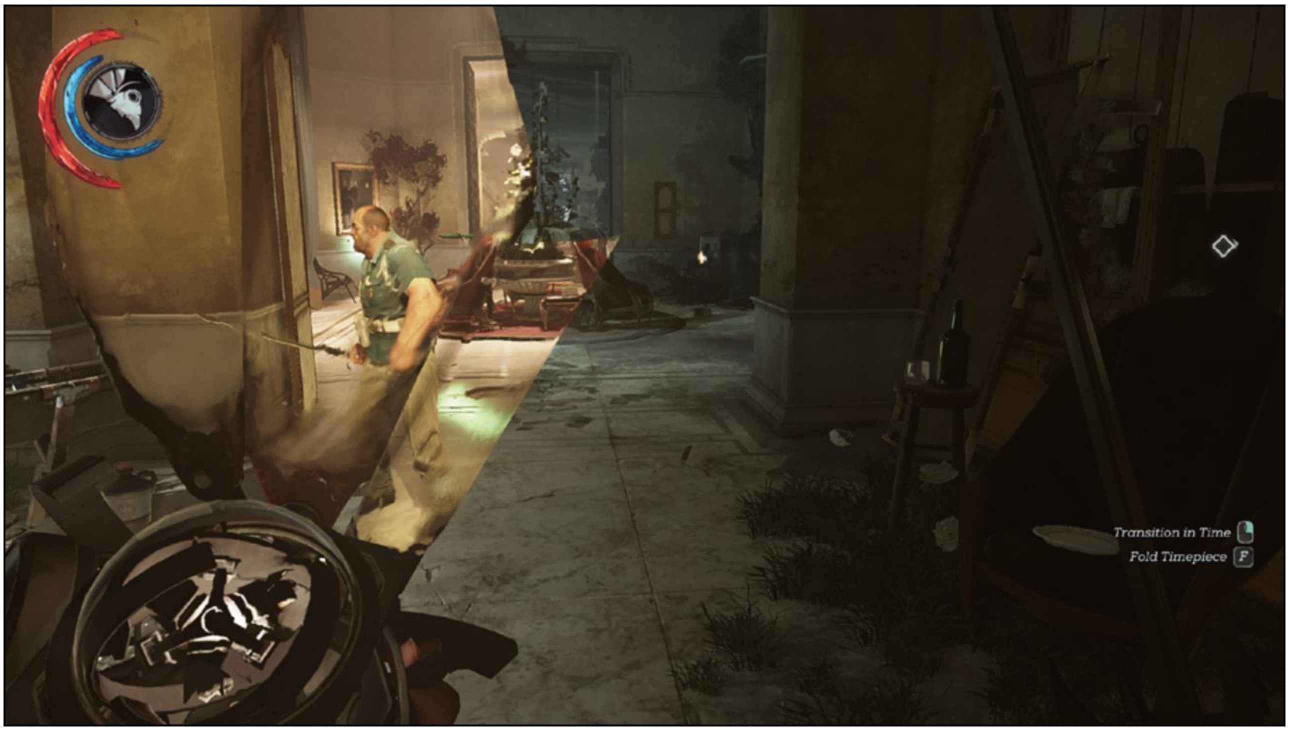
Teleporting NPCs



Ambush behavior

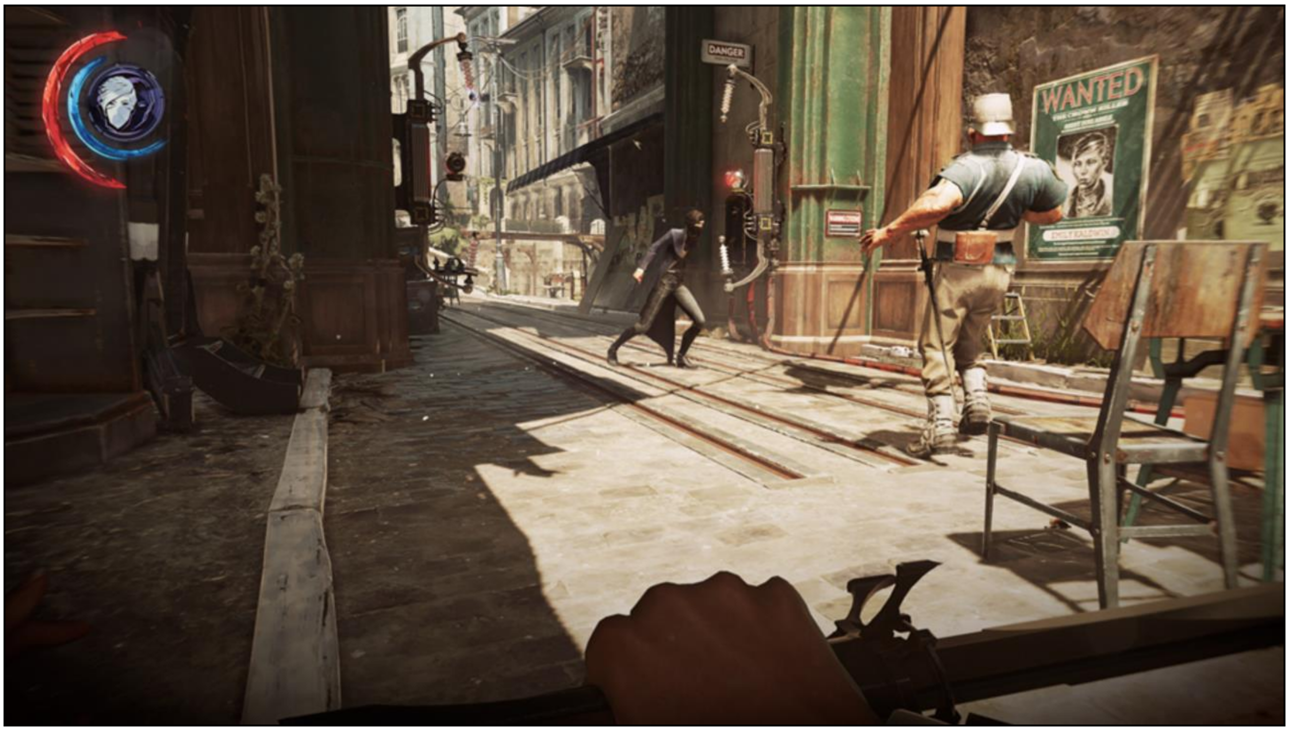


NPCs interacting with bloodflies

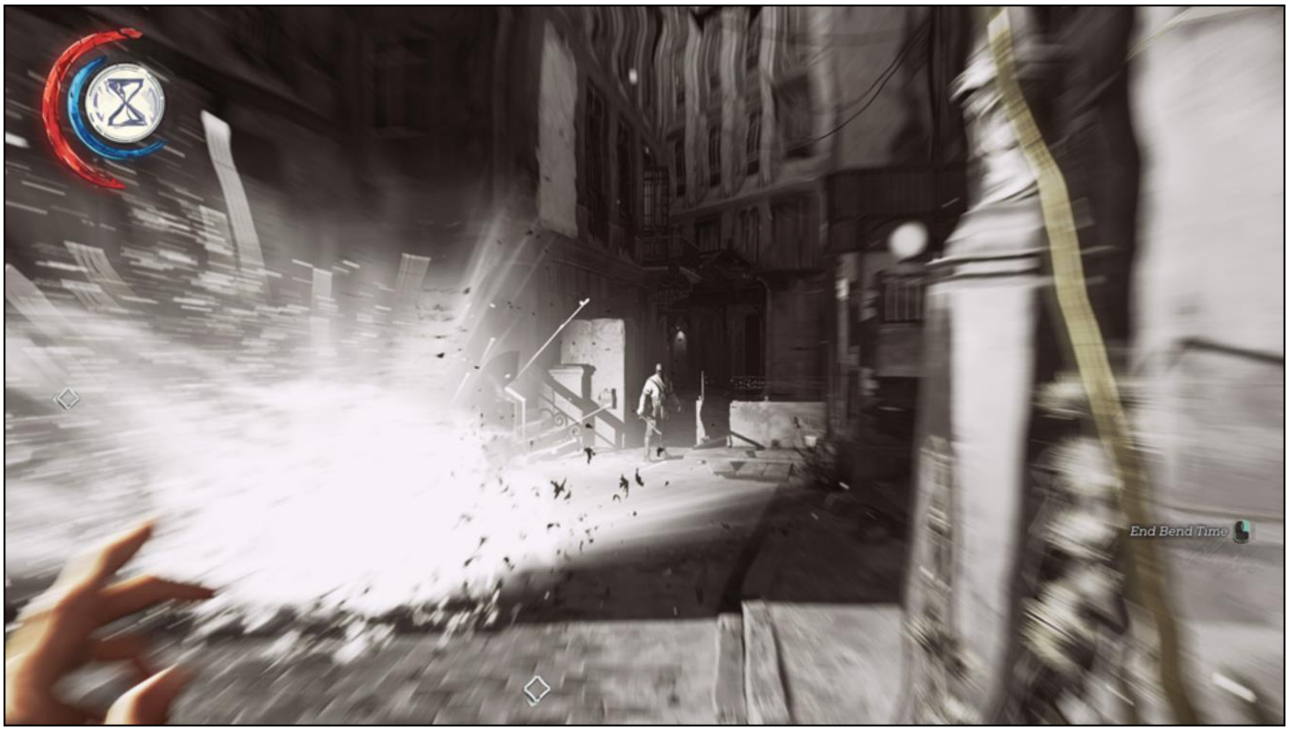


Time travel

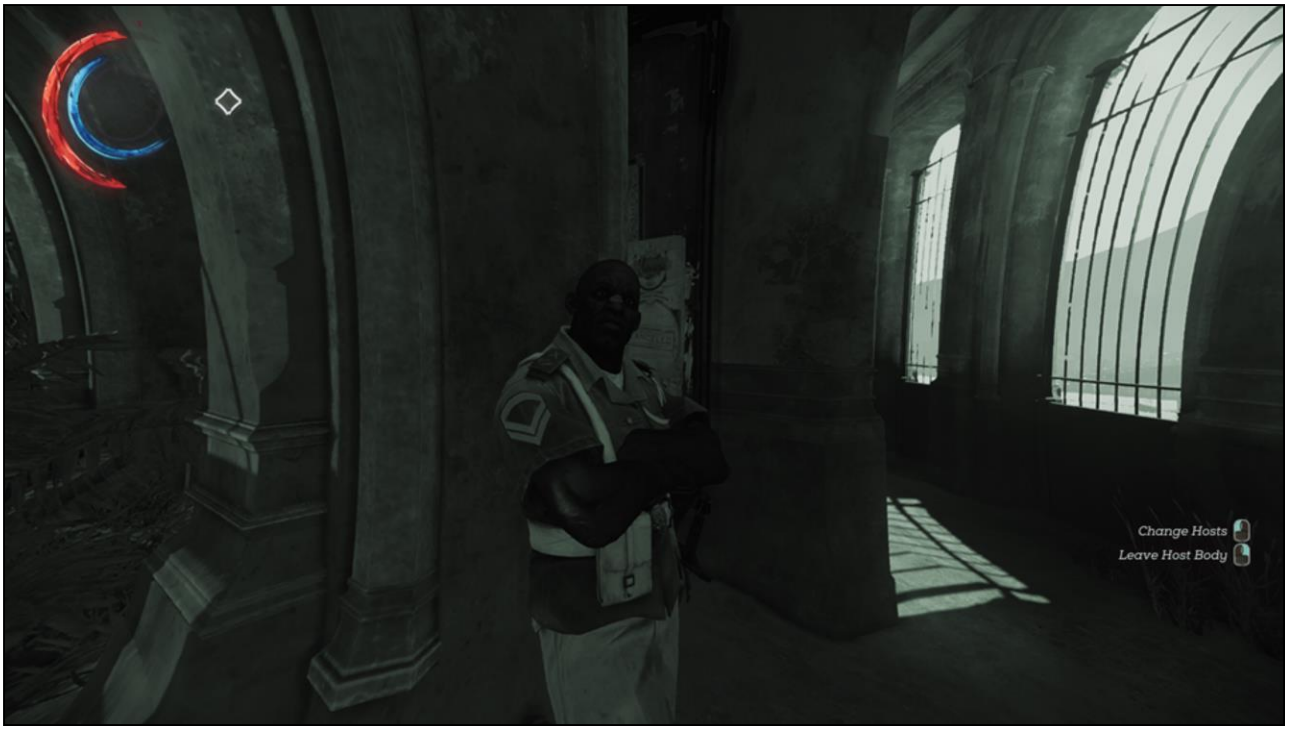
Moving walls, and floors and roof!



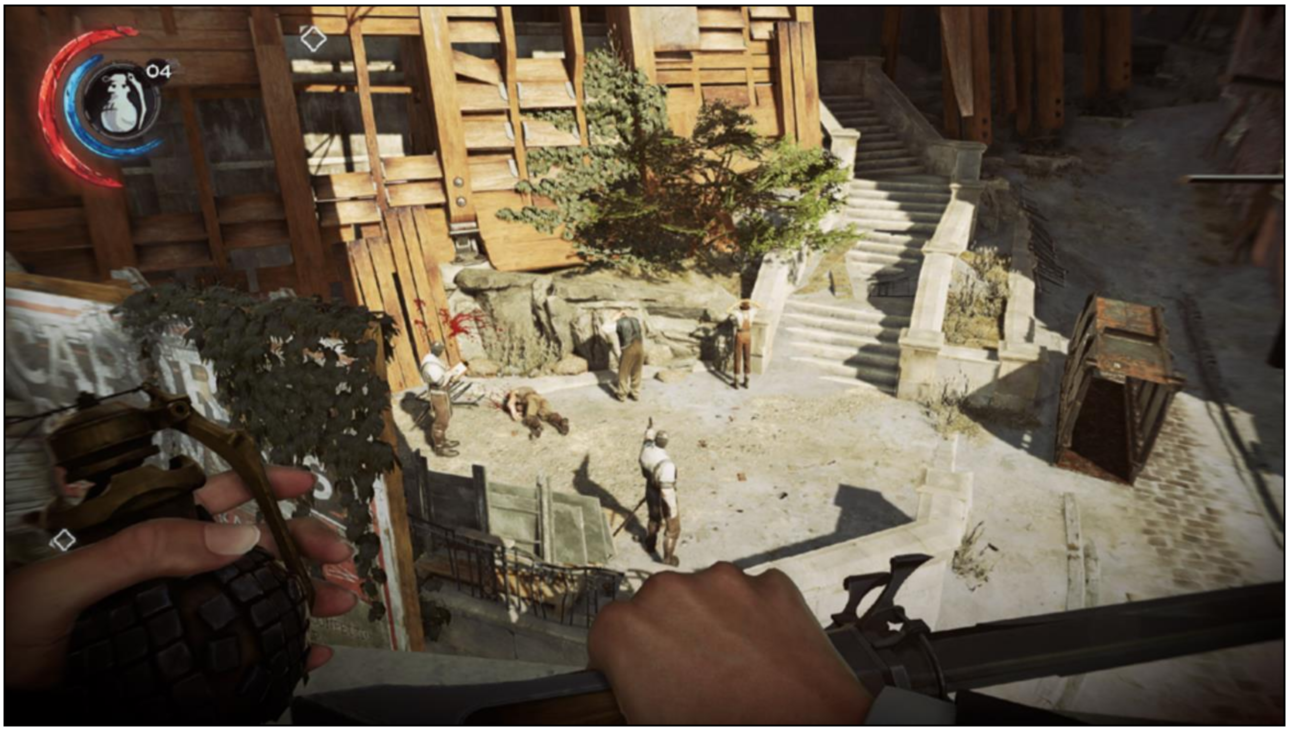
Player clone



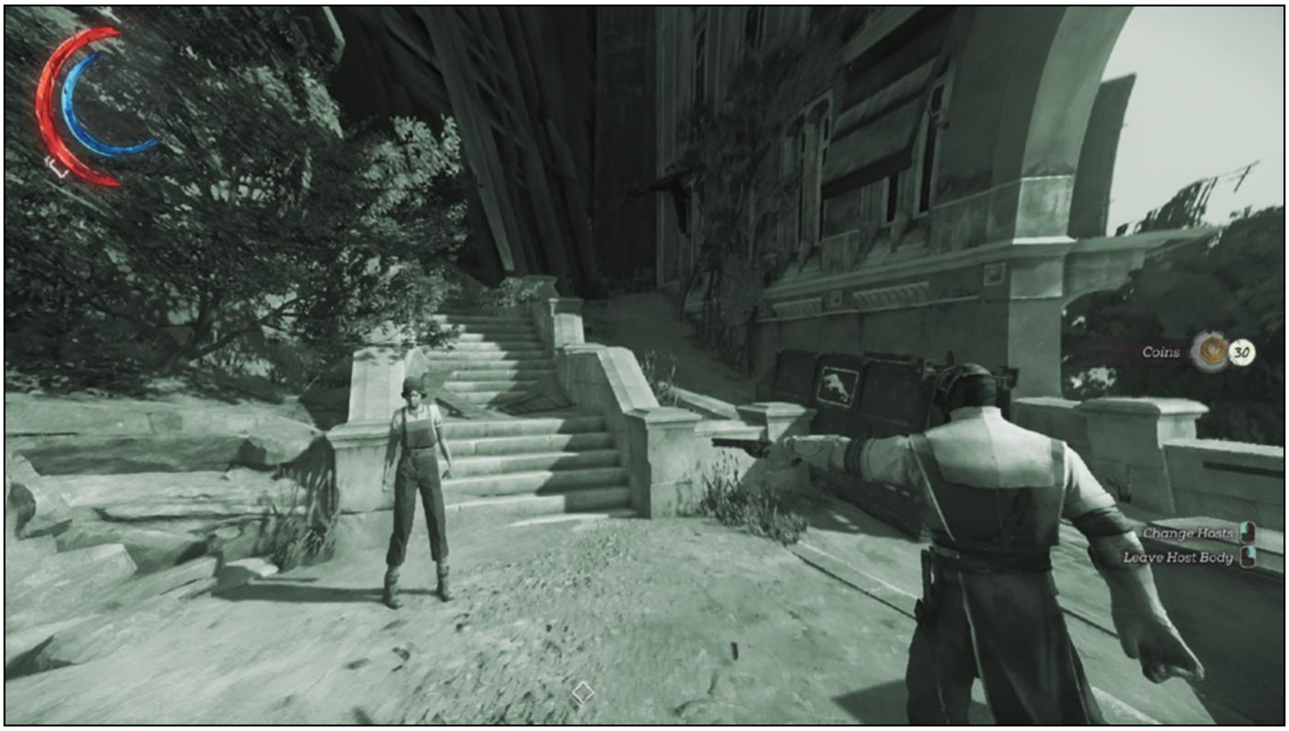
Time stop



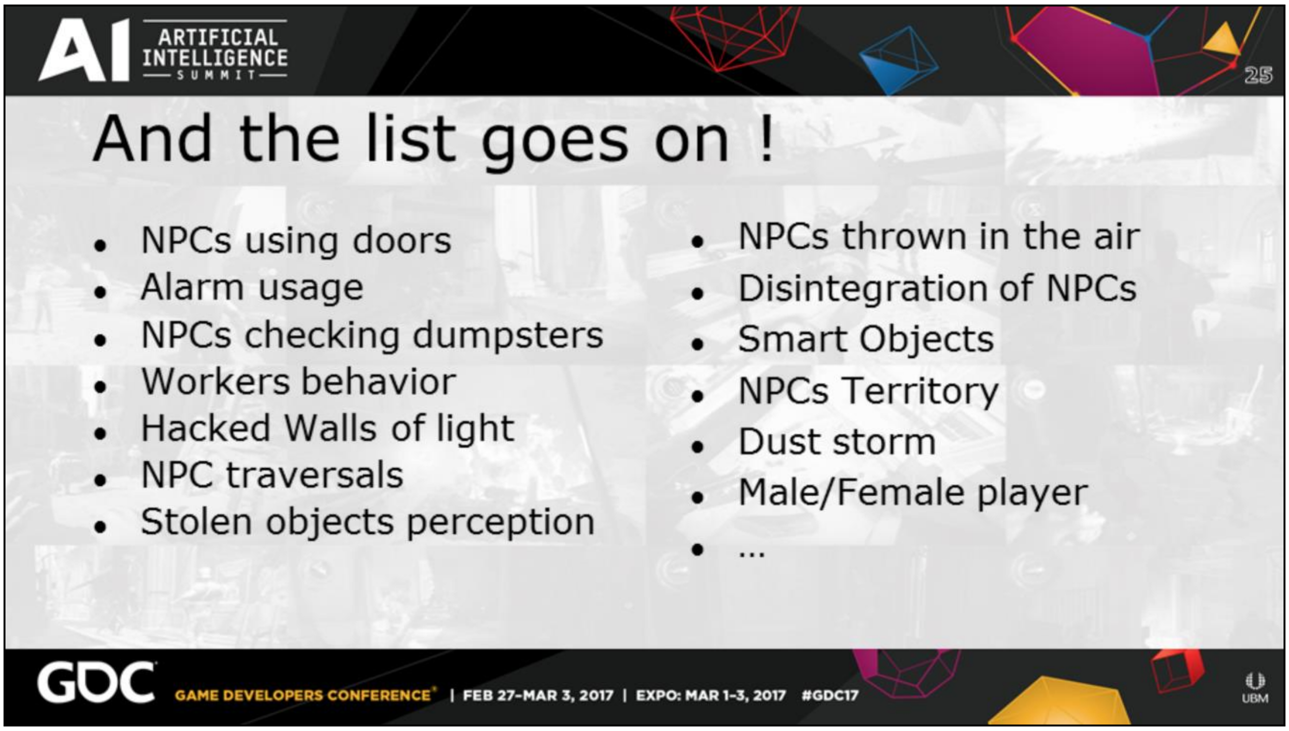
Player taking control of NPCs



Fully interactable scripted scenes



And what about fully interactable scripted scenes, while the player takes control of a participating NPC?


ARTIFICIAL INTELLIGENCE SUMMIT

25

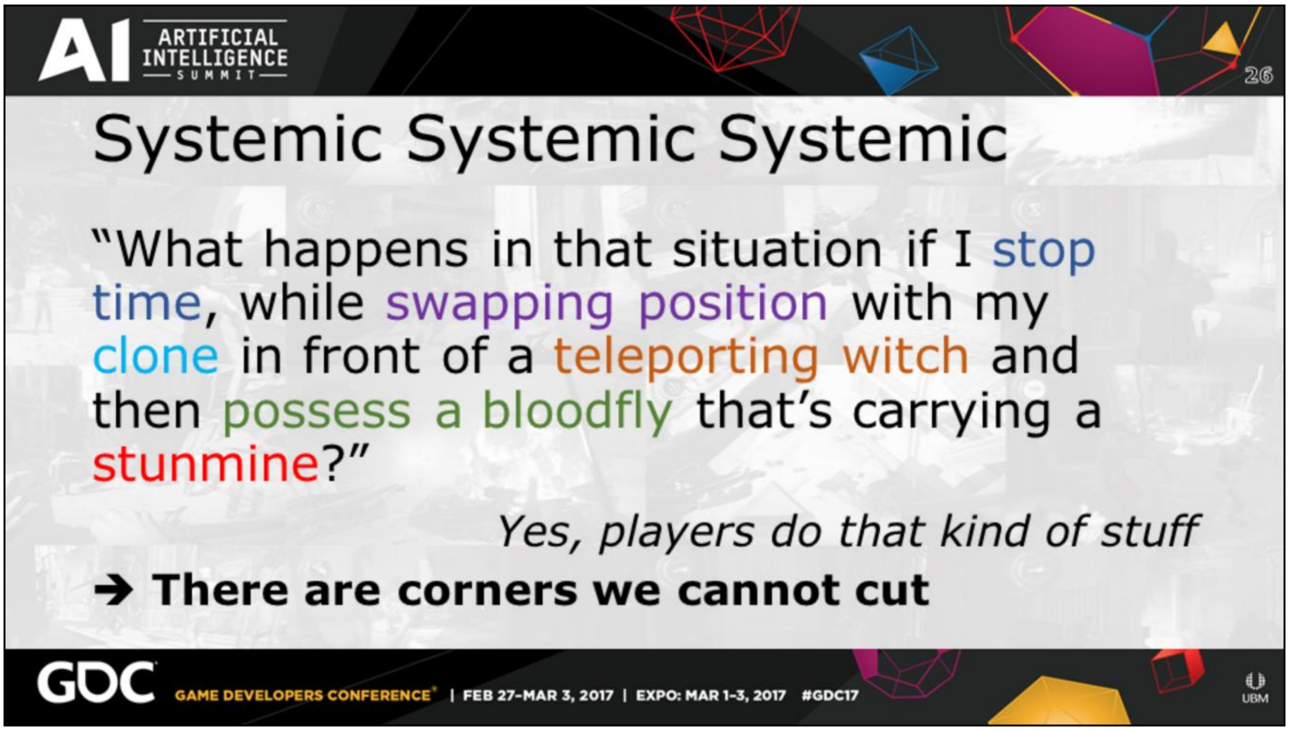
And the list goes on !

- NPCs using doors
- Alarm usage
- NPCs checking dumpsters
- Workers behavior
- Hacked Walls of light
- NPC traversals
- Stolen objects perception
- NPCs thrown in the air
- Disintegration of NPCs
- Smart Objects
- NPCs Territory
- Dust storm
- Male/Female player
- …

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

Basically we have all the problems of action/adventure games, and much more


ARTIFICIAL INTELLIGENCE SUMMIT

Systemic Systemic Systemic

"What happens in that situation if I stop time, while swapping position with my clone in front of a teleporting witch and then possess a bloodfly that's carrying a stunmine?"

Yes, players do that kind of stuff

➔ **There are corners we cannot cut**

Thus we had to rely very heavily on simulation, because in every feature review, there is always some guy in the team, to ask something like this:

"Well this is cool, but: <enounce>"

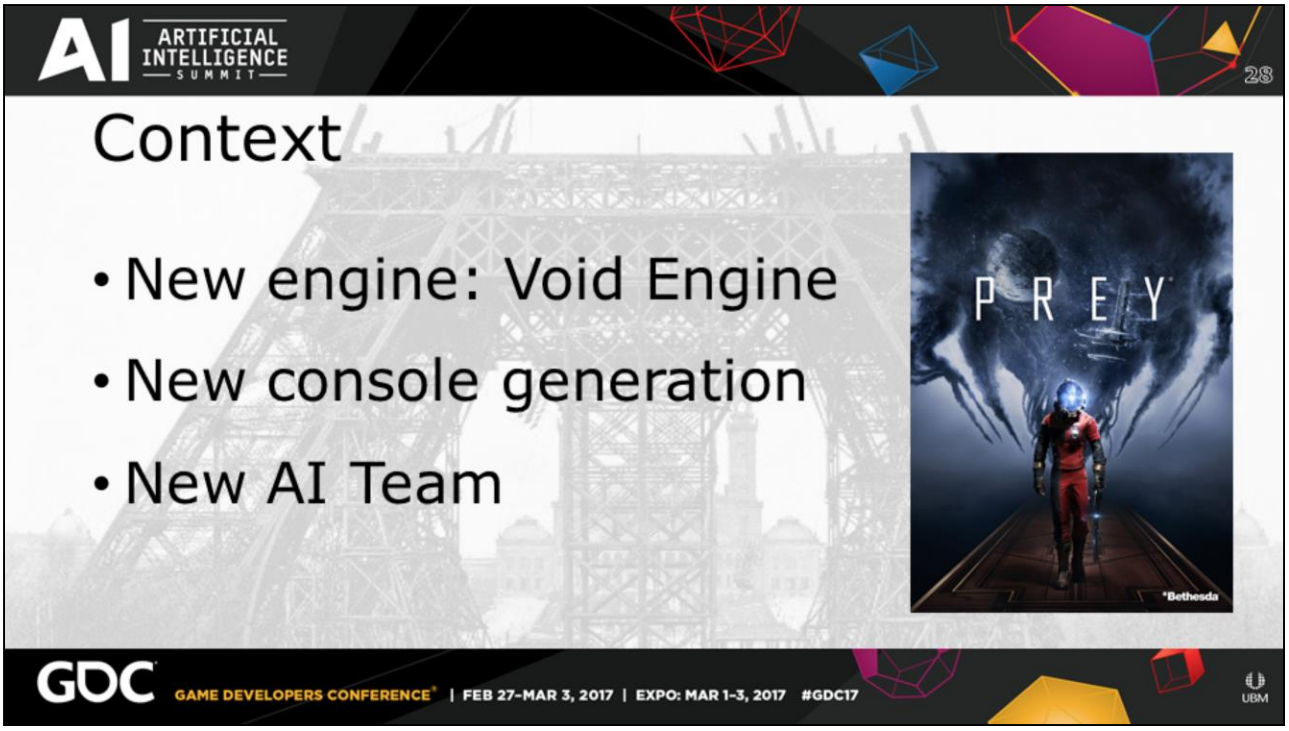
Every. Time.

And the thing is that it's totally possible to do that kind of crazy stuff with our game. The bottom line being: there are some corners that we simply cannot afford to cut. There are many, many edge cases. We had to do many things the hard way, to ensure all the systems worked together.

And in order to implement all those complicated systems, we were starting from…

Built from the ground up

…nothing.



Context

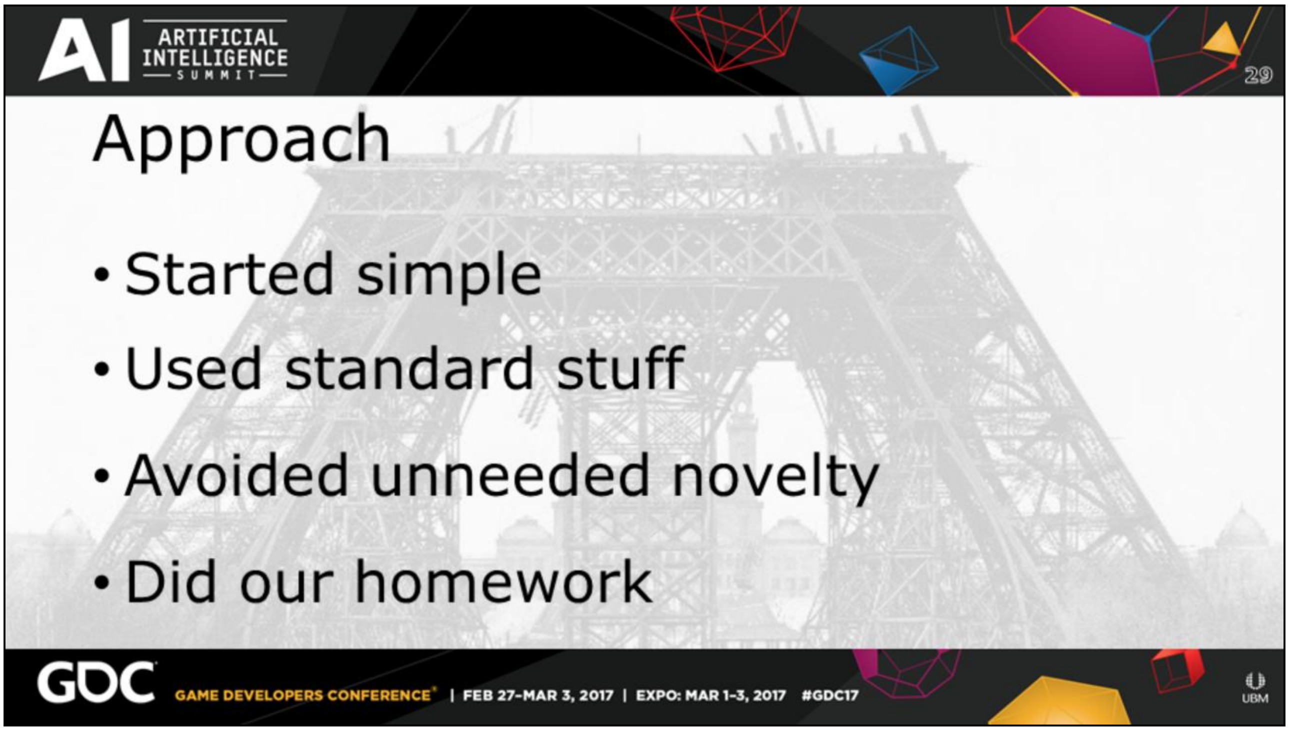
- New engine: Void Engine
- New console generation
- New AI Team

AI ARTIFICIAL INTELLIGENCE SUMMIT
28
GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
UBM

Beginning Dishonored 2 we chose to build a new engine, starting from id tech 5, and reworking it to fit our kind of games. We rewrote the AI part from scratch.

We also planned to take full advantage of the new generation hardware, Dishonored 1 being initially last gen only.

And as the Arkane Austin guys went to another project (which is a pretty cool game called Prey), there was only one AI programmer remaining from Dishonored 1, so we basically had to rebuild an AI team.



Approach

- Started simple
- Used standard stuff
- Avoided unneeded novelty
- Did our homework

ARTIFICIAL INTELLIGENCE SUMMIT
29
GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

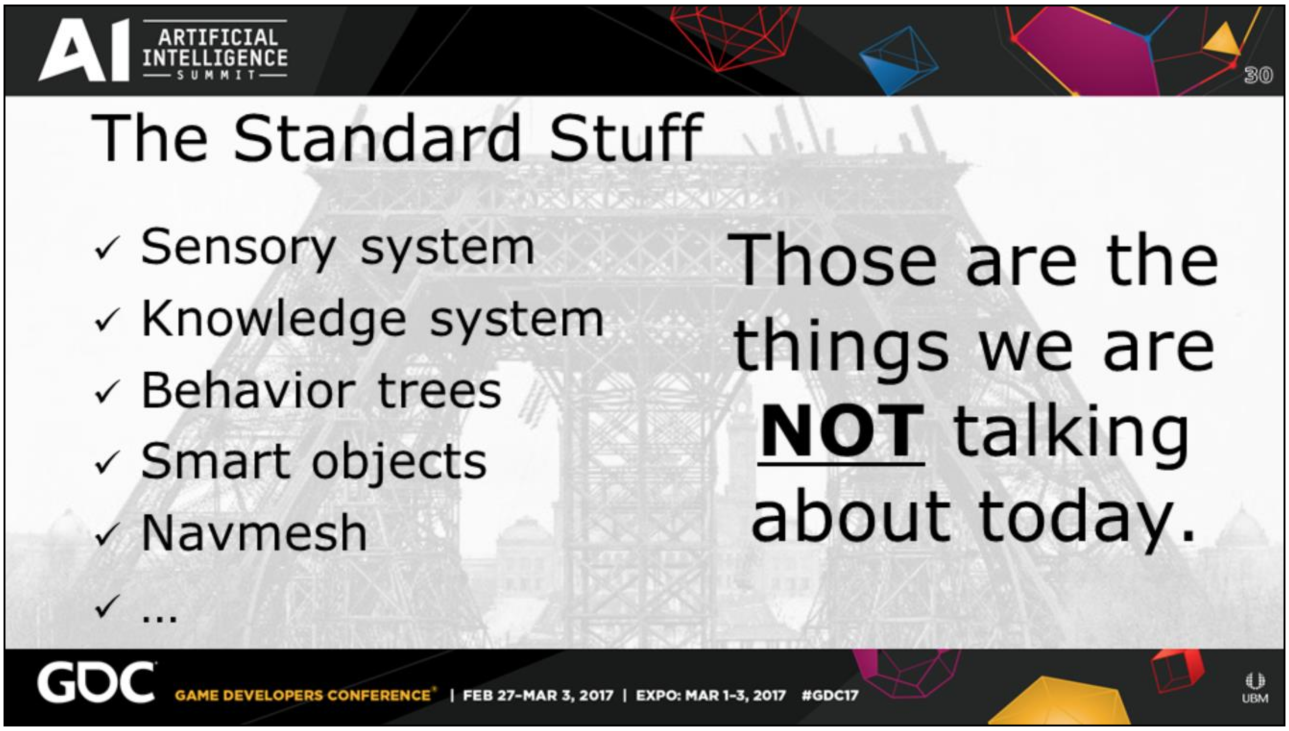
In that context, we decided to always try simple things first, to avoid over complication at all cost.

We chose tried and true techniques, standard ways of organizing our AI systems, known stuff with ample literature available.

We tried to step away from unneeded new shiny things.

And very importantly we did our homework, spent time writing technical documents, thinking on paper, writing unit tests for our systems.

Speaking about systems…


ARTIFICIAL INTELLIGENCE SUMMIT

30

The Standard Stuff

- ✓ Sensory system
- ✓ Knowledge system
- ✓ Behavior trees
- ✓ Smart objects
- ✓ Navmesh
- ✓ …

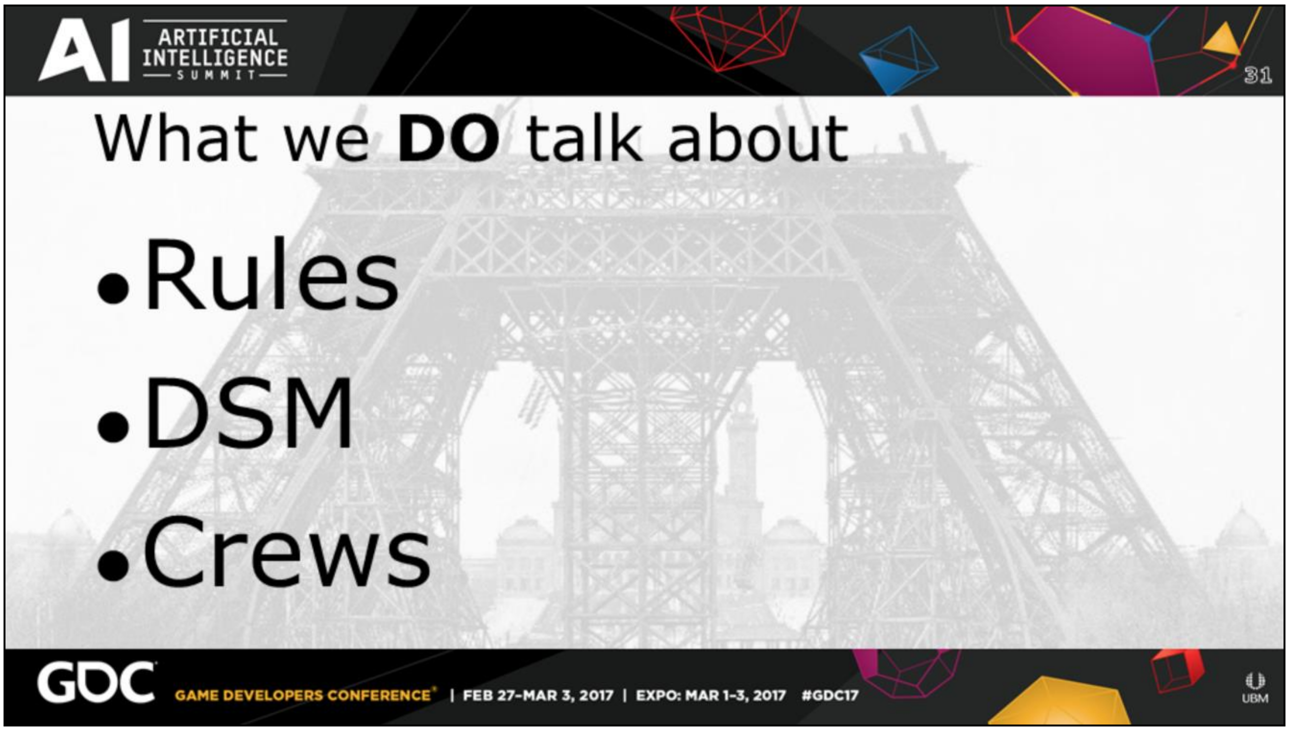
Those are the things we are **NOT** talking about today.

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

.. and standard stuff, we’ve got it all, <enumerate>, everything.

But guess what… those are not within the scope of this talk.



ARTIFICIAL INTELLIGENCE SUMMIT

31

What we **DO** talk about

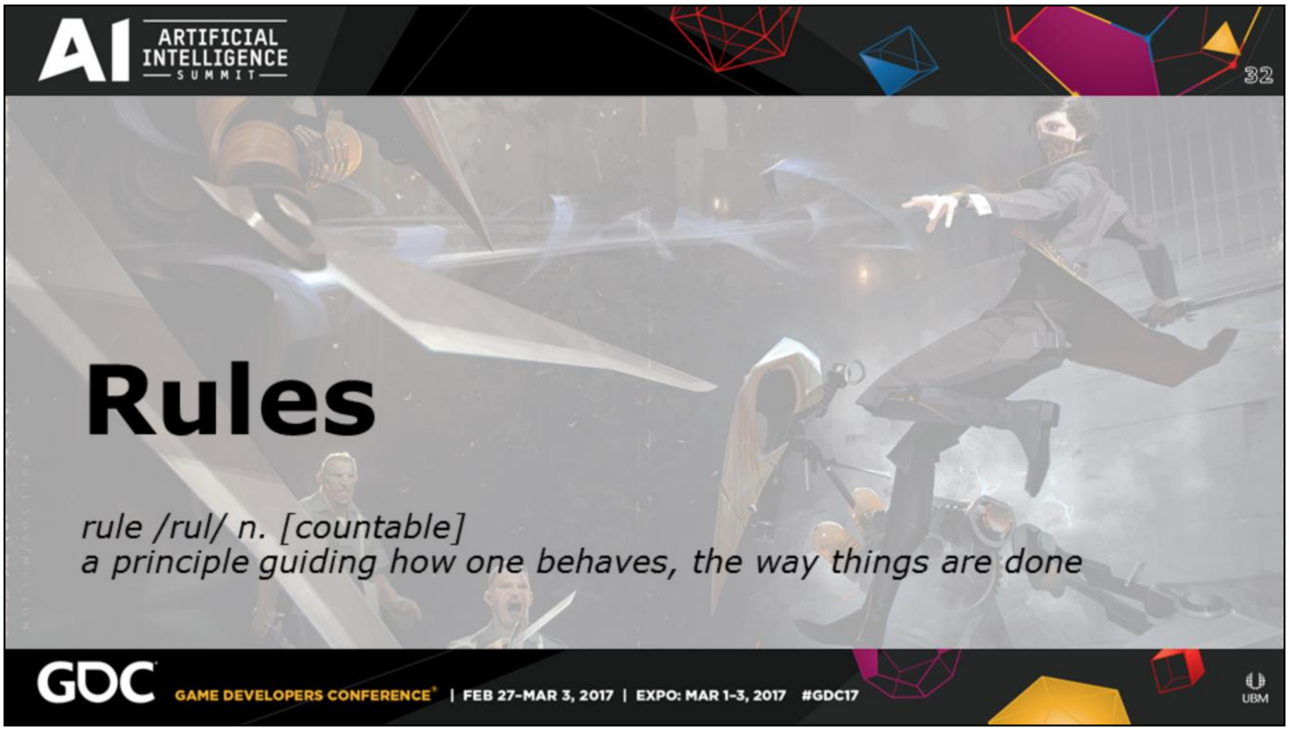
- Rules
- DSM
- Crews

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

What we are going to discuss today, are 3 systems where we did things a little differently.

Let’s see one of those.



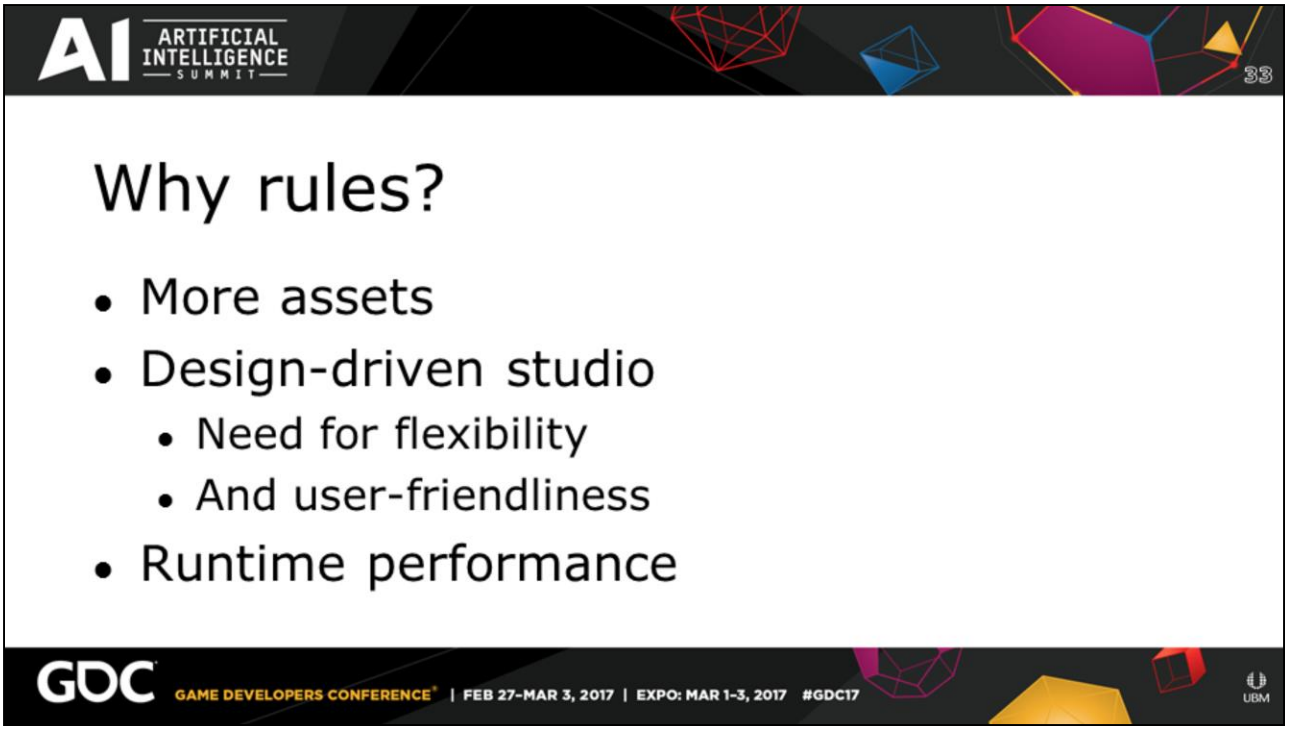
Rules

rule /rul/ n. [countable]
a principle guiding how one behaves, the way things are done

ARTIFICIAL INTELLIGENCE SUMMIT
32
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

<Laurent>

Hi everyone! I'm going to first present a few things about our rule system.


ARTIFICIAL INTELLIGENCE SUMMIT

33

Why rules?

- More assets
- Design-driven studio
  - Need for flexibility
  - And user-friendliness
- Runtime performance

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

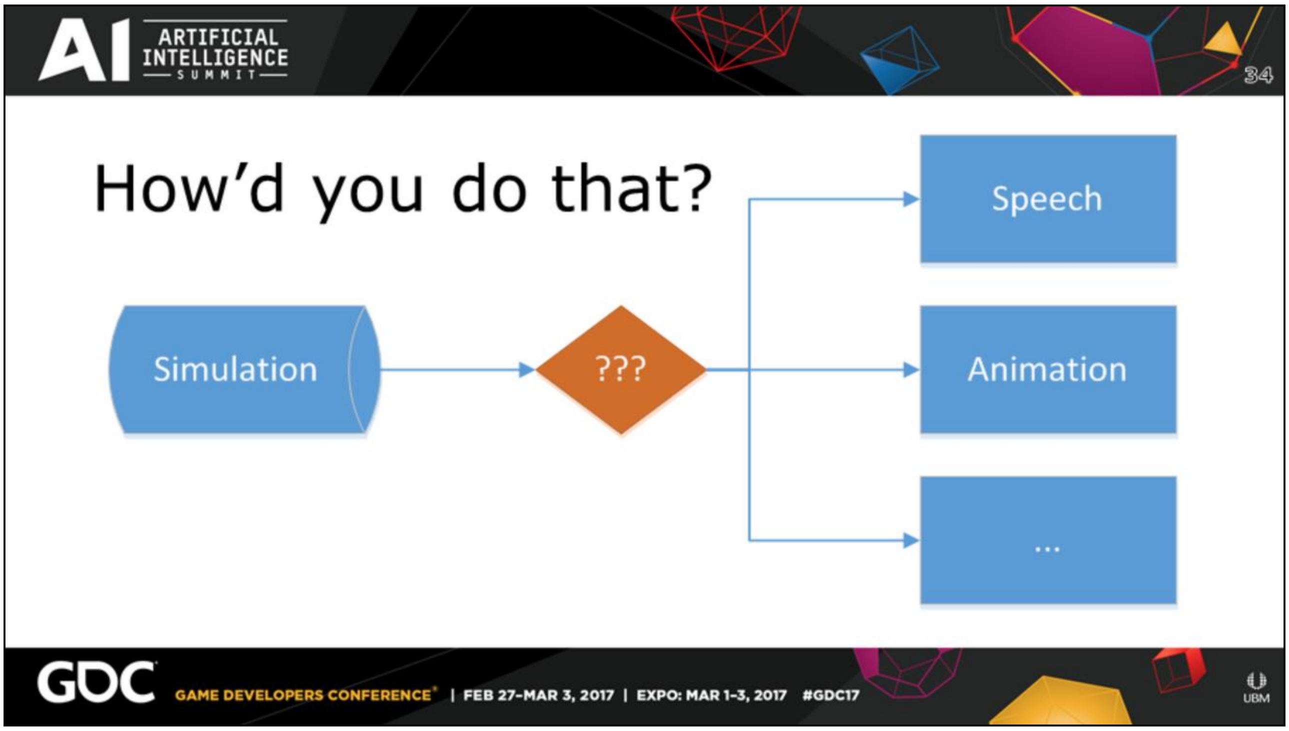
UBM

So: why rules?

Very trivially making a next-gen Dishonored which such a large scope meant more assets to present.

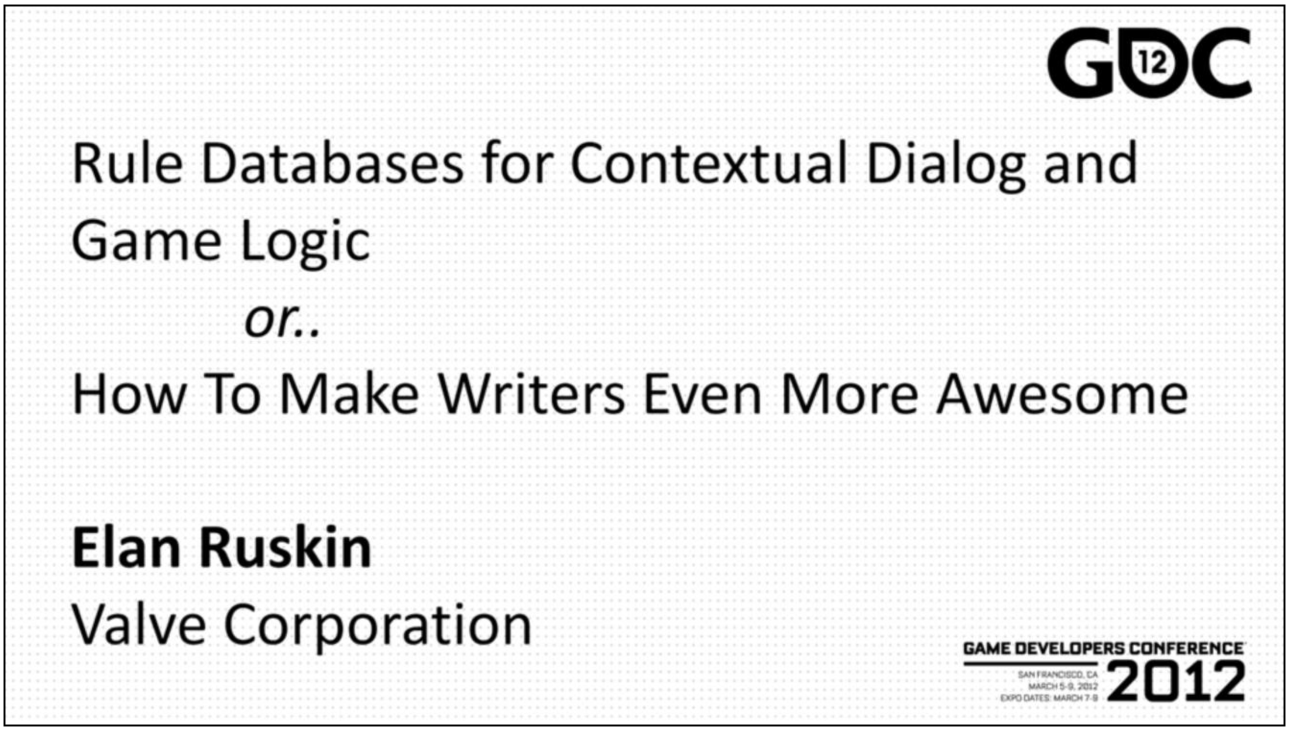
And we’re a very design-driven studio. Something our designers expressed was their desire to have more control over how voice recordings and animations are used by the game. Having to make a code change each time an asset needs to be integrated is very bad. And as we didn’t want to turn them into programmers either, so we had to make sure we’d come up with something user-friendly.

Also last but not least, we needed to find a solution with a performance cost that fits the hardware of the current generation of consoles.


ARTIFICIAL INTELLIGENCE SUMMIT
34
How'd you do that?
Simulation
???
Speech
Animation
...
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

So how do we track all the simulation state, to present these assets in a varied and meaningful way?

This was an open question during pre-production. And because we started our AI tech from scratch, we had the complete freedom to try a new technique.


GDC 12

Rule Databases for Contextual Dialog and Game Logic

or..

How To Make Writers Even More Awesome

Elan Ruskin

Valve Corporation

GAME DEVELOPERS CONFERENCE

SAN FRANCISCO, CA
MARCH 5-9, 2012
EXPO DATES: MARCH 7-9

2012

Well this is where this GDC talk by Elan Ruskin came to help: he presented a rule system used notably in Left4Dead 1 & 2*.

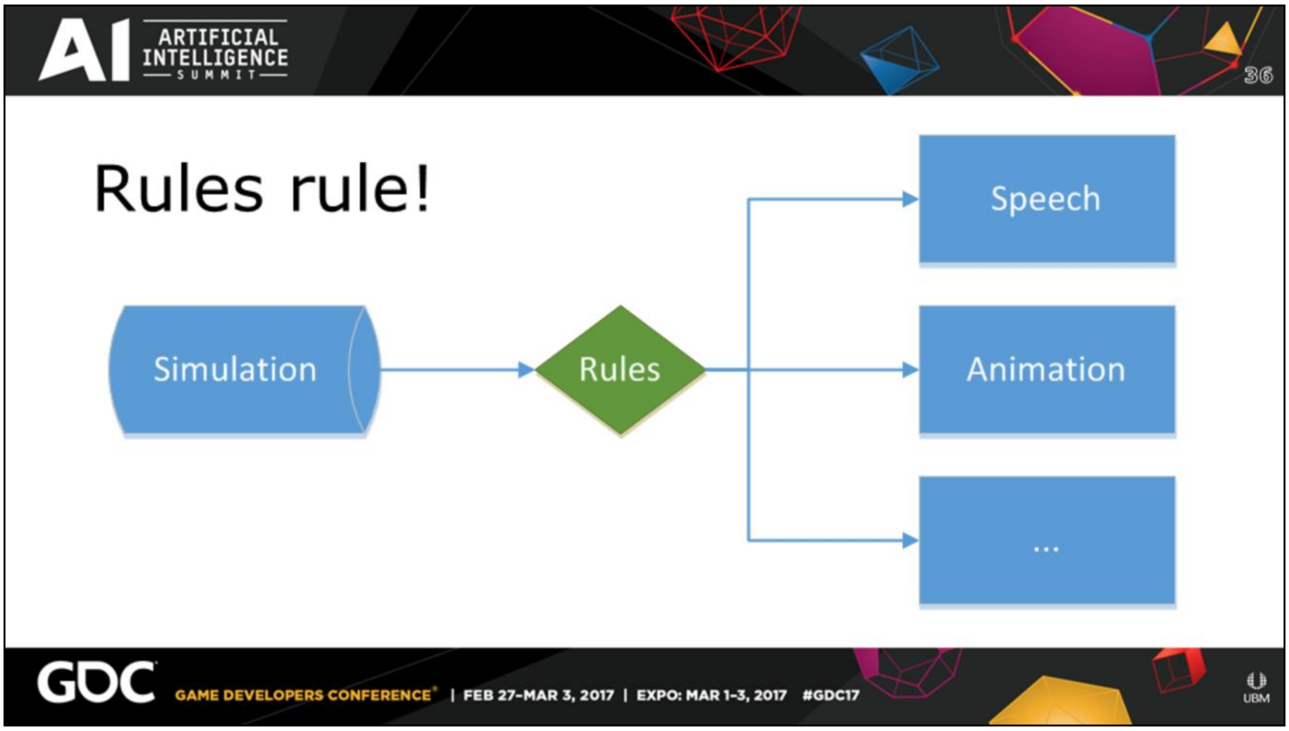
It inspired us, and seemed like a good candidate to help us overcome the challenges I just mentioned. Do yourself a favor and go check it some day.

So just to make things clear from the beginning: we basically stole all of his ideas ☺

---

[*] AI-driven Dynamic Dialog through Fuzzy Pattern Matching. Empower Your Writers!

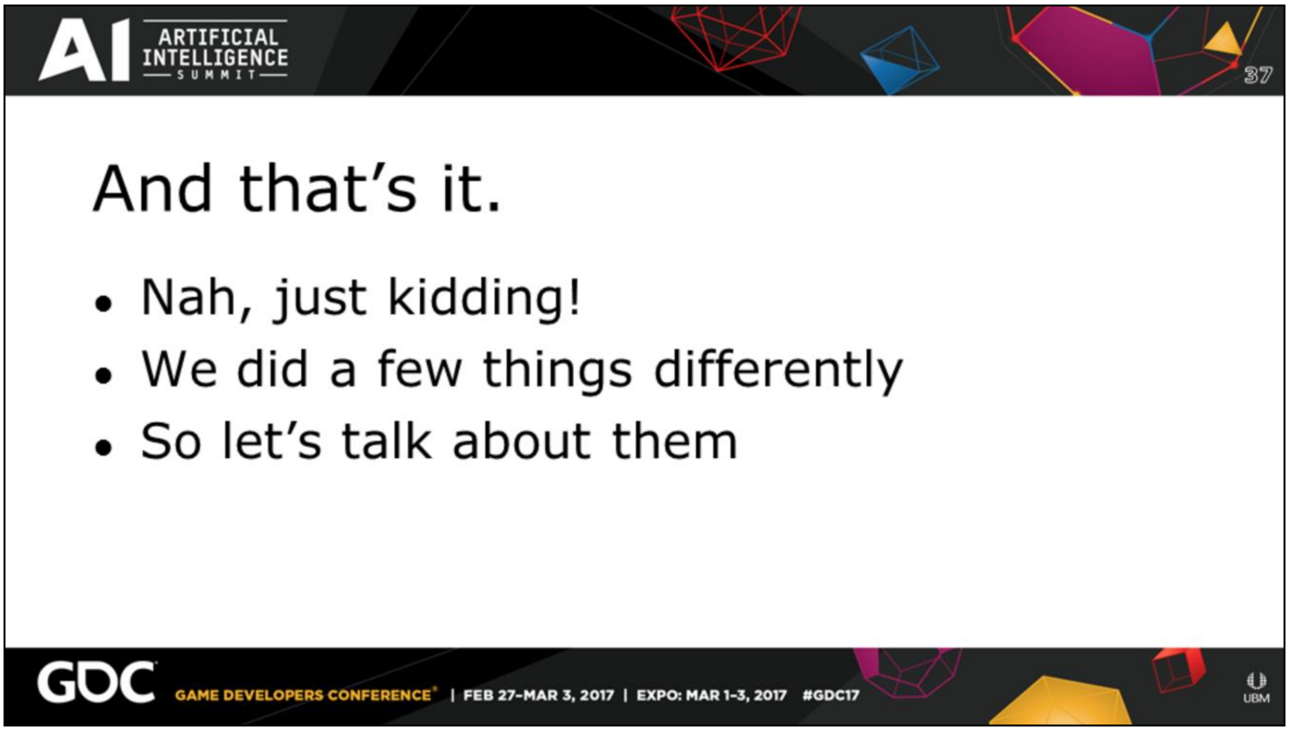
http://www.gdcvault.com/play/1015317/AI-driven-Dynamic-Dialog-through



Rules rule!

Rules were something that met our requirements.

We went on and implemented a similar system in our new engine, the Void Engine.


ARTIFICIAL INTELLIGENCE SUMMIT

37

And that’s it.

- Nah, just kidding!
- We did a few things differently
- So let’s talk about them

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

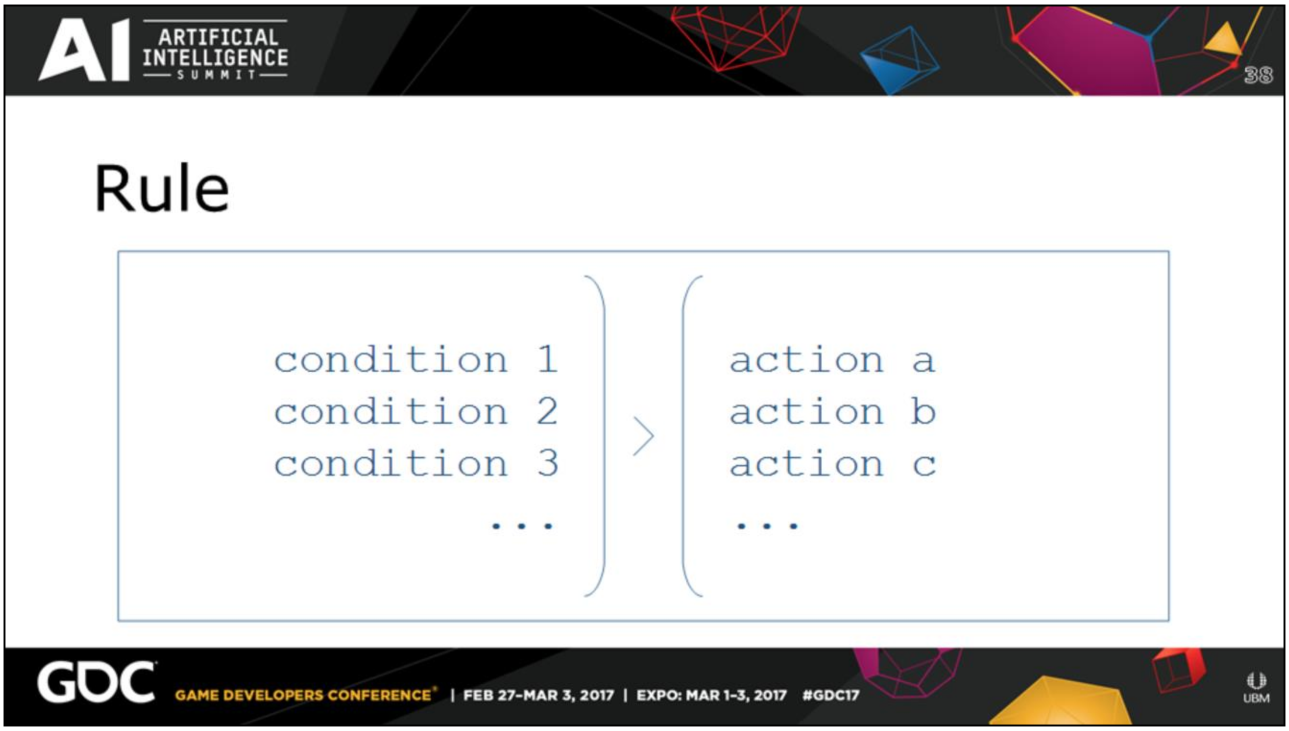
So that’s it for rules. Thank you for listening.

…

Nah just kidding!

I said a similar system, not identical. Like every team, we have our own requirements so we did a few things differently.

So my goal here is to give you details on the specifics of our system.



ARTIFICIAL INTELLIGENCE SUMMIT

38

Rule

```
condition 1  \    /  action a
condition 2   >  (   action b
condition 3  /    \  action c
        ...           ...
```

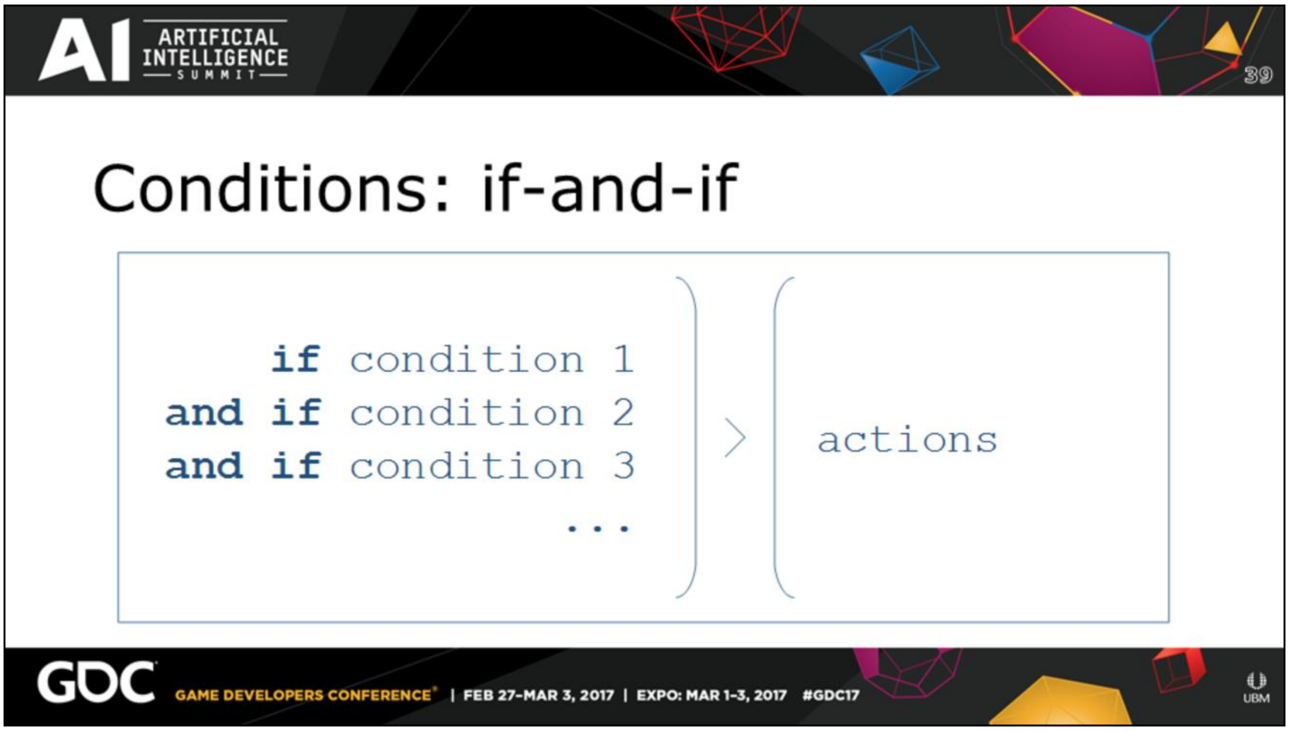
GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

But first, let's go over the terminology and basic principles of our system.

We define a rule as the combination of two collections:

- a list of conditions,
- and a list of actions.



ARTIFICIAL INTELLIGENCE SUMMIT

39

Conditions: if-and-if

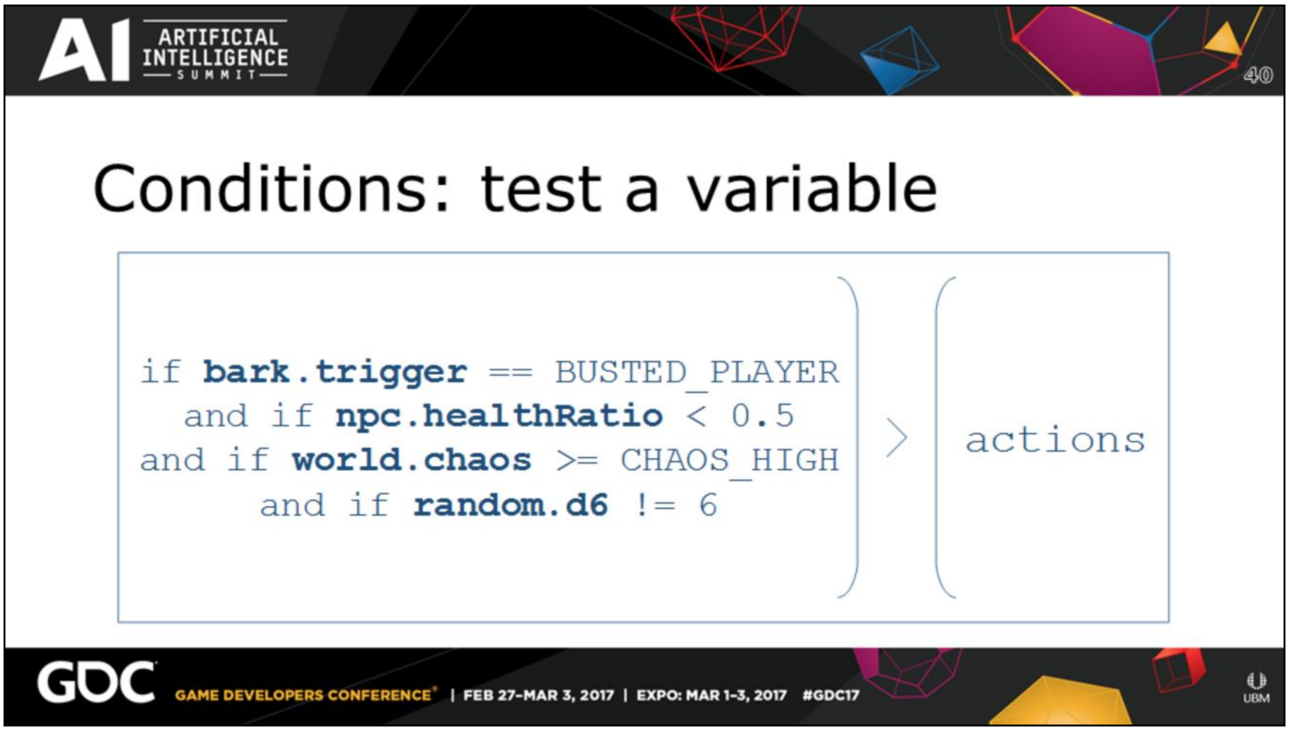
```
    if condition 1
and if condition 2      actions
and if condition 3
               ...
```

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

Conditions are predicates. We combine them in what is just a glorified if-and-if conditional.

They poll the state of the simulation, in other words the current game context.



Conditions: test a variable

```
if bark.trigger == BUSTED_PLAYER
  and if npc.healthRatio < 0.5
and if world.chaos >= CHAOS_HIGH
      and if random.d6 != 6
```

> actions

GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Concretely, a condition is the comparizon between a named variables and a constant value.

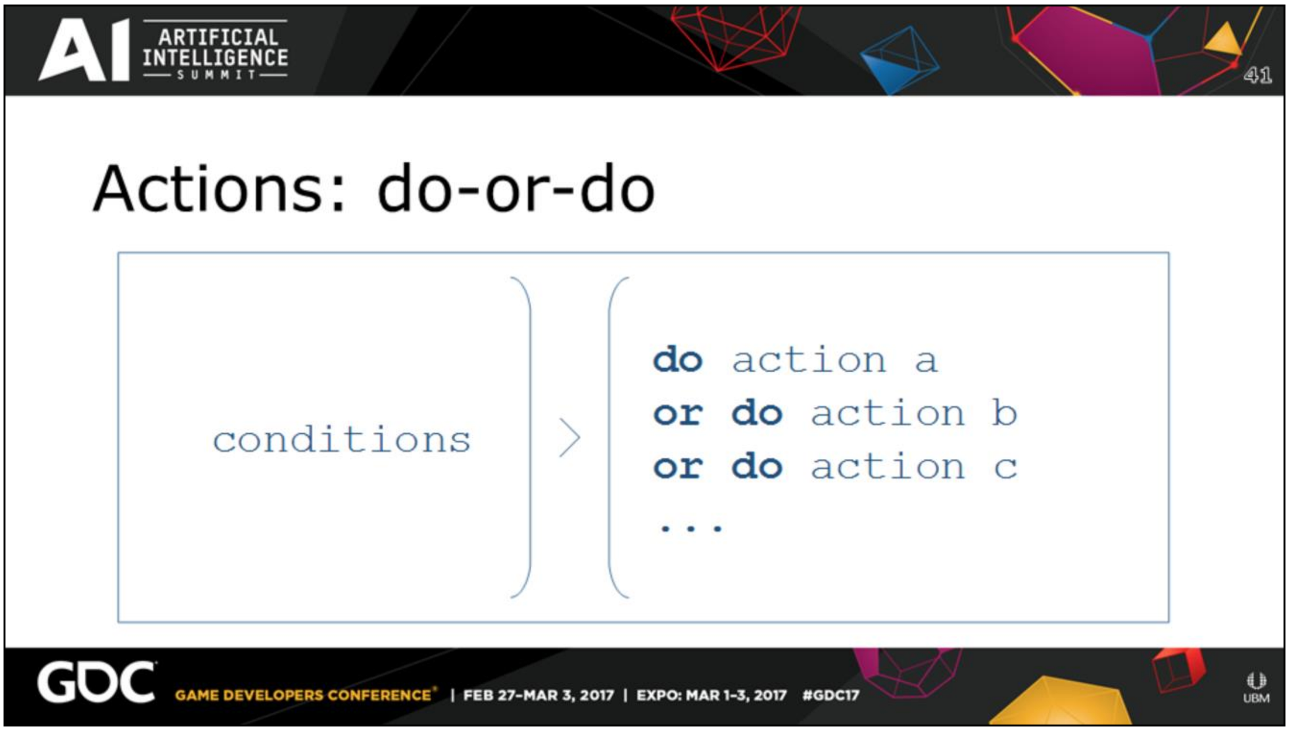
Here's an example.

We can have a rule matching when an NPC barks because they just busted the player. So we're testing the `bark.trigger` variable that tells us why we're trying to bark now.

And let's test a few more variables:

- Is the current HP a bit low?
- Is the world is in high or very high chaos?
- And let's match this only 5 times out of 6.

Note that we only have logical "ands". This removes the need to handle operator precedence. It also makes evaluating the result trivial: a rule matches when every condition matches.



ARTIFICIAL INTELLIGENCE SUMMIT

41

Actions: do-or-do

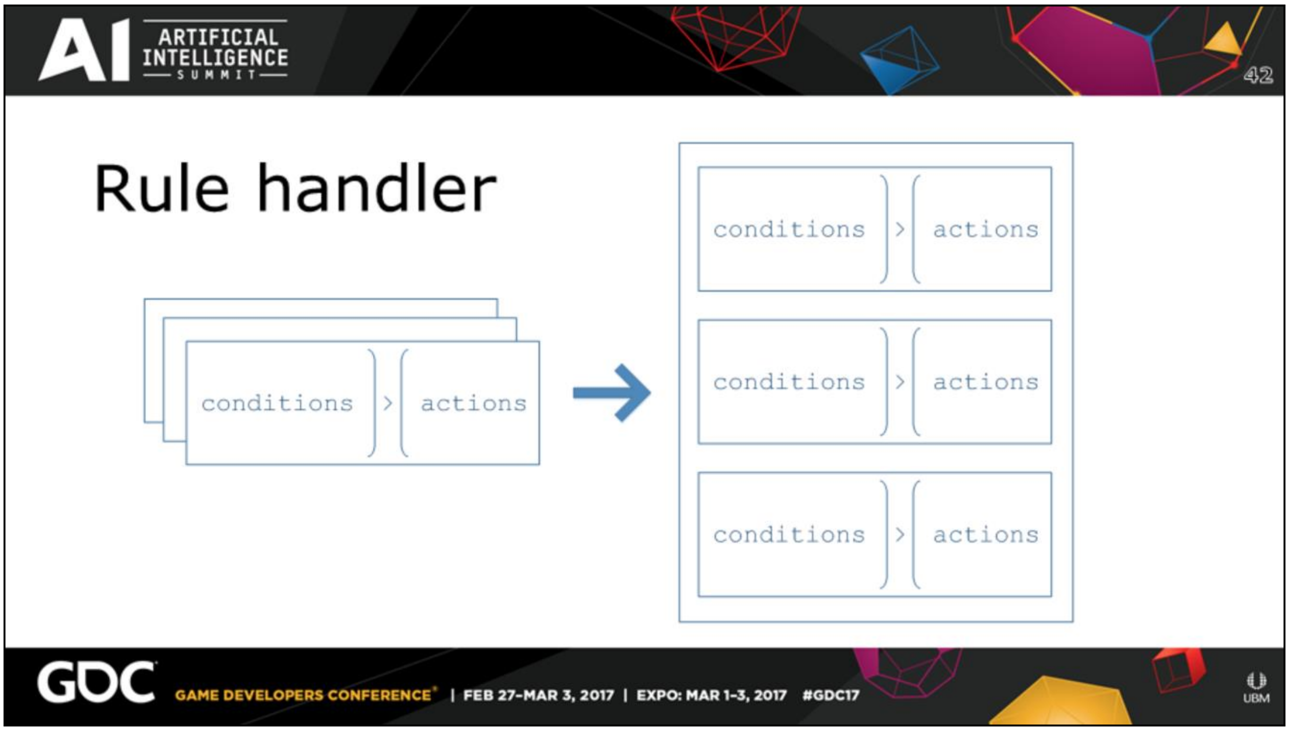
```
                 do action a
conditions  >    or do action b
                 or do action c
                 ...
```

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

Now to actions: they can be virtually anything. As you saw earlier, for us it’s mostly playing lines of speech or animations, but it could be anything else.

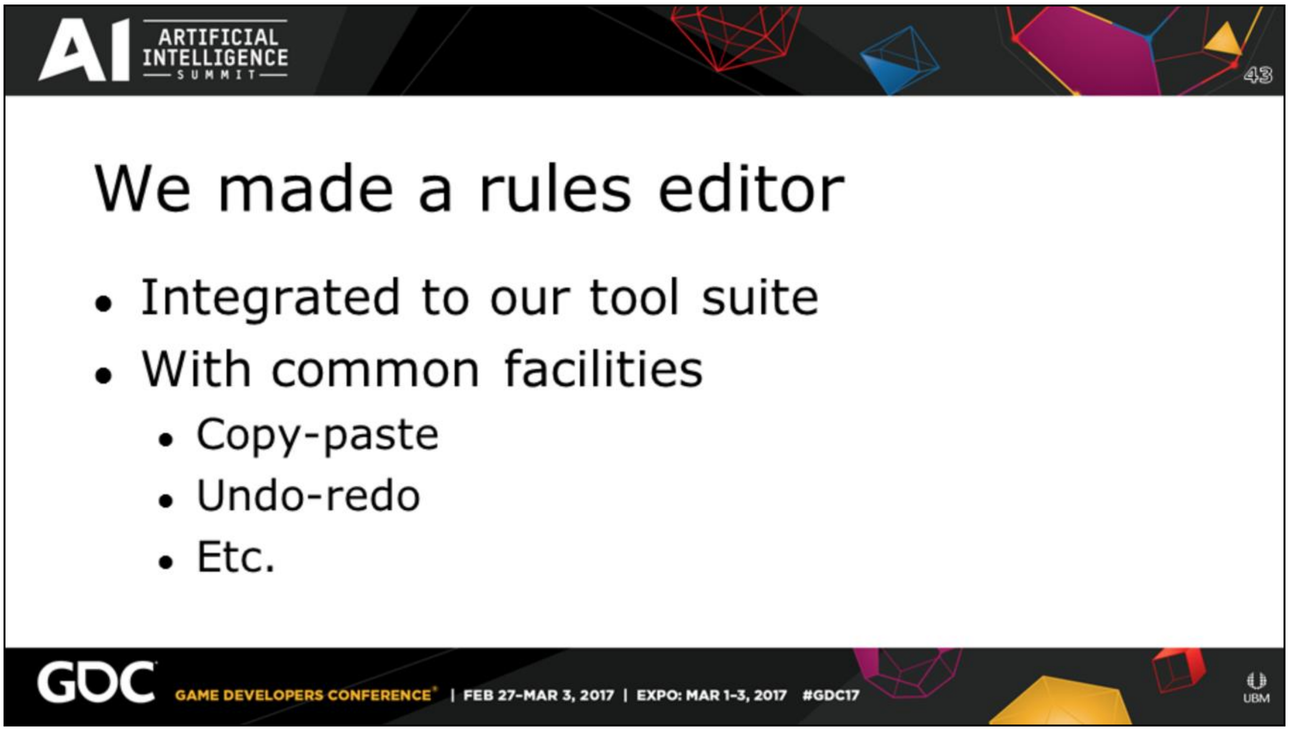
When there are several actions, they’re picked one at a time. Every time a rule matches, only one *single* action is executed. When there’s enough assets, it’s a great tool to provide variety in a an explicit way.



Rule handler

Finally, if you take a bunch of these rules and stick them together, you get a rule handler.

This is just a set of rules, our editable resource.



We made a rules editor

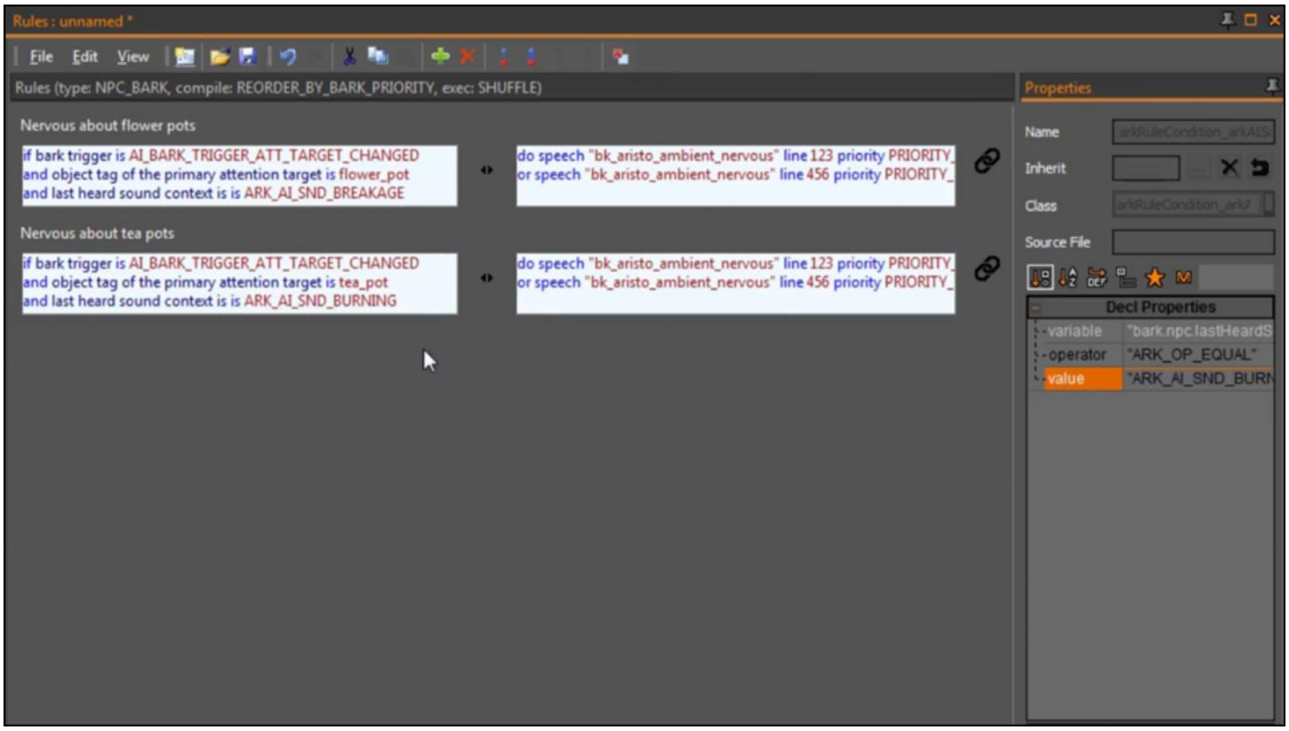
- Integrated to our tool suite
- With common facilities
  - Copy-paste
  - Undo-redo
  - Etc.

So let’s talk about editing. One thing we made is an editor.

Elan: we followed your advice, and listened to our designers. Their wish was a tool integrated into our in-house editor.

And I already told you about designers, right? They want everything. So it had to come with all the handy features that make such a tool usable: copy-paste, undo-redo, and so on.

So we created just that. Let’s see the rules editor in action quickly.



```
Rules (type: NPC_BARK, compile: REORDER_BY_BARK_PRIORITY, exec: SHUFFLE)

Nervous about flower pots
if bark trigger is AI_BARK_TRIGGER_ATT_TARGET_CHANGED
and object tag of the primary attention target is flower_pot
and last heard sound context is is ARK_AI_SND_BREAKAGE
do speech "bk_aristo_ambient_nervous" line 123 priority PRIORITY_
or speech "bk_aristo_ambient_nervous" line 456 priority PRIORITY_

Nervous about tea pots
if bark trigger is AI_BARK_TRIGGER_ATT_TARGET_CHANGED
and object tag of the primary attention target is tea_pot
and last heard sound context is is ARK_AI_SND_BURNING
do speech "bk_aristo_ambient_nervous" line 123 priority PRIORITY_
or speech "bk_aristo_ambient_nervous" line 456 priority PRIORITY_
```

Decl Properties	
variable	"bark.npc.lastHeardS
operator	"ARK_OP_EQUAL"
value	"ARK_AI_SND_BURN

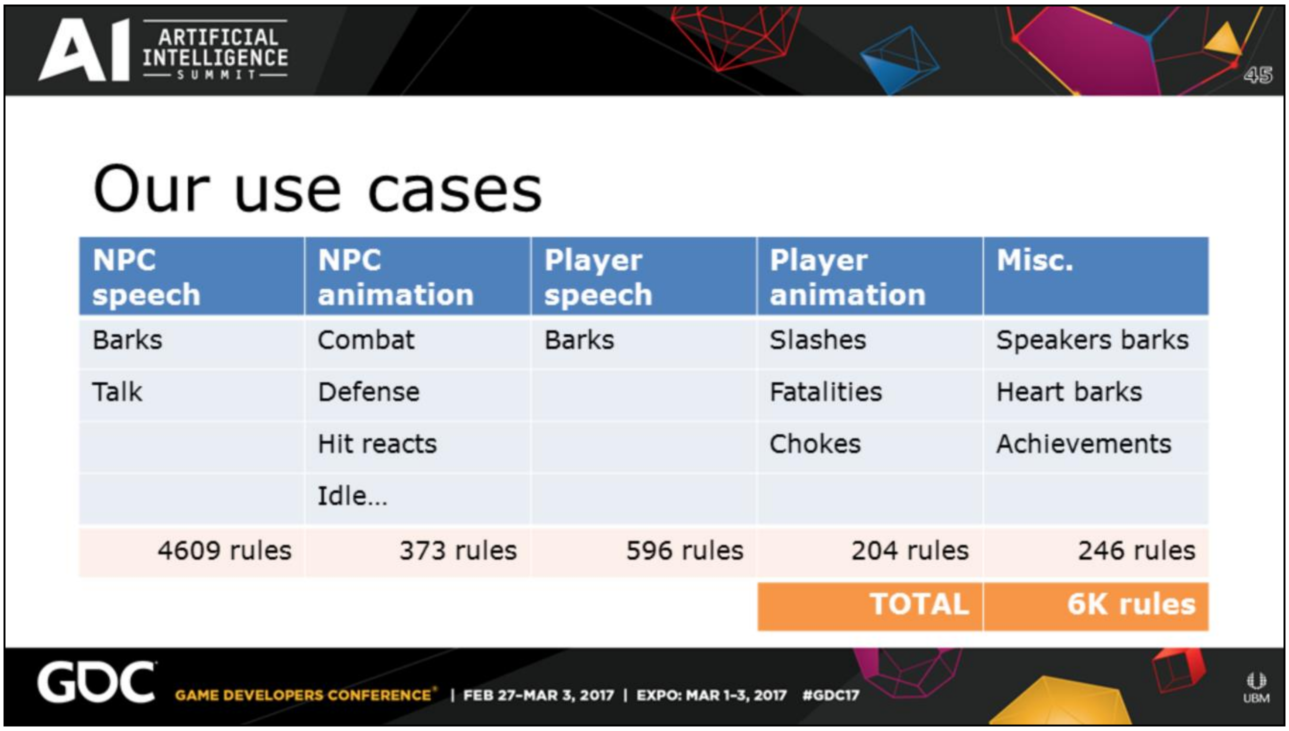
<originally a video>

This is me creating a couple of NPC bark rules... Let's say I want to play some lines when an aristocrat notices something… For example a flower pot or a tea pot.

I won't give you a full tutorial, but what I want to show here is the type of editor we have.

You can see that we have two rules: here and here. And that we have sets of conditions and sets of actions.

So, a respectable editor. Maybe not totally mature, but this is the actual tool we used for Dishonored 2.


ARTIFICIAL INTELLIGENCE SUMMIT
45
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Our use cases

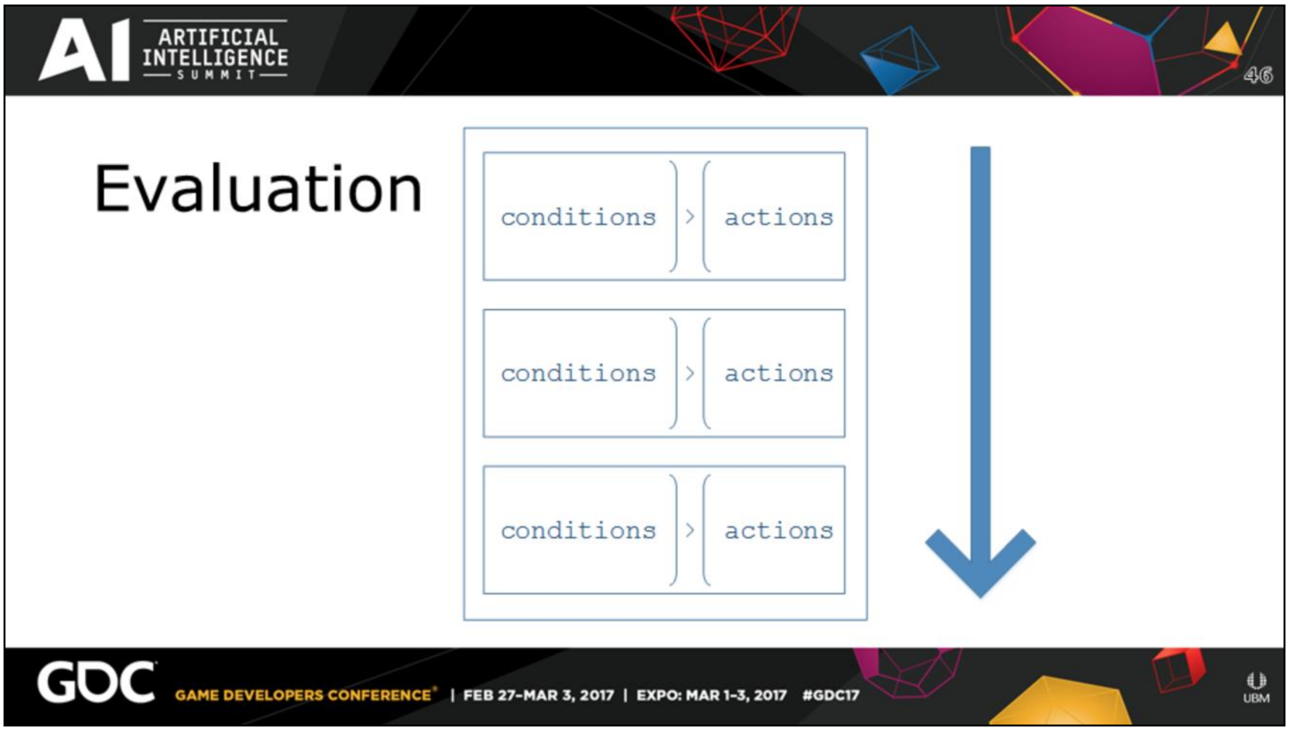
NPC speech	NPC animation	Player speech	Player animation	Misc.
Barks	Combat	Barks	Slashes	Speakers barks
Talk	Defense		Fatalities	Heart barks
	Hit reacts		Chokes	Achievements
	Idle...			
4609 rules	373 rules	596 rules	204 rules	246 rules
			TOTAL	6K rules

Now what did we create with it?

I won't detail everything but as you can see we use them not only for NPCs, but also for the player. And mostly for speech and animations. So this proved to be quite a versatile system.

One interesting extra use case is achievements. The rule system was a very good fit for things like "check that X coins were looted to grant achievement Y".

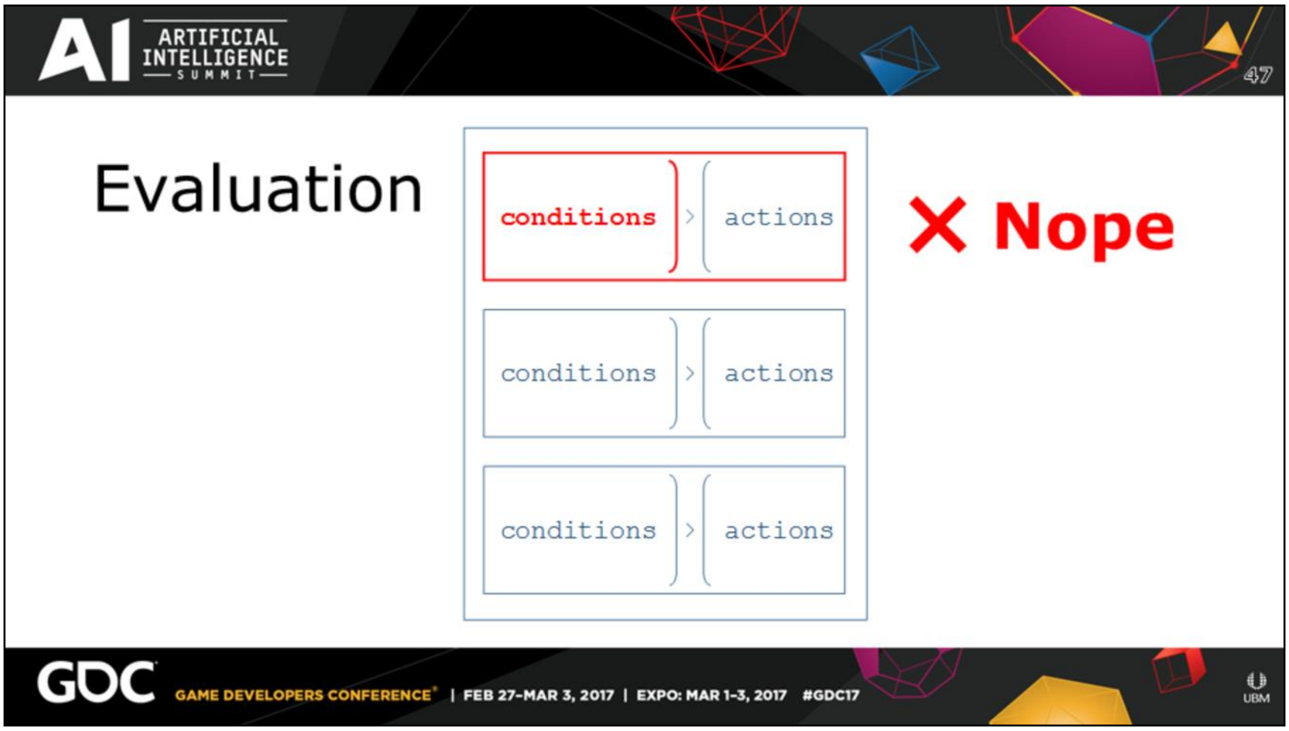
And since we all love numbers: this is how many rules were created for each category. The grand total is about six thousand rules.


ARTIFICIAL INTELLIGENCE SUMMIT
46
conditions
actions
conditions
actions
conditions
actions
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Evaluation

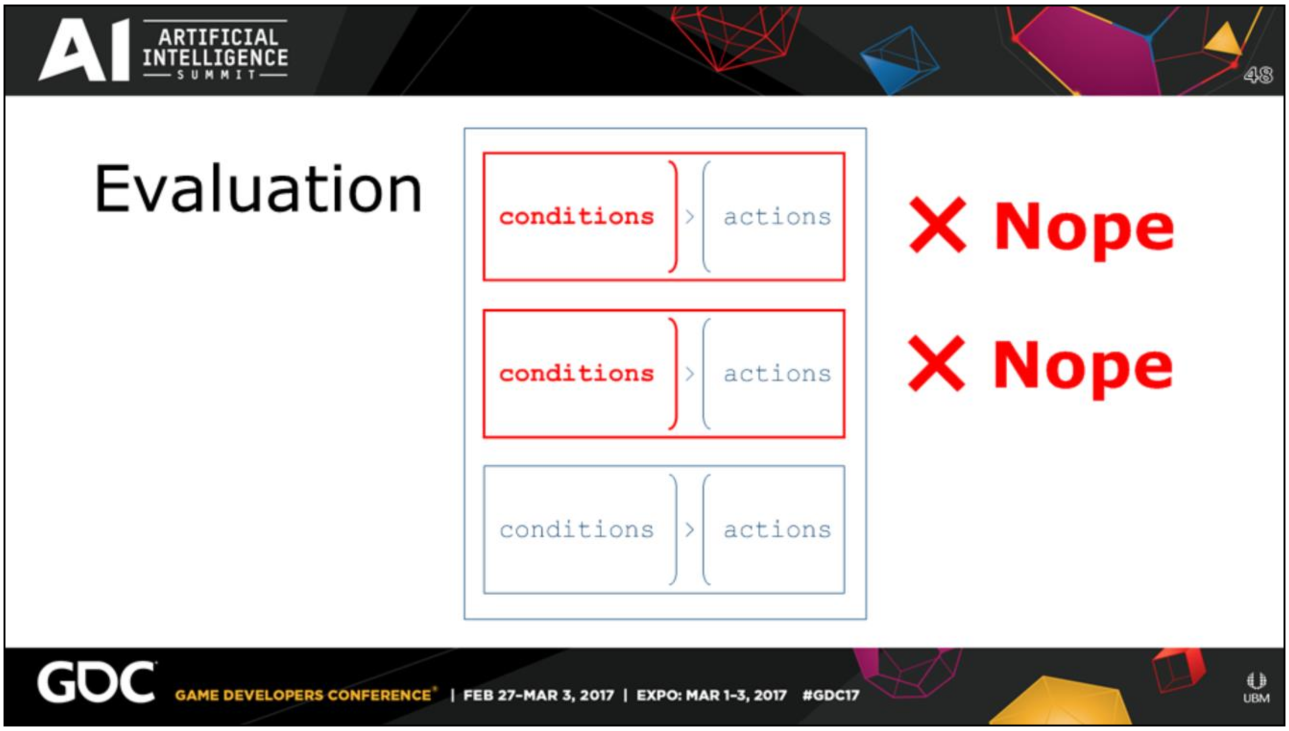
At runtime, rule handlers are instantiated and will be used for evaluation.

This means evaluating all rules it contains one by one, in order, and executing the first one that gets selected.



Evaluation

So here we evaluate the first rule: it doesn’t match so we discard this.

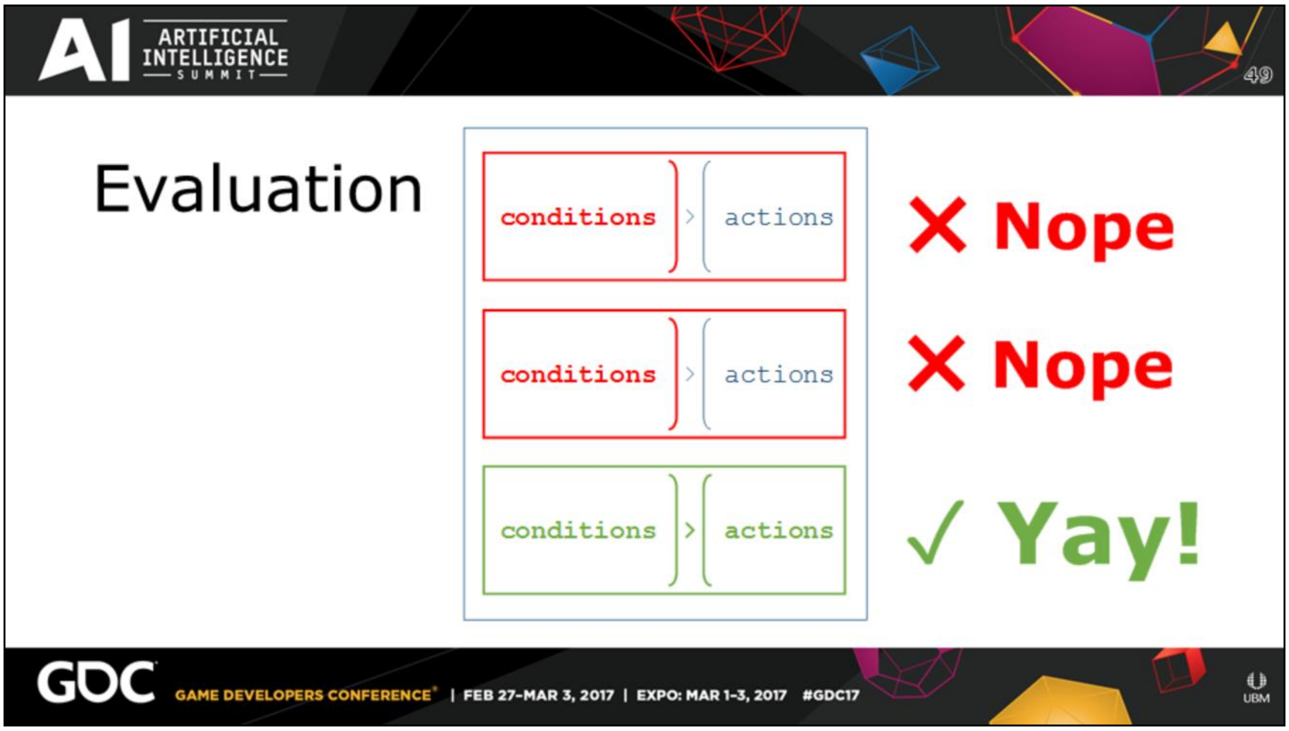


ARTIFICIAL INTELLIGENCE SUMMIT

Evaluation

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

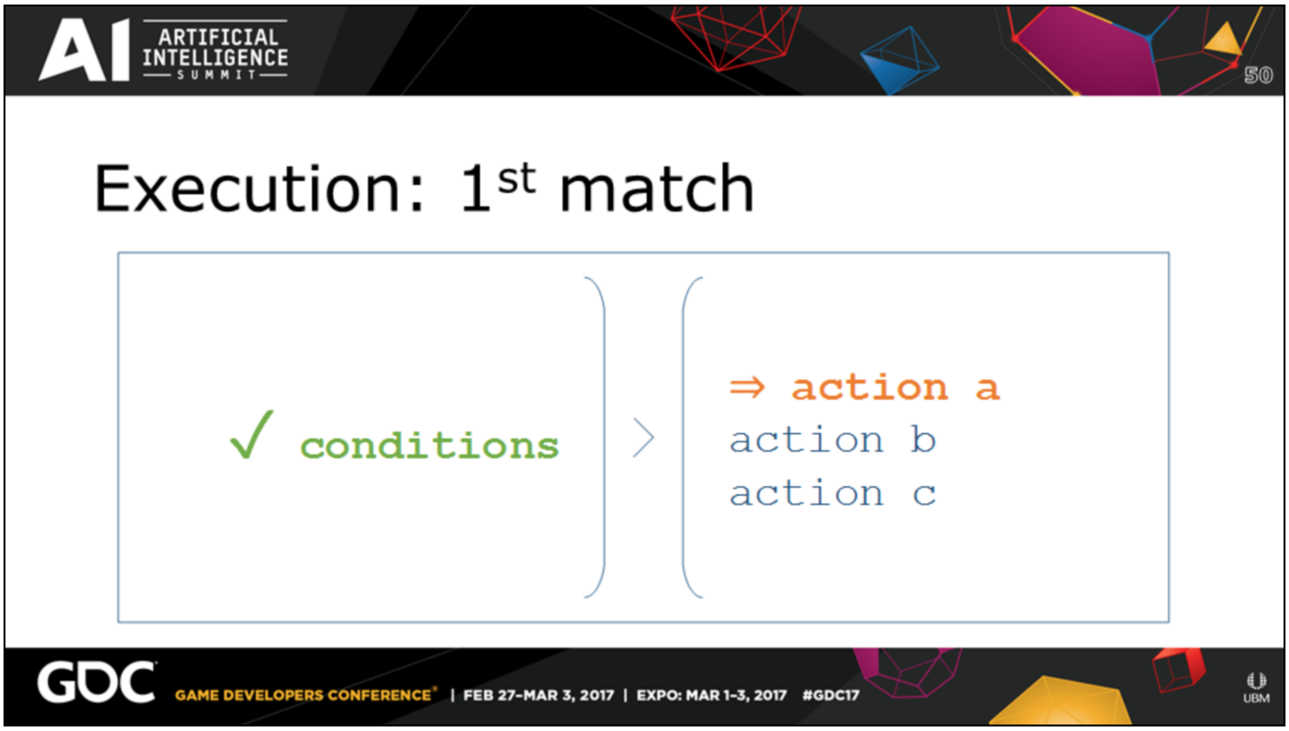
Same for the second rule…



Evaluation

So we try yet another rule: this time all conditions match so it's picked for execution.

We never needed to match more rules than only one, so the evaluation stops here. That's some CPU time saved for other stuff.



Execution: 1st match

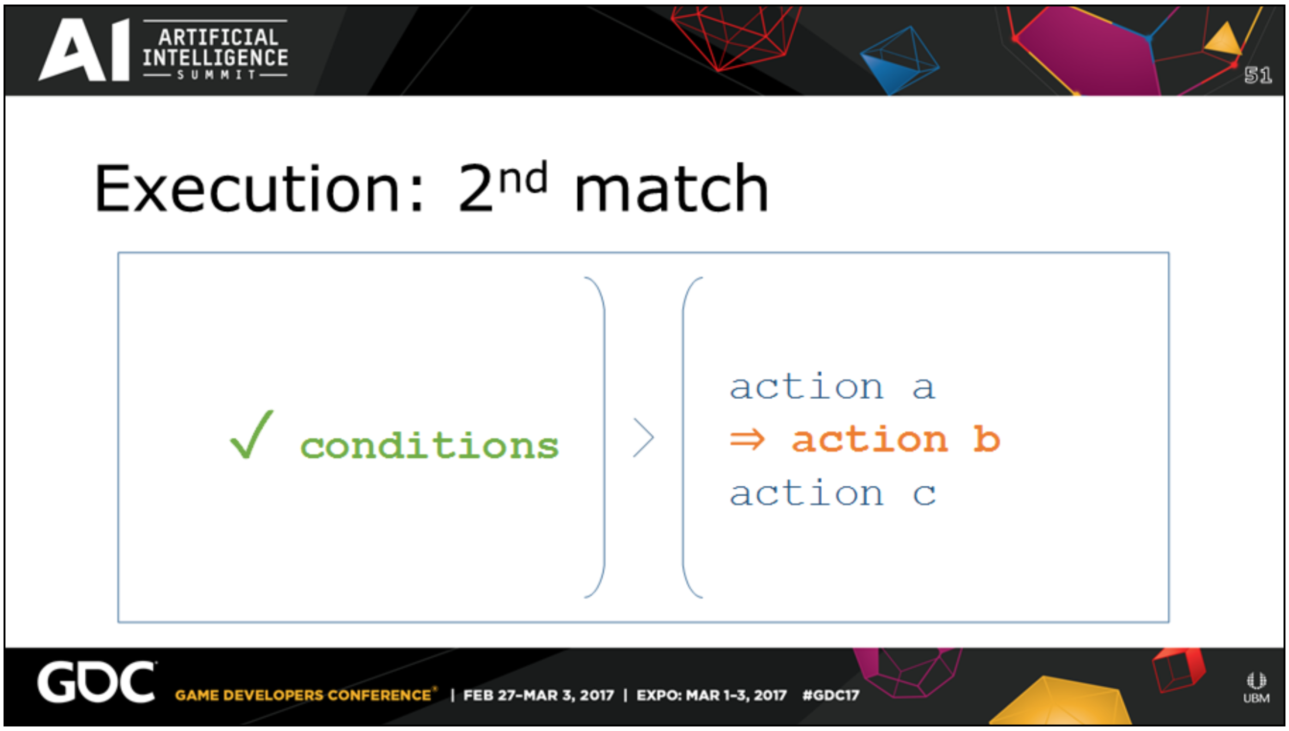
✓ conditions

⇒ action a
action b
action c

Now what do I mean by “execution”?

Every time a rule is selected, one action is executed. There are several ways you can handle this.

One way is to go over the actions sequentially. The first time a rule is selected we pick the first action and execute it…



ARTIFICIAL INTELLIGENCE SUMMIT

51

Execution: 2nd match

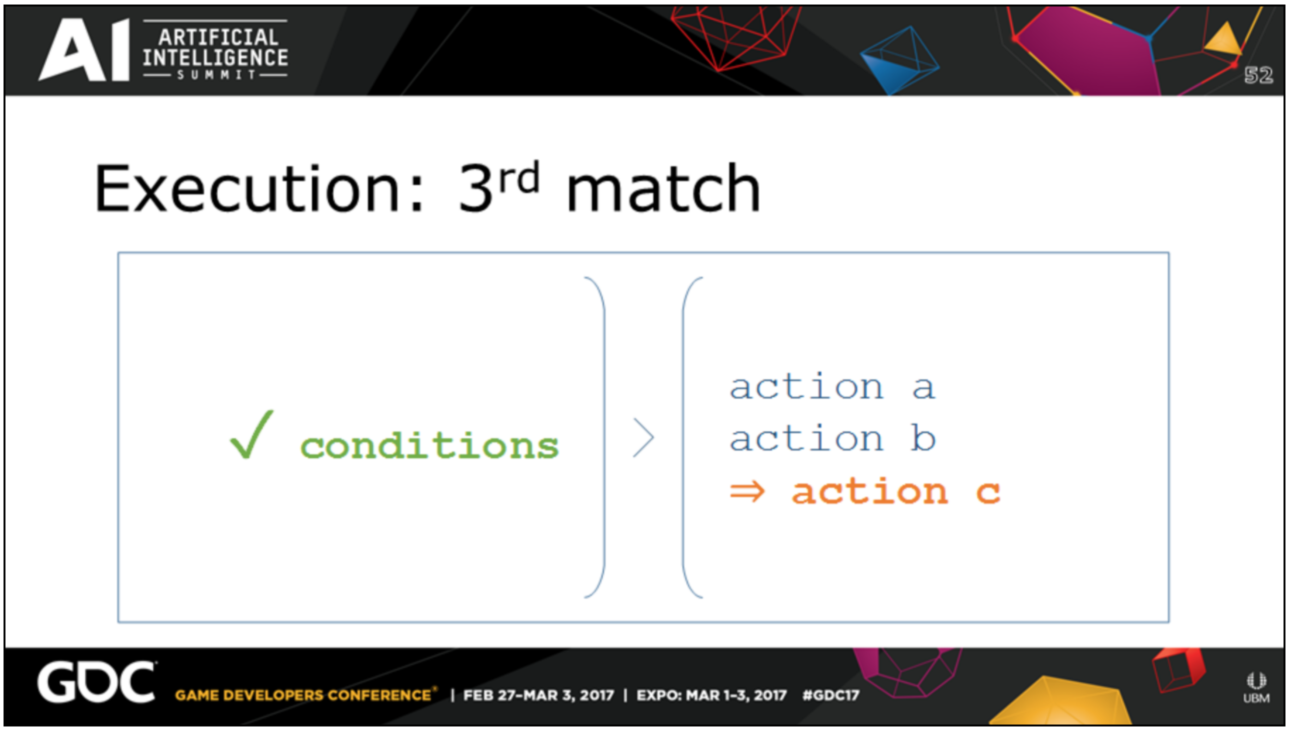
✓ conditions

```
action a
⇒ action b
action c
```

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

… and then the next time this rule matches, we execute the second action…



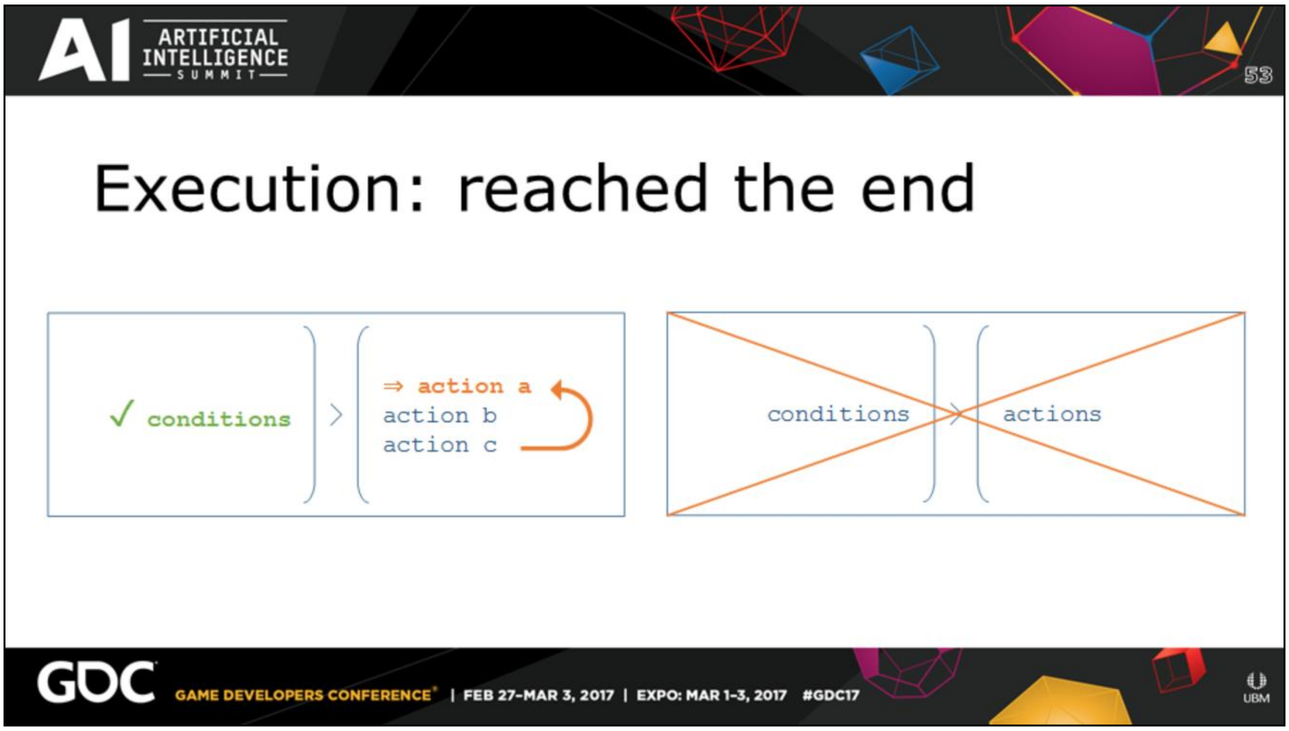
ARTIFICIAL INTELLIGENCE SUMMIT

Execution: 3rd match

✓ conditions 〉 action a
action b
⇒ action c

… until we run out of actions.

Things have been pretty straightforward until now, but please bear with me because this is where things start to be interesting.



Execution: reached the end

At this point we can either start again from the first action.

Or we can simply say these actions are depleted and never select this rule again.

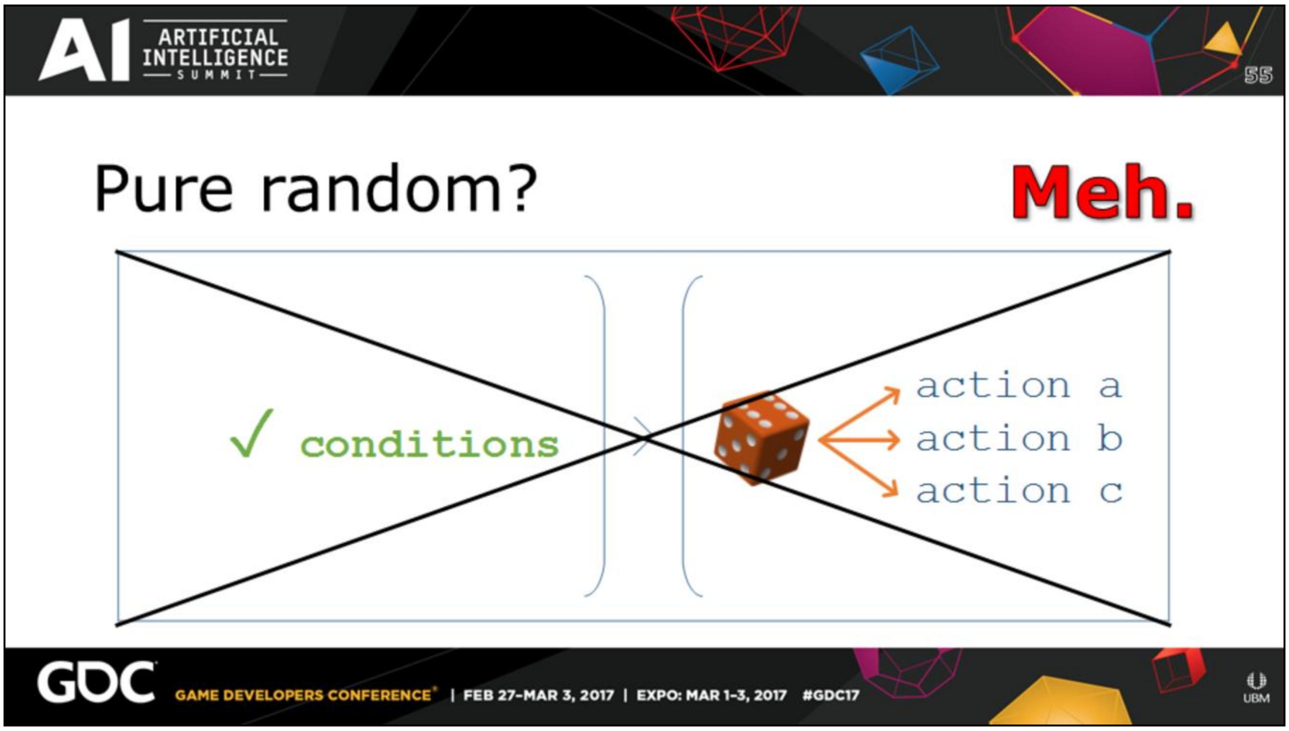
We allow both, and we call these two options “execution policies”.

OK but I can haz random?

… Sequential execution is handy, but sometimes what you really want is random variations.

For instance for ambient barks, you don't want NPCs to say the same lines in the same order all the time.

This gives a "robot effect" pretty fast, and is distracting for players.


ARTIFICIAL INTELLIGENCE SUMMIT
55
Pure random?
Meh.
✓ conditions
action a
action b
action c
GDC GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
UBM

The naïve approach to randomizing things, is to pick a random outcome every time the rule is selected.

The problem with this kind of random is that it doesn't prevent repetitions, which is another annoying "robot effect".

So we don't do this.

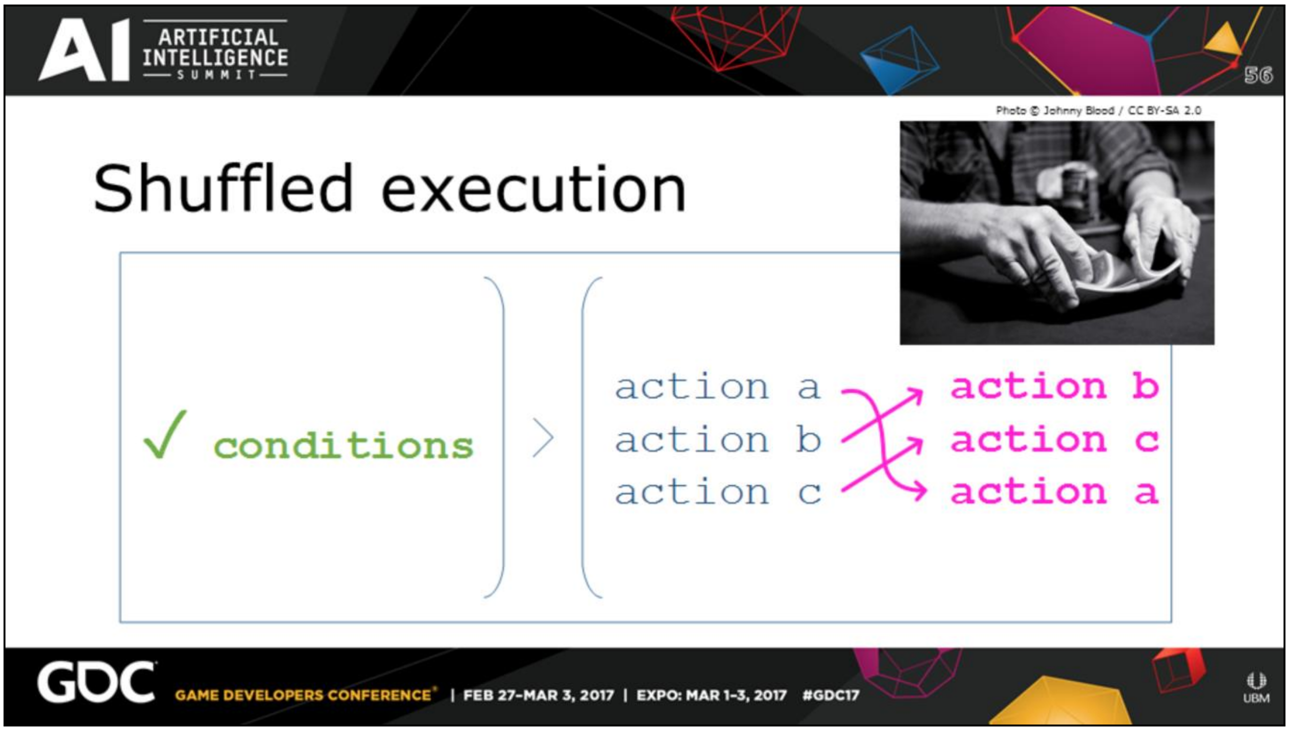
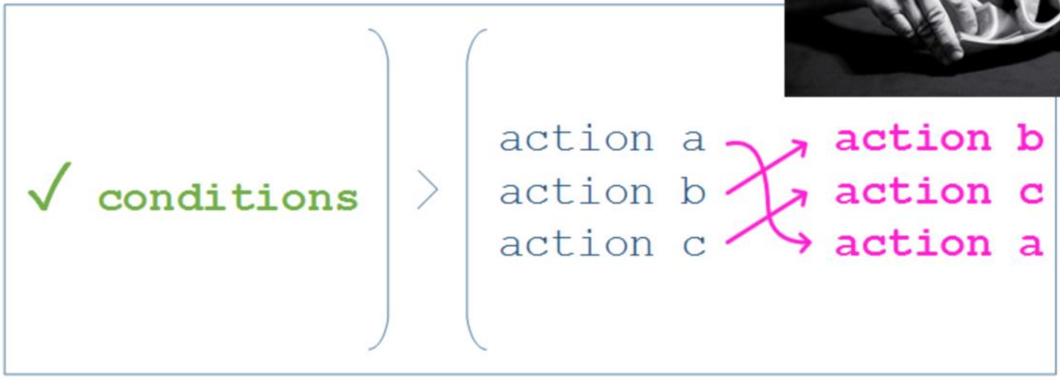


Photo © Johnny Blood / CC BY-SA 2.0

Shuffled execution

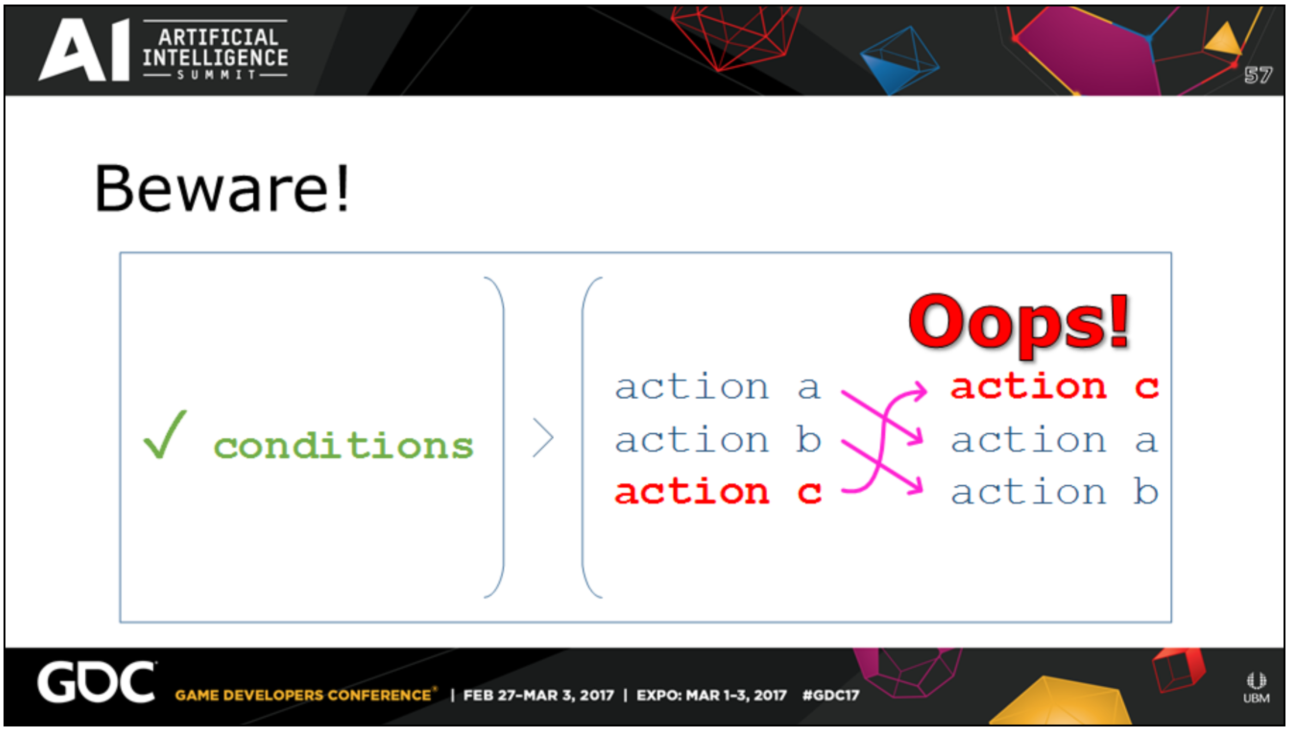


What we do instead is that we shuffle the action list during initialization, and then shuffle again every time we reach the end.

Just like you would shuffle a deck of cards before playing a game of poker.

This is called a shuffle bag. Given a list of possible options, it's the most natural way to use them all while avoiding common pitfalls.

Photo © Johnny Blood / CC BY-SA 2.0
https://commons.wikimedia.org/wiki/File:Riffle_shuffle.jpg


Beware!

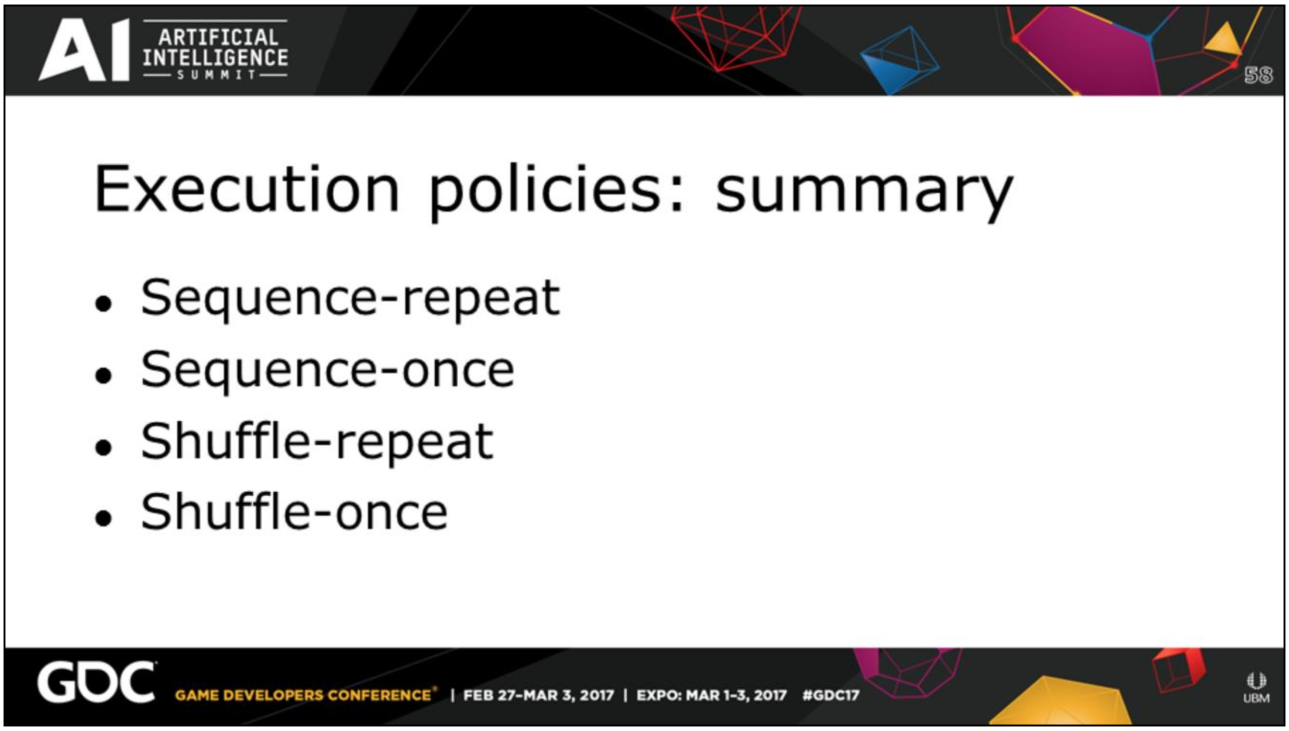
Beware though!

When shuffling, you should never put back the *last* action in the *first* spot, otherwise you'll get a nasty repetition.

One work around is to send it to any random spot but the first one*.

---

[*] Details here:
http://gamedev.stackexchange.com/questions/29743/how-do-i-produce-enjoyably-random-as-opposed-to-pseudo-random/29747#29747



ARTIFICIAL INTELLIGENCE SUMMIT

58

Execution policies: summary

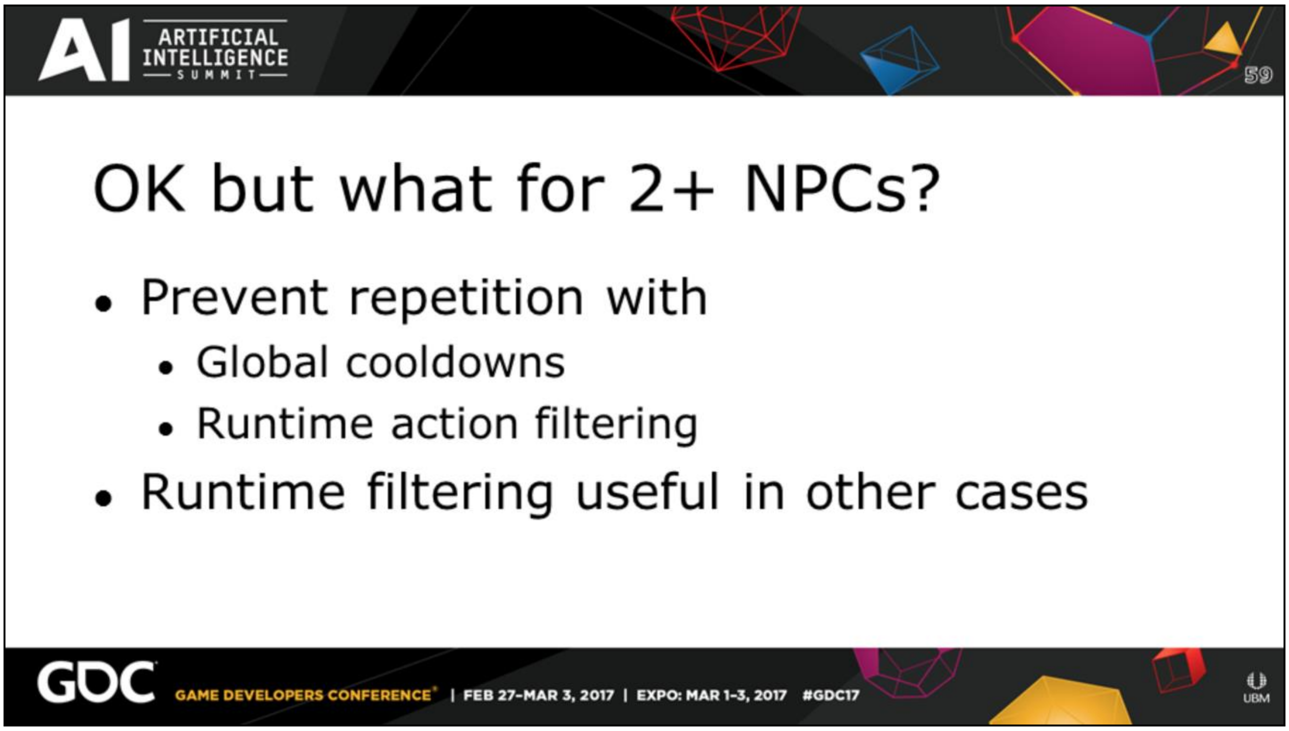
- Sequence-repeat
- Sequence-once
- Shuffle-repeat
- Shuffle-once

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

So that's a total of four execution policies that we use in the final game.

Sequential or shuffled execution of actions works great to prevent repetition for one entity.



OK but what for 2+ NPCs?

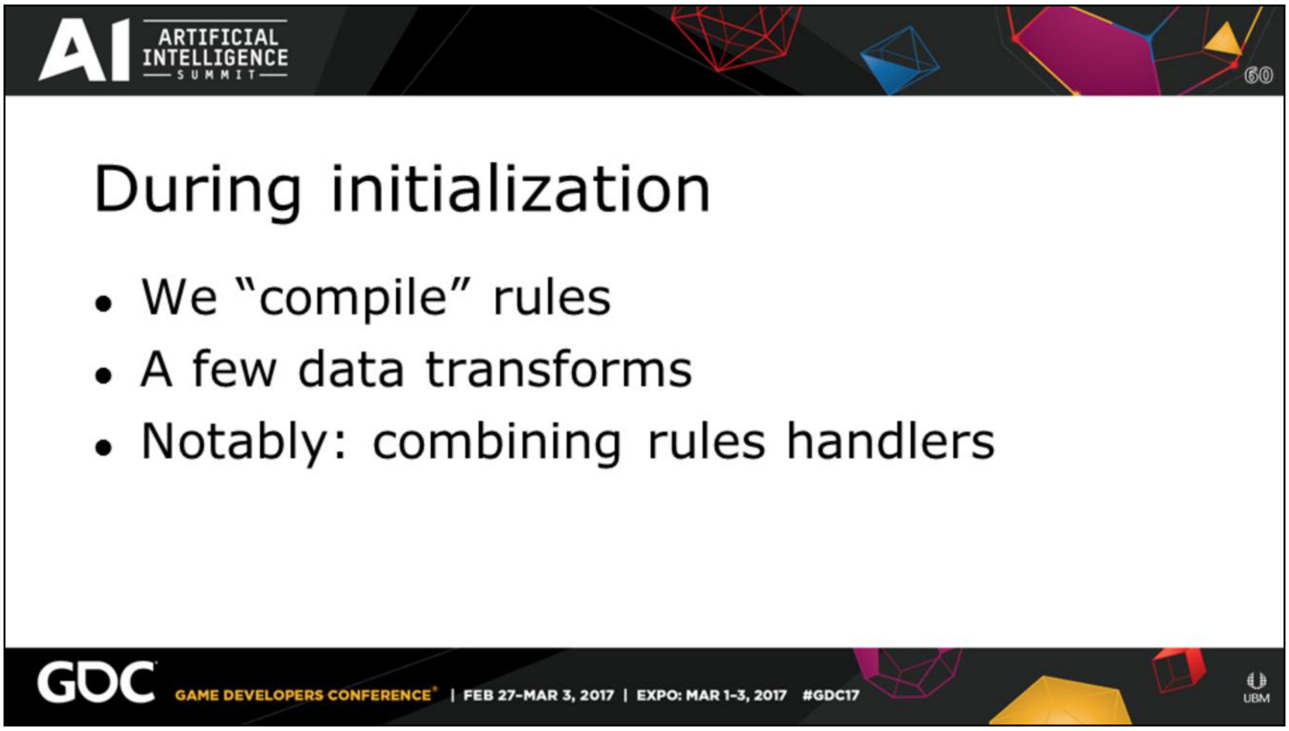
- Prevent repetition with
  - Global cooldowns
  - Runtime action filtering
- Runtime filtering useful in other cases

ARTIFICIAL INTELLIGENCE SUMMIT
59
GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17
UBM

But what about *several* entities?

We still wanted to prevent the same bark or attack from being played twice in a row by different NPCs. We obtained this by putting recently executed actions in a global list, forbidding other NPCs to execute it until a cooldown depletes.

Actually, once we had this option to filter actions at runtime, we found out other uses for it. We also ended up implementing it for conditions. We didn't plan for this initially and this evolved a bit organically, but it's probably something interesting to explore further.


ARTIFICIAL INTELLIGENCE SUMMIT

During initialization

- We “compile” rules
- A few data transforms
- Notably: combining rules handlers

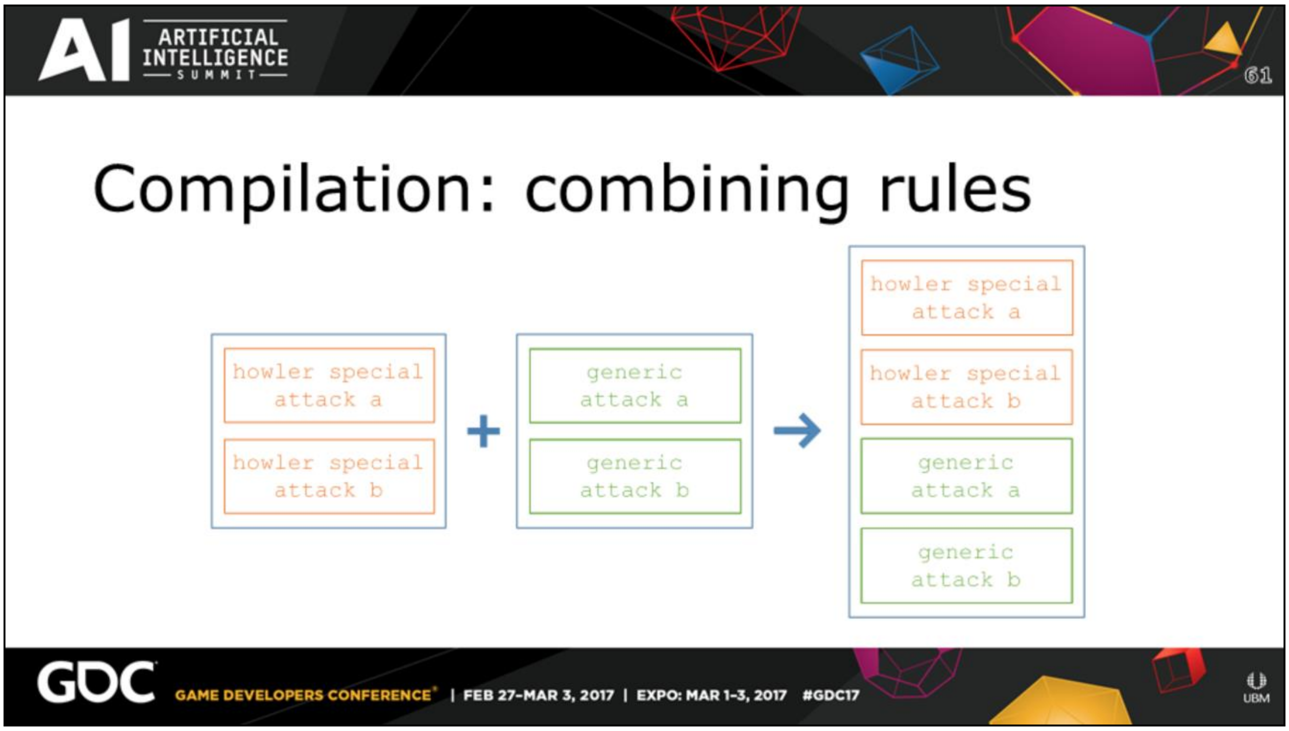
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Earlier, I mentioned briefly that we instantiate our rule handlers during initialization.

We actually say that we “compile” them.

That’s because we perform some significant data transforms at this stage, mostly for performance reasons.

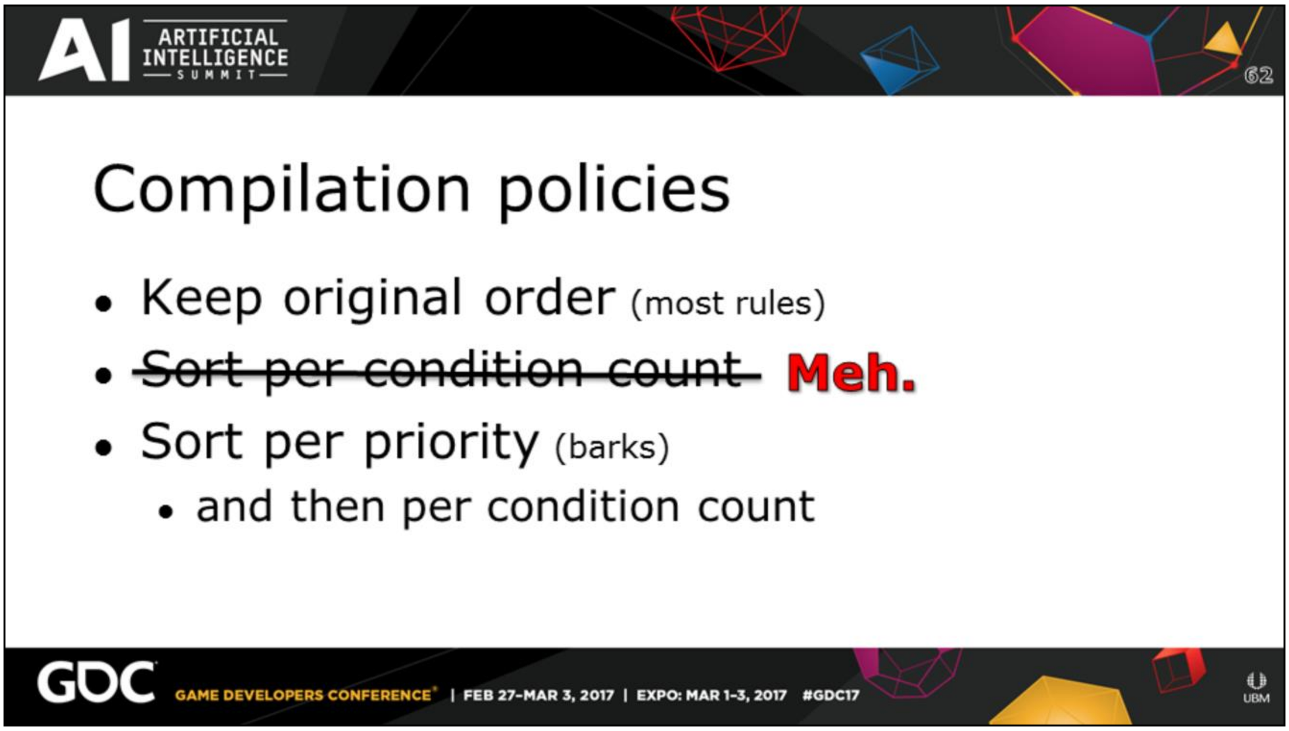
One interesting thing that we do is that we combine rule handlers. This allows to add a custom set of rules for specific NPCs.



Compilation: combining rules

For instance members of the Howlers gang have a bunch of specific attacks.

We can take these attacks and combine them with the generic attacks to create one single rule handler instance.


ARTIFICIAL INTELLIGENCE SUMMIT
62
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Compilation policies

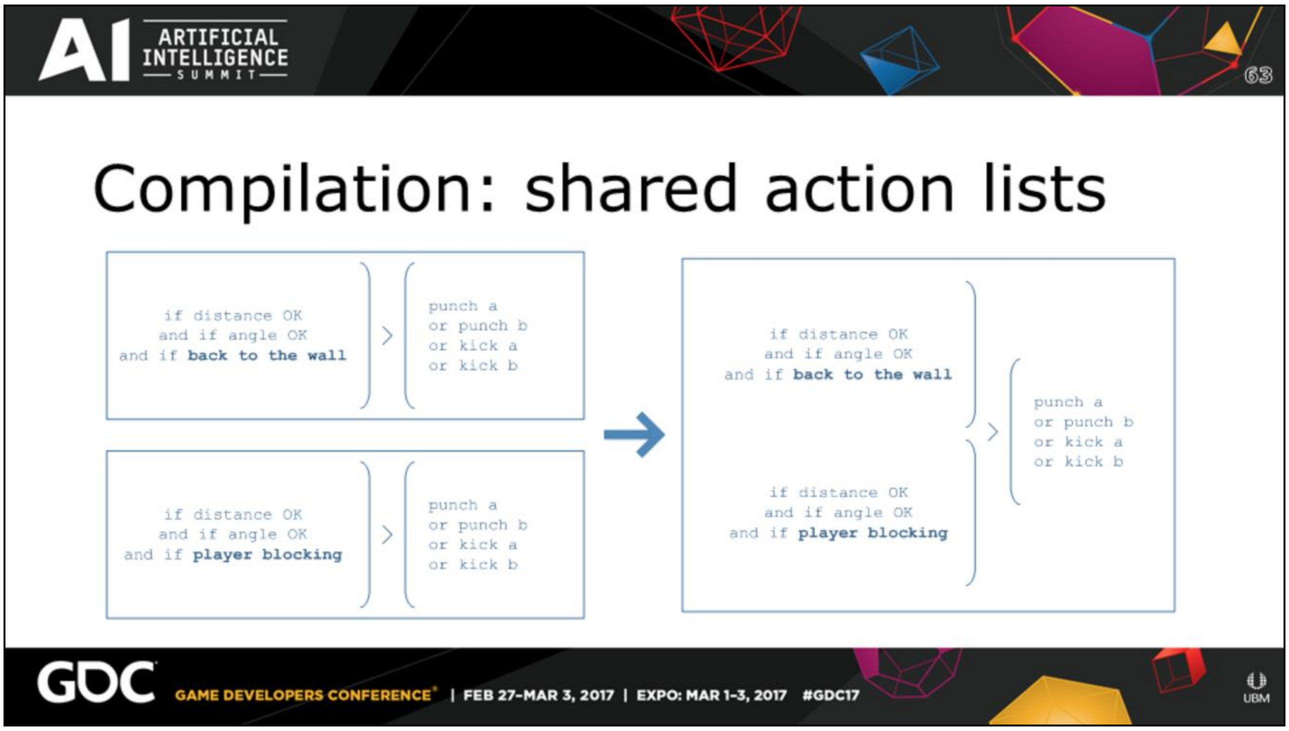
- Keep original order (most rules)
- ~~Sort per condition count~~ **Meh.**
- Sort per priority (barks)
  - and then per condition count

Since we combine rule handlers we need to determine how to sort the resultant set of rules. We call this a compilation policy.

One solution is to simply *not* sort, and just concatenate everything. This is actually a very sensible solution for small rule sets: what you see in editor is what you get in game. Clear and simple: we use this for most rules. But for us, barks were the exception. We have too many of them and really needed more levers to get to the desired results.

We first tried to sort rules per condition count as Elan Ruskin advises in his talk, but it didn’t cut it for us. Our Lead Narrative Designer Sachka often found herself adding new empty conditions just to get the outcome she wanted.

So we changed our approach near the end of the production. The solution we shipped with is that we give every bark a priority and we use this as the first criteria for sorting, keeping the condition count only as a second criteria.



Compilation: shared action lists

```
if distance OK
and if angle OK
and if back to the wall
```

```
punch a
or punch b
or kick a
or kick b
```

```
if distance OK
and if angle OK
and if player blocking
```

```
punch a
or punch b
or kick a
or kick b
```

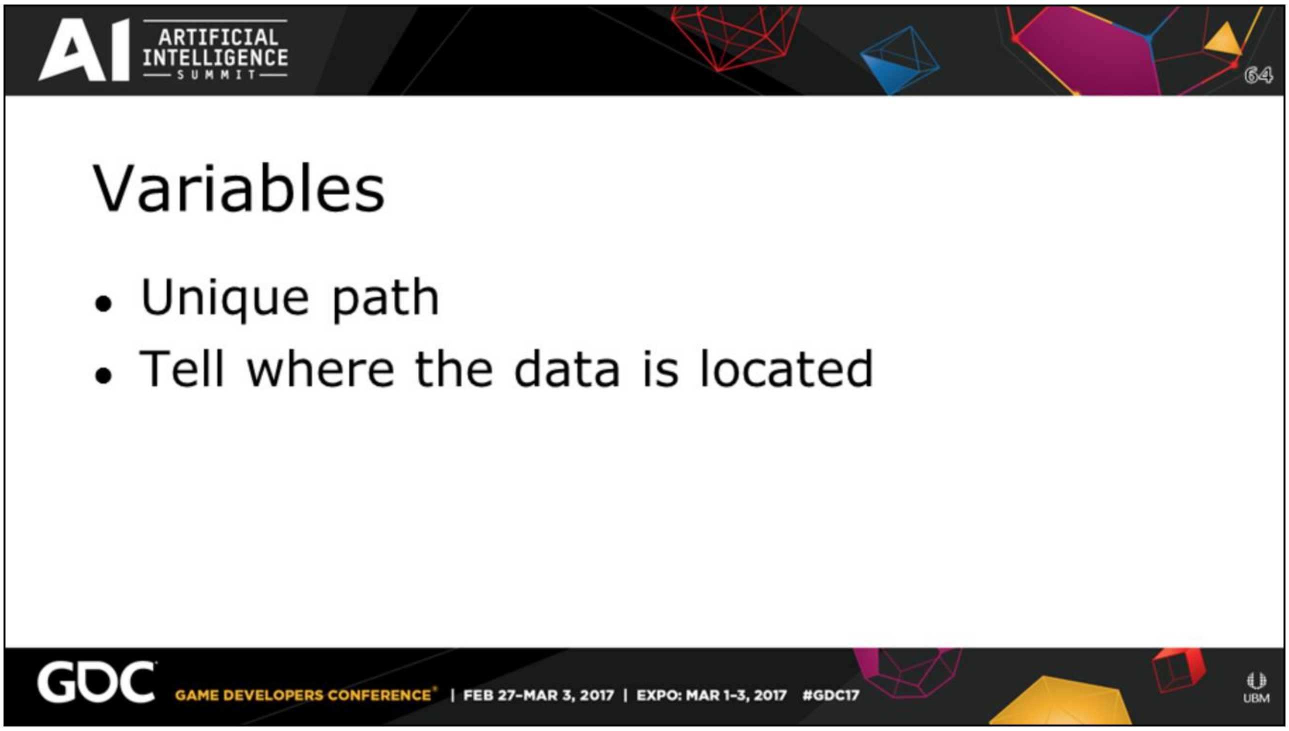
ARTIFICIAL INTELLIGENCE SUMMIT
63
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Something else we do during compilation is that we share identical action lists. Let me explain this with another example.

Our fighting NPCs can kick or punch under different sets of conditions. Offline, we have two rules: one to trigger close combat attacks when back to the wall (literally), and another one for a player blocking too much.

At runtime these two rules point to the same action list, so the sequential execution shares the same counter. In other words you won't observe the same punch animation twice in a row just because it was triggered under different conditions.

And yes, in case you wonder, that does give us a logical "or" for the few cases where we actually need it.



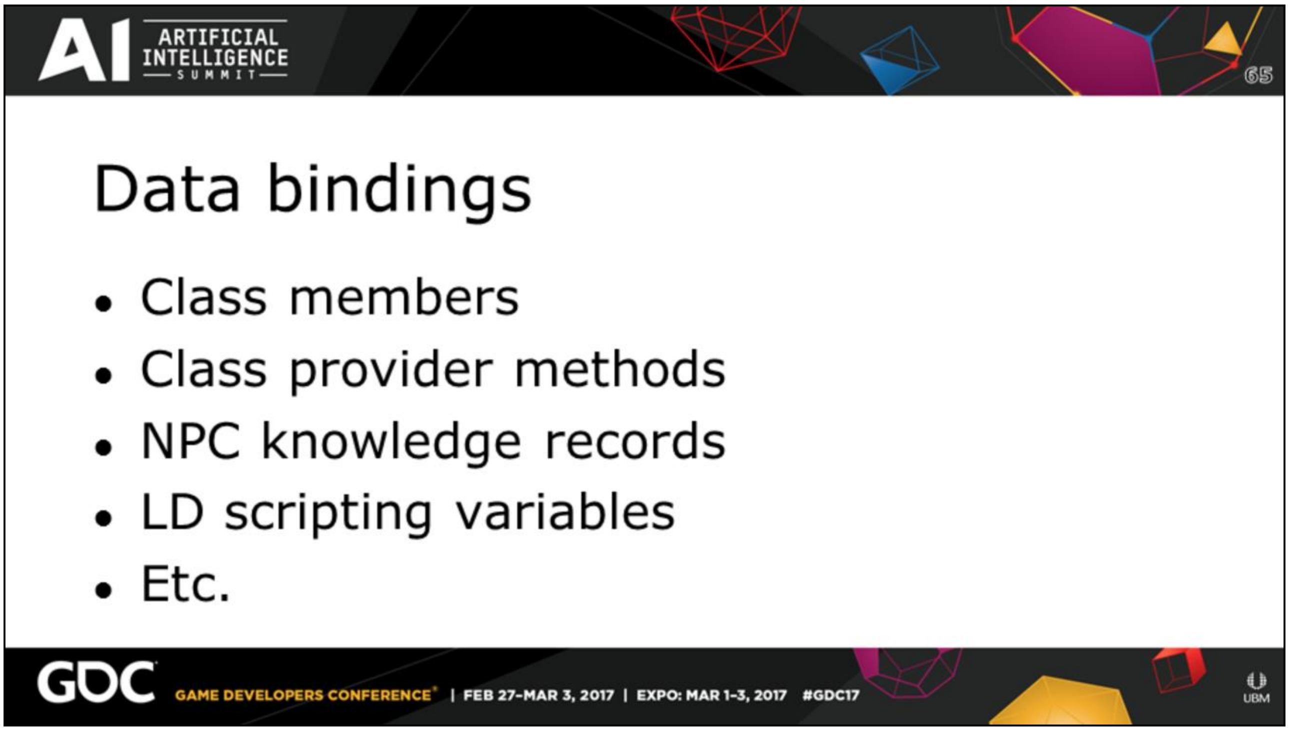
Variables

- Unique path
- Tell where the data is located

Now let me come back to variables for a moment, and give you more juicy details.

These variables have a unique path to keep things simple and ordered in the editor.

Their role is to tell us how to get to the actual data.

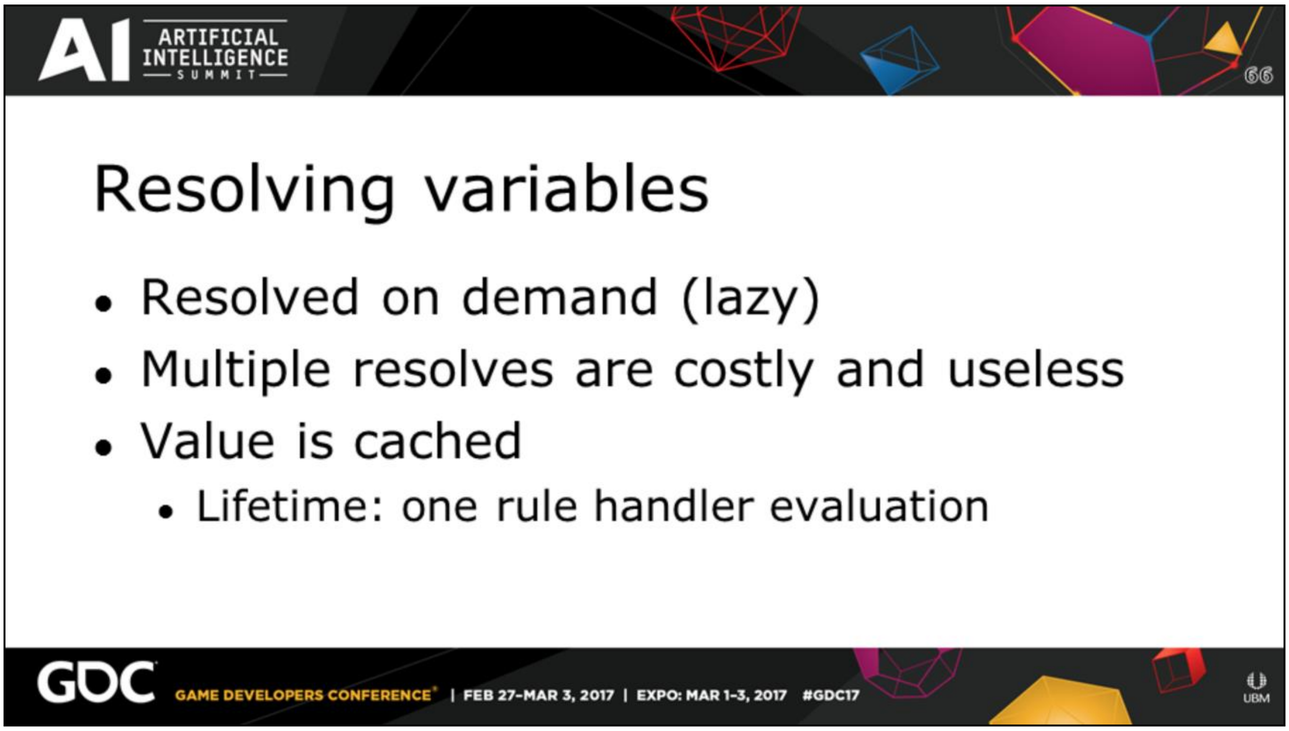


Data bindings

- Class members
- Class provider methods
- NPC knowledge records
- LD scripting variables
- Etc.

So let me list some data bindings that we have.

A variable can give the value of a C++ class member… Or be computed by a method (we call these “providers”)… It can point directly at an NPC knowledge record… Or at a variable created in data by level designers… And we also have a few more bindings.



ARTIFICIAL INTELLIGENCE SUMMIT

66

Resolving variables

- Resolved on demand (lazy)
- Multiple resolves are costly and useless
- Value is cached
  - Lifetime: one rule handler evaluation

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

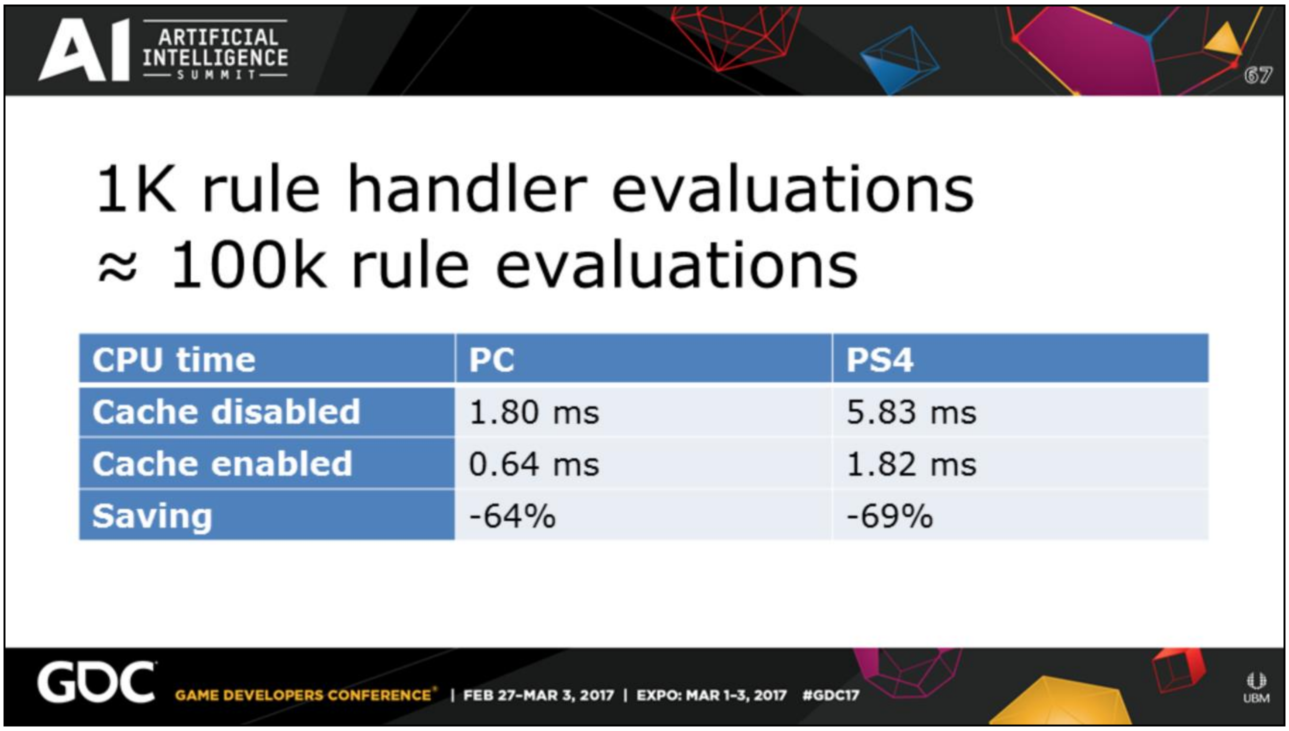
UBM

When do we resolve variables?

We resolve these variables on demand. Most of the times, we only need a few variable to be actually resolved during evaluation. For instance let's say that the first rule of a set matches. We won't evaluate the remainder, so we don't have to pay the cost of resolving the variables we will never check.

Still, resolving a variable *every time* would be very damageable for performance. We have variables that are actually method pointers, and that do perform quite a bit of computations before their return value. Since resolving twice or more is useless in practice, we don't want to pay that cost each time a variable is tested during one evaluation.

So we cache the resolved values for the time of an evaluation.



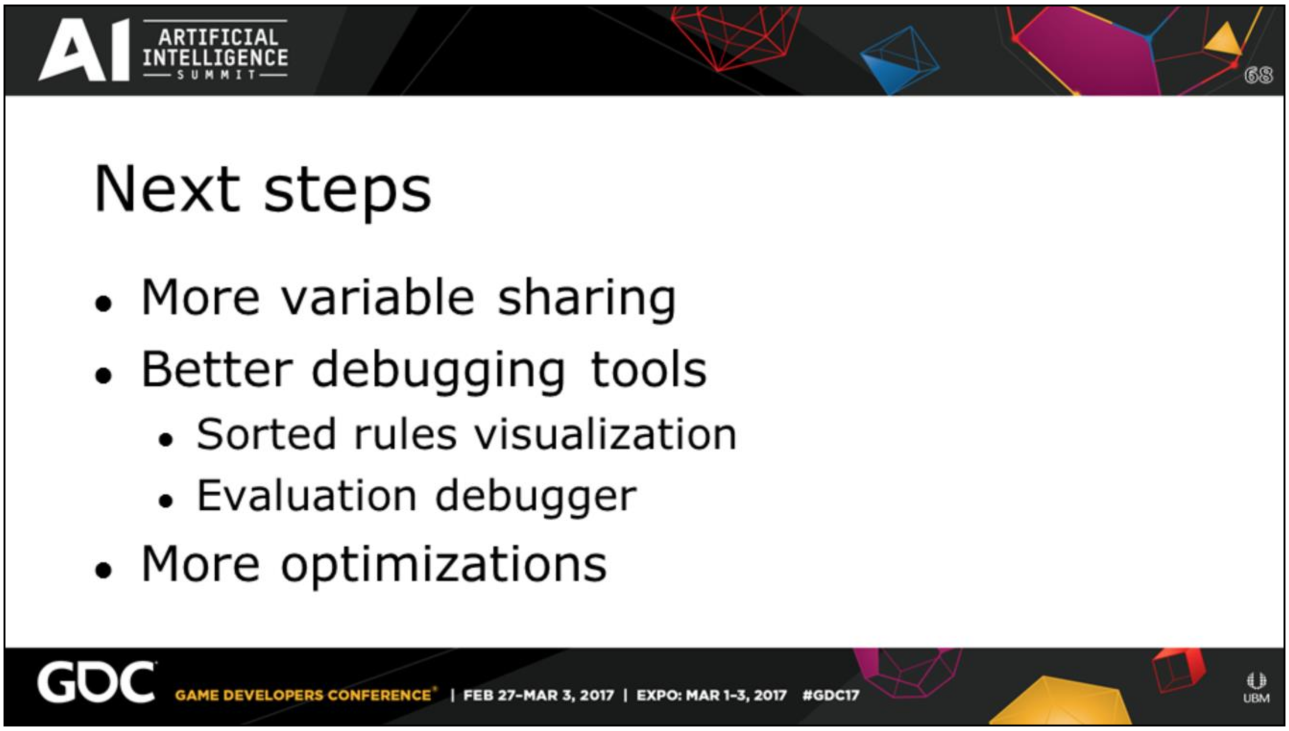
1K rule handler evaluations ≈ 100k rule evaluations

CPU time	PC	PS4
Cache disabled	1.80 ms	5.83 ms
Cache enabled	0.64 ms	1.82 ms
Saving	-64%	-69%

Having this cache does shave about two thirds of the time spent evaluating rules, so it’s a win.

Note that these figures are for a thousand rule handlers, but we evaluate just a few of them per frame.

So our performance metrics are OK for this generation of consoles and onwards.


ARTIFICIAL INTELLIGENCE SUMMIT

68

Next steps

- More variable sharing
- Better debugging tools
  - Sorted rules visualization
  - Evaluation debugger
- More optimizations

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

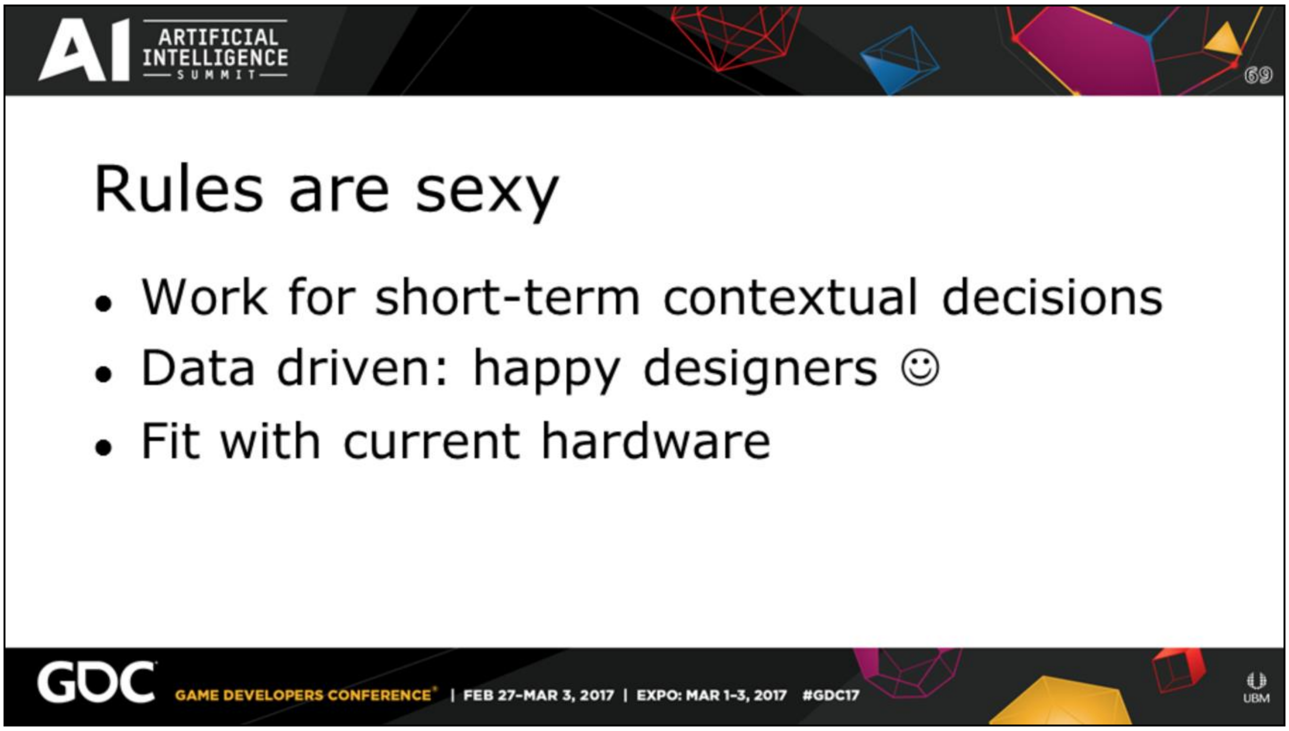
UBM

Time to close this chapter with what we envision for the future.

One thing that is a bit clumsy with our current system is that some variables are only available for some type of rule, for no particular reasons. For instance a variable exposed for barks can be missing for attacks. If you are going to use rules in many different contexts just like we do: make sure you take this into account, and share your variables as much as possible.

Of course you always need better debugging tools, and we make no exception. Something we miss is a visualization tool for the combined and sorted rules. Also at runtime, all we have to debug an evaluation is a good old dump: we'll try to improve this in the future.

Finally we've talked about our cache system but we know there's more we can do on the performance side. So if we expand our usage of rules, we know there's room for it.


ARTIFICIAL INTELLIGENCE SUMMIT

69

Rules are sexy

- Work for short-term contextual decisions
- Data driven: happy designers ☺
- Fit with current hardware

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

So to sum up, what makes rules so sexy?

Firstly they are very appropriate for simple and short-term contextual decisions.

Secondly they're a data-driven tool. In the end they made our designers happy. I believe. Or at least mildly satisfied ☺ It should be obvious to everybody: if you have to make a code change for each and every new asset in a AAA game, well you're just never going to ship it.

And finally, you saw the metrics: we now have more than enough hardware resources to afford such a system. So there's no reason not to do it.

Your turn to rule Xavier!

DSM

Or why a good acronym is important

ARTIFICIAL INTELLIGENCE SUMMIT
70
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Thanks Laurent, you rule ☺

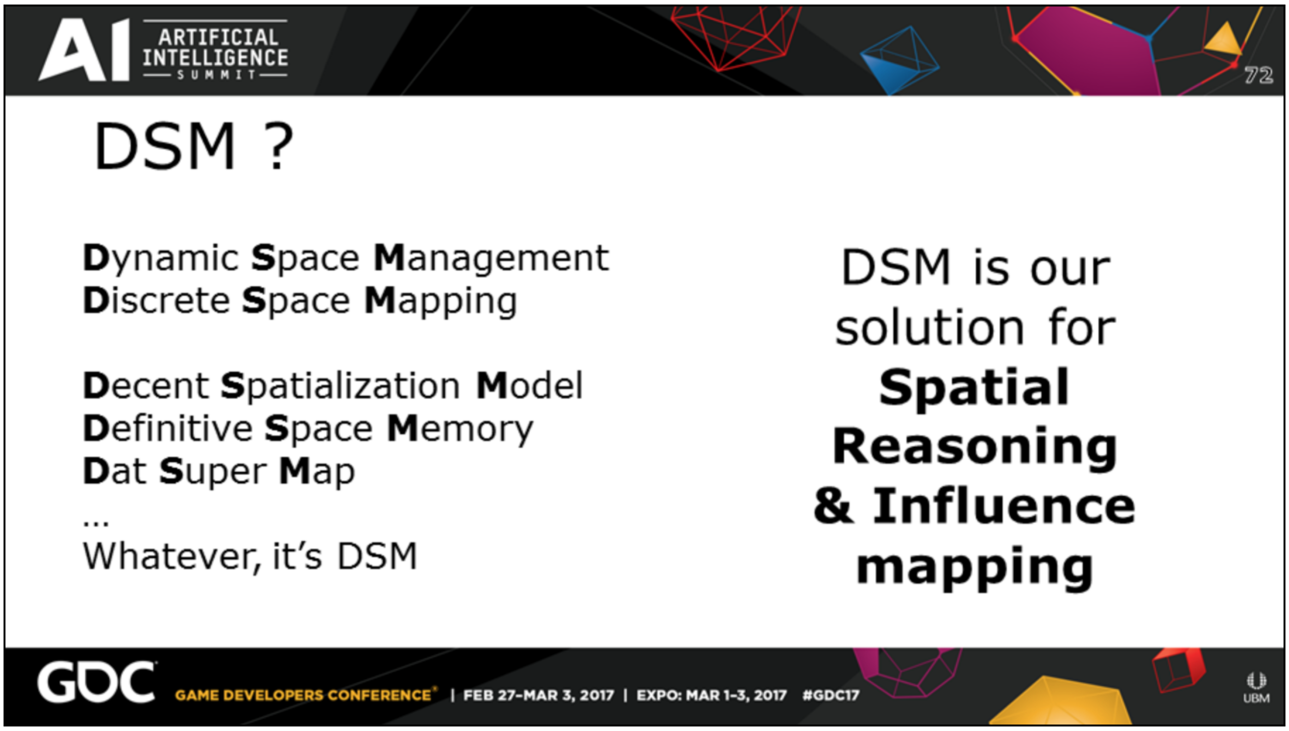
OK now, let’s speak DSM.

I do realize that I'm not making things easy on myself by starting a section of this talk with some obscure acronym.

Let's try to explain a little bit.

Photo © Takashi Hososhima / CC BY-SA 2.0

https://commons.wikimedia.org/wiki/File:What_are_you_doing_here%3F_(7046417743).jpg



DSM ?

Dynamic **S**pace **M**anagement
Discrete **S**pace **M**apping

Decent **S**patialization **M**odel
Definitive **S**pace **M**emory
Dat **S**uper **M**ap
…
Whatever, it's DSM

DSM is our solution for **Spatial Reasoning & Influence mapping**

ARTIFICIAL INTELLIGENCE SUMMIT
72
GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

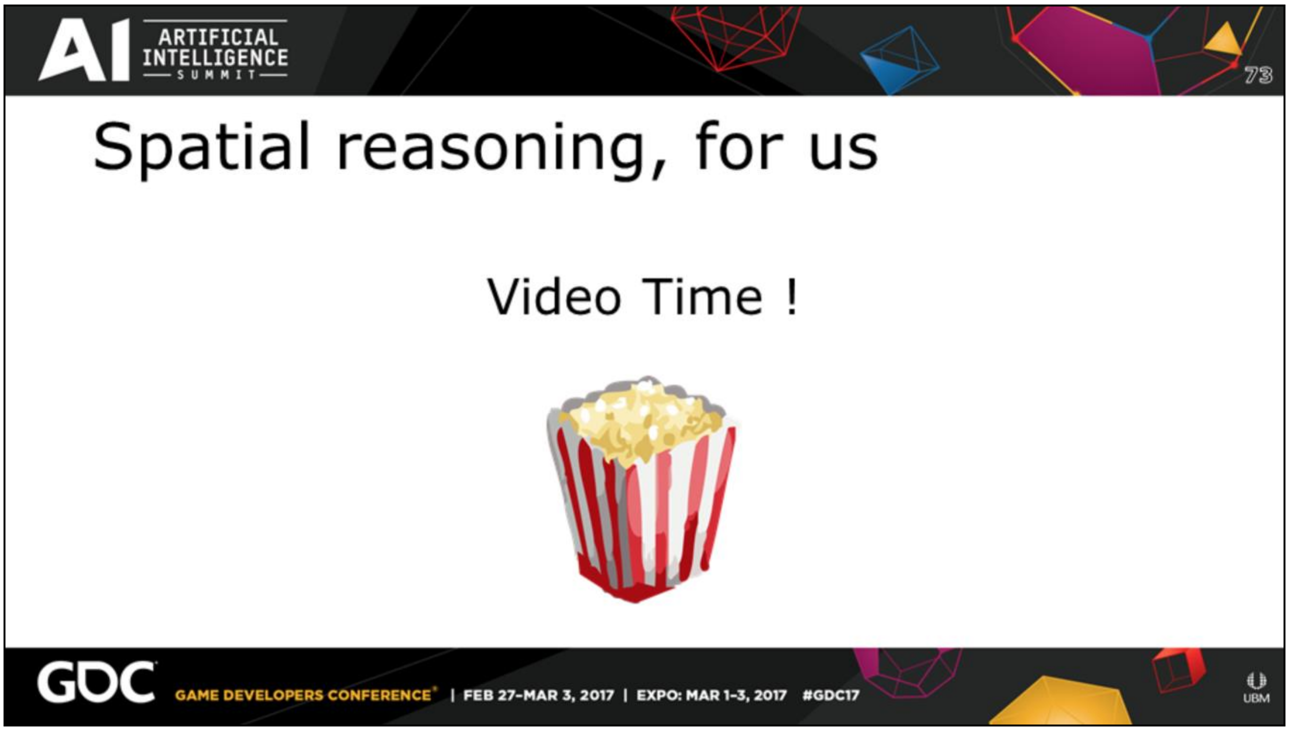
Originally DSM meant Dynamic Space management, or maybe Discrete Space Mapping.

But you could take any of those other meanings I came up with while writing this part of the talk.

And yes, there is a pun ;)

The point here is that it's not about what it means, it's about what it does.

DSM is our solution for spatial reasoning & influence mapping.



Spatial reasoning, for us

Video Time !

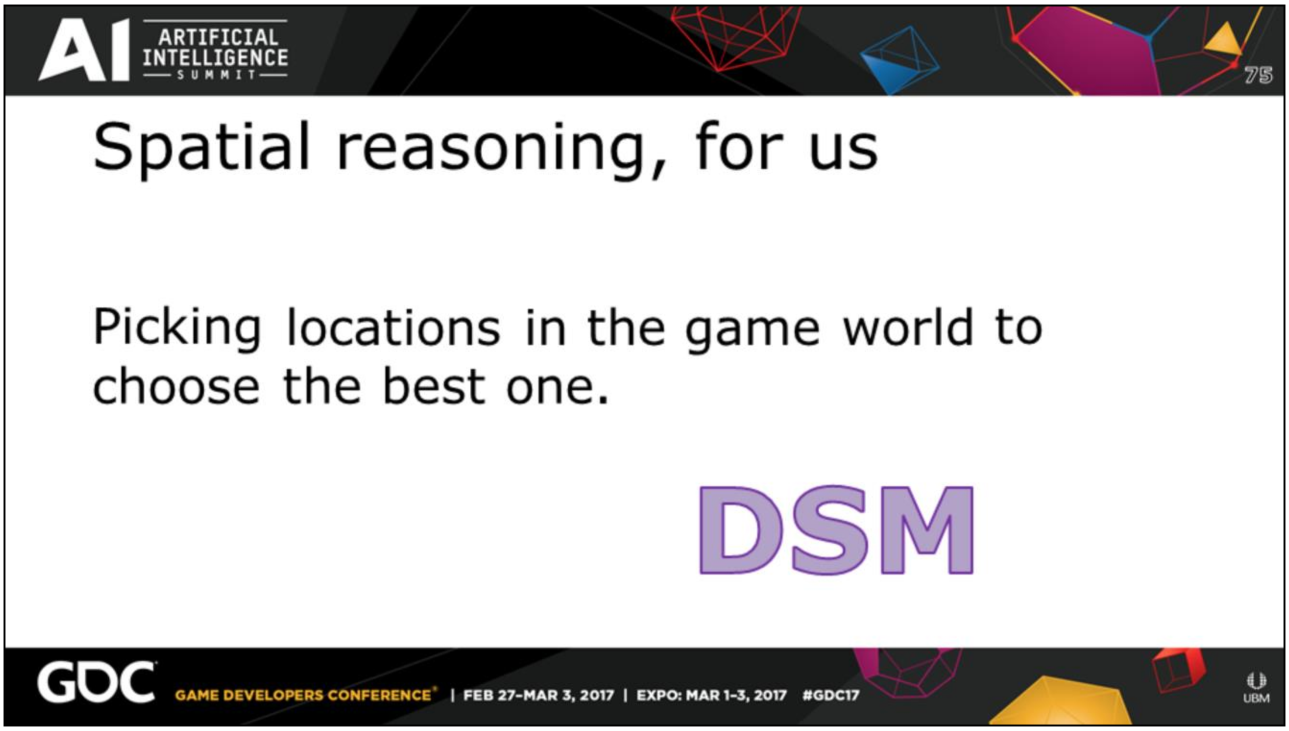
First let me show you what we use spatial reasoning for in our game.

<originally a video>

There you can see Witches teleporting themselves and summoning a vine, using dynamic positions.

Then we have the always funny case of civilians running away in panic, and choosing destinations dynamically.

And lastly we have the player losing NPCs in combat, having them decide where they should go to chase him.



ARTIFICIAL INTELLIGENCE SUMMIT

75

Spatial reasoning, for us

Picking locations in the game world to choose the best one.

DSM

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

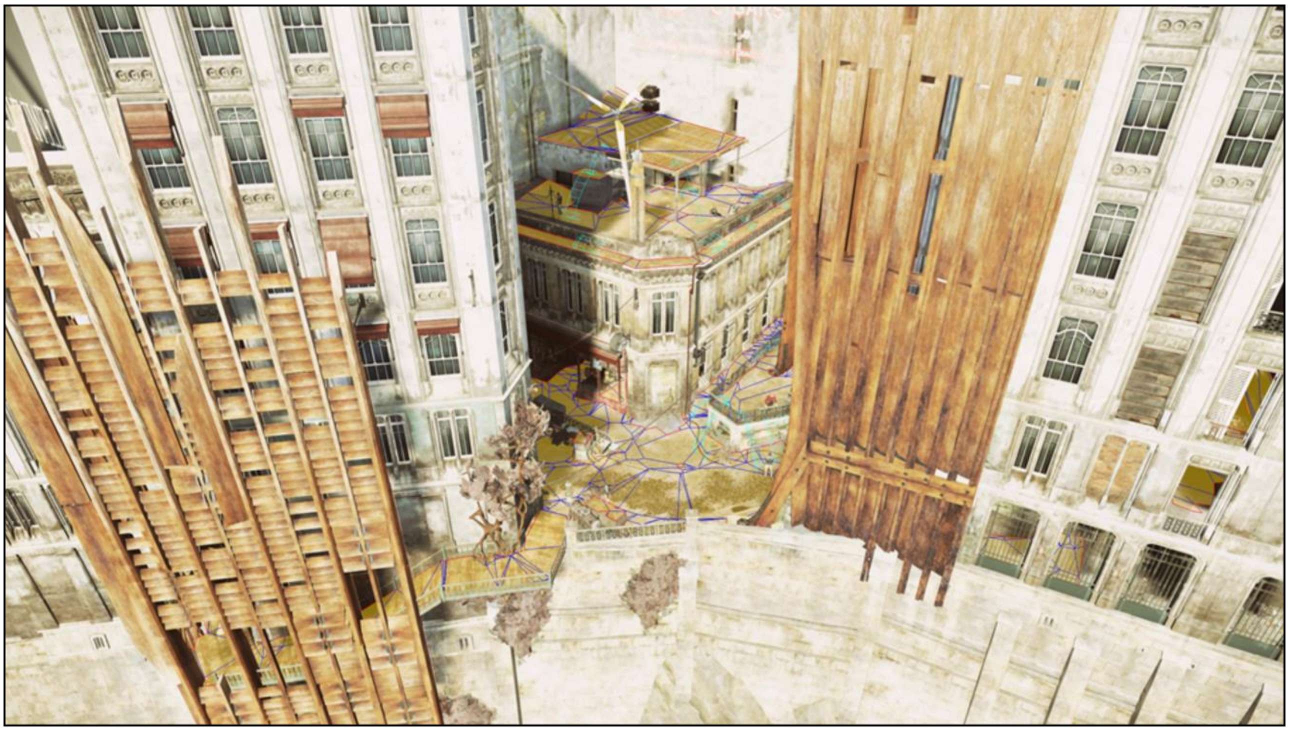
Now to summarize all this, this is what the spatial reasoning is about for us: scoring positions in the world.

And in order to do that, you have to have a finite number of positions, and then to evaluate them.

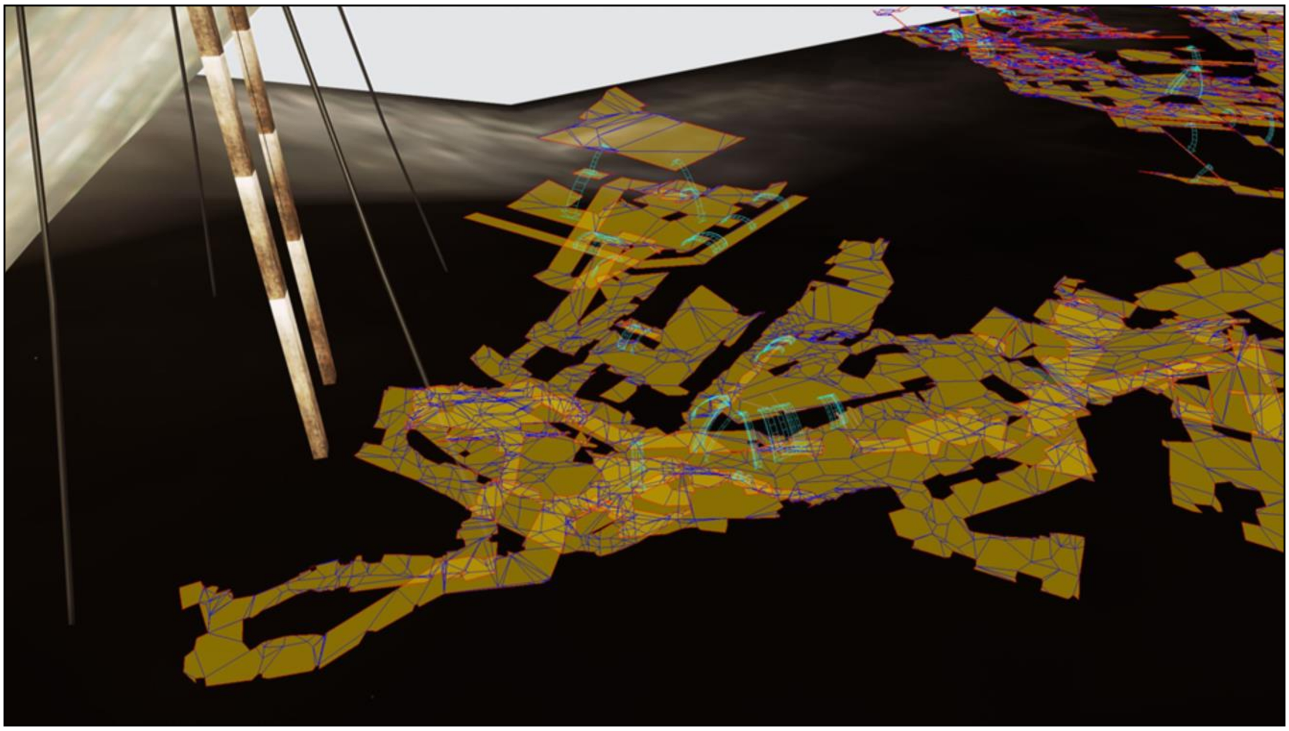
DSM helps for both of those, so let’s start with the first part.

Let me introduce the main element of the DSM, the layout.

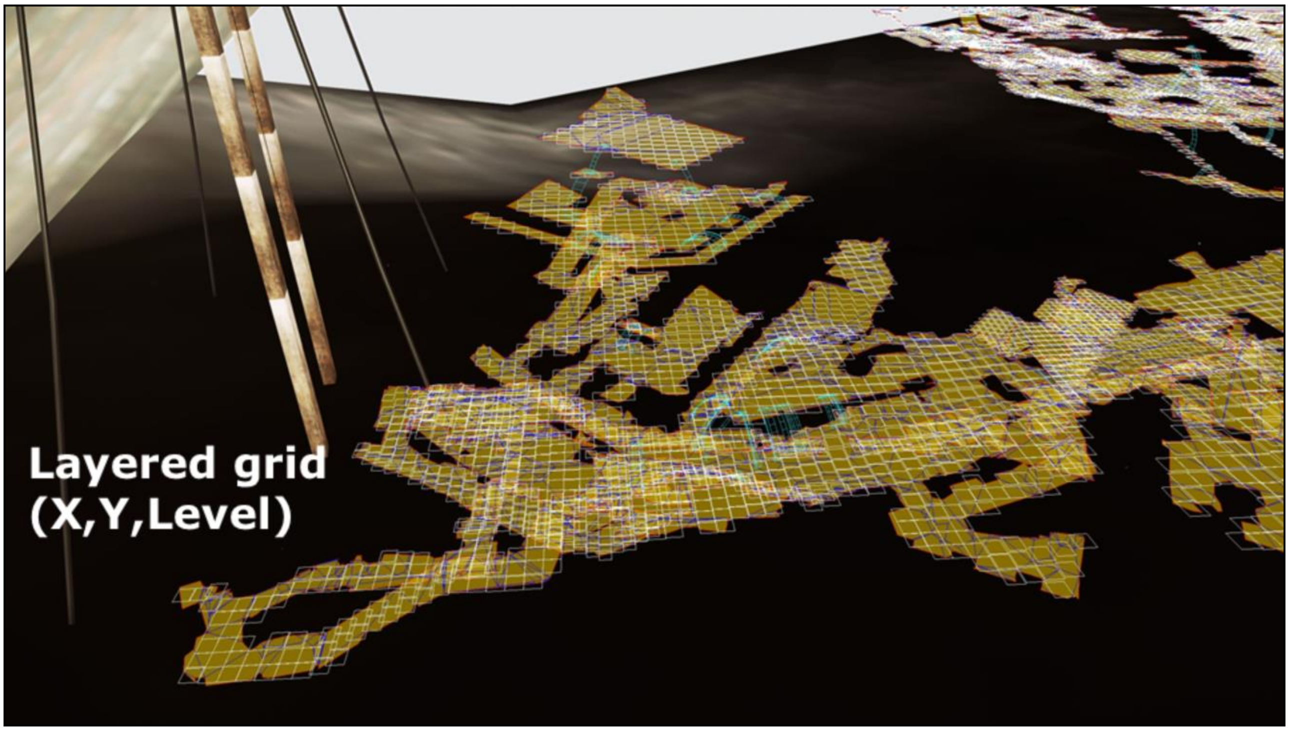
The whole point of the layout is discretizing the geometry of the map



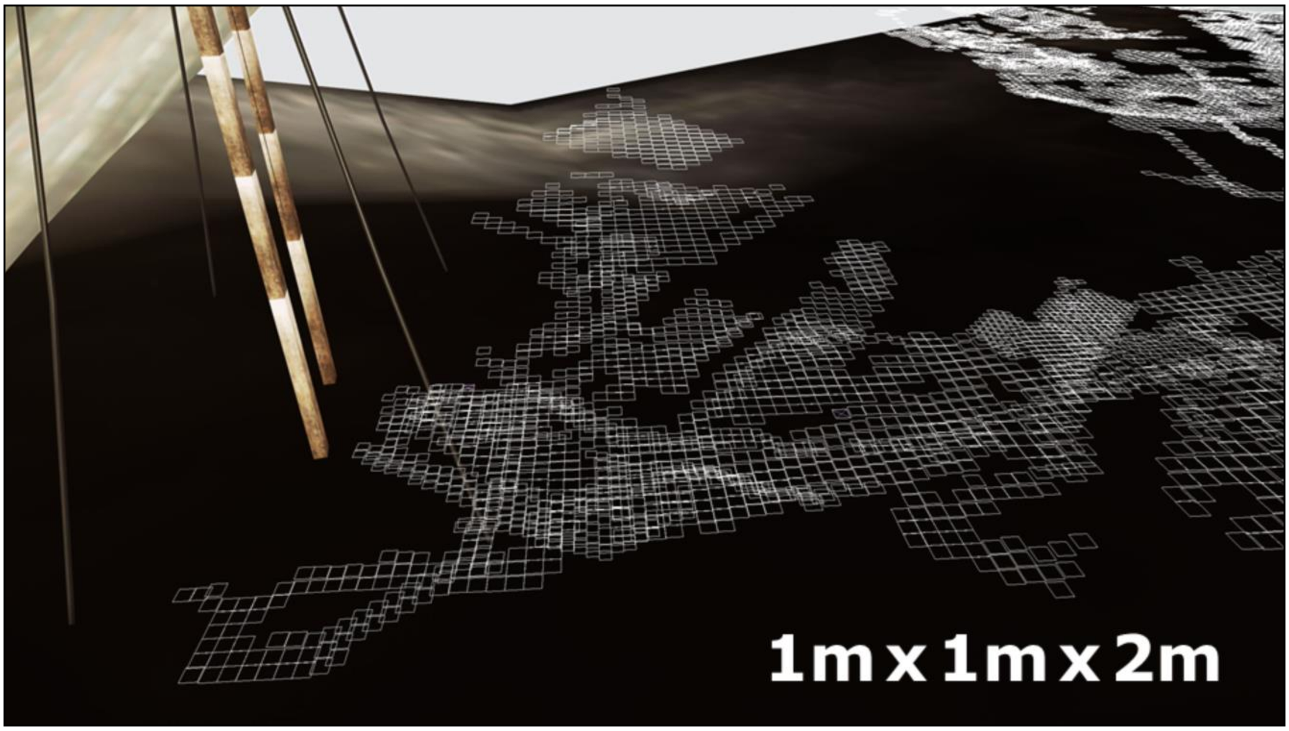
Or more precisely, discretizing the navmesh, which depends on the map geometry



We do computation offline, going from each navmesh face…

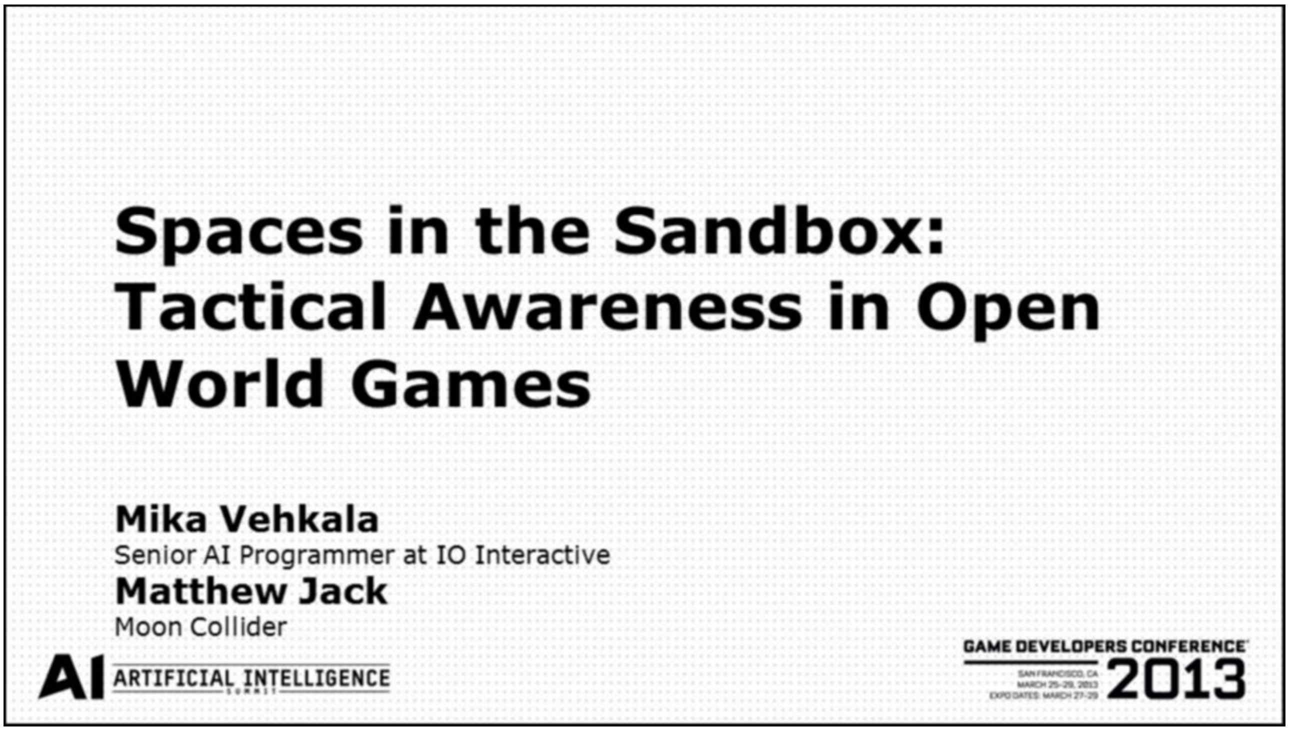


… to a bunch of cells forming a layered grid…



Et voilà !

The grid resolution is 1m x 1m x 2m and its origin is the minimum point of the navmesh bounding box.



Spaces in the Sandbox: Tactical Awareness in Open World Games

Mika Vehkala
Senior AI Programmer at IO Interactive
Matthew Jack
Moon Collider

AI ARTIFICIAL INTELLIGENCE SUMMIT

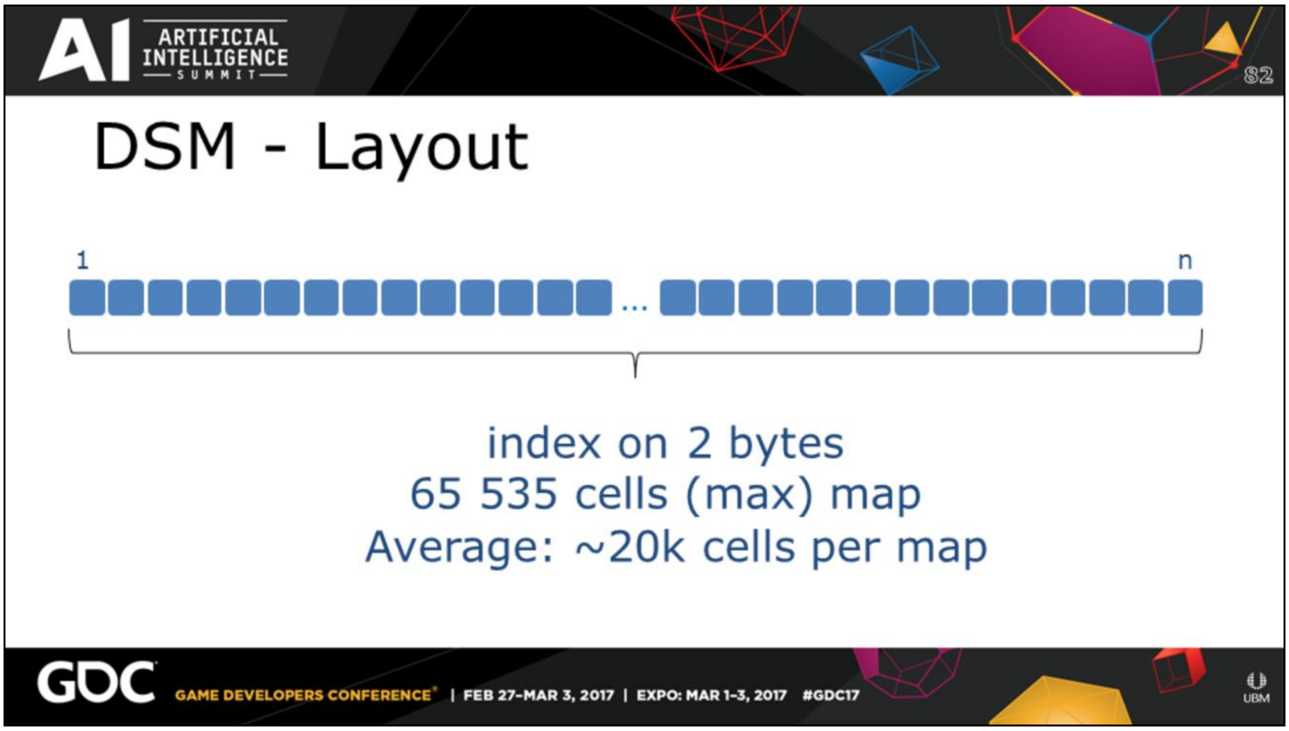
GAME DEVELOPERS CONFERENCE 2013
SAN FRANCISCO, CA
MARCH 25-29, 2013
EXPO DATES: MARCH 27-29

There is ample literature around on how to build a grid from a navmesh. We can recommend this talk* among other.

Thus I’m not going to detail those techniques.

---

[*] Spaces in the Sandbox: Tactical Awareness in Open World Games
http://gdcvault.com/play/1018136/Spaces-in-the-Sandbox-Tactical



DSM - Layout

1
n
...

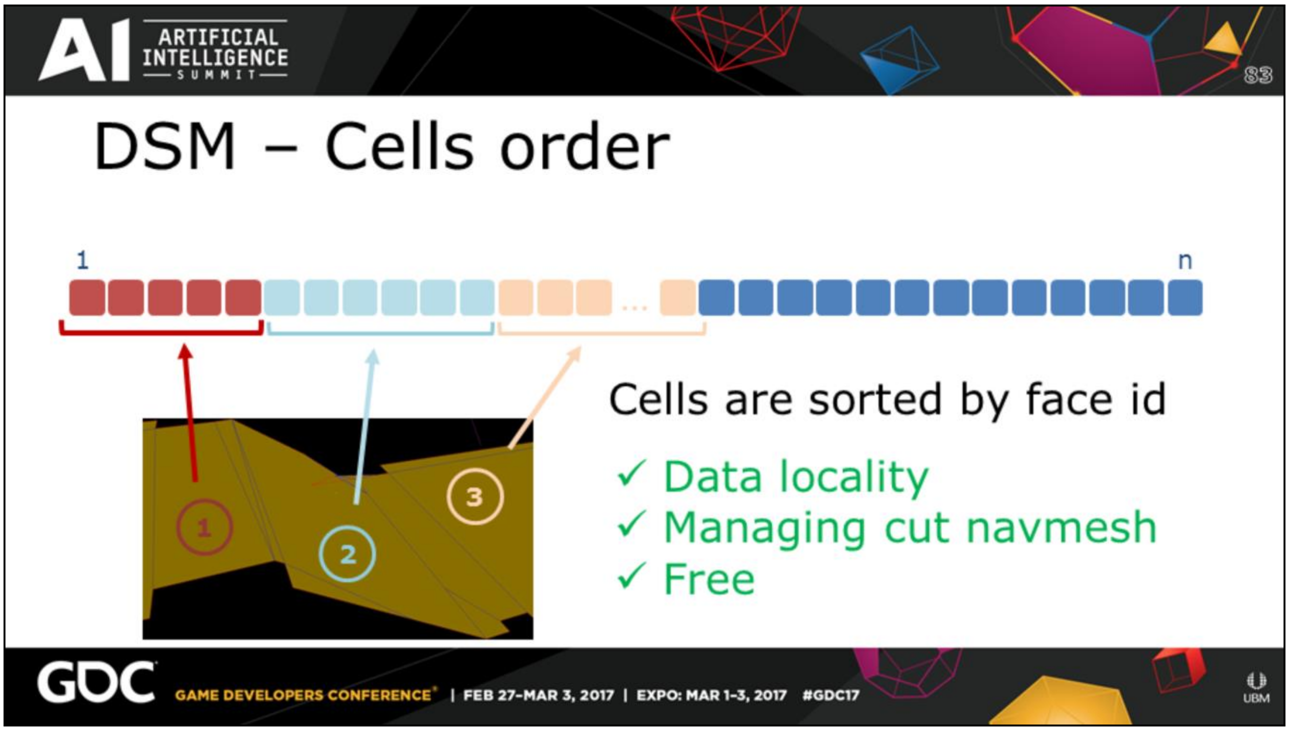
index on 2 bytes
65 535 cells (max) map
Average: ~20k cells per map

ARTIFICIAL INTELLIGENCE SUMMIT
82
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

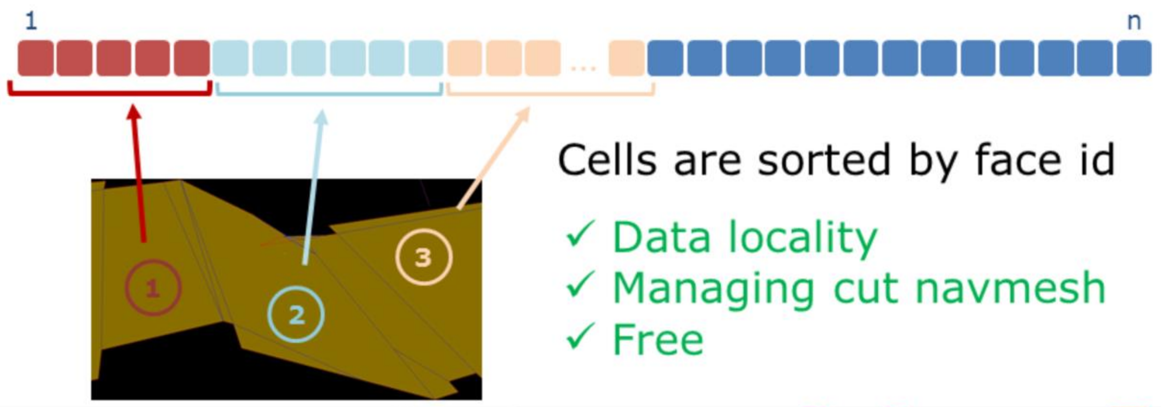
The data representation of the DSM grid is a big array of cells.

We encode the array index on 2 bytes, which allows more than sixty thousands cells in a map.

This is sufficient for us, as in average we have twenty thousands cells.



DSM – Cells order



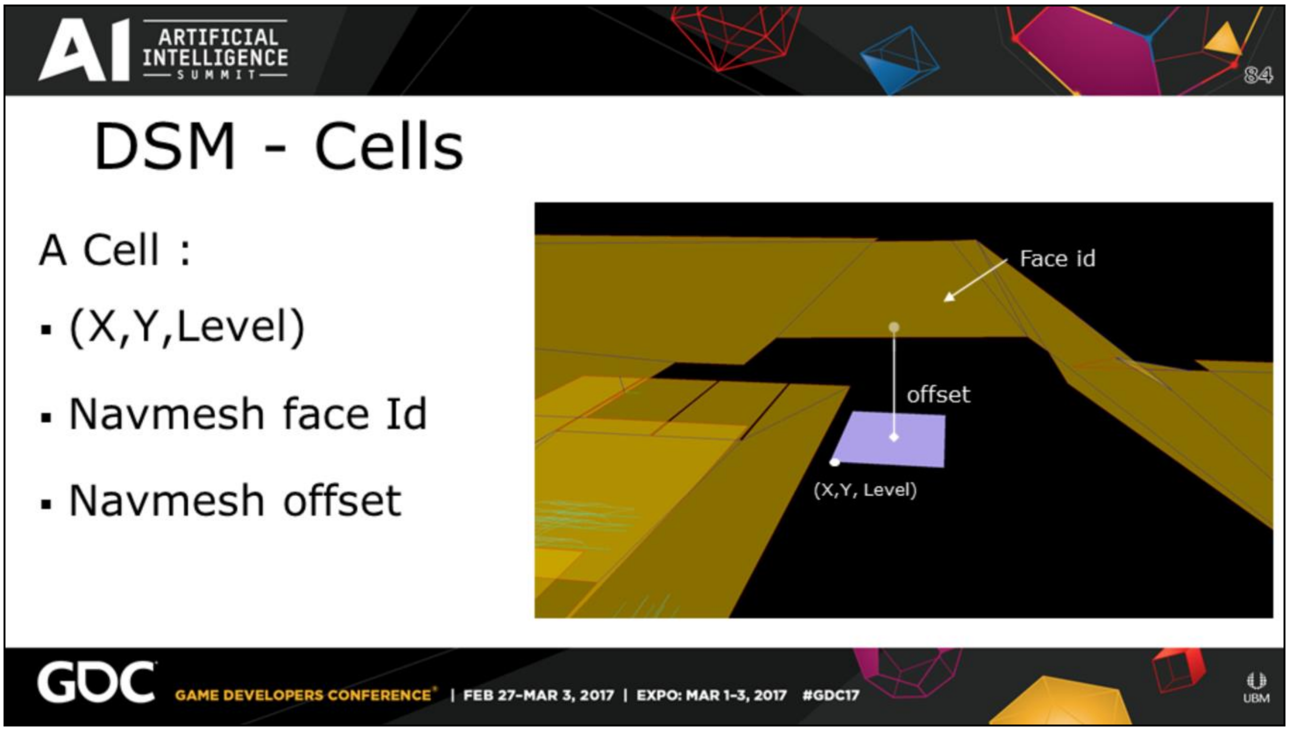
Cells are sorted by face id

✓ Data locality
✓ Managing cut navmesh
✓ Free

Now let me talk real quick about the order of the cells in the array.

They are naturally grouped by navmesh faces. This is an interesting property:

- First it enhances data locality, cells close in the world tend to be close in memory
- It's also interesting when the navmesh is getting cut, we'll touch on that a little bit later
- Lastly it's basically free, because it's just a consequence of the way we construct the layout, iterating over all navmesh faces



DSM - Cells

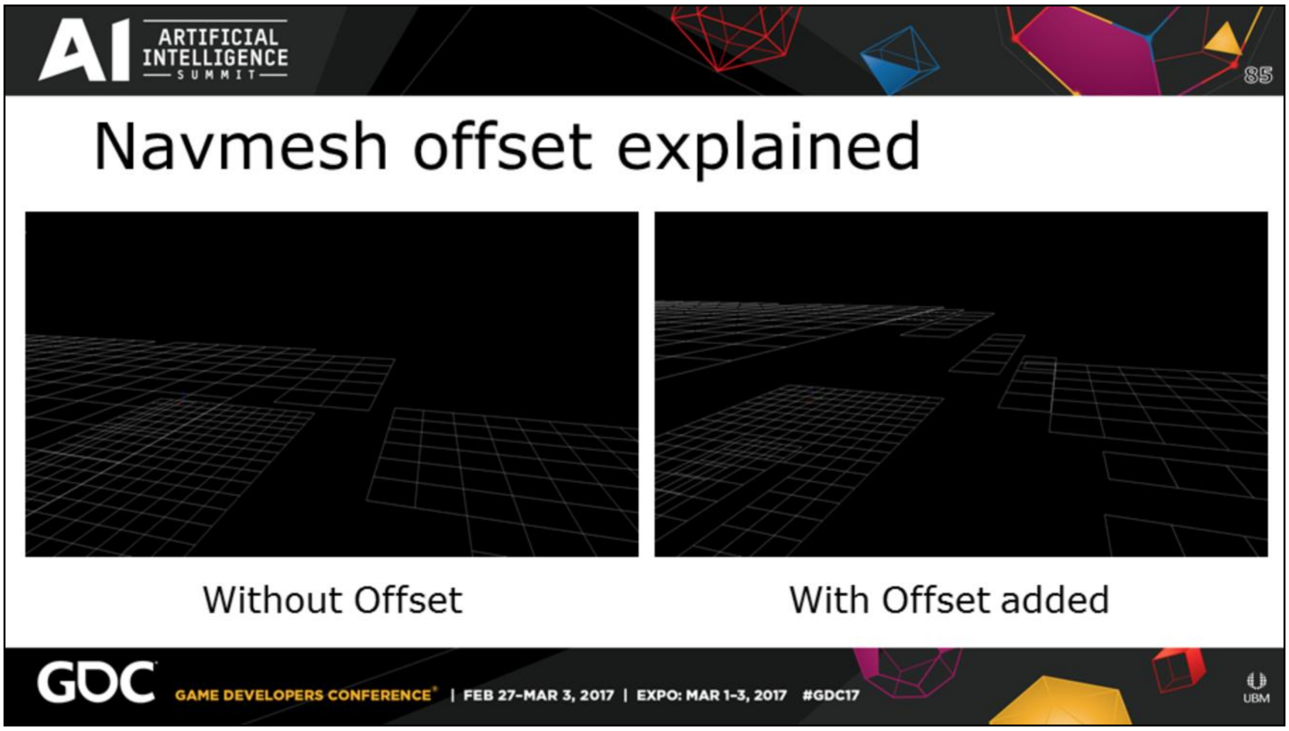
A Cell :

- (X,Y,Level)
- Navmesh face Id
- Navmesh offset

Face id
offset
(X,Y, Level)

Each cell is a data structure containing:

- The grid “coordinates” (X, Y, Level)
- The navmesh face id it was created from
- The vertical offset from their center to this navmesh face


AI ARTIFICIAL INTELLIGENCE SUMMIT
85
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Navmesh offset explained

Without Offset

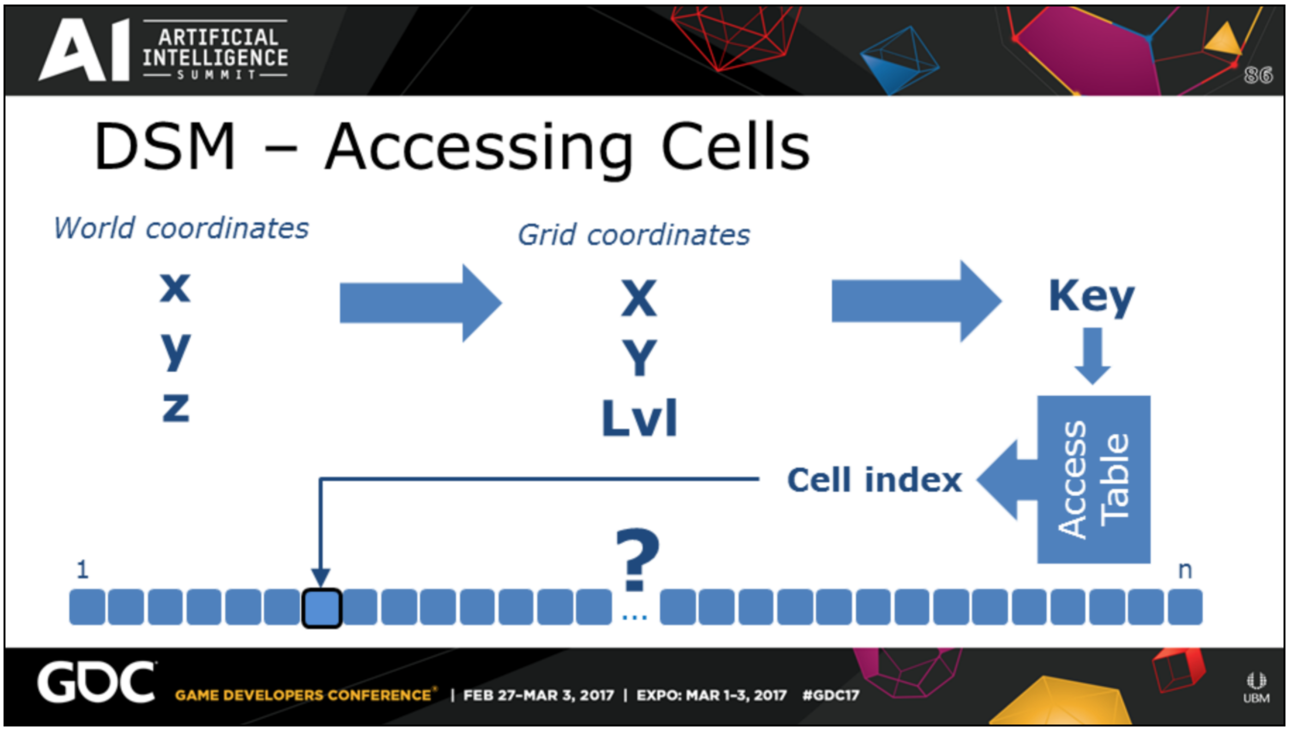
With Offset added

Navmesh offset is important, because without it, we only have a bunch of flat grid layers to play with, which is not an ideal representation of the world, unless you're making a game on Apple II.

As you can see on the right, when taking navmesh offset into account, our cells are far closer from what the world actually looks like.

And as this is all about picking locations in the world, it's better if those locations actually make sense right ?

Using this offset, we can go from a cell to a real world location, but what about doing it the other way around?



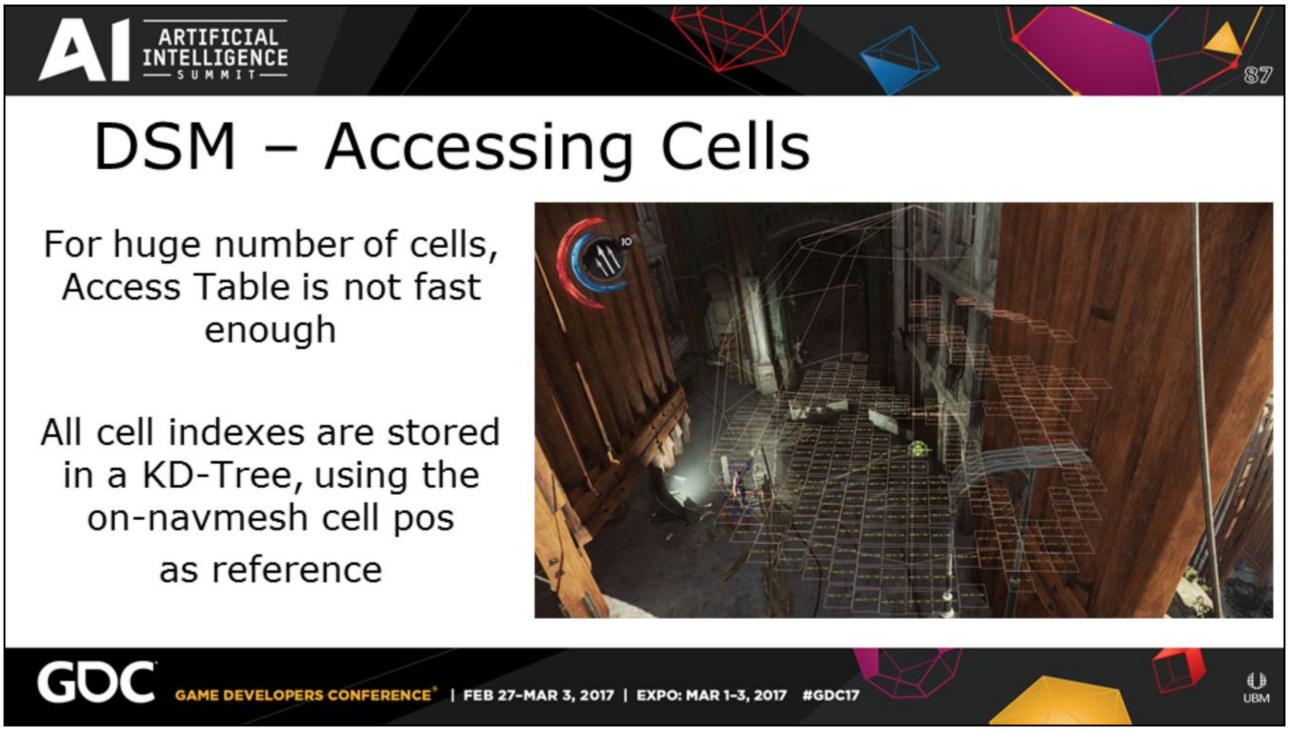
DSM – Accessing Cells

World coordinates
x
y
z
Grid coordinates
X
Y
Lvl
Key
Access Table
Cell index
?
1
n
...
86
ARTIFICIAL INTELLIGENCE SUMMIT
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Let's say you have a location in the world, and wants to find which cell it belongs to.

Knowing the origin and the resolution of the DSM grid, we know we can translate from world coordinates, to grid coordinates.

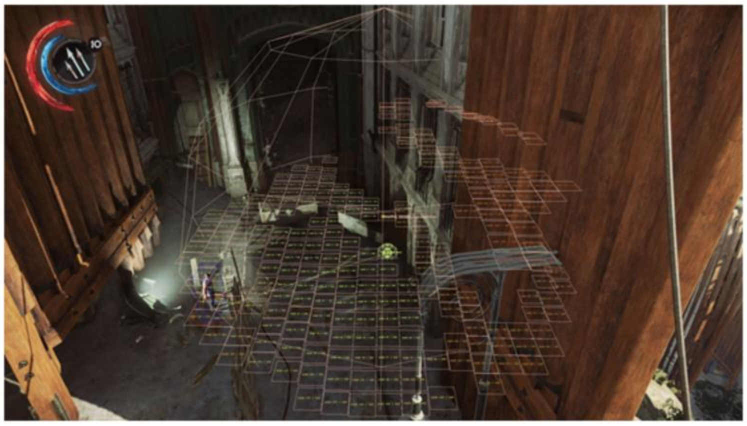
Then from those coordinates we can construct a Key, that we then feed to a hash table we call "Access table", and get the resulting cell index. Easy.



DSM – Accessing Cells

For huge number of cells, Access Table is not fast enough

All cell indexes are stored in a KD-Tree, using the on-navmesh cell pos as reference



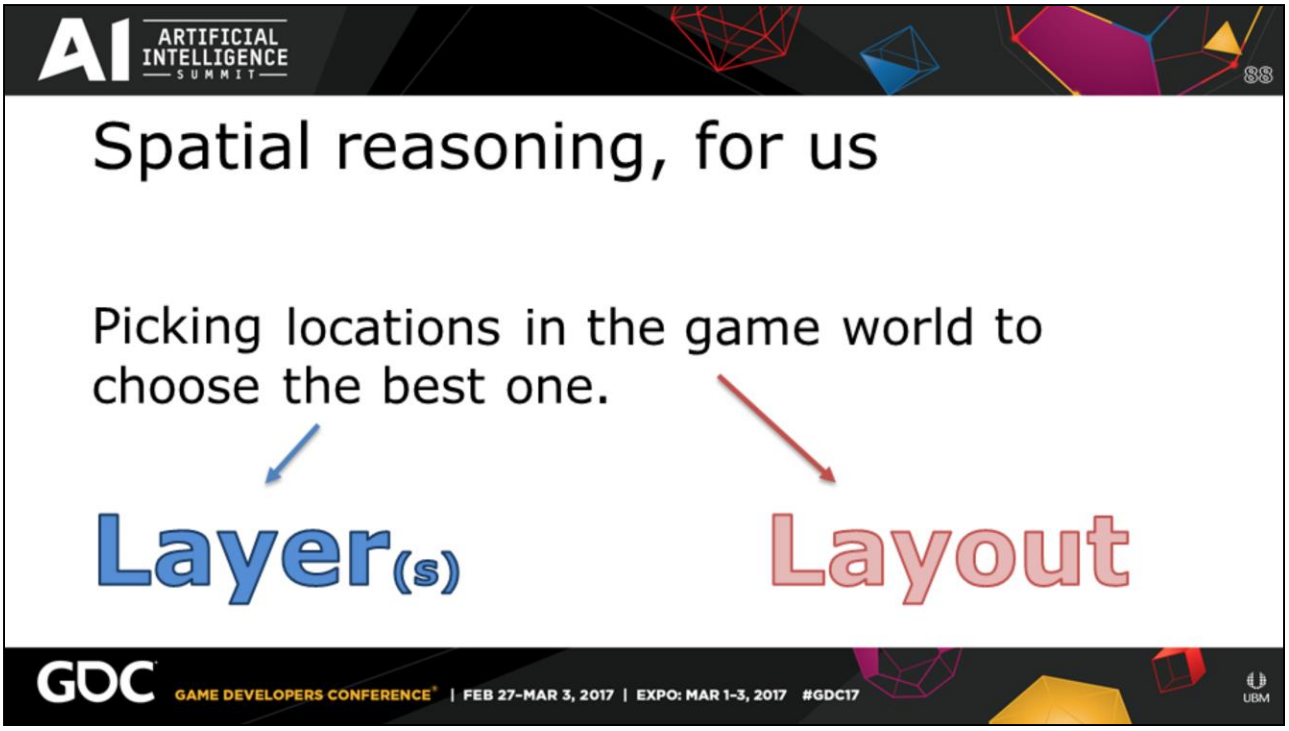
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Sometime the access table is not fast enough, when you need to access a lot of cells at once.

Thus we also had cell indexes stored in what we call the access tree, which is in fact a k-d tree.

With this we can formulate queries like “give me the list of cells that are within a 5 meter radius from this point in the world”. Pretty handy.

OK, with those two ways of collecting cells, we’re pretty much covered.

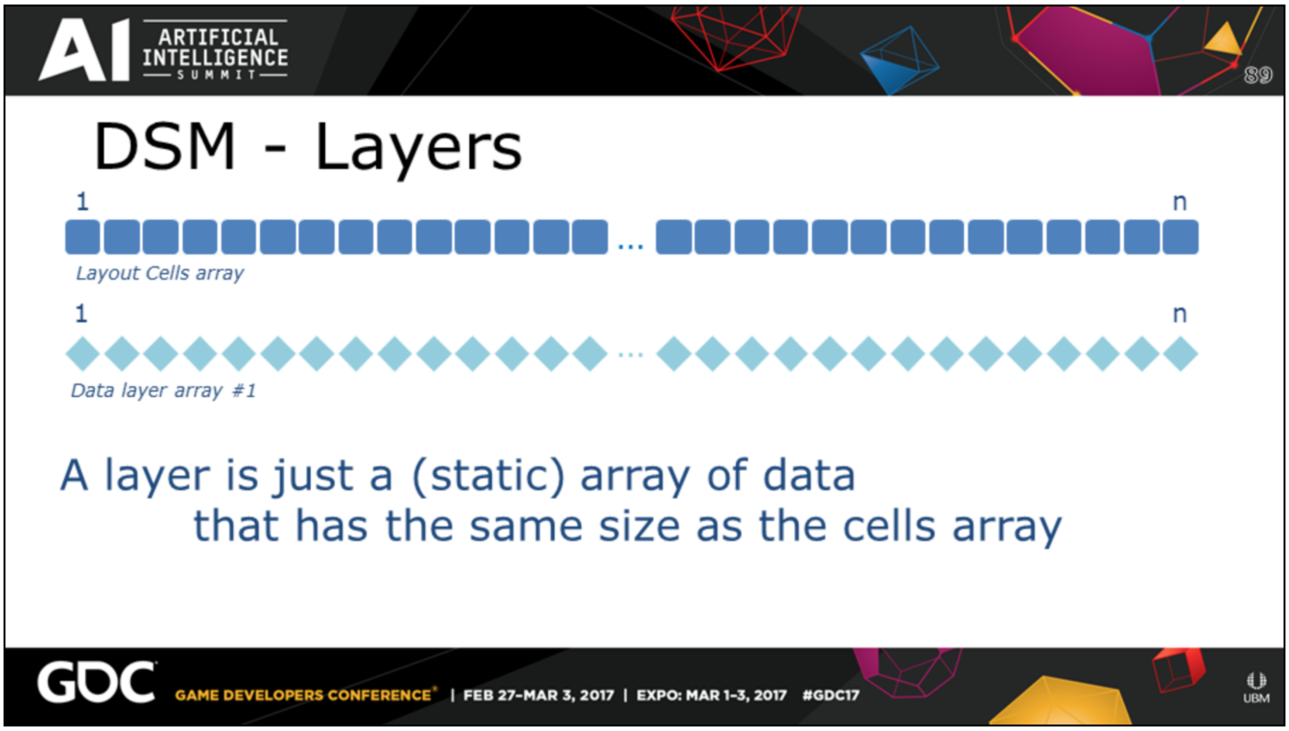


Spatial reasoning, for us

Picking locations in the game world to choose the best one.

ARTIFICIAL INTELLIGENCE SUMMIT
88
Layer(s)
Layout
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

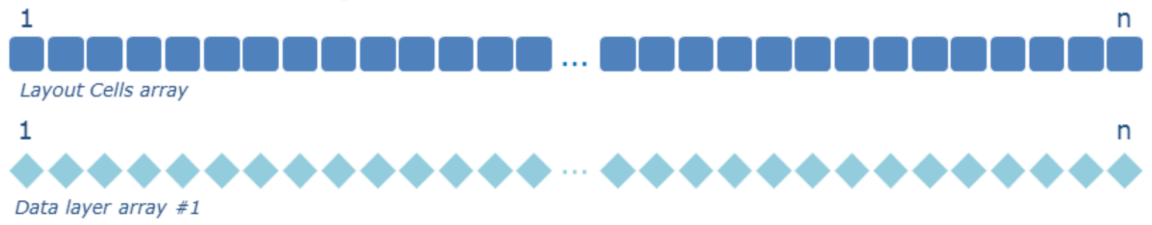
If we go back to our statement earlier, we saw that the world discretization part is supported by the layout. Now let’s discuss the evaluation part, that is supported by what we call layers.



ARTIFICIAL INTELLIGENCE SUMMIT

89

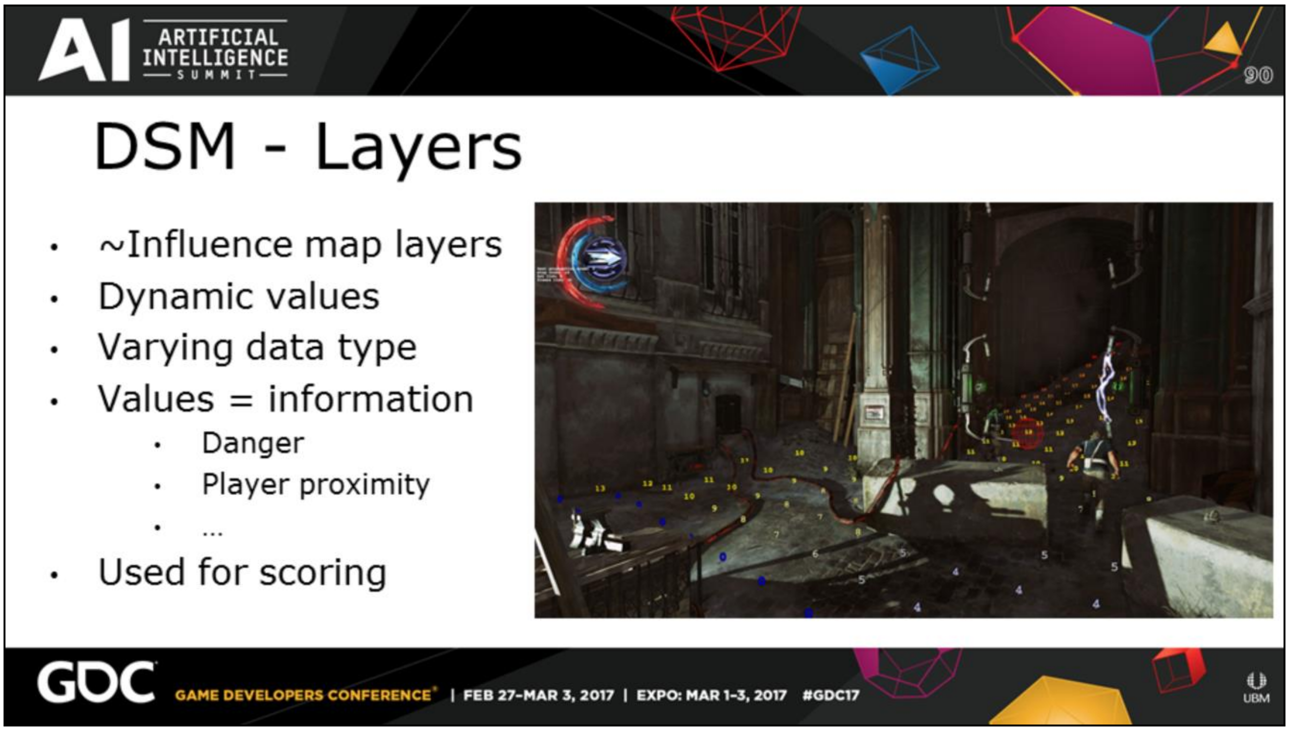
DSM - Layers



A layer is just a (static) array of data
that has the same size as the cells array

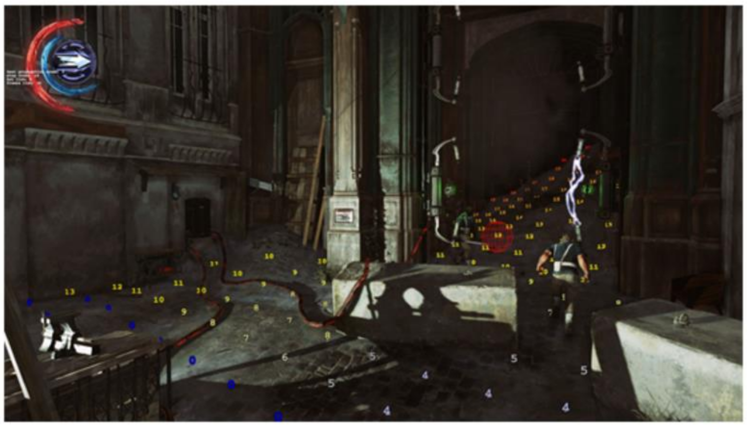
GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

A layer is simply an array of values. The array is allocated in one block, and is the same size as the layout cells array



DSM - Layers

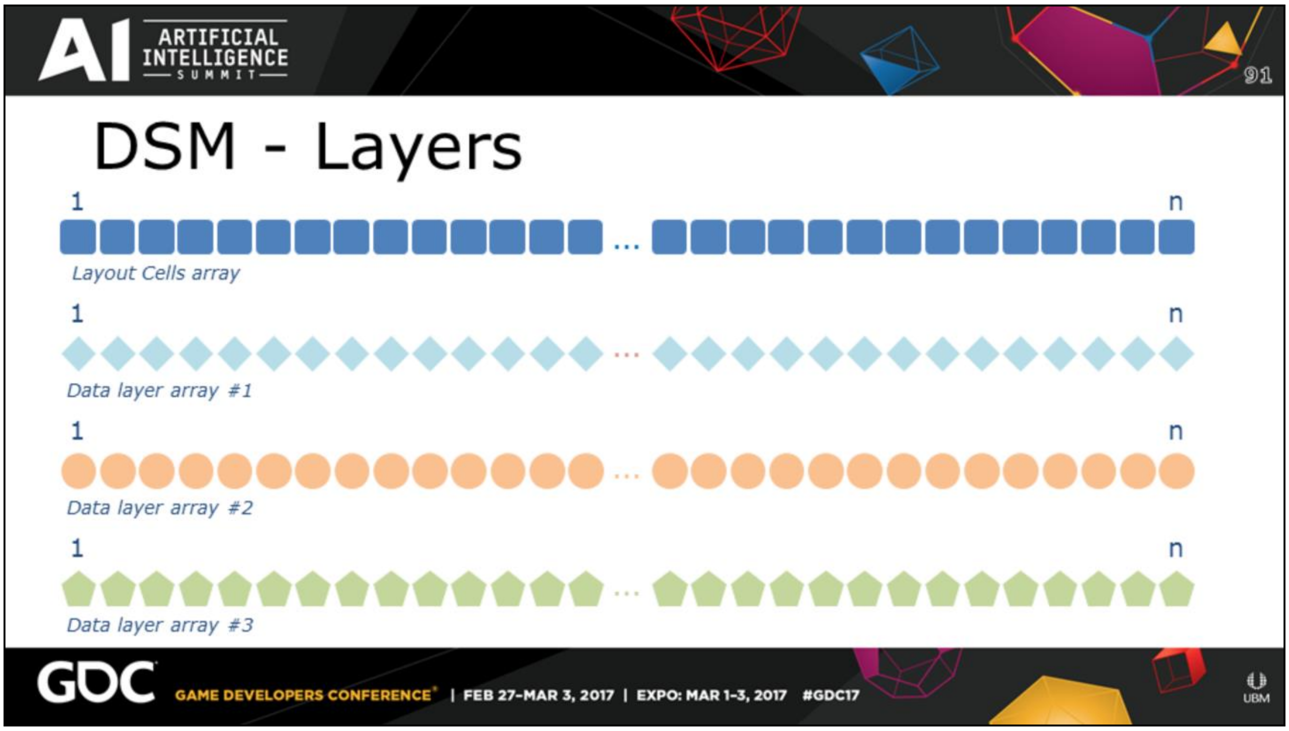
- ~Influence map layers
- Dynamic values
- Varying data type
- Values = information
    - Danger
    - Player proximity
    - …
- Used for scoring



Layers are the dynamic part of the DSM system, they are equivalent to the classical influence map layers, for those familiar with this concept.

Unlike the layout, layer data is allowed to change at runtime.

It can store any data type. Those values are typically information on the game world, like danger rating of a cell or proximity to the player, etc. Their main function is to support scoring of each cell.



ARTIFICIAL INTELLIGENCE SUMMIT

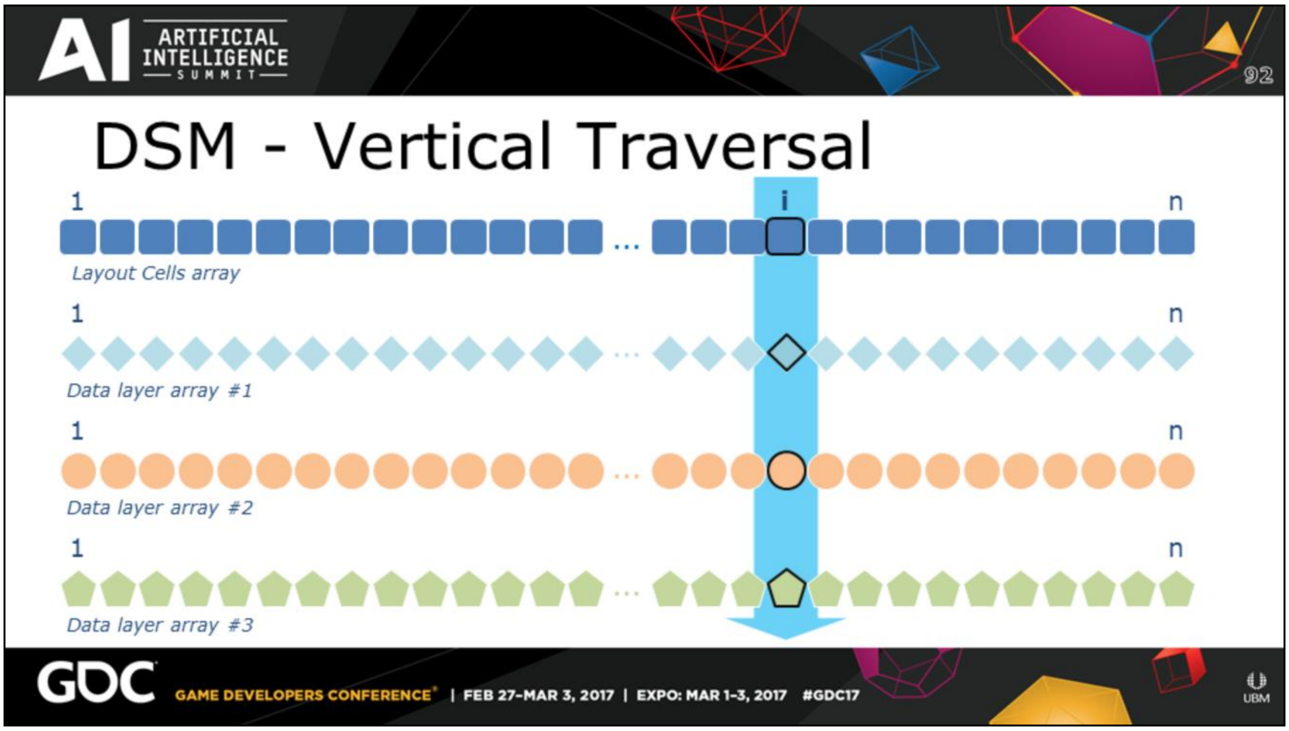
91

DSM - Layers

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

The nice thing is that you can add as many layers as you need, and each layer memory cost is mainly dependent on the type of data you choose to store in it.

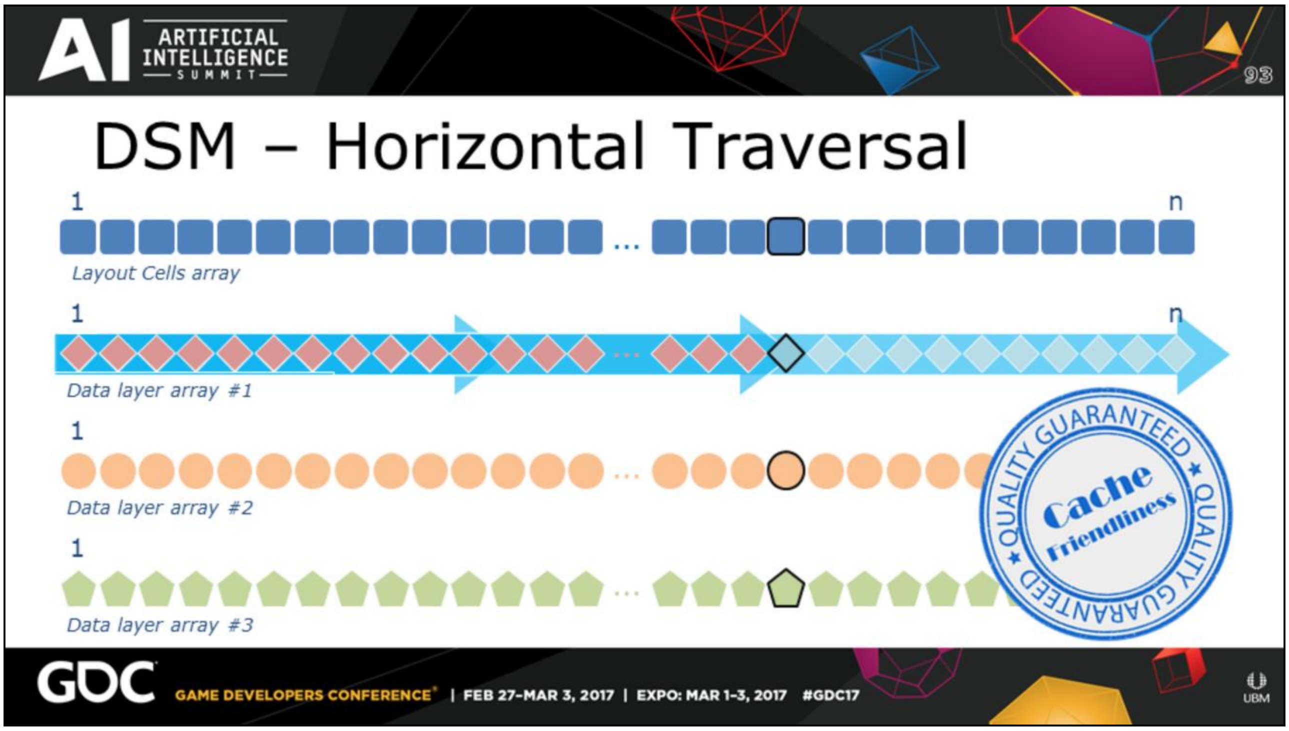


DSM - Vertical Traversal

Once you have one cell index, you can instantly access any piece of data about that cell, in any layer, constant time.

Hence we can combine layer values easily for our scoring need.

Let’s call this Vertical Traversal.

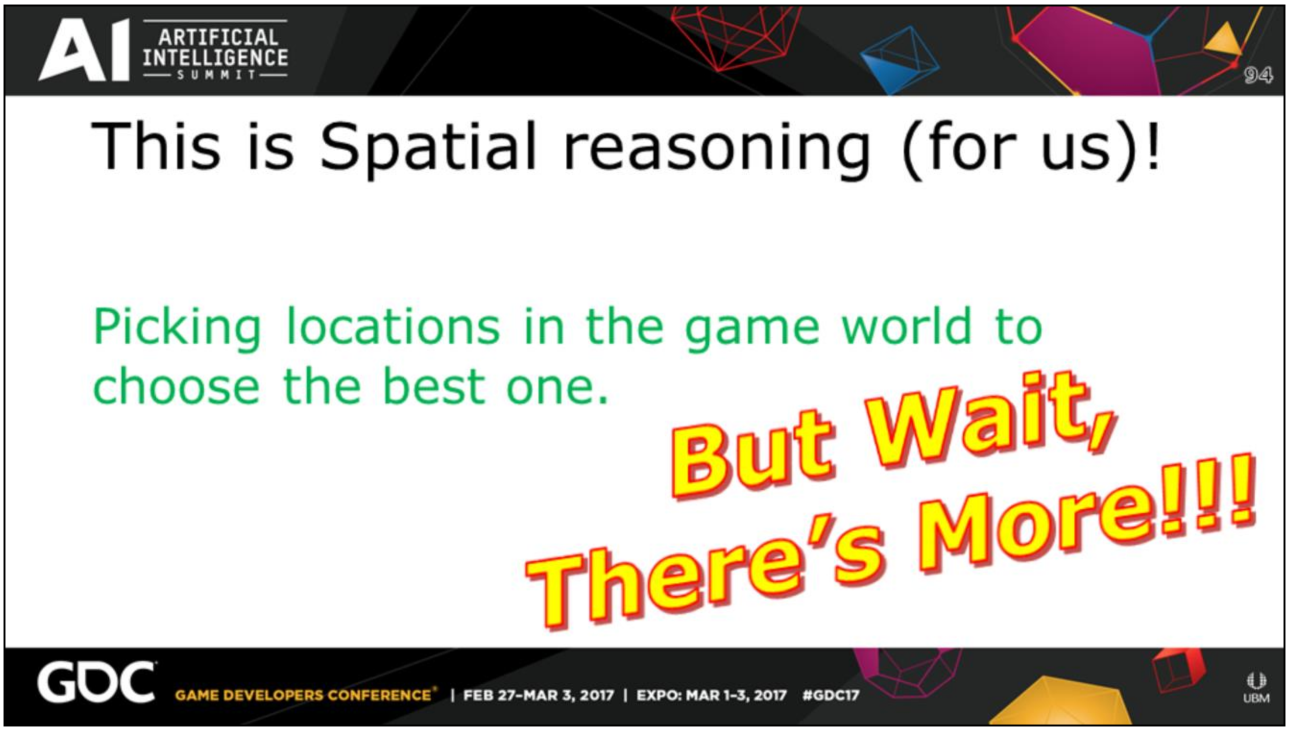


DSM – Horizontal Traversal

Another interesting things to do with layers, is traversing them sequentially, until finding a cell that has the correct value.

Once we found one, then we have a cell index and thus we can directly access data about that very cell, be it in the layout or in another layer. Then we can carry on.

Let's call this Horizontal Traversal, and please do note, good friends, that this technique was awarded the totally unofficial Cache Friendliness Seal of Quality. Traversing layers that way is really, really fast. Provided you put reasonably-sized data in them of course.



ARTIFICIAL INTELLIGENCE SUMMIT

94

This is Spatial reasoning (for us)!

Picking locations in the game world to choose the best one.

But Wait, There's More!!!

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

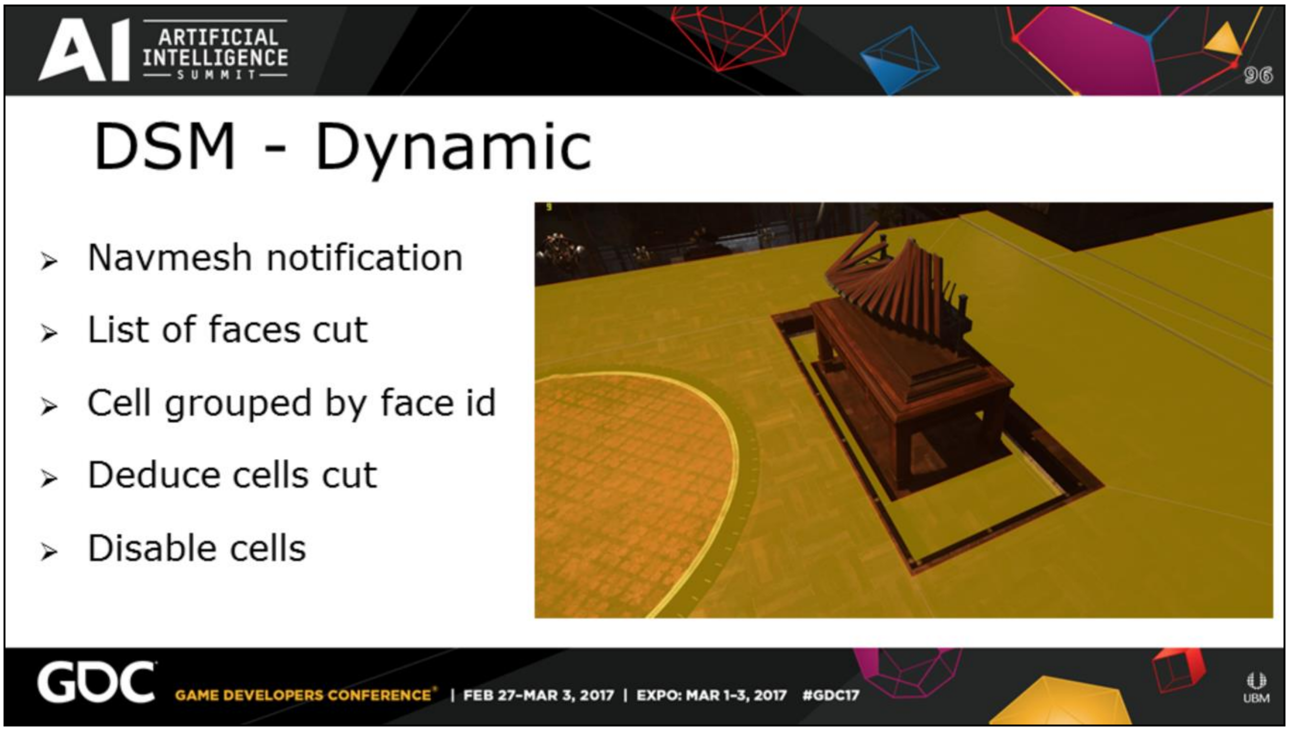
Ok we now can pick locations in the game world, and score them, great.

We're ready to take a look at a concrete example. But before that, we need to discuss a particular problem we had to solve.

<originally a video>

Remember the moving floors and walls of clockwork mansion?

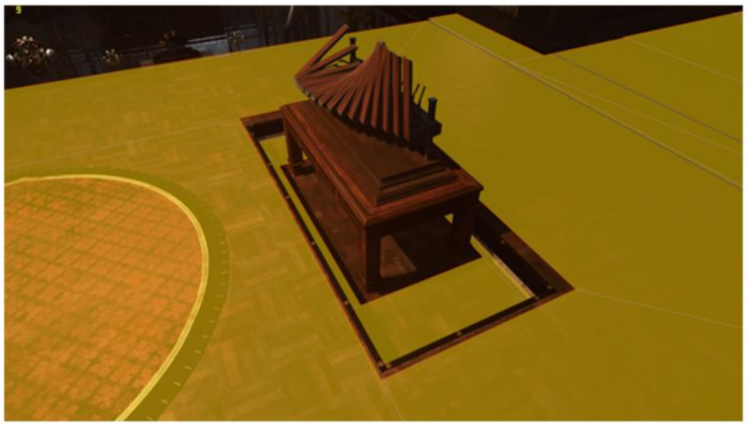
We obviously had many cases of navmesh being cut dynamically. How did we manage that?



ARTIFICIAL INTELLIGENCE SUMMIT

DSM - Dynamic

- Navmesh notification
- List of faces cut
- Cell grouped by face id
- Deduce cells cut
- Disable cells

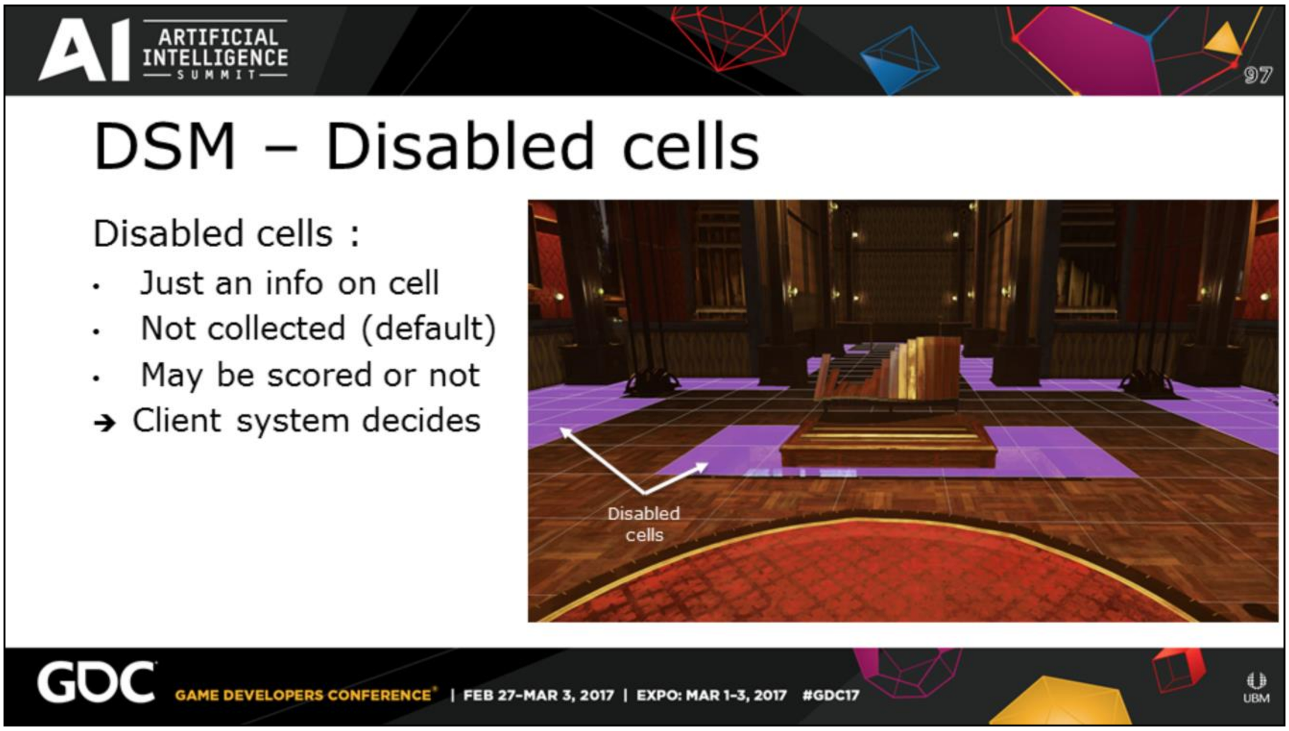


When the navmesh gets cut by any kind of object (like a door or this piece of furniture), the DSM gets notified with the list of the navmesh faces affected.

Remember how cells are grouped by face id in the layout array?

This allows us to know which cells belong to the list of cut navmesh faces, very rapidly.

From there it's easy to test which cells are actually cut, and disable them.



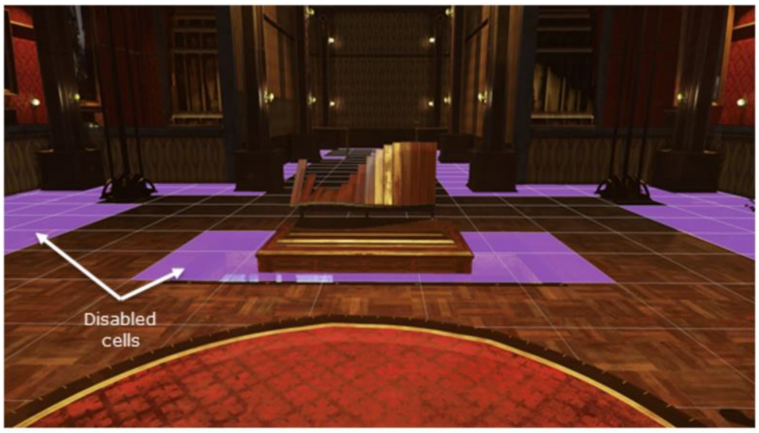
ARTIFICIAL INTELLIGENCE SUMMIT

DSM – Disabled cells

Disabled cells :

- Just an info on cell
- Not collected (default)
- May be scored or not

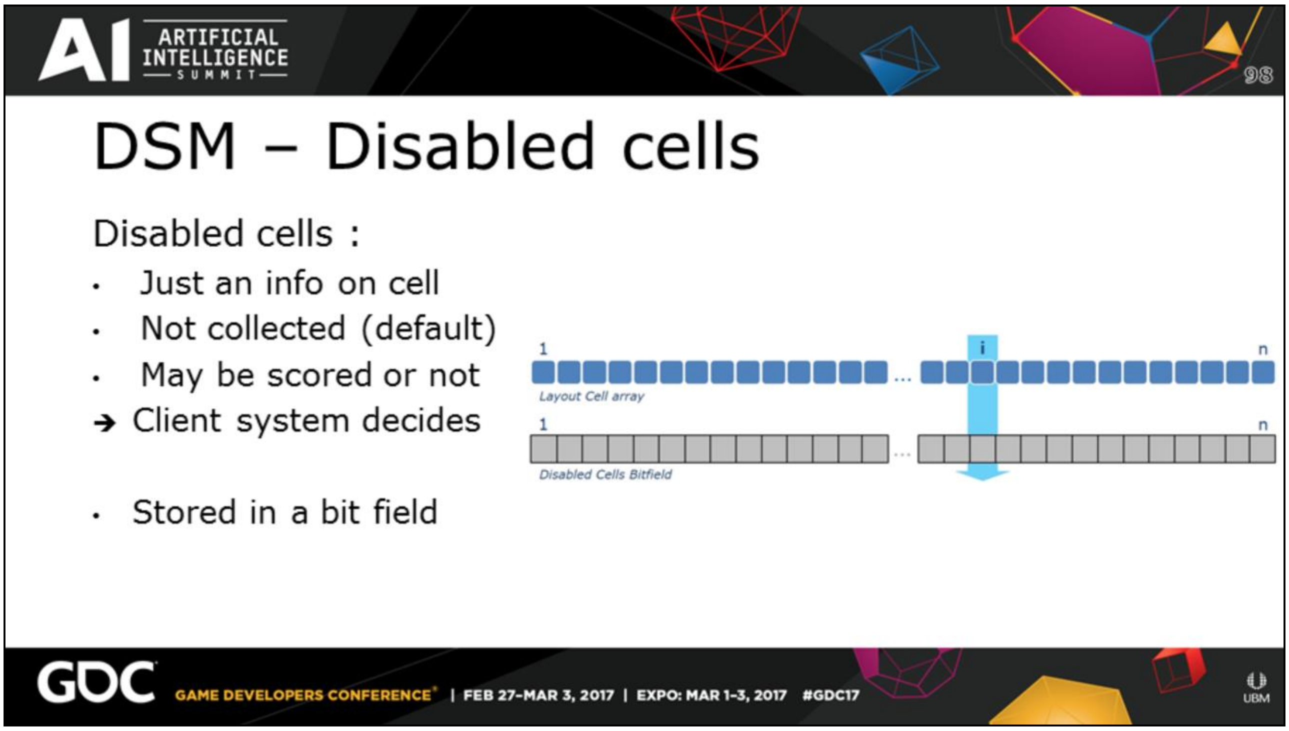
➔ Client system decides



GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Disabling a cell is merely another information on it.

The access table, and the access tree do not return disabled cells, by default. But in the end it really is up to client code to decide if the disabled status is taken into account or not.



ARTIFICIAL INTELLIGENCE SUMMIT

98

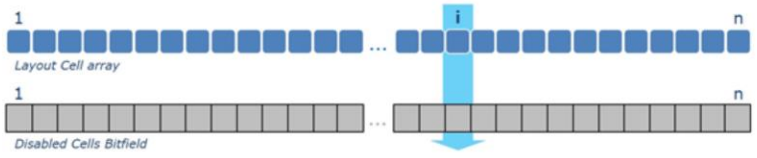
DSM – Disabled cells

Disabled cells :

- Just an info on cell
- Not collected (default)
- May be scored or not

➔ Client system decides

- Stored in a bit field



GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

The disabled status itself is stored in a bit field containing as many bits as the layout has cells.

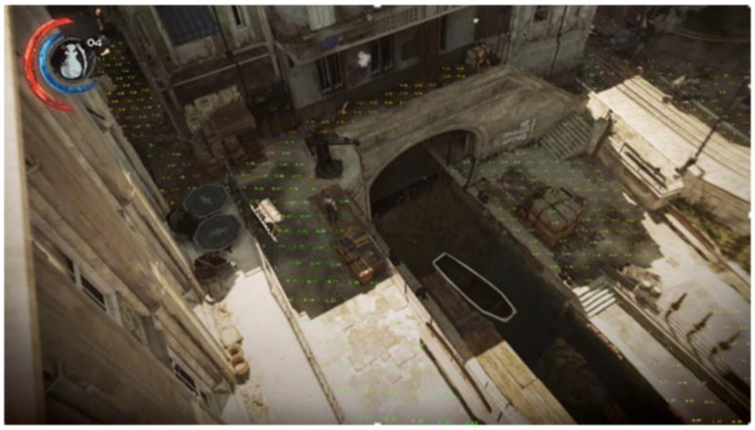
We use vertical traversal to check if a cell is enabled or not.

ARTIFICIAL INTELLIGENCE SUMMIT

In Practice - Searching

Search basic principle:

- Choose a destination
- Go to it
- Choose another one
- And so on ...
- Until end of timer

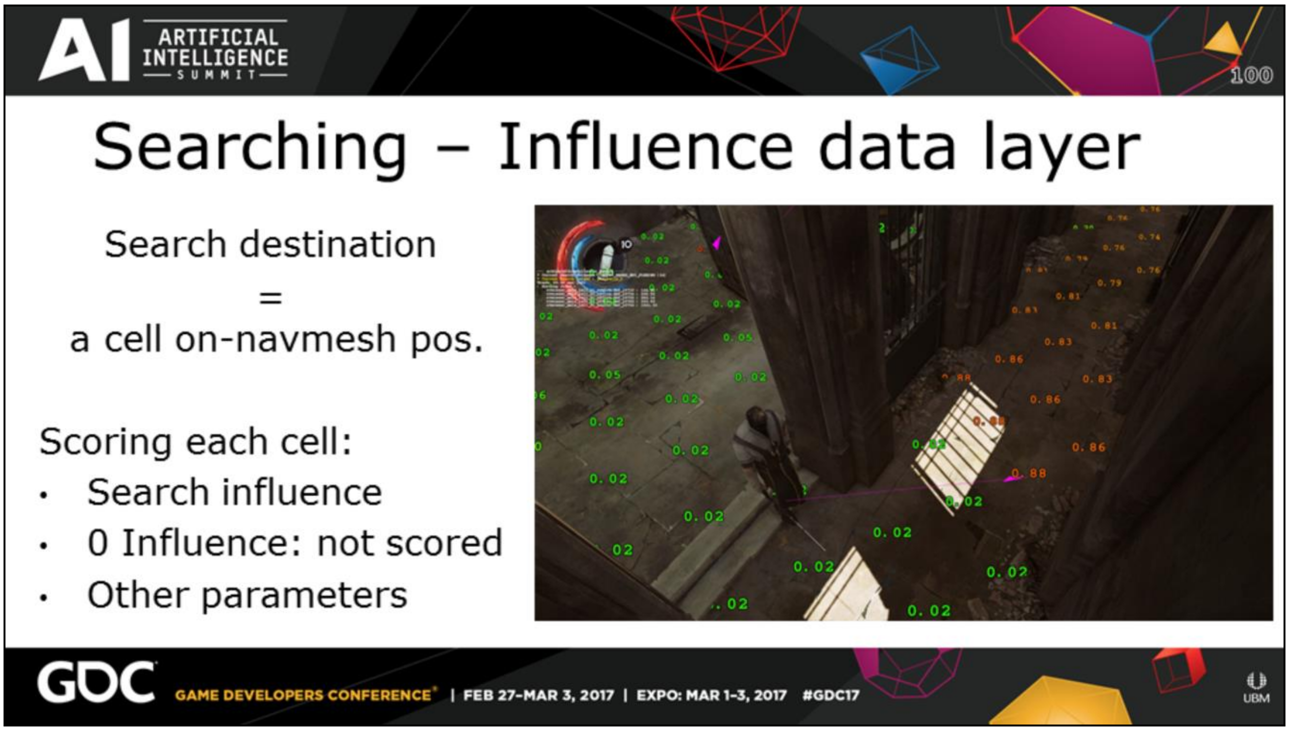


<originally a video>

Alright, let’s talk about an actual example of DSM usage in the game, which is search destination selection.

The basic principle of search is that NPCs choose one destination, go to it, and then choose another one, rinse and repeat until the end of a timer.

We use the DSM to decide which destination is selected.

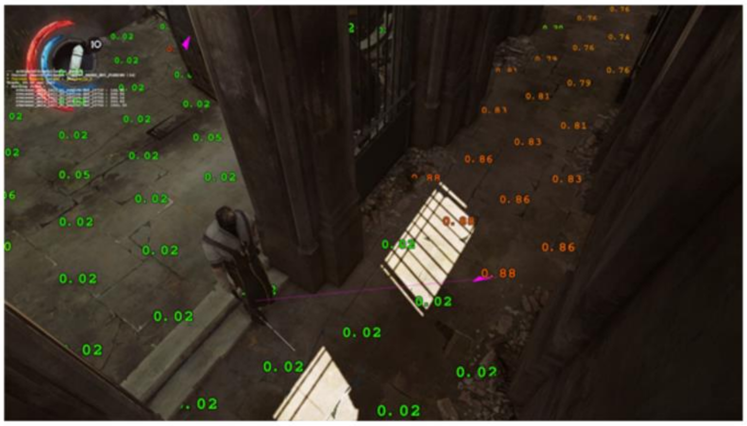


Searching – Influence data layer

Search destination
=
a cell on-navmesh pos.

Scoring each cell:

- Search influence
- 0 Influence: not scored
- Other parameters



GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

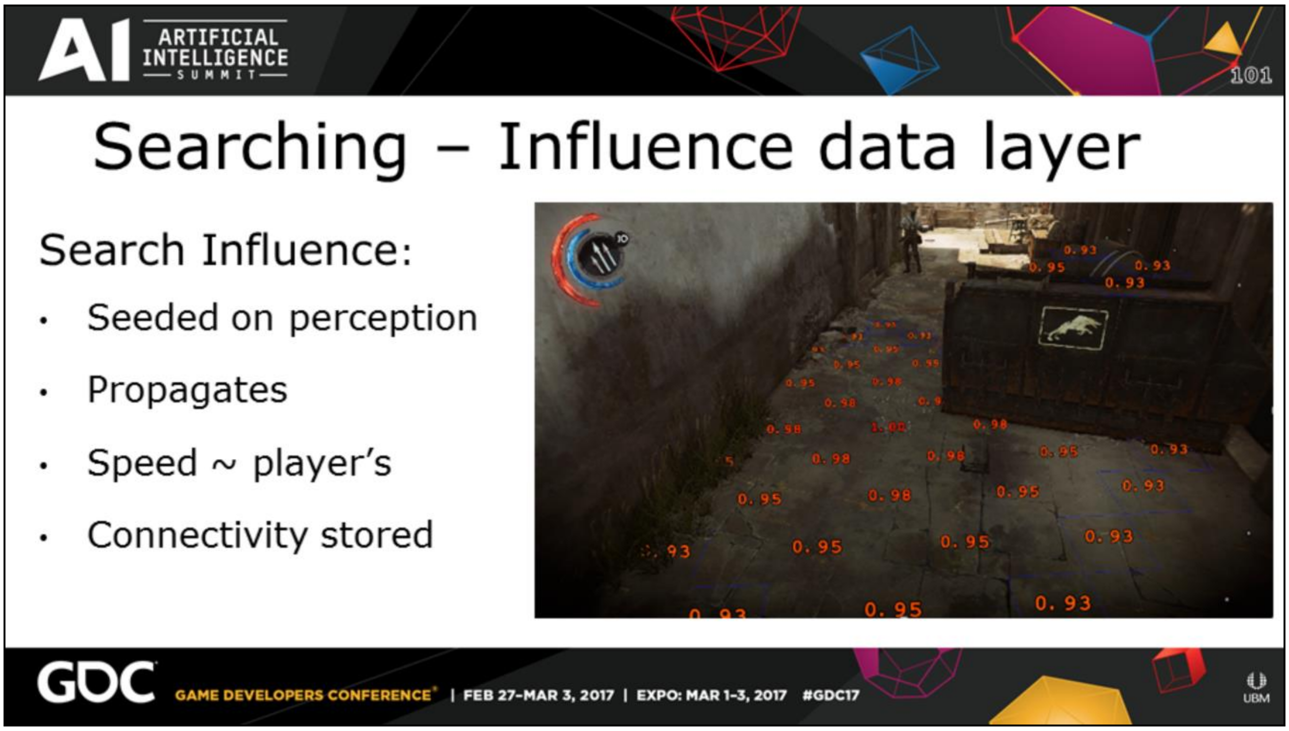
We decided potential search destination would be the on-navmesh position of a DSM cell.

Thus we just need to score each cell, and retrieve the best one. Simple right?

To drive this scoring we use a specific data layer we call Search influence layer.

Its value is used as the main factor for search destination scoring (cells without influence are not even considered).

We also use other factors to ponder this one, so that for instance NPCs are spread out enough, and choose cells that are not too close or too far from them etc.



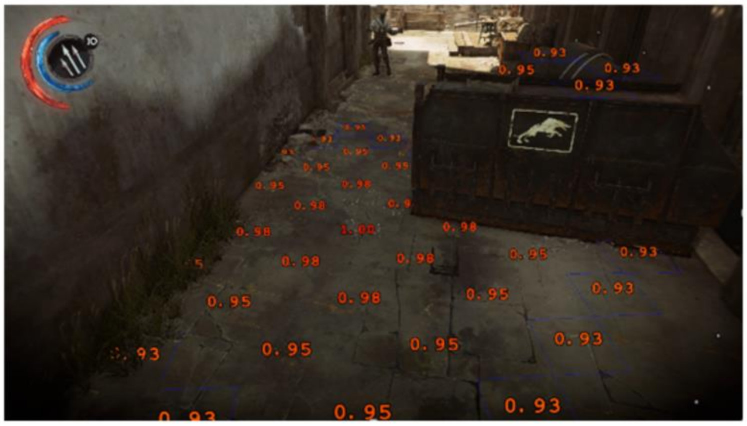
ARTIFICIAL INTELLIGENCE SUMMIT

101

Searching – Influence data layer

Search Influence:

- Seeded on perception
- Propagates
- Speed ~ player's
- Connectivity stored



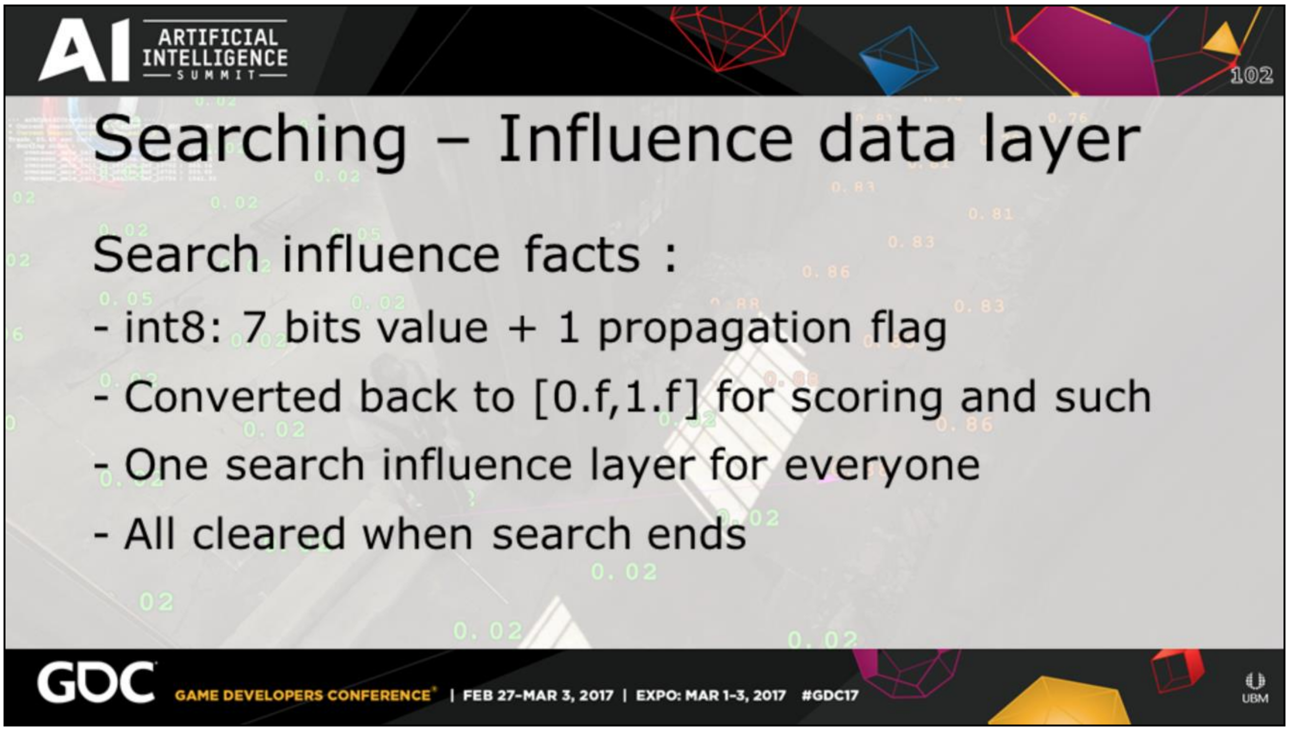
GAME DEVELOPERS CONFERENCE | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

<originally a video>

Whenever something is perceived by a NPC, influence is seeded on the layer, then propagated along neighboring cells, in all directions.

The propagation speed is roughly the same as the player speed.

In order to support propagation we keep connectivity information between cells in the layout.



Searching – Influence data layer

Search influence facts :

- int8: 7 bits value + 1 propagation flag
- Converted back to [0.f,1.f] for scoring and such
- One search influence layer for everyone
- All cleared when search ends

The influence value itself is stored on 8 bits, using 7 bits for the value, and 1 for propagation. [0-127]

Please note, than when manipulating influence, it is converted into a floating point number between 0.f and 1.f, it’s more convenient for things like scoring.

We chose to have one search layer for everyone, so that the data is actually shared.

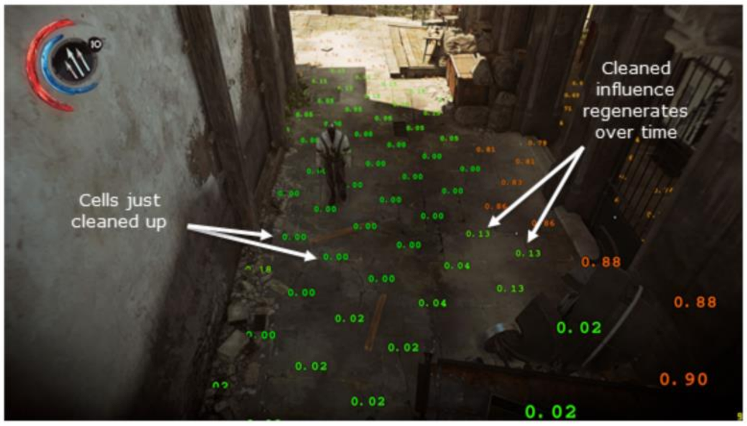
When search is done, all search influence values are reset to invalid.

Searching – Influence Cleaning

NPCs clean cells up from their influence so they becomes less desirable.

Cleaning ➔ Influence = 0

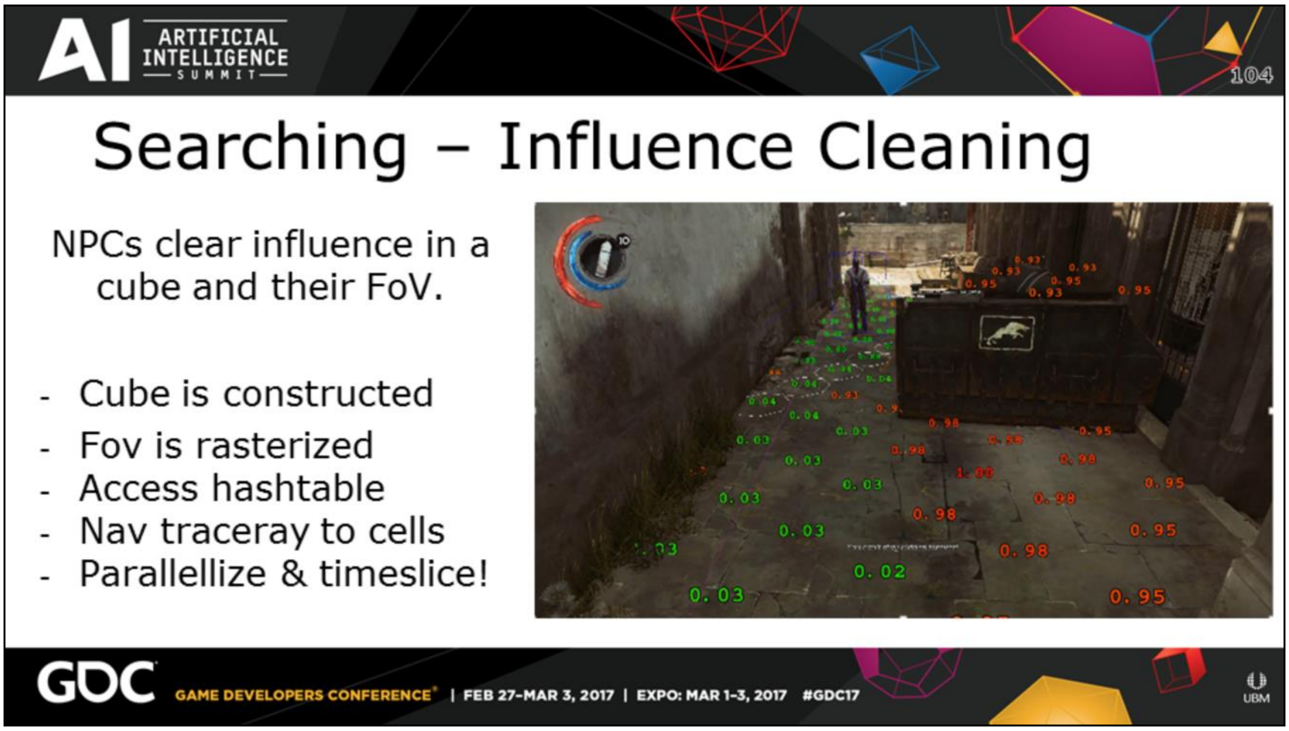
Influence regenerates over time, up to a low threshold



NPCs “clean” influence from cell once inspected so that it becomes less desirable.

Cleaning means setting the cell’s influence to 0.

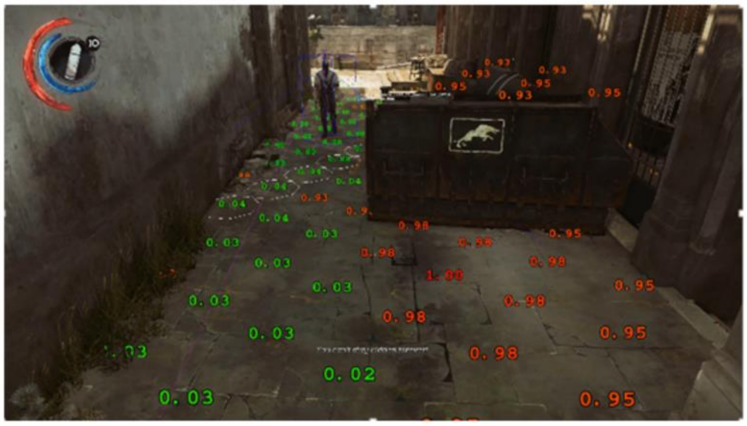
However it regenerates over time up to a very low value: this is to avoid the search being ever finished before the end of the set timer.


ARTIFICIAL INTELLIGENCE SUMMIT
104
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Searching – Influence Cleaning

NPCs clear influence in a cube and their FoV.

- Cube is constructed
- Fov is rasterized
- Access hashtable
- Nav traceray to cells
- Parallellize & timeslice!



<originally a video>

NPCs clean influence in two ways:

1. In a “cube” of cells we construct around the NPC location
2. Within their inner field of view: we rasterize the triangle on the DSM grid.

We use the access table to retrieve the cells in each case. To be sure the NPC actually “sees” the cell, we check there is a straight line on navmesh from NPC to the cell center.

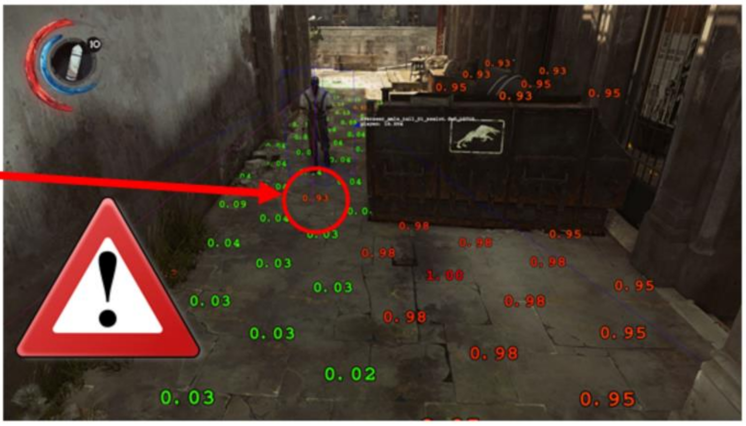
This is expensive! So we made sure to multithread all this properly, and used time slicing as we don’t need instant results in that case

To conclude on search, there is something I’d like to confess…

Searching – Bug !!!

This cell is not cleaned up properly !!!

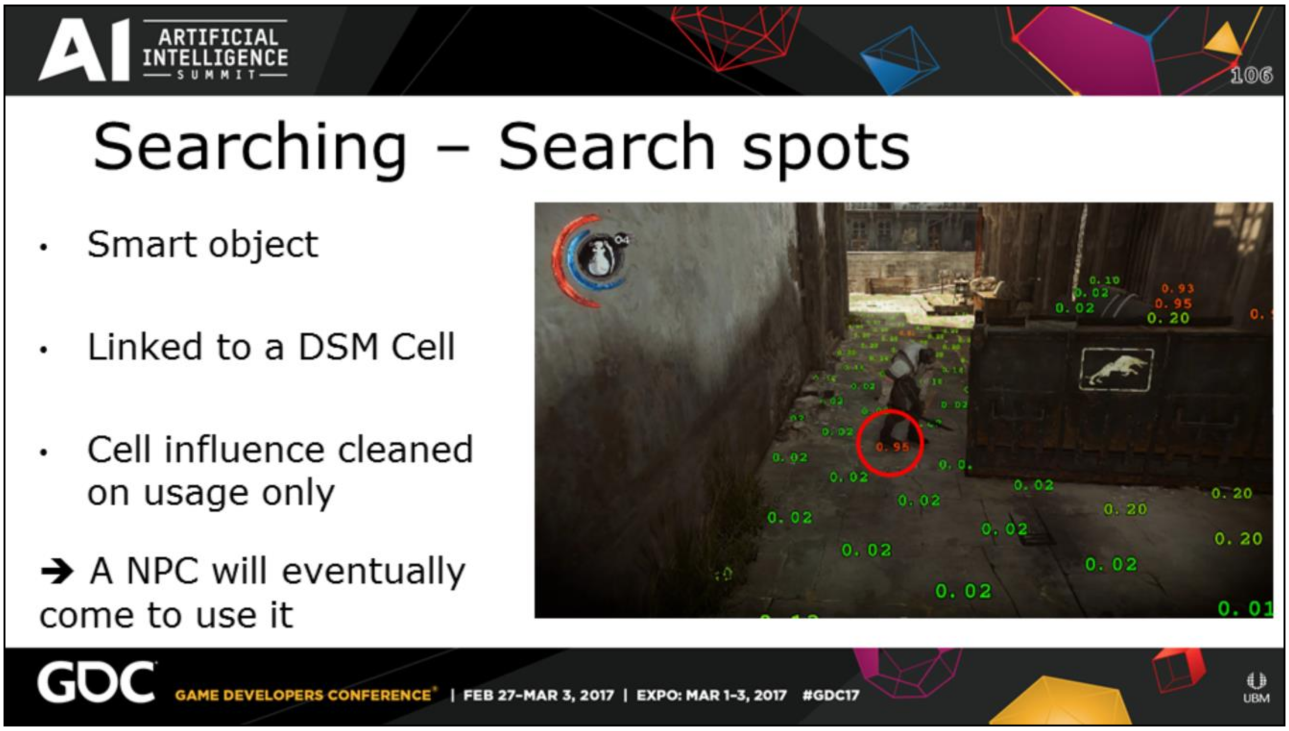
Just kidding ;)



Some among you probably noticed this bug in the previous video.

The influence is not cleaned properly on this cell. That’s clearly a bug right?

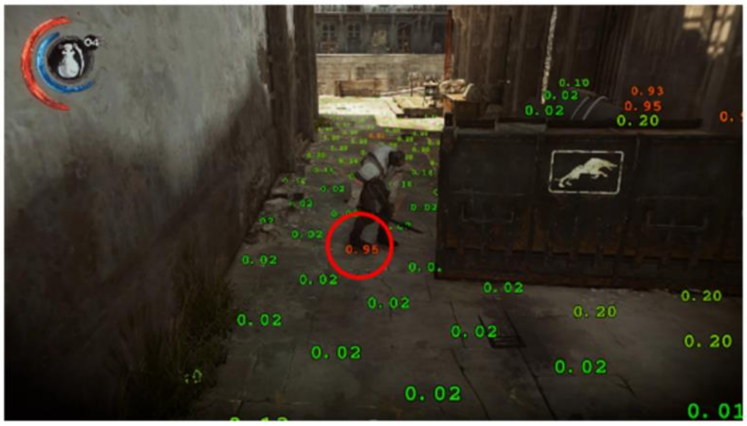
No obviously I’m just kidding ☺



Searching – Search spots

- Smart object
- Linked to a DSM Cell
- Cell influence cleaned on usage only

➔ A NPC will eventually come to use it

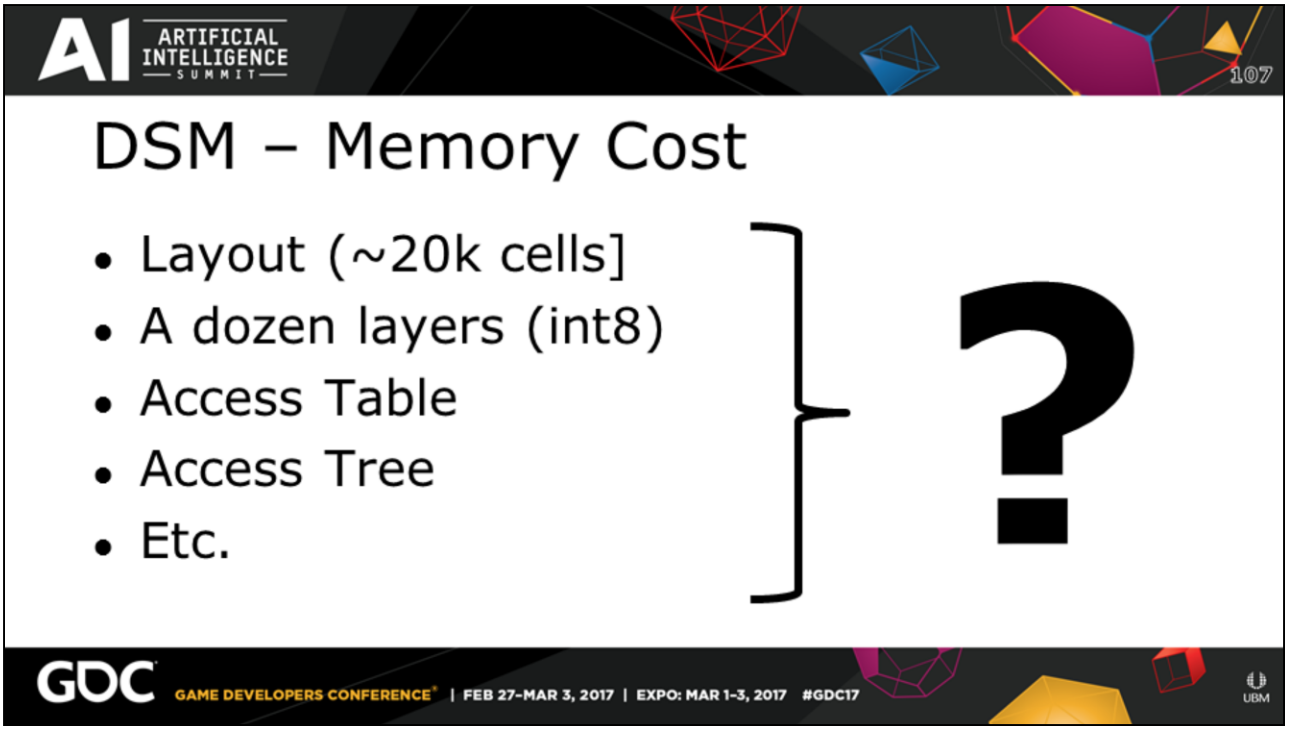


<originally a video>

In fact we have smart objects affecting search. We call them search spots and each one is linked to a cell.

On those we forbid search influence cleaning until the smart object usage actually takes place. That way a NPC will eventually go the smart object to use it.

Now we're really done with the search part.



DSM – Memory Cost

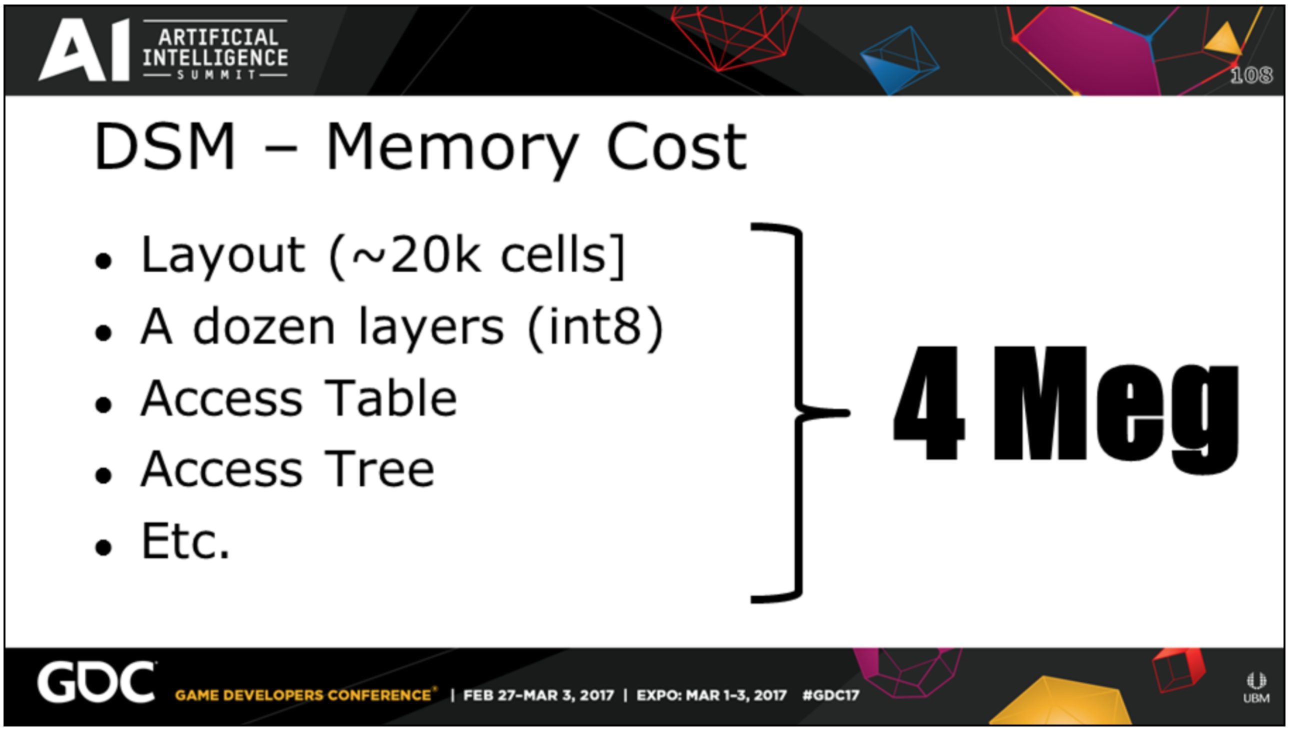
- Layout (~20k cells]
- A dozen layers (int8)
- Access Table
- Access Tree
- Etc.

?
107
ARTIFICIAL INTELLIGENCE SUMMIT
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Now let’s talk about cold hard facts.

During runtime, counting the layout, our dozen layers, access table, access tree, etc. What would the memory cost for the whole DSM system be?

<Survey: less than 1 meg, more than 5 meg?, “All you guys that did not raise your hand, you win.”>



DSM – Memory Cost

- Layout (~20k cells]
- A dozen layers (int8)
- Access Table
- Access Tree
- Etc.

4 Meg

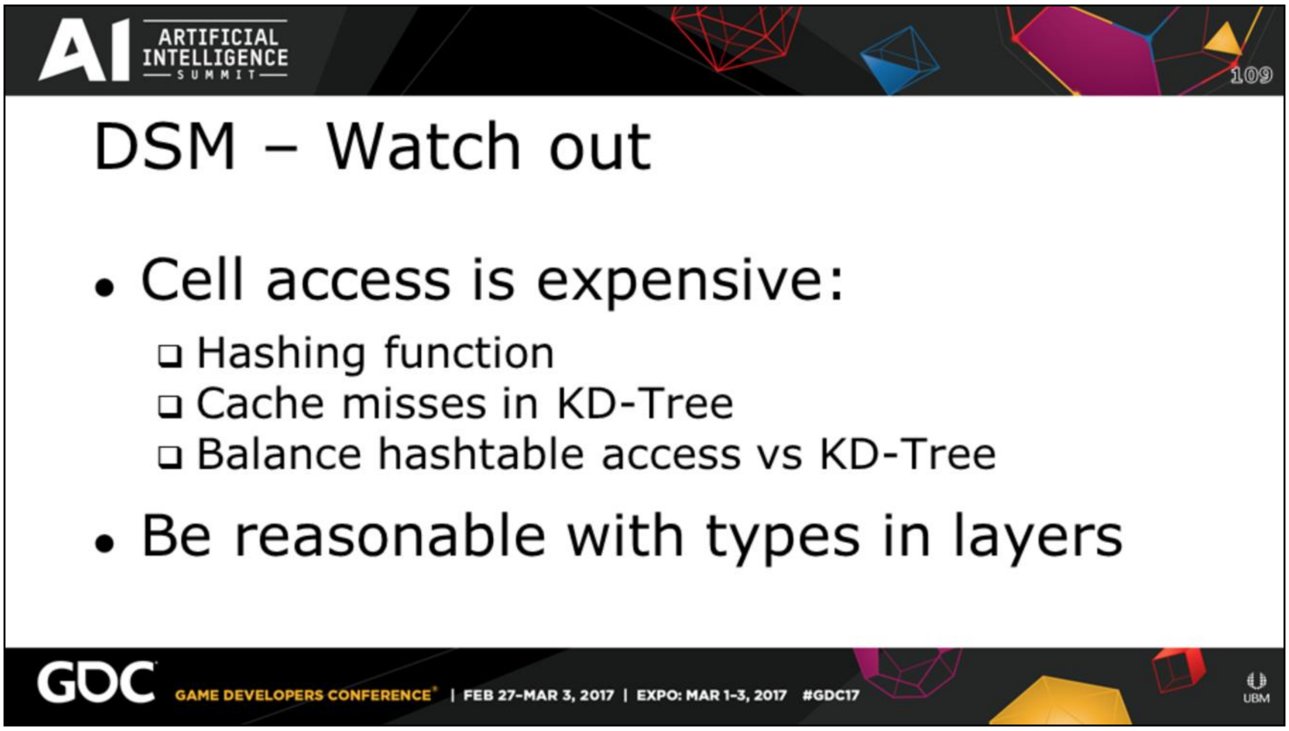
DSM memory consumption was profiled around 4 megabytes

Some years ago, there was an interview from Naughty Dog folks, stating that they were allowed 3 megs on Uncharted 2* ... for the whole game state.

But with current gen console, guess what… 4 meg is perfectly reasonable.

---

[*] Climbing and Sneaking Behind UNCHARTED 2: AMONG THIEVES's AI
https://aigamedev.com/premium/interview/uncharted2-among-thieves/



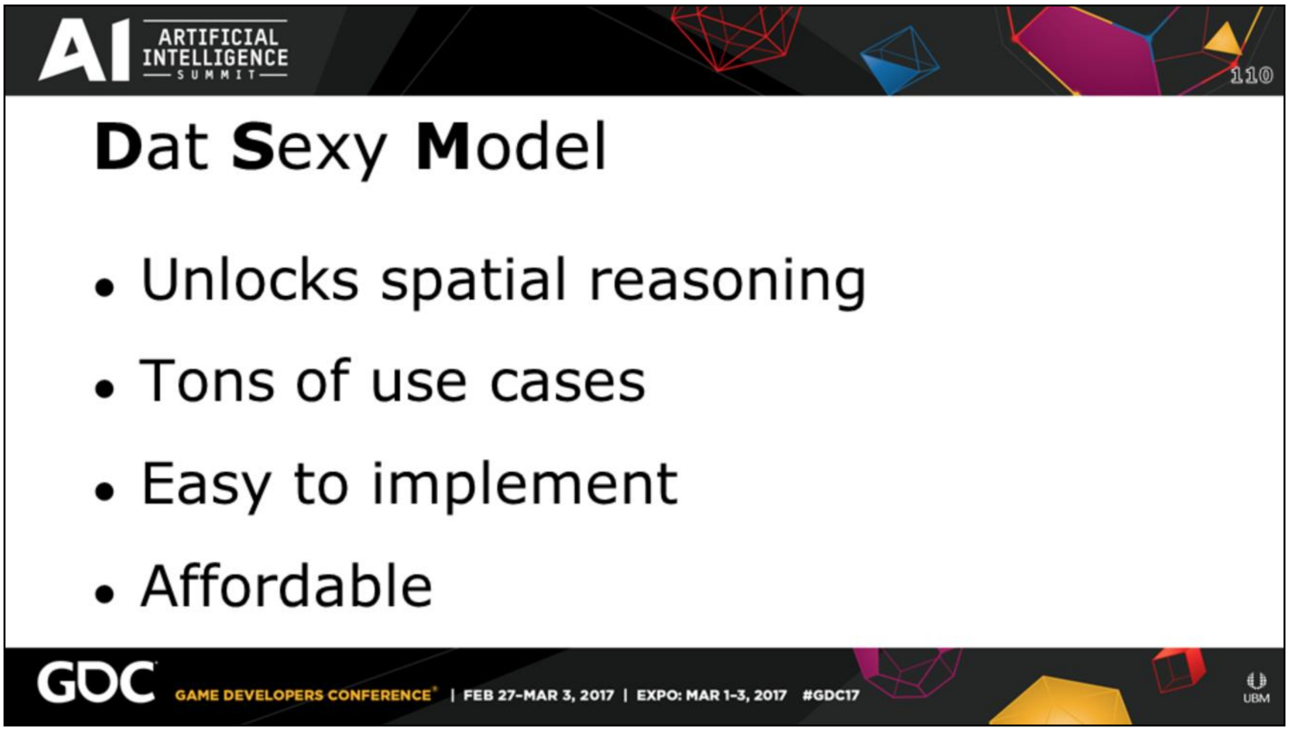
DSM – Watch out

- Cell access is expensive:
  - ❑ Hashing function
  - ❑ Cache misses in KD-Tree
  - ❑ Balance hashtable access vs KD-Tree
- Be reasonable with types in layers

Before concluding, let’s talk about some things you want to watch our for when using DSM.

First thing, the big performance consumer for us has been cell accesses. This means you need to watch out for the performance of your hashing function, make sure your kd-tree implementation generates as few cache misses as possible, and really think about the proper way to collect cells for each use case.

Second thing, we can’t stress enough that you don’t want to store a 4 bytes float in a layer when all you really need is one byte. I guess Dave would agree <gesture towards M. Mark>.


Dat Sexy Model

- Unlocks spatial reasoning
- Tons of use cases
- Easy to implement
- Affordable

ARTIFICIAL INTELLIGENCE SUMMIT
110
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

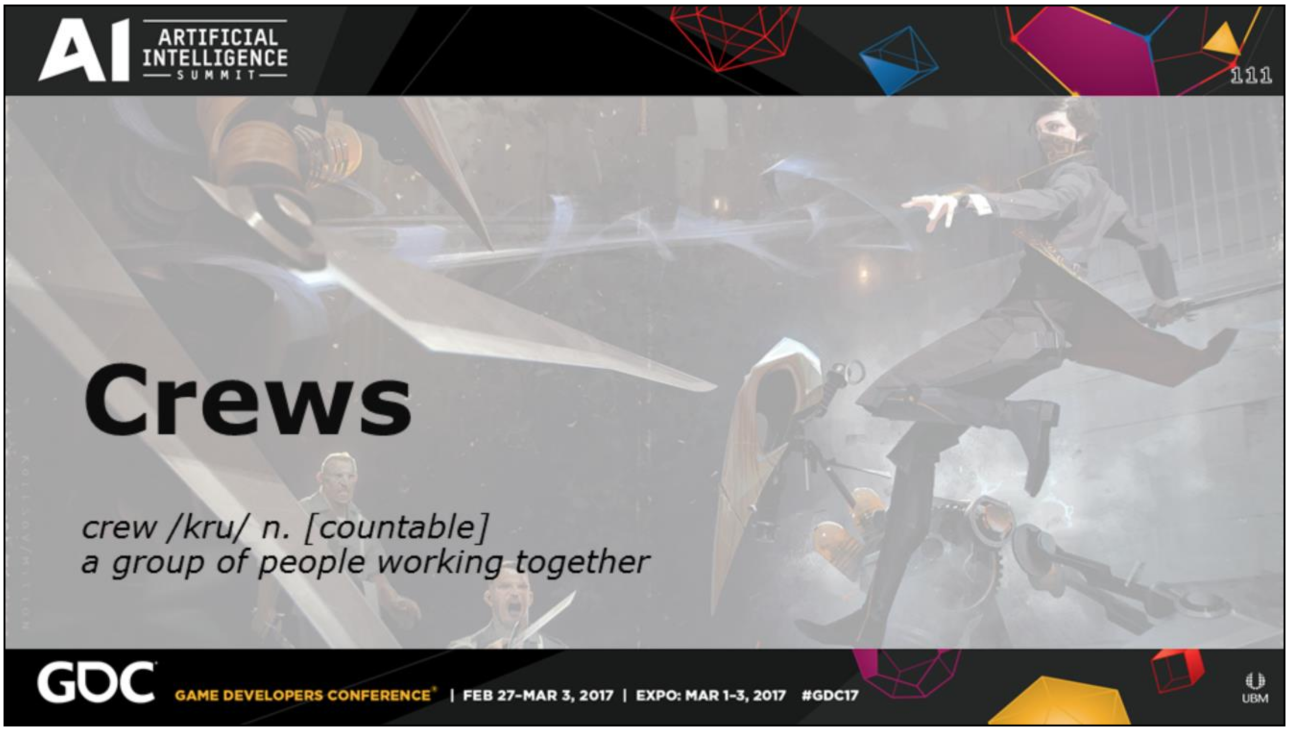
To finish on this part, let me tell you why DSM is sexy.

For us, the system has been a breakthrough as far as spatial reasoning is concerned, it unlocked selecting positions dynamically when all we had was pre-placed points before (which we had in Dishonored 1 for flee destinations or blood vines summon positions)

The second thing is that there are a lot of use cases for this system, far more than what we originally envisioned.

Finally, try it, it's rather simple to implement, and provided you're reasonable, it's affordable.

Now let's talk about another affordable system, coordinated by Laurent.



ARTIFICIAL INTELLIGENCE SUMMIT

111

Crews

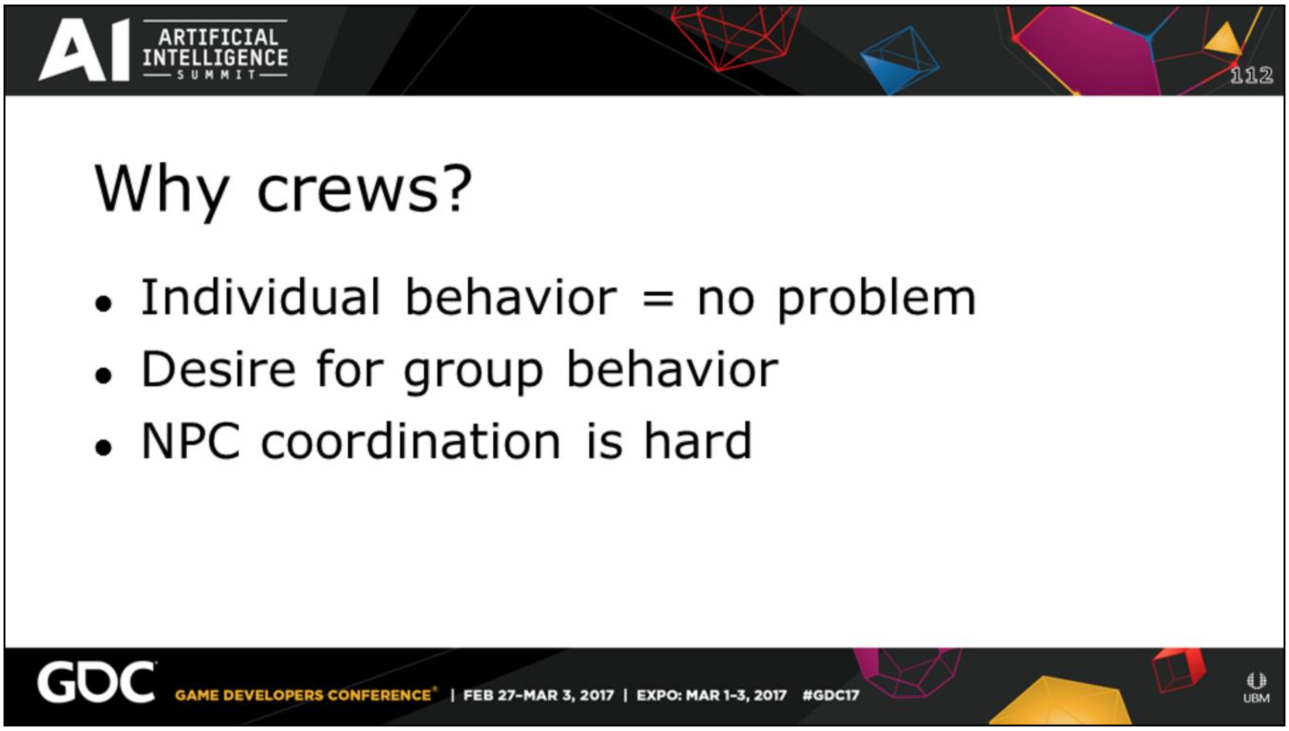
crew /kru/ n. [countable]
a group of people working together

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

Now let's talk about crews.

It's our way of handling NPC coordination.


Why crews?

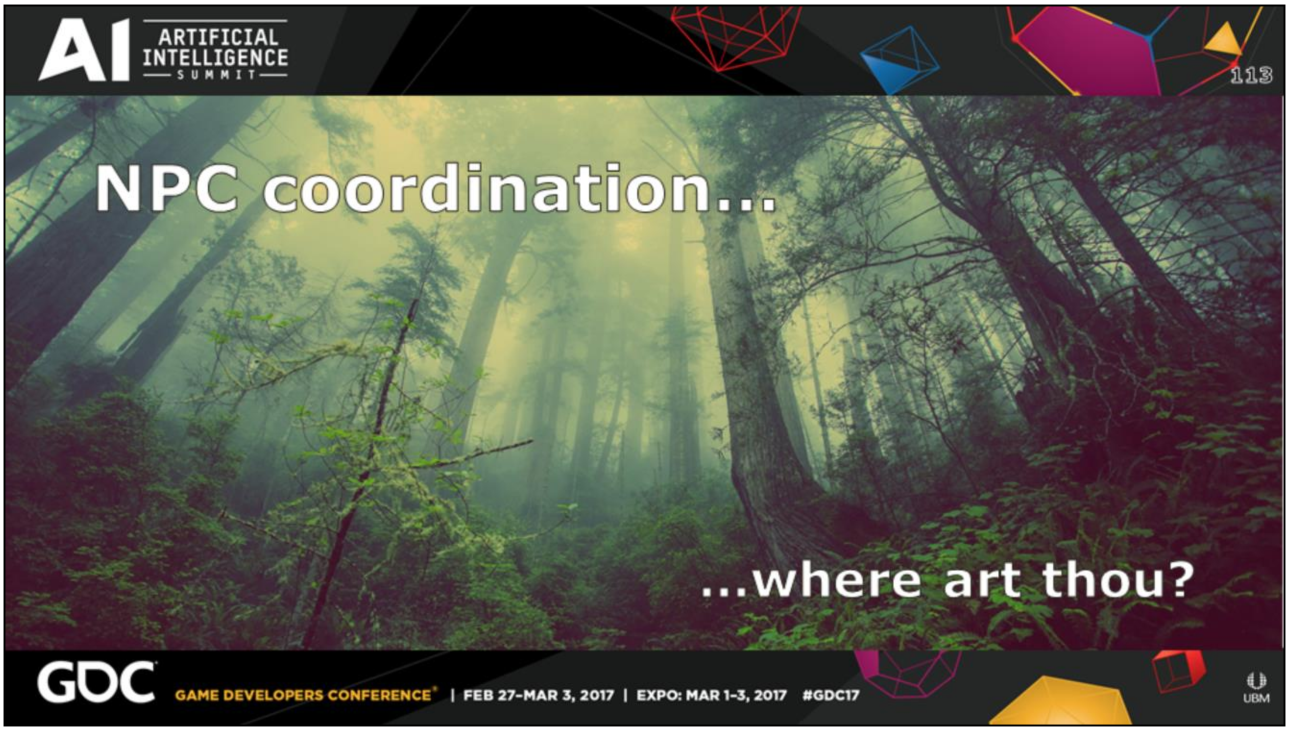
- Individual behavior = no problem
- Desire for group behavior
- NPC coordination is hard

So where does this come from?

During pre-production we knew that were going to have standard tech fitting for *individual* NPC behavior.

But our systems designers also had a strong desire to push the envelope and implement some ambitious *group* behaviors. Designers, right? ☺

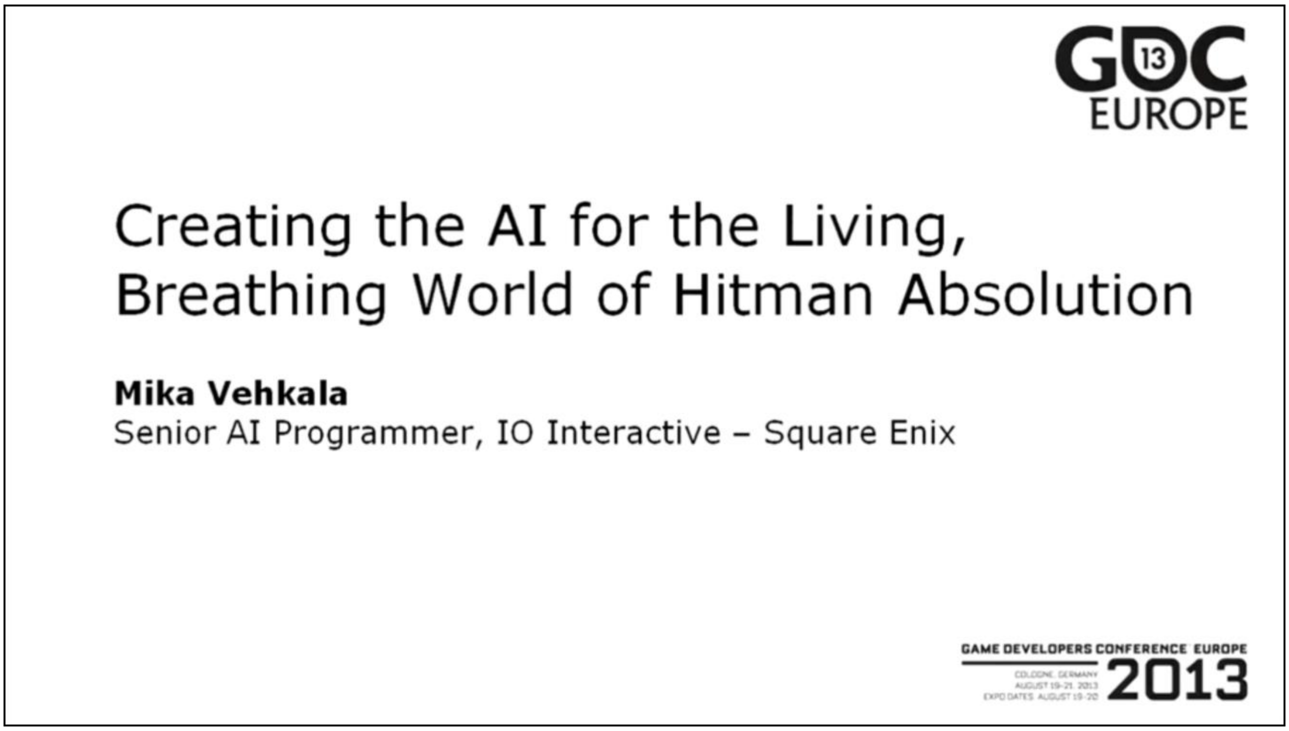
Well we knew from previous experiences that this one desire was going to be hard to satisfy. You can get some basic things working with inter-NPC communication, but it’s hard to get a fully coordinated group behavior. There’s a point where you need another approach to avoid getting trapped in a monstrous plate of spaghetti code.



NPC coordination...

...where art thou?

And even if there's already lots of literature about NPC coordination, we didn't find the perfect match for our desires so we had to venture into uncharted territory.


GDC 13 EUROPE

Creating the AI for the Living, Breathing World of Hitman Absolution

Mika Vehkala
Senior AI Programmer, IO Interactive – Square Enix

GAME DEVELOPERS CONFERENCE EUROPE
COLOGNE, GERMANY
AUGUST 19-21, 2013
EXPO DATES: AUGUST 19-20
2013

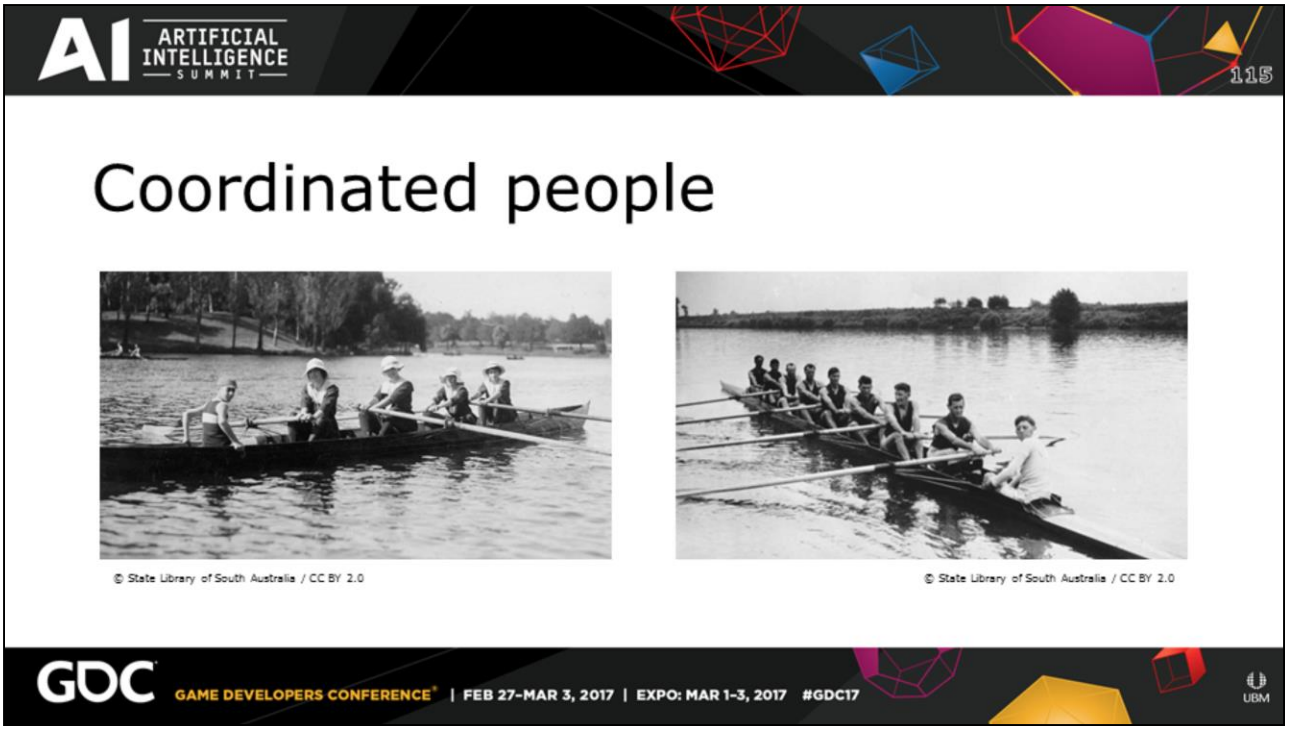
But once more, the light came from a GDC talk, this time from Mika Vehkala*.

Among other things, he introduced the concept of "situations", a technique used in Hitman Absolution. So again, go check it if you have any interest in this topic.

Thank you Mika! Our AI crews are largely inspired by your ideas. But again, we feel that we did some interesting things on our own.

---

[*] Creating the AI for the Living, Breathing World of Hitman: Absolution
http://www.gdcvault.com/play/1019353/Creating-the-AI-for-the



Coordinated people

© State Library of South Australia / CC BY 2.0

© State Library of South Australia / CC BY 2.0

So what's a crew? It's a group of people working together towards a common objective.

Take this crew, or that crew. Quite literally, they're in the same boat. Clearly, they will have to coordinate if they want to go anywhere. They need to row together.

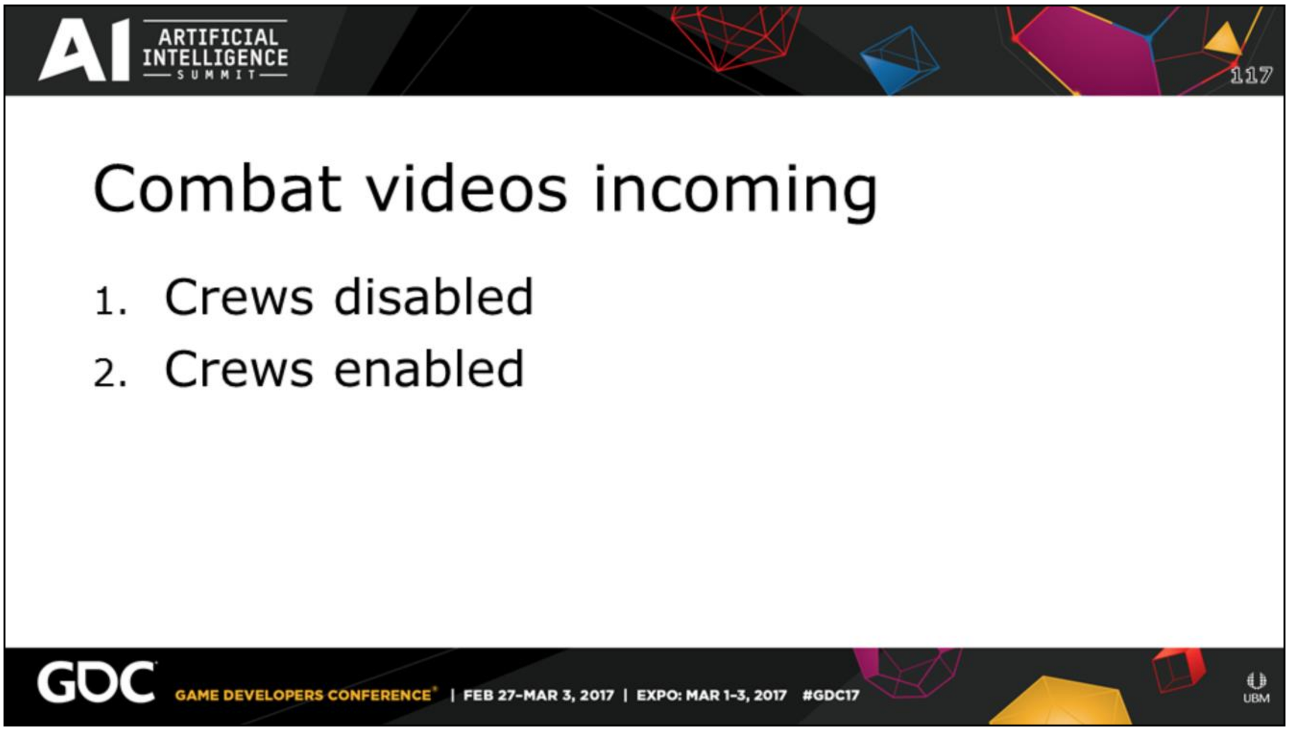
Photos © State Library of South Australia / CC BY 2.0
https://www.flickr.com/photos/state_library_south_australia/14595109718
https://www.flickr.com/photos/state_library_south_australia/4539666042

Not-so-coordinated people

Although they can row together, these people are still individuals.

Free them up, let the evening come, add some music and beverages… And soon you end up with this absolute mess.

Agents making their own decisions, sometimes interacting, sometimes not.


ARTIFICIAL INTELLIGENCE SUMMIT

117

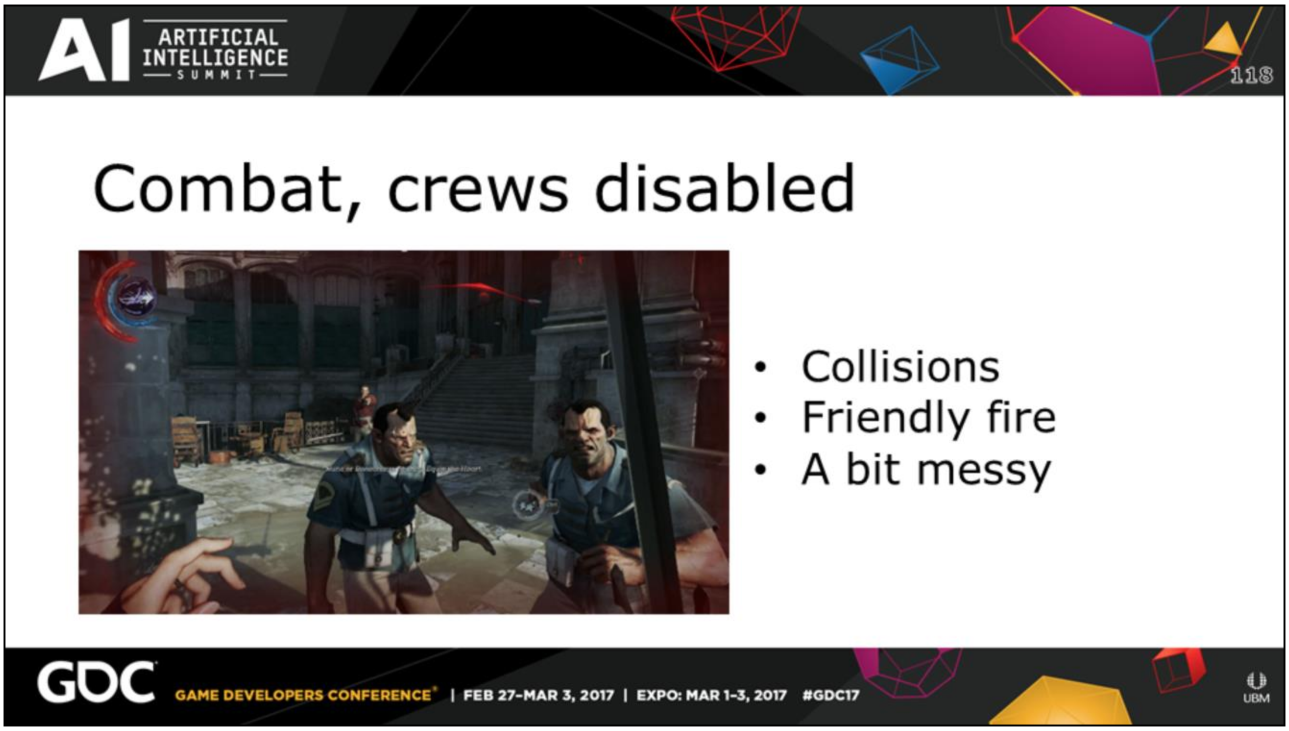
Combat videos incoming

1. Crews disabled
2. Crews enabled

GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

In practice, what does this mean?

Let me illustrate this with two videos of what we call the “regular combat situation”. Don’t be afraid: that’s our jargon for a bunch of guards directly fighting the player.


ARTIFICIAL INTELLIGENCE SUMMIT

118

Combat, crews disabled

- Collisions
- Friendly fire
- A bit messy

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

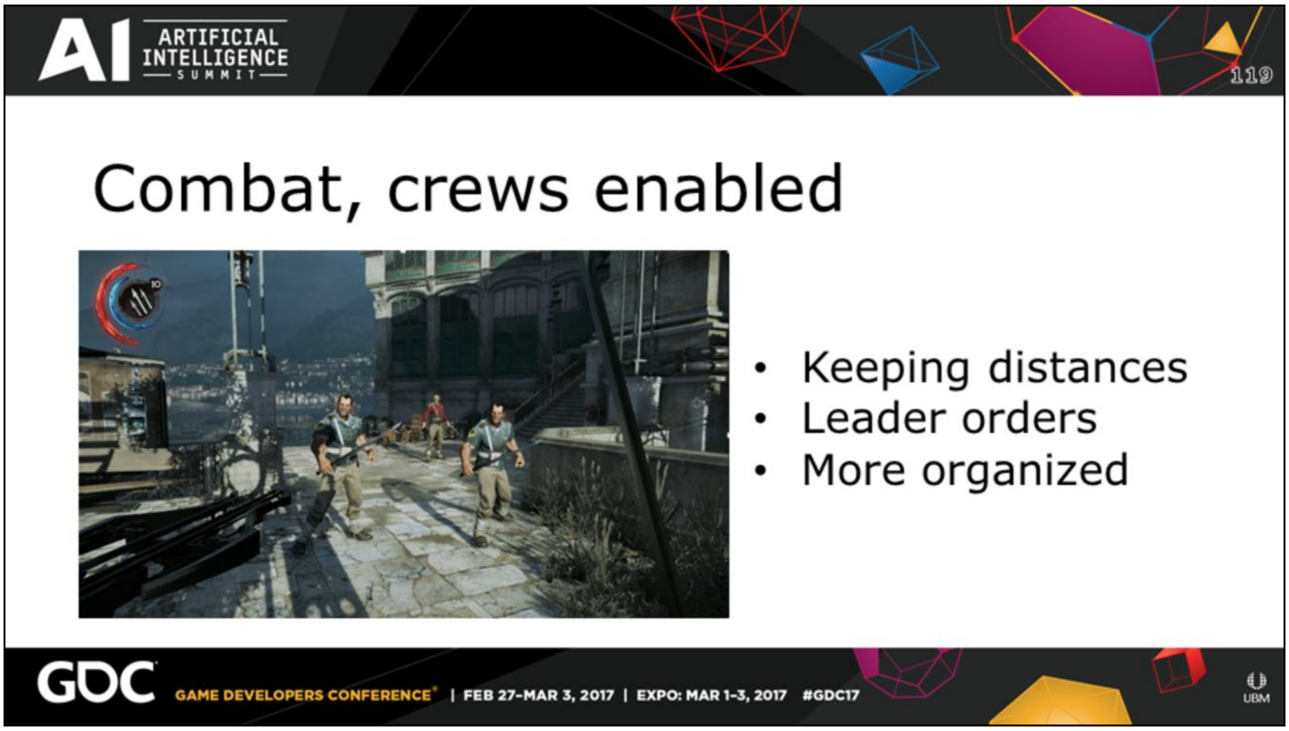
UBM

<originally a video>

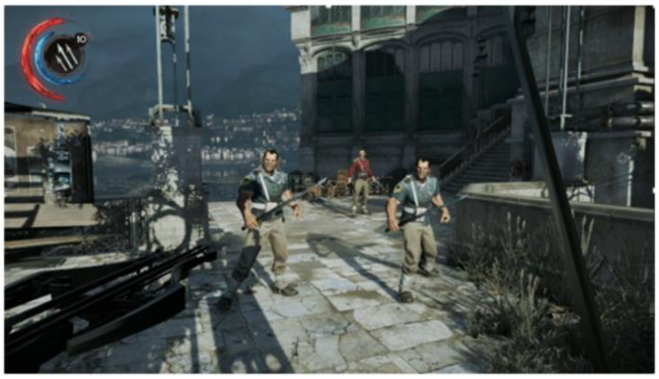
So here you can see that the NPCs are fighting, but things are a bit disorganized.

They're struggling... They all try to reach the player at the same time... Sometimes bumping into each other... There's even friendly fire going on.

Not very pretty.



Combat, crews enabled



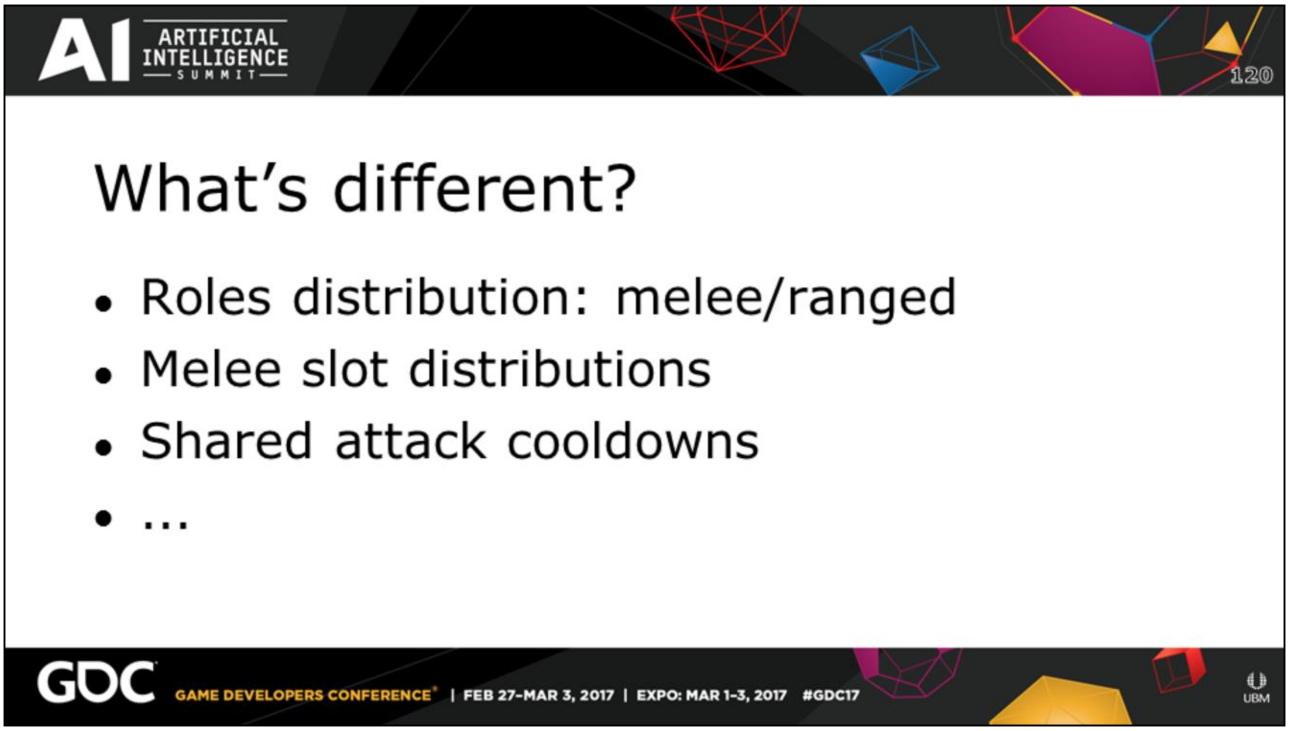
- Keeping distances
- Leader orders
- More organized

<originally a video>

So this time the crew is enabled.

Guards are keeping their distances a bit more... The elite guard in the background is shouting orders... He use his pistol a little more effectively...

Things look a little more organized.


ARTIFICIAL INTELLIGENCE SUMMIT

120

What's different?

- Roles distribution: melee/ranged
- Melee slot distributions
- Shared attack cooldowns
- ...

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

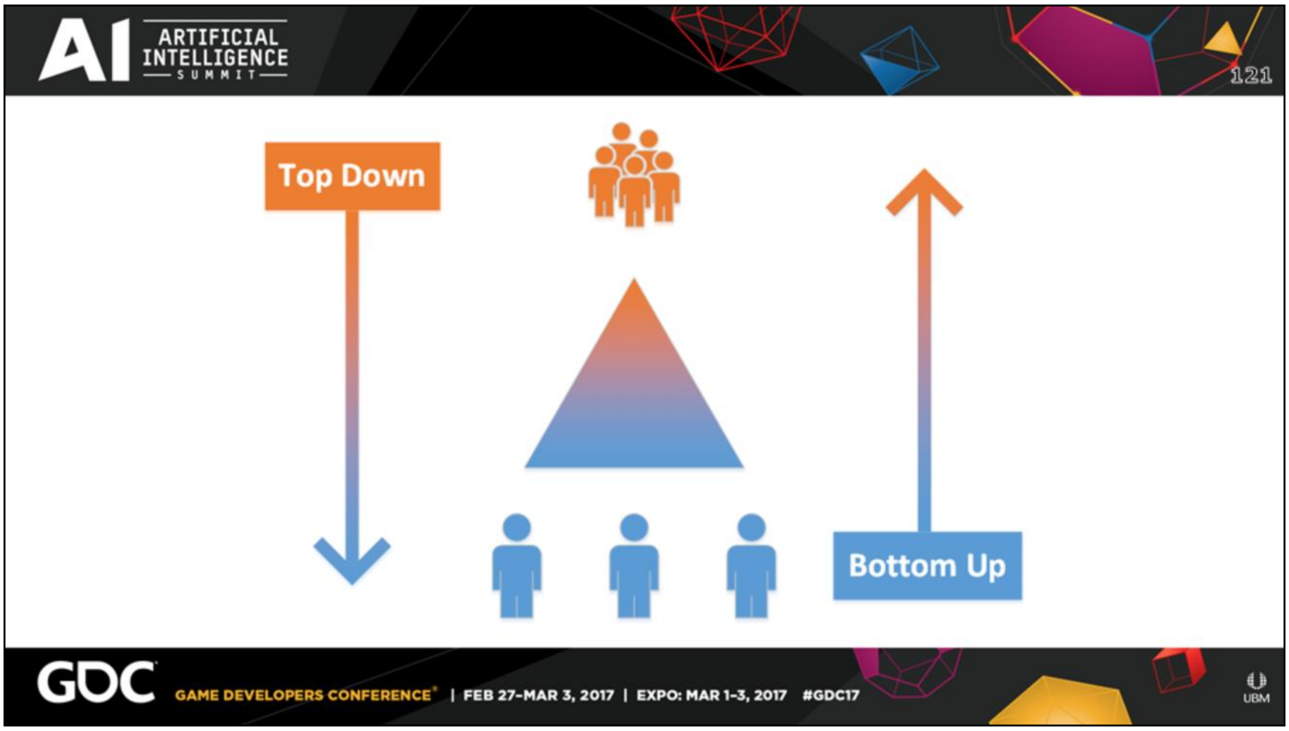
What makes things different in the second video? There's a combat crew working behind the scene to coordinate NPCs.

It assigns them a certain role in combat, for instance melee fighter or ranged fighter.

For NPCs in melee, we have a bunch of slots that poll the navmesh around the player. It's the crews' function to find a proper distribution across these slots.

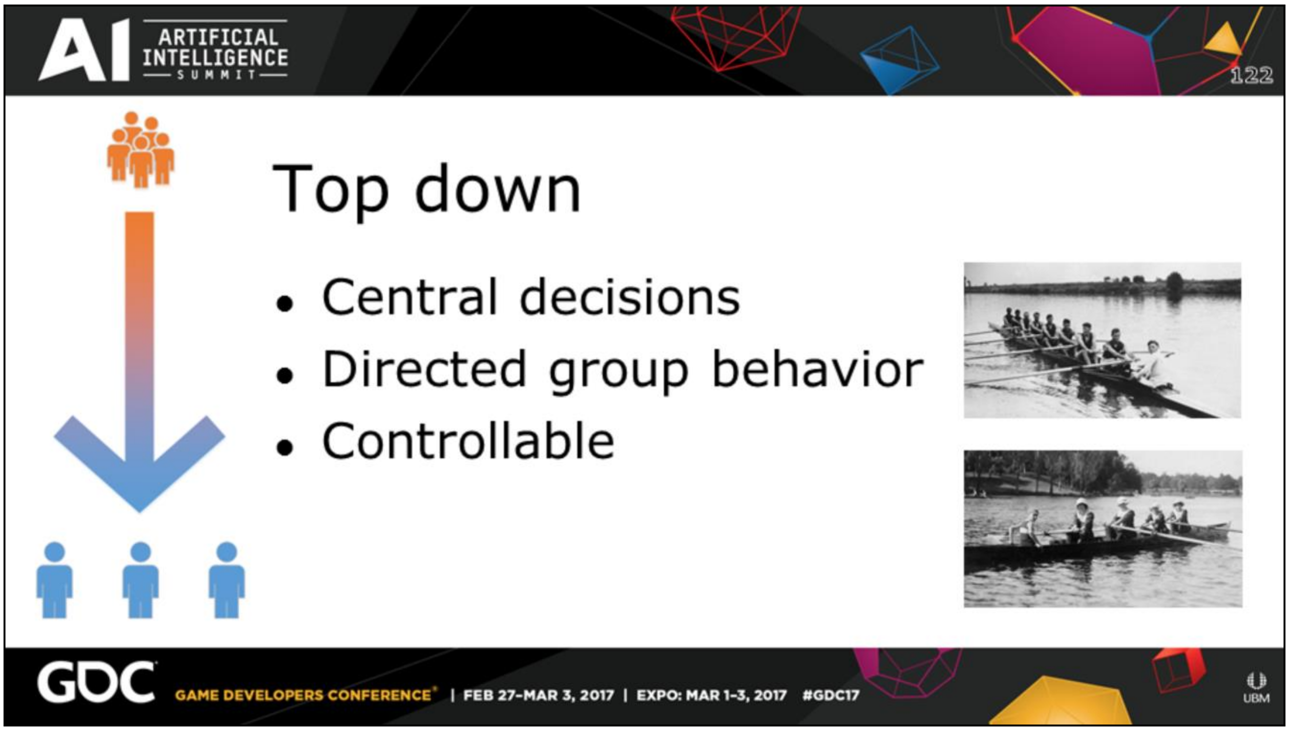
The crew also controls the attack rate through shared attack cooldowns, as we don't want the player to be completely overwhelmed.

And so on.


ARTIFICIAL INTELLIGENCE SUMMIT
121
Top Down
Bottom Up
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Let's step back a bit. I might be stating the obvious here but I want to drive this home.

Two opposite paradigms exist, two approaches to group behavior: top down or bottom-up.

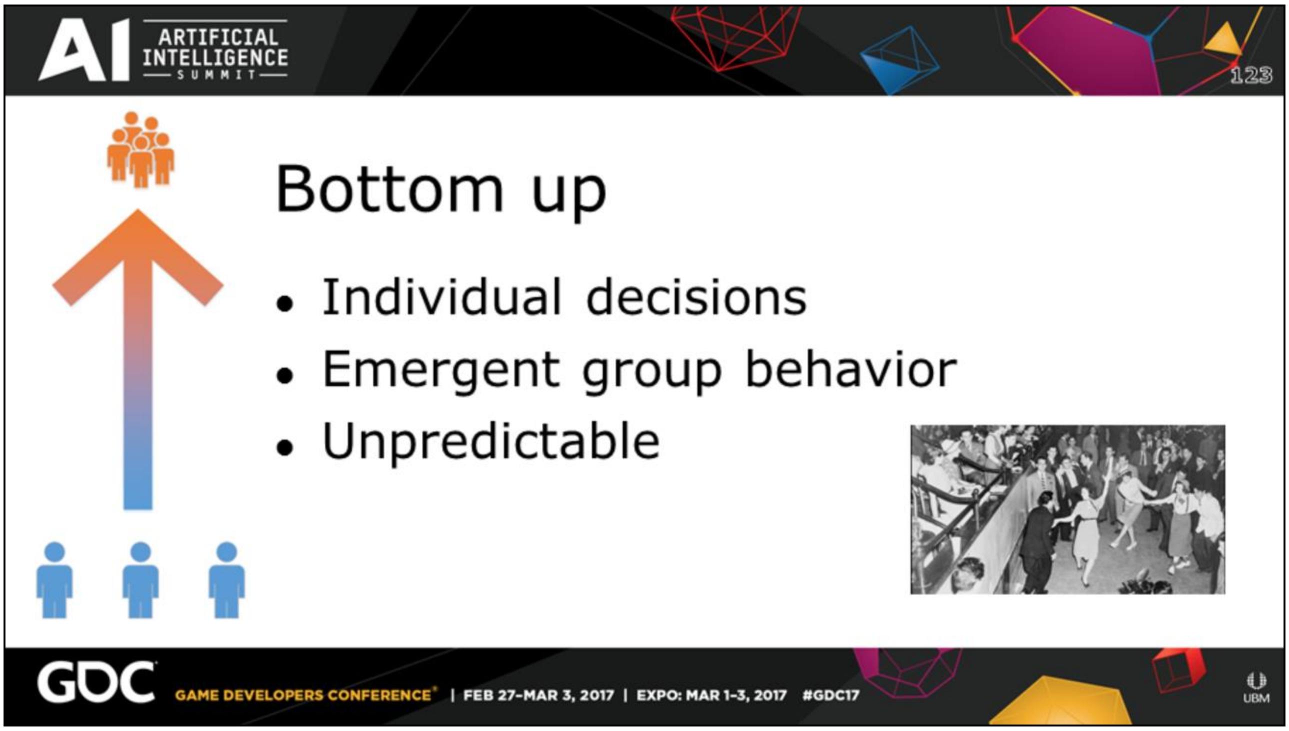

ARTIFICIAL INTELLIGENCE SUMMIT
122
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Top down

- Central decisions
- Directed group behavior
- Controllable

The top down approach is to have a central coordinating entity that takes decisions and gives directions. A ruling mastermind.

This allows for pre-determined group behavior.



ARTIFICIAL INTELLIGENCE SUMMIT

123

Bottom up

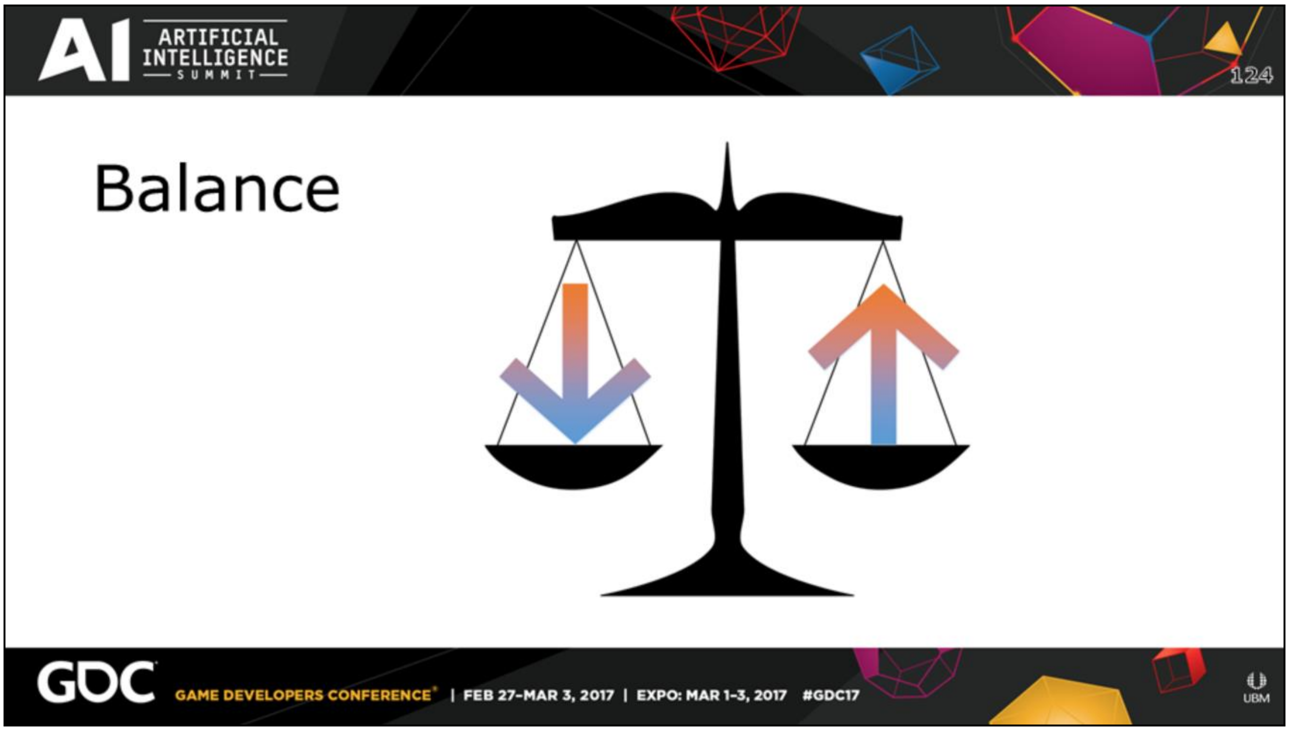
- Individual decisions
- Emergent group behavior
- Unpredictable

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

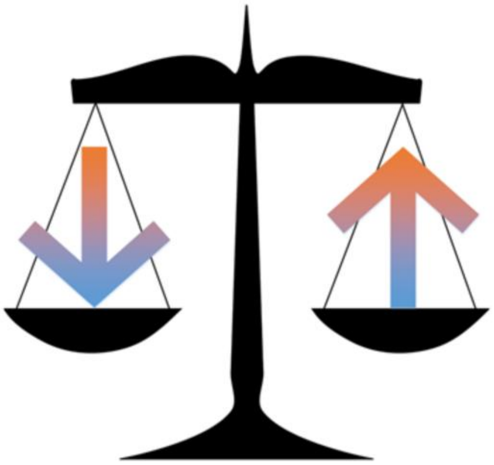
UBM

The bottom up approach lets all individuals take their decisions, and the occasional interactions create an emergent group behavior.

The outcome is unpredictable.



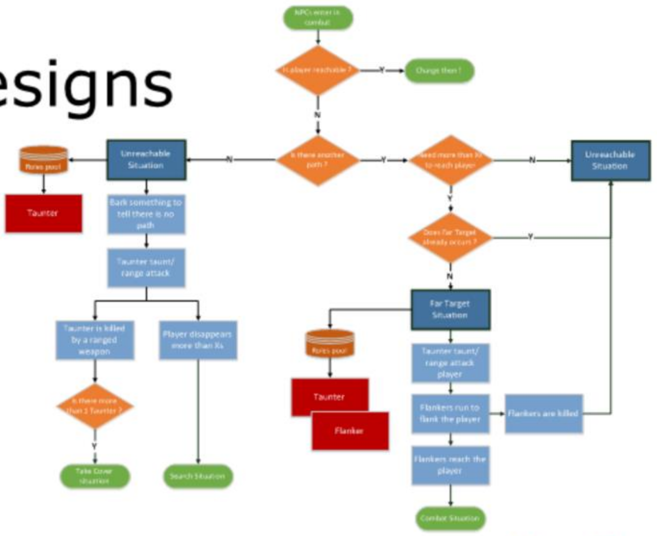
Balance



I won’t tell you that one approach is better than the other. There are pros and cons on both sides.

So my point is: you have to find a balance between the two that fits the game you’re making.

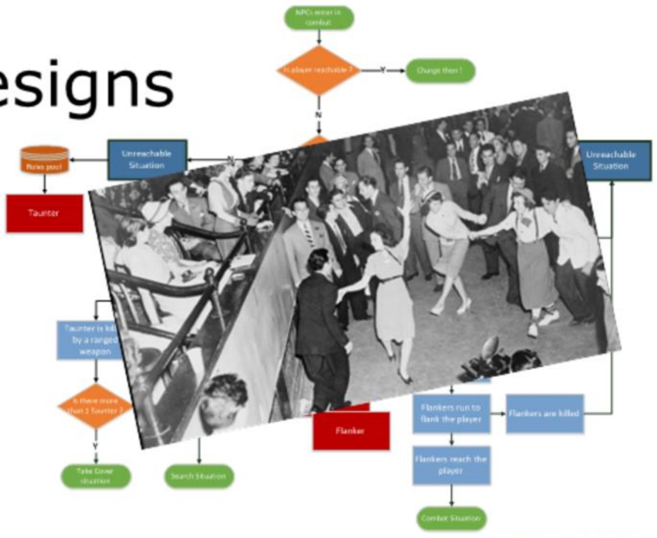
Early designs


NPCs enter in combat
Is player reachable ?
Y
Charge then !
N
Is there another path ?
N
Unreachable Situation
Roles pool
Taunter
Bark something to tell there is no path
Taunter taunt/ range attack
Taunter is killed by a ranged weapon
Player disappears more than Xs
Is there more than 1 Taunter ?
Y
Take Cover situation
Search Situation
Y
Need more than Xs to reach player
N
Unreachable Situation
Y
Does Far Target already occurs ?
Y
N
Far Target Situation
Roles pool
Taunter
Flanker
Taunter taunt/ range attack player
Flankers run to flank the player
Flankers are killed
Flankers reach the player
Combat Situation

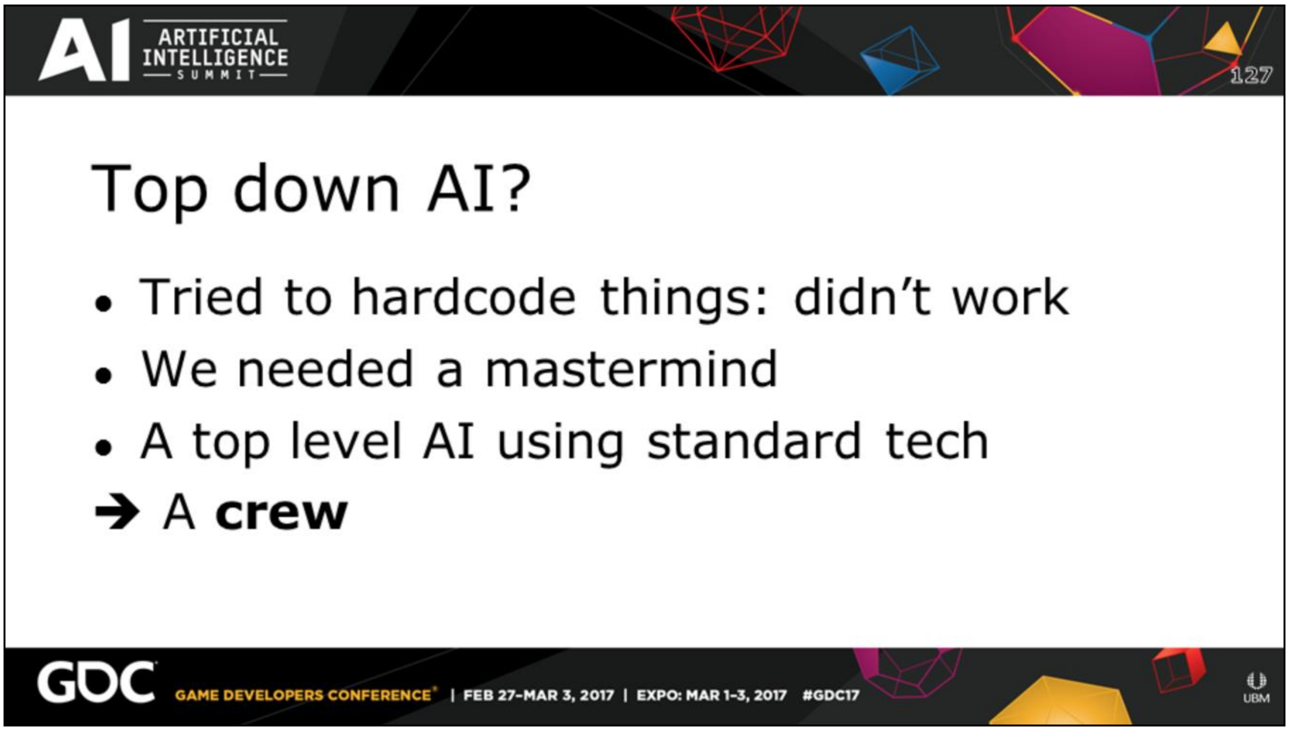
So what about us, and Dishonored 2? In our case, we had early designs such as this one.

This is the kind of group behavior our designers envisioned. This one’s from our combat designer Jonathan. Don’t worry about the details: just note that this represents evolving situations, with NPCs dynamically taking different roles depending on the environment and player actions. My point here is that it’s very difficult to implement something along those lines…

Early designs



… When all you have is this mess. That is just impossible.


Top down AI?

- Tried to hardcode things: didn't work
- We needed a mastermind
- A top level AI using standard tech

➔ A **crew**

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

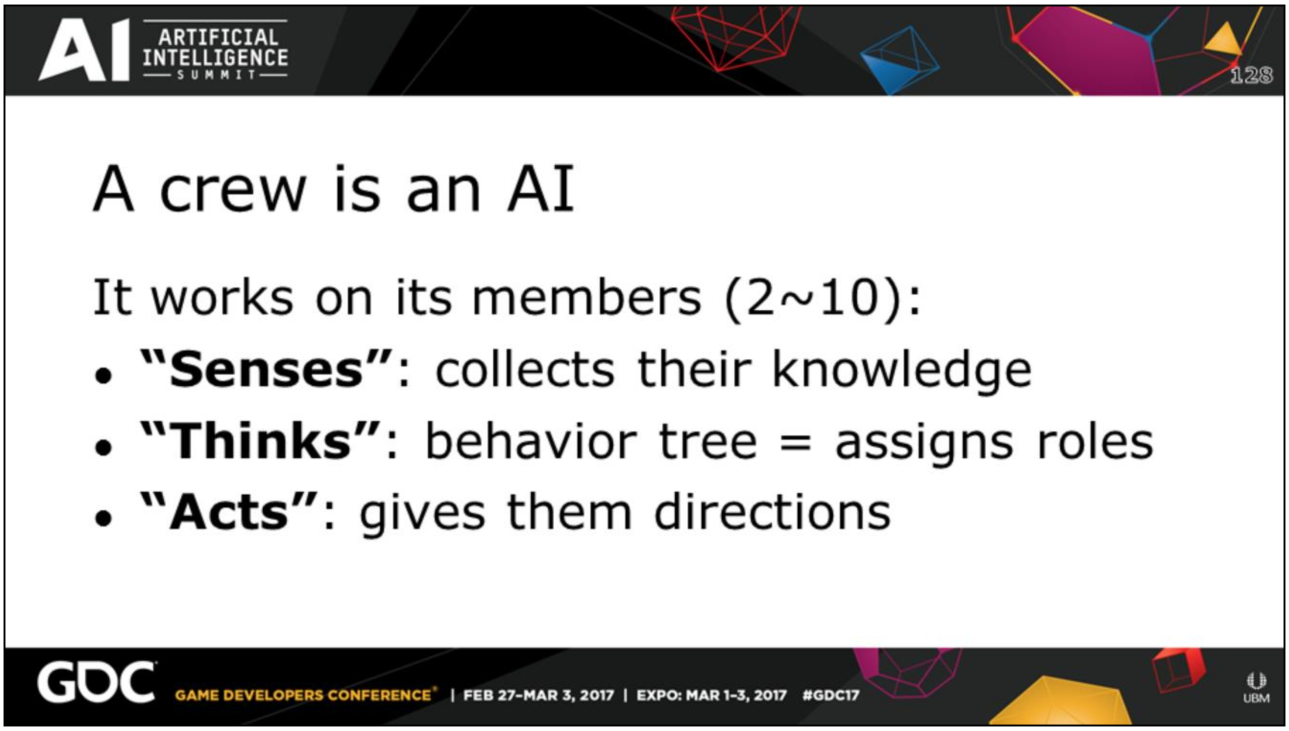
So we needed a top down approach to AI.

We first tried a prototype of a situation system, inspired by the Hitman talk. But things were very hardcoded, and too rigid.

Since what we needed was a mastermind taking its own decisions, it became obvious that what we needed was an actual AI.

Instead of reinventing the wheel, we figured we could simply reuse our standard AI tech. Our mastermind was going to have sensors, a knowledge blackboard, and a behavior tree.

So this is what we call a crew.



A crew is an AI

It works on its members (2~10):

- **"Senses"**: collects their knowledge
- **"Thinks"**: behavior tree = assigns roles
- **"Acts"**: gives them directions

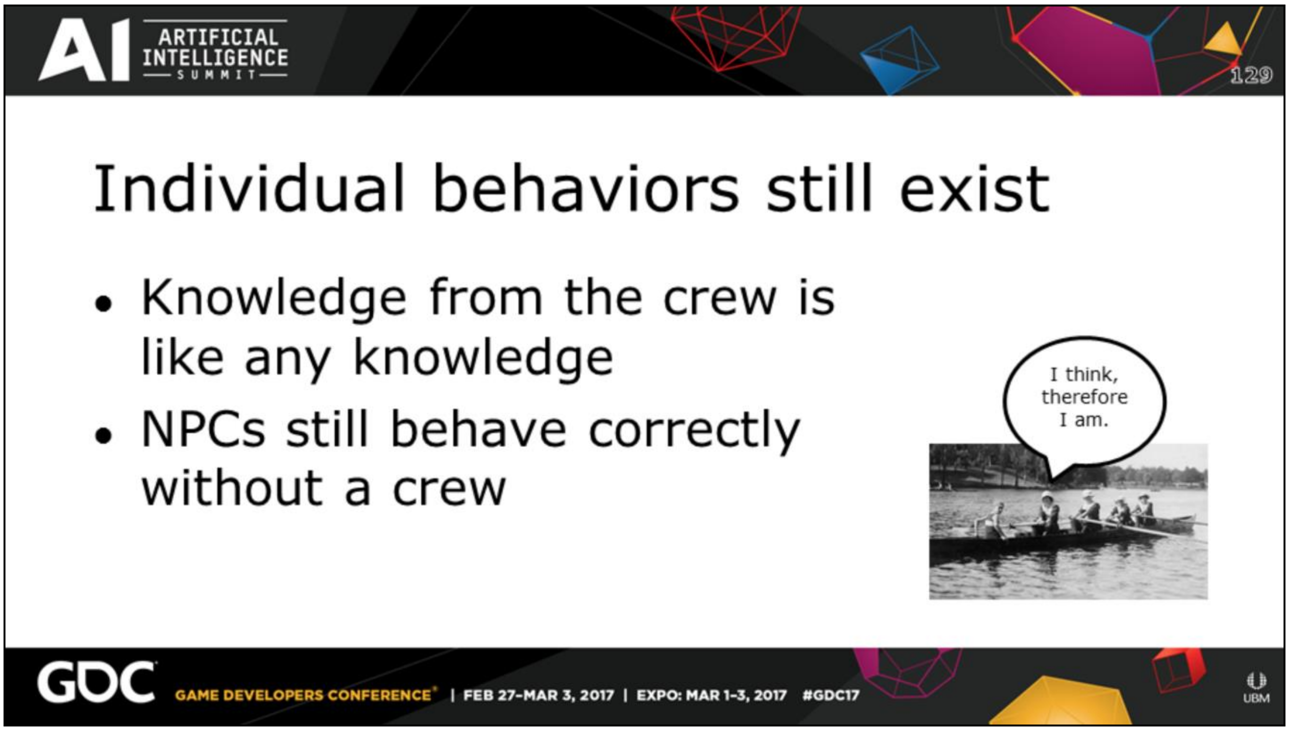
What's the job of a crew?

It works like any AI, but it's main interest is the members that compose it. Just to give you an idea: typically, we're talking about two to ten members.

So when we update a crew, it first "senses", which means that is starts by collecting the knowledge of each member.

It then "thinks", which means updating a behavior tree that decides which role to assign to which member.

Finally, it "acts". This mostly consists in writing directions in the NPCs individual knowledge, so they play their role correctly when it's their turn to update.



ARTIFICIAL INTELLIGENCE SUMMIT

Individual behaviors still exist

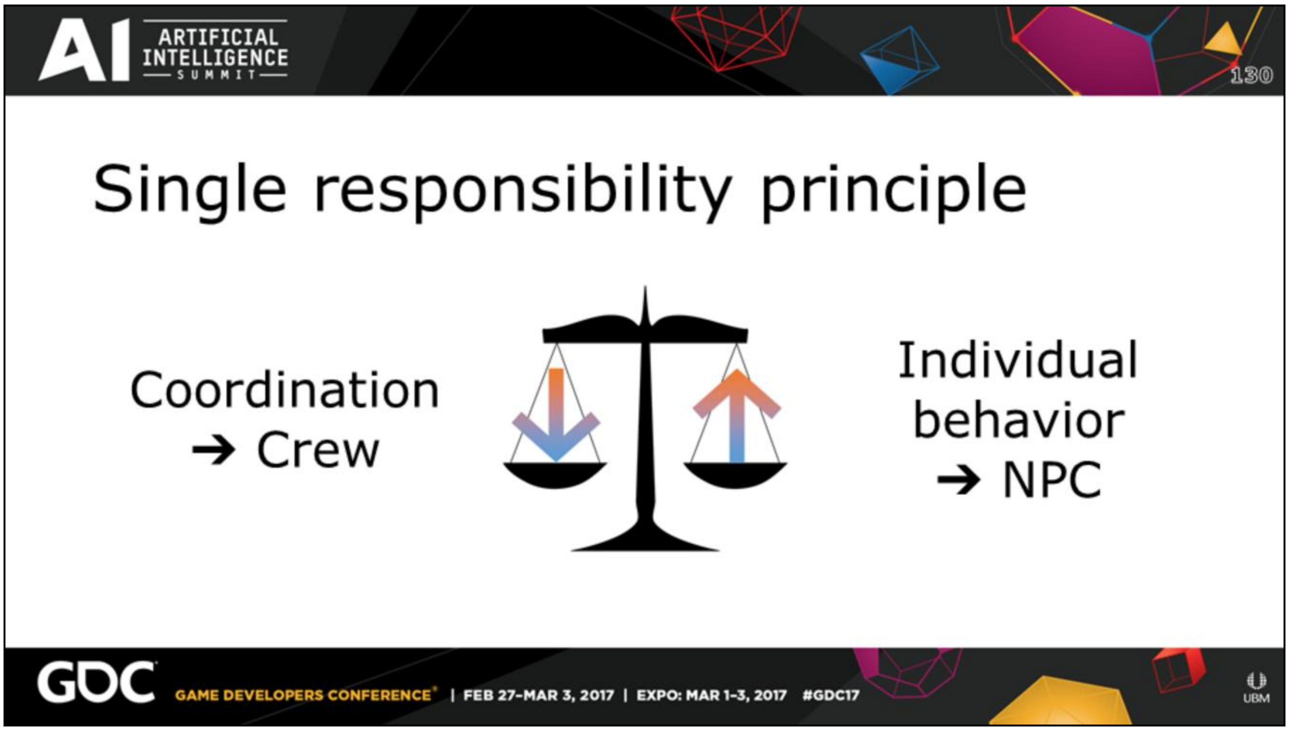
- Knowledge from the crew is like any knowledge
- NPCs still behave correctly without a crew

GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

That’s one important point: individual behaviors still exist.

NPCs use the knowledge coming from the crew as any other source of knowledge (for example their sensors).

So if you disable the crews, NPCs just act as if they were alone, but they’re not completely shut down.



Single responsibility principle

Coordination ➔ Crew

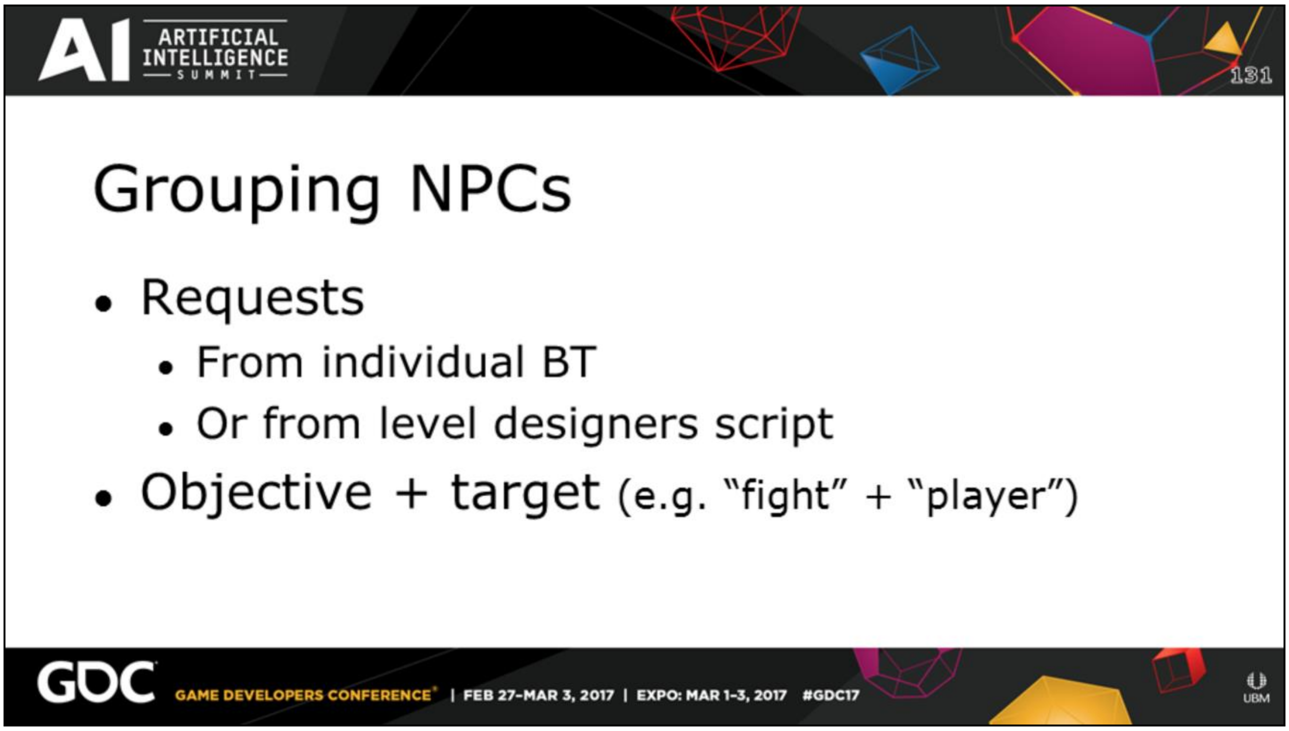
Individual behavior ➔ NPC

So this is what I meant by finding a balance. If something good came out of the early 2000s craze of object-oriented programming, it’s the single responsibility principle.

On one side of the balance, crews should be responsible for the coordination only. As soon as the coordination logic starts to drive NPCs directly, things starts to smell.

On the other side of the balance, the individual behavior logic should always work on a single NPC. As soon as it tries to perform coordination, it becomes a mess.

Our rule of thumb is that we try to avoid using any crew logic when a behavior involves one and only one NPC.



Grouping NPCs

- Requests
  - From individual BT
  - Or from level designers script
- Objective + target (e.g. "fight" + "player")

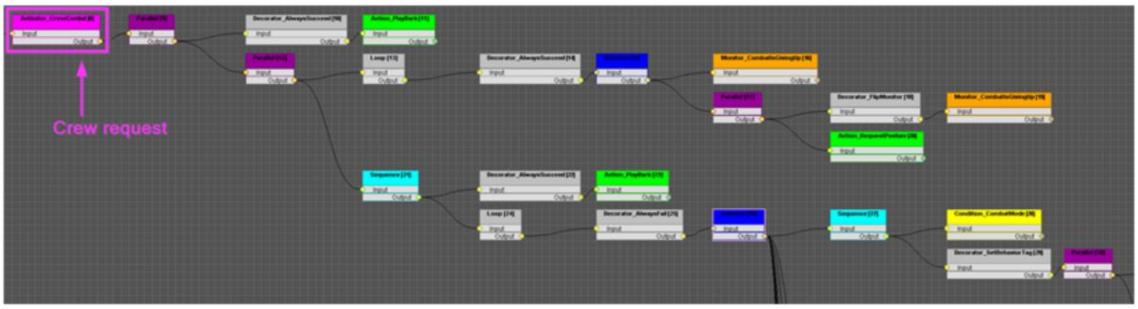
ARTIFICIAL INTELLIGENCE SUMMIT
131
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Now you might wonder: how do we group NPCs together?

We use a very basic request system. NPCs can emit a request to join a crew directly from their behavior tree. We also have crews controlled by the level designers, so they can create crew requests from our in-house visual scripting tool.

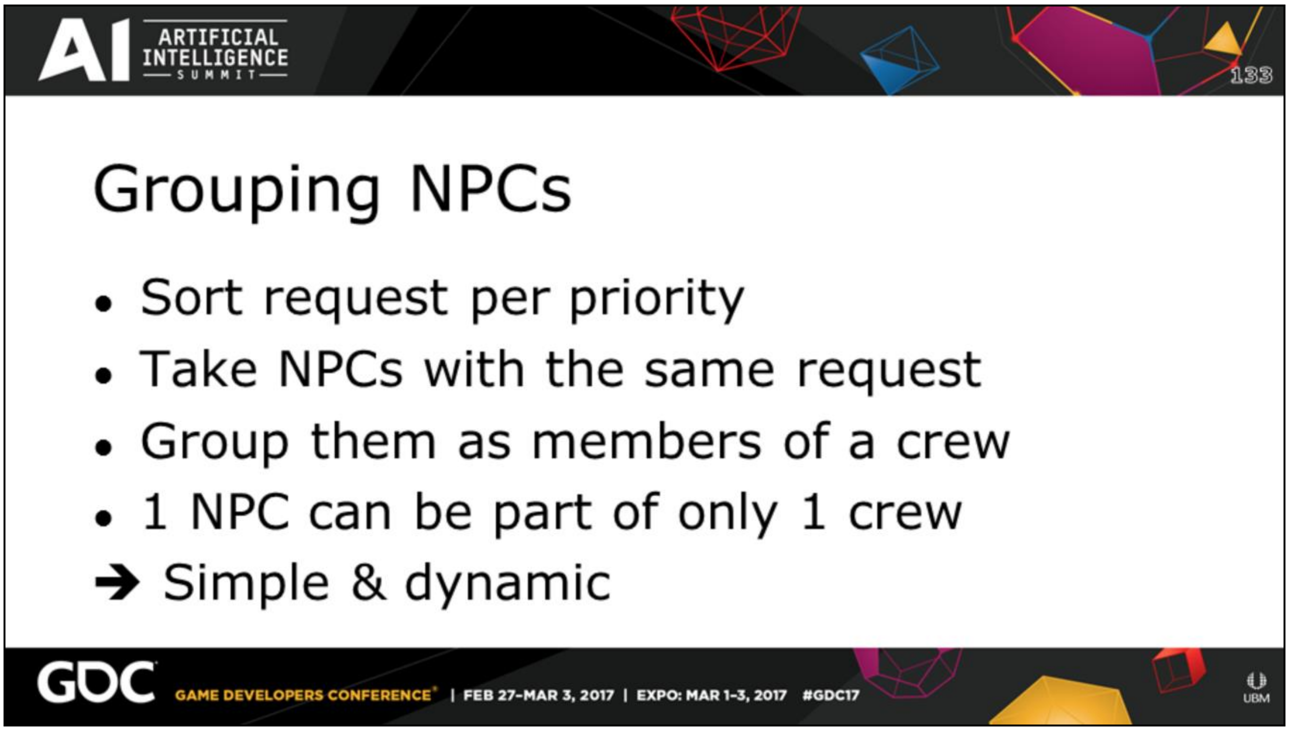
This request is made of an objective and a target. For instance, "fight the player".

Grouping NPCs


Activator_CrewCombat [8]
Parallel [9]
Decorator_AlwaysSucceed [10]
Action_PlayBark [11]
Parallel [12]
Loop [13]
Decorator_AlwaysSucceed [14]
Monitor_CombatIsGoingUp [16]
Parallel [17]
Decorator_FlipMonitor [18]
Monitor_CombatIsGoingUp [19]
Action_RequestPosture [20]
Sequence [21]
Decorator_AlwaysSucceed [22]
Action_PlayBark [23]
Loop [24]
Decorator_AlwaysFail [25]
Sequence [27]
Condition_CombatMode [28]
Decorator_SetBehaviorTag [29]
Parallel [30]
Input
Output
Crew request

So just to show you what it looks like, this is a part of our individual combat behavior tree.

The highlighted node at the tree root is what posts a request to join a combat crew.



Grouping NPCs

- Sort request per priority
- Take NPCs with the same request
- Group them as members of a crew
- 1 NPC can be part of only 1 crew

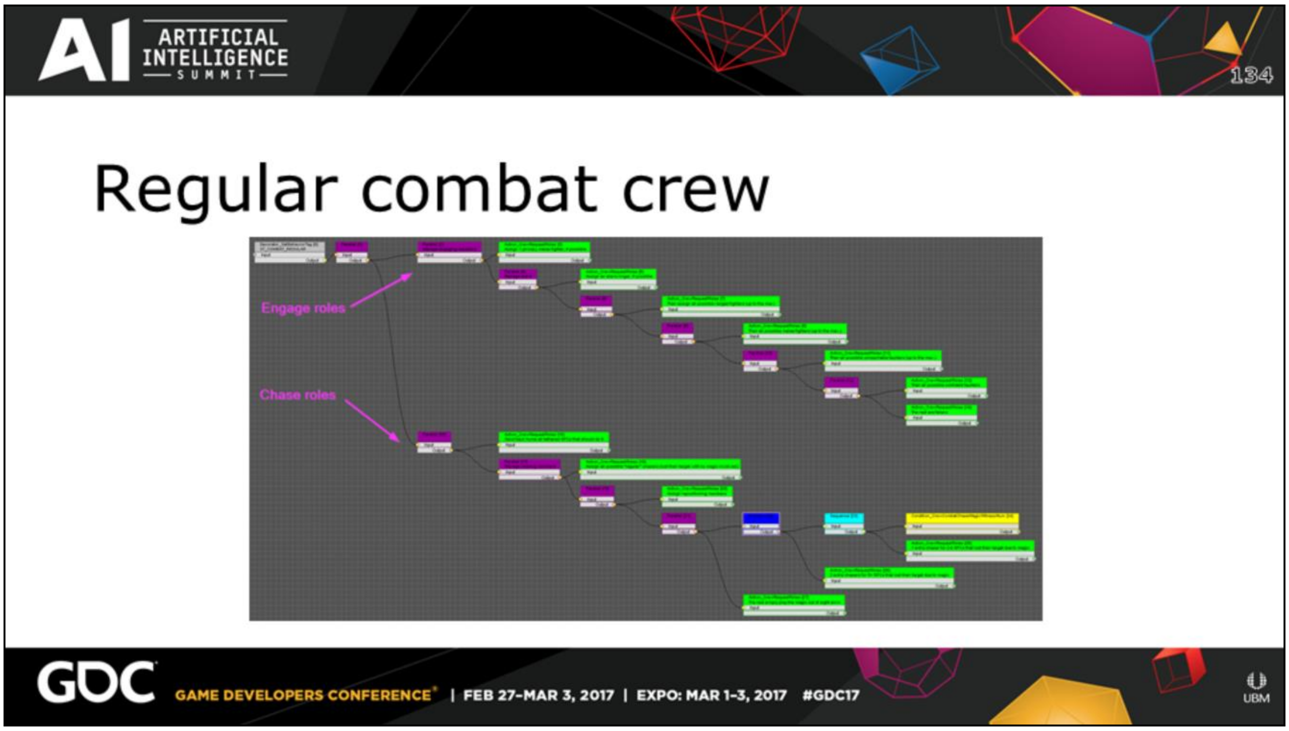
➔ Simple & dynamic

At any given time, we sort requests per priority, and all NPCs sharing the same objective and target simply become members of the same crew.

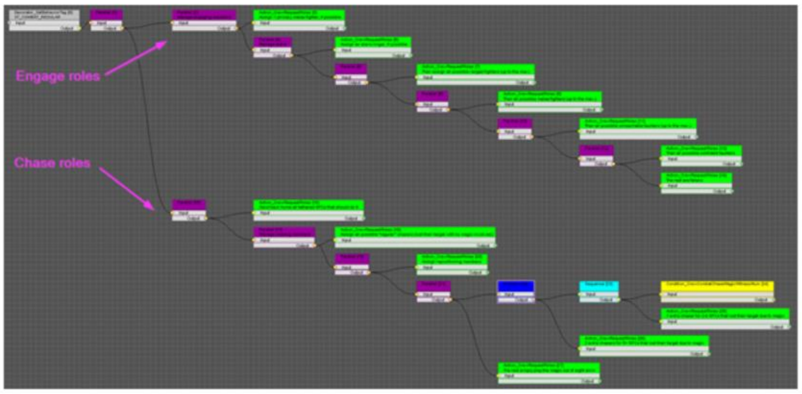
We allow an NPC to be part of only one crew.

This is both simple and dynamic, and avoids to put too much complexity at a very high level.

Now that we grouped NPCs together, let’s see an illustration of group behavior.



Regular combat crew

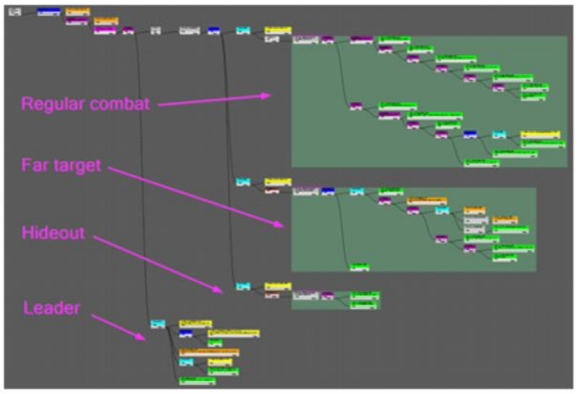


This is the *crew* behavior tree for regular combat. This is what drives the coordinated combat video I've shown earlier.

It's nothing fancy. You see two branches, one for the roles assigned to NPCs currently engaging the player, another one for the roles assigned to NPCs chasing the player. So this is just a definition of what roles are available for this part of combat.

But that's only one part of the crew combat behavior.

All combat



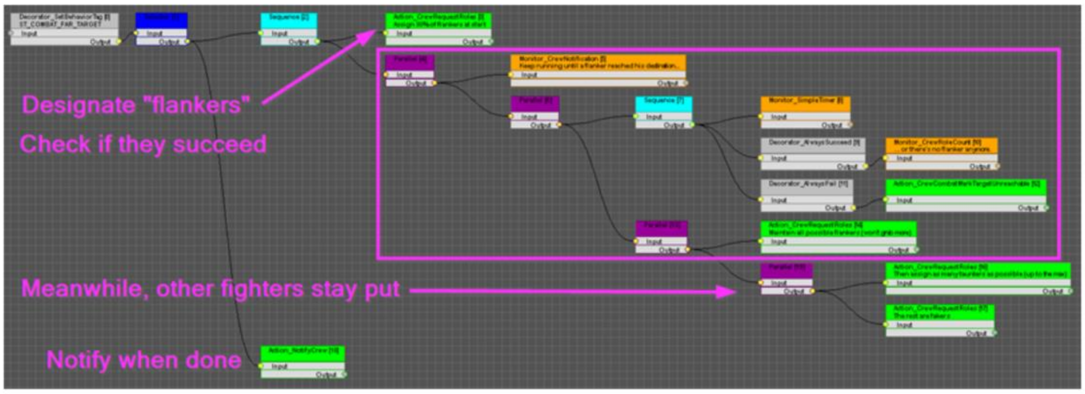
Here's what you get if you zoom out a bit: the big picture.

So as you can see there's more than "regular combat". Under certain conditions, the crew behavior changes and roles are assigned differently.

Now let me focus on the part labelled "far target".

ARTIFICIAL INTELLIGENCE SUMMIT

Far target


Decorator_SetBehaviorTag [1] ST_COMBAT_FAR_TARGET
Sequence [2]
Action_CrewRequestRoles [3]
Designate "flankers"
Check if they succeed
Parallel [4]
Monitor_CrewNotification [5] Keep running until a flanker reached his destination...
Parallel [6]
Sequence [7]
Monitor_SimpleTimer [8]
Decorator_AlwaysSucceed [9]
Monitor_CrewRoleCount [10] ... or there's no flanker anymore.
Decorator_AlwaysFail [11]
Action_CrewCombatMarkTargetUnreachable [12]
Action_CrewRequestRoles [14] Maintain all possible flankers (won't grab more)
Meanwhile, other fighters stay put
Action_CrewRequestRoles [16] Then assign as many taunters as possible (up to the max)
Action_CrewRequestRoles [17] The rest are fakers
Notify when done
Action_NotifyCrew [18]
Input
Output

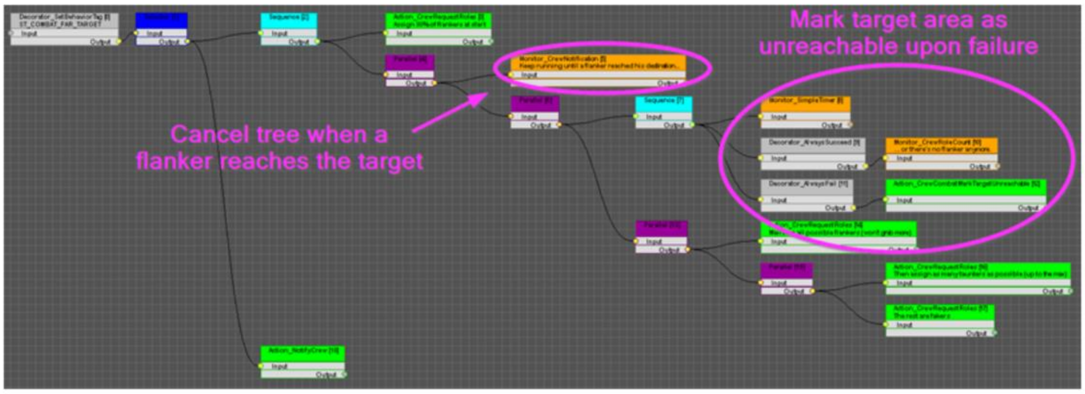
GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

UBM

This is activated when one member of the crew notices that their path to the player is way longer than the actual distance. That means that there's some piece of level that needs to be bypassed to reach them.

So what the crew does in this situation is that it splits the combat in two. A few members are sent to flank the player, trying to reach the melee, while the others simply stay put, taunting and using ranged attacks.

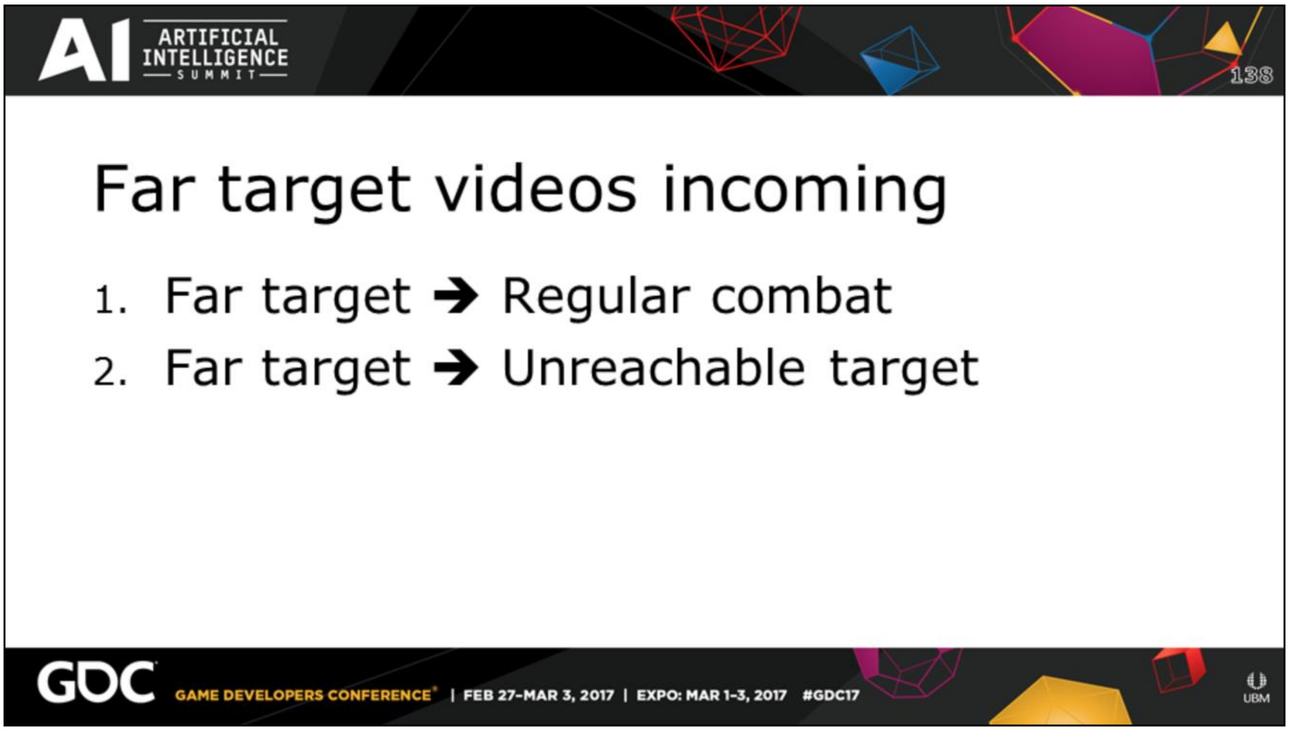
Far target


Mark target area as unreachable upon failure
Cancel tree when a flanker reaches the target
Decorator_SetBehaviorTag [1] ST_COMBAT_FAR_TARGET
Sequence [2]
Parallel [4]
Monitor_CrewNotification [5] Keep running until a flanker reached his destination...
Parallel [6]
Sequence [7]
Monitor_SimpleTimer [8]
Decorator_AlwaysSucceed [9]
Monitor_CrewRoleCount [10] ...or there's no flanker anymore
Decorator_AlwaysFail [11]
Action_CrewCombatMarkTargetUnreachable [12]
Parallel [13]
Parallel [15]
Action_NotifyCrew [18]

Note that we have two end conditions here, depending on what happens to the flanking NPCs.

As soon as any of them manages to reach the player, the far target tree is cancelled and the crew switches back to regular combat, which means that more NPCs will join the melee.

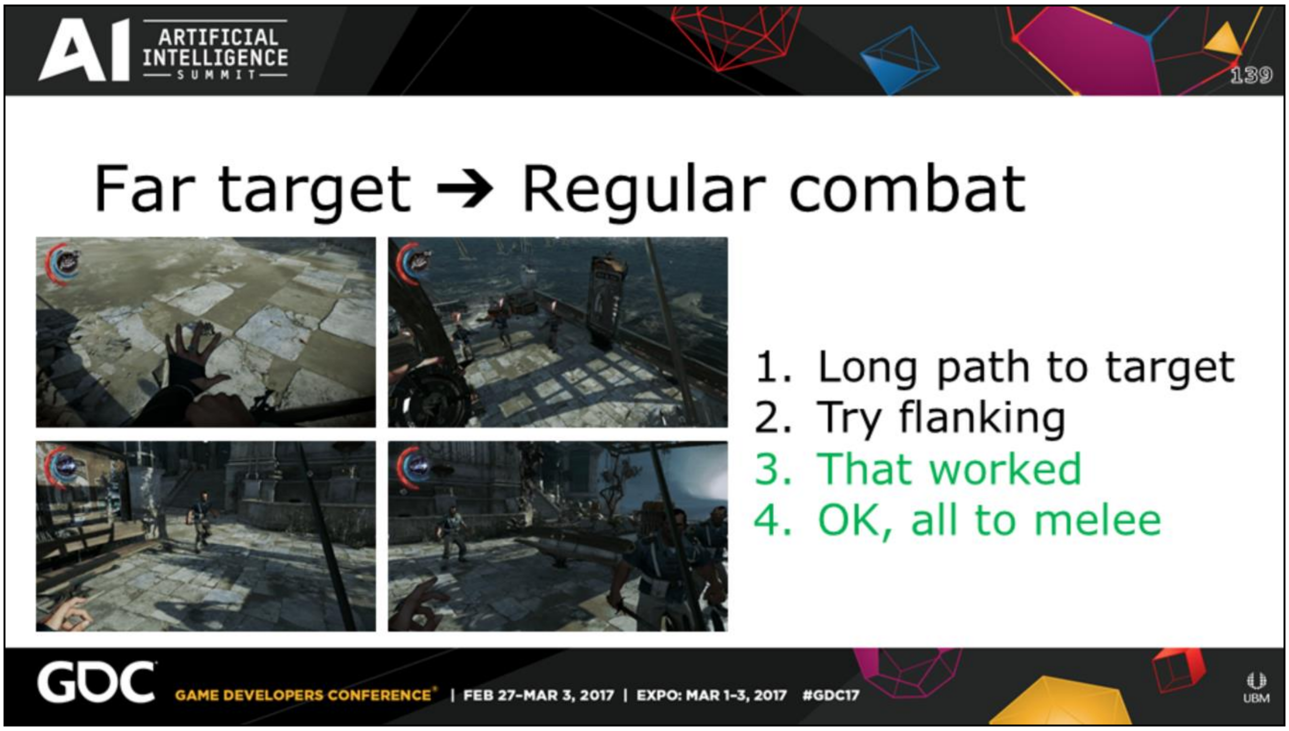
But if too much time passes, or if any of the flankers gets killed, the crew marks the area around the target as unreachable. In this case it’s deemed too dangerous to try flanking the player so we’ll just stick to ranged combat.



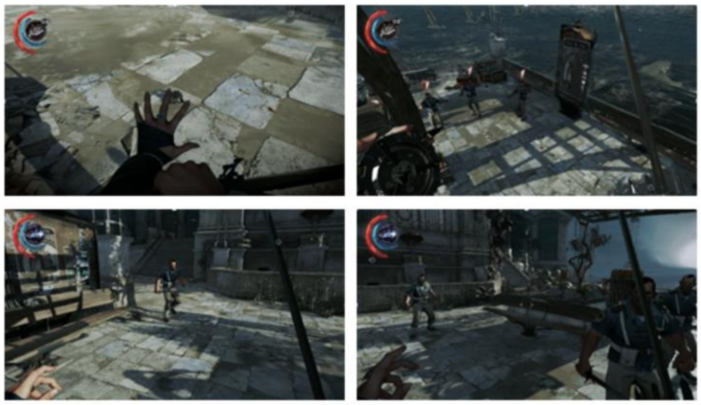
Far target videos incoming

1. Far target ➔ Regular combat
2. Far target ➔ Unreachable target

So let me show you these two outcomes in video.



Far target ➔ Regular combat



1. Long path to target
2. Try flanking
3. That worked
4. OK, all to melee

<originally a video>

Here we have a few guards...

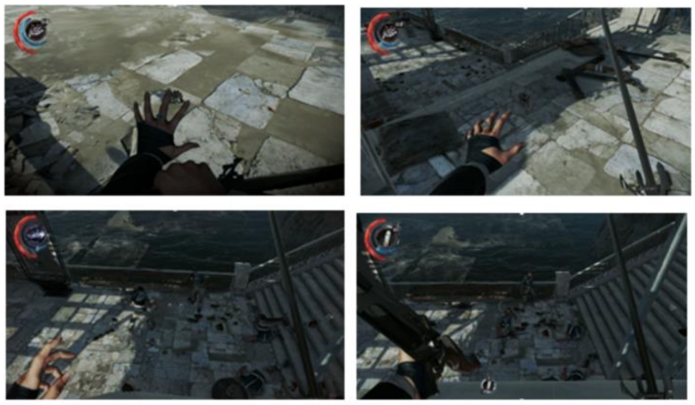
I'm setting up a trap, and I'll try to lure them into it...

They busted me! The combat starts, and I get to a point where they'll have to navigate a long path to reach me...

Now one of them is trying to flank me... and gets killed.

But I didn't set enough traps so another one reaches me and they all come in melee...

Far target ➔ Unreachable target



1. Long path to target
2. Try flanking
3. Oops!
4. Let's stay here

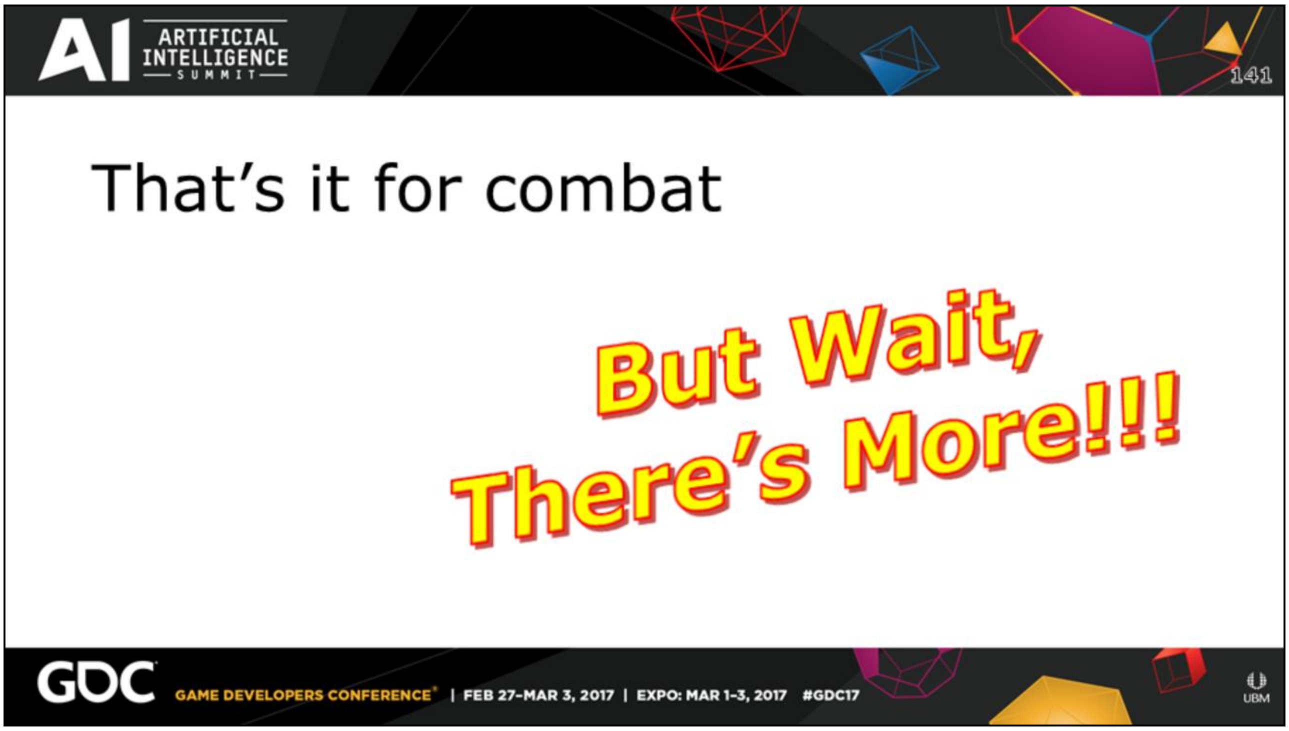
<originally a video>

Apparently one trap wasn't enough, let's try with two of them...

Same thing: I'm getting aggro, trying to lure them into these traps....

Two of them are trying to flank me, but this time both get killed.

So at this point, the remaining guards just decide it's safer to stay where they are and to pick me from a distance.

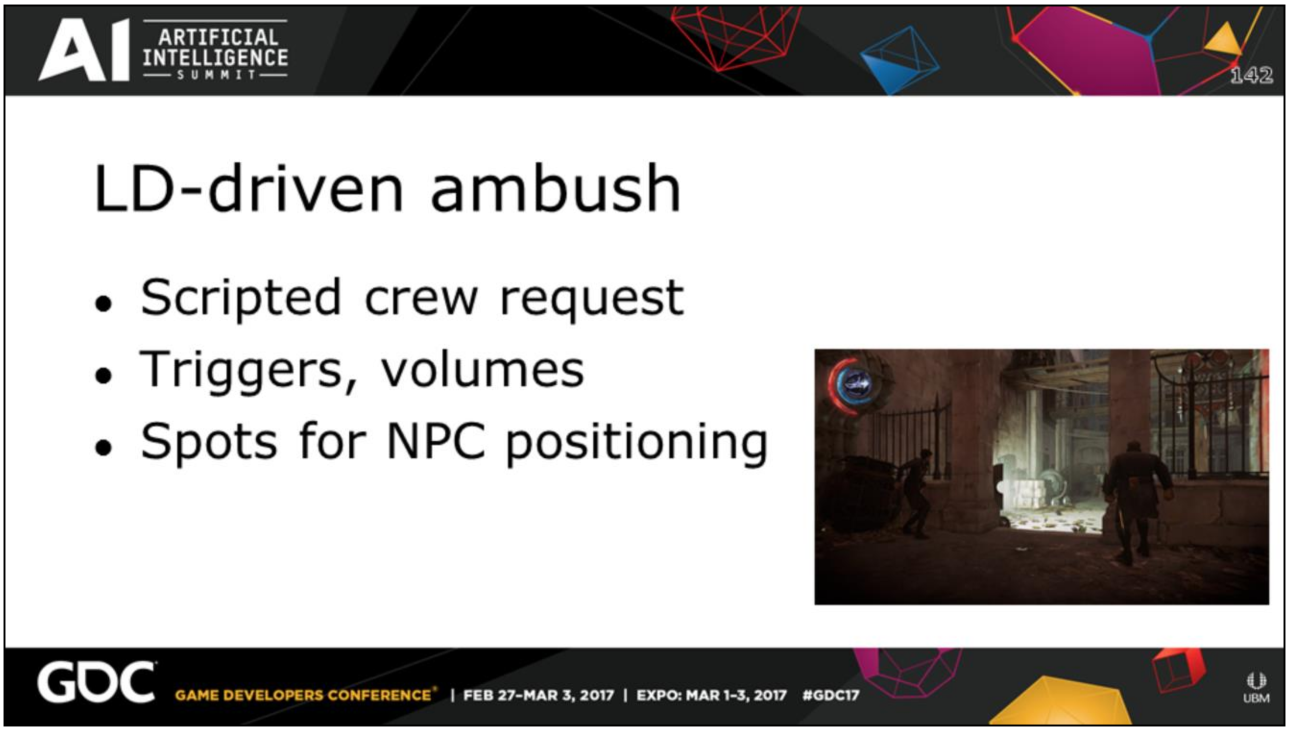


That’s it for combat

But Wait,
There’s More!!!

So that’s it for combat crews.

But we do use crews for more stuff.



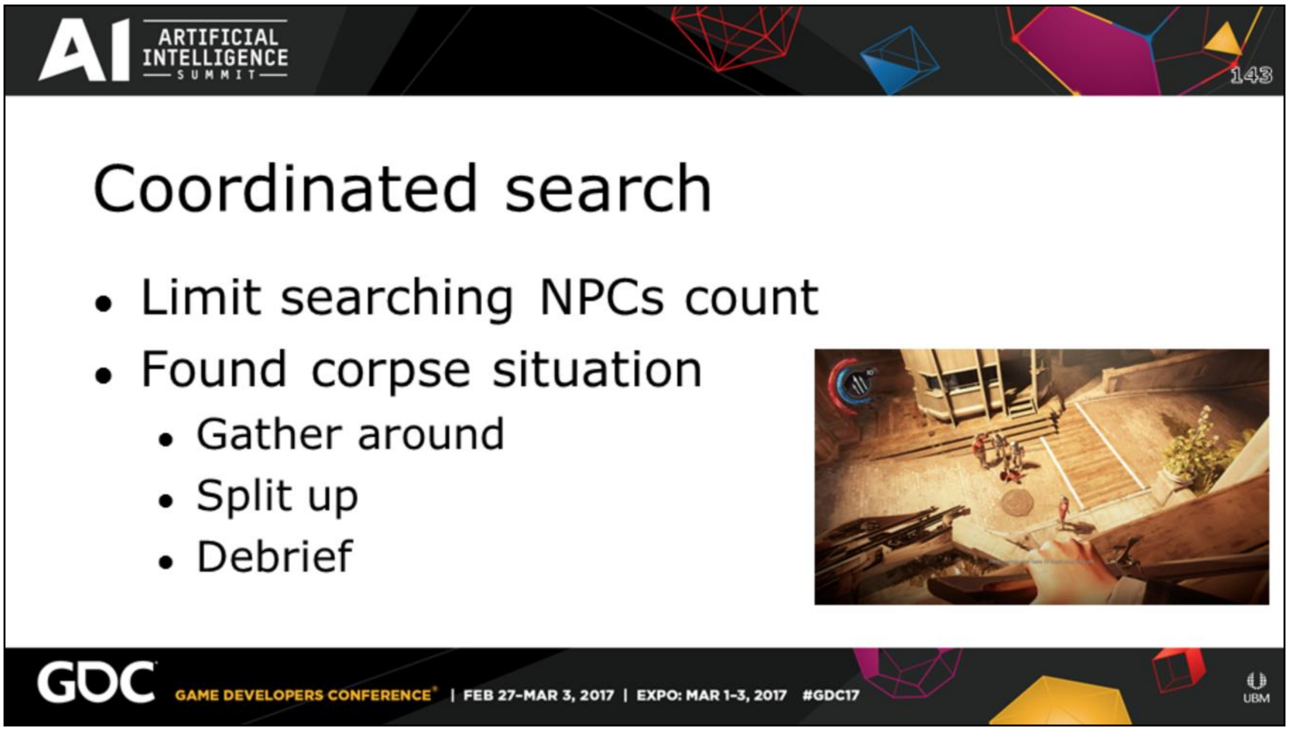
LD-driven ambush

- Scripted crew request
- Triggers, volumes
- Spots for NPC positioning

Everything I've shown so far is level independent, it can trigger anywhere in the game.

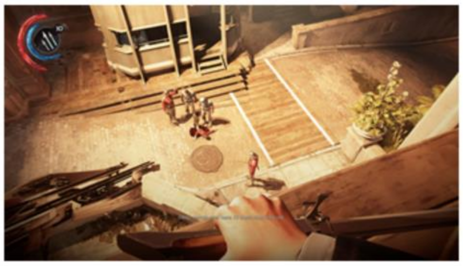
But we also wanted to try to use crews for ambushes, and to let level designers set them up. There was a few cases in the first Dishonored where scripted ambushes were messing up with the systemic AI, so this was an attempt to fix this.

In Dishonored 2 level designers can create an ambush crew from script, start and stop it as they want. On top of this they manage the triggers and volumes that define the ambush area, and place spots that NPCs use for positioning.



Coordinated search

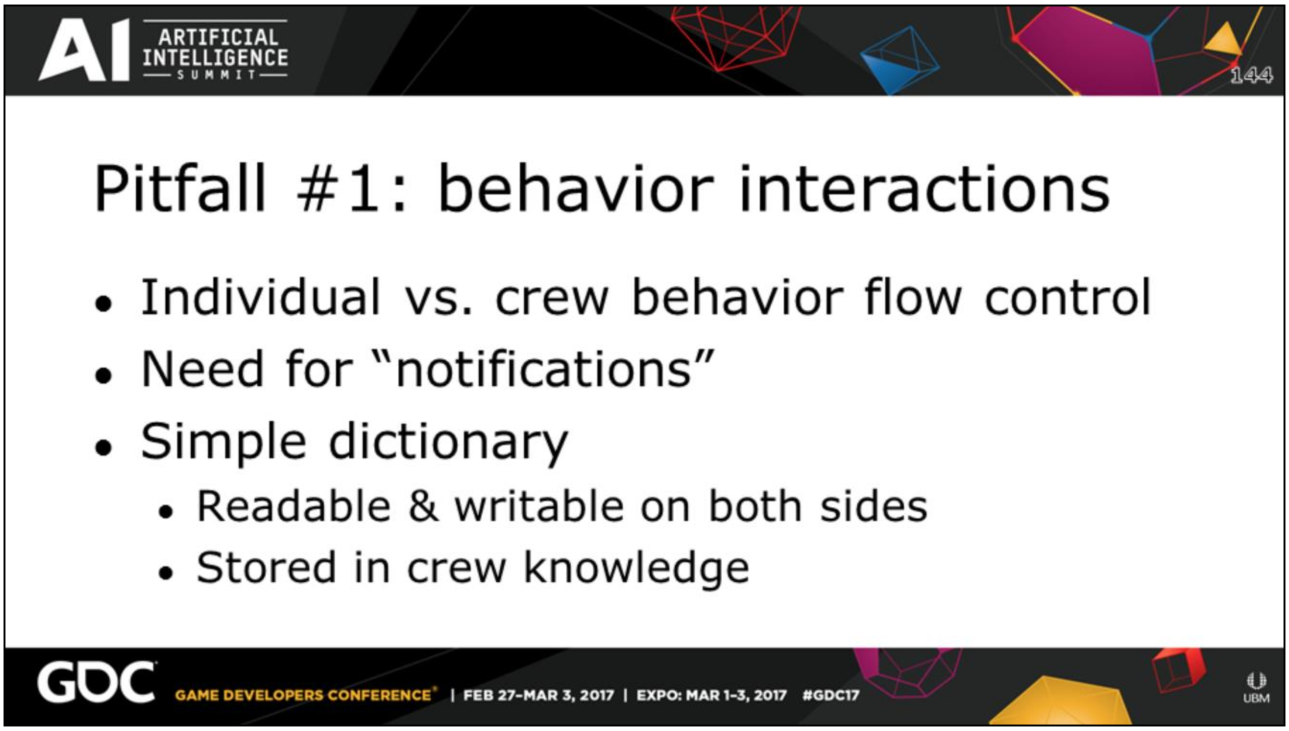
- Limit searching NPCs count
- Found corpse situation
  - Gather around
  - Split up
  - Debrief



Yet another case of coordinated behavior is search.

First, we use the crew to limit the number of NPCs searching actively, in some cases we even allow only one of them to search.

There’s also more evolved coordinated behavior that might trigger when guards discover a dead body. They first gather around the corpse. One of them will shout a few orders, and then they’ll split up in search for the culprit. If they fail in finding anything, they’ll gather round again to debrief the situation before falling back to patrol. These phases are also handled by the crew behavior tree.


ARTIFICIAL INTELLIGENCE SUMMIT

Pitfall #1: behavior interactions

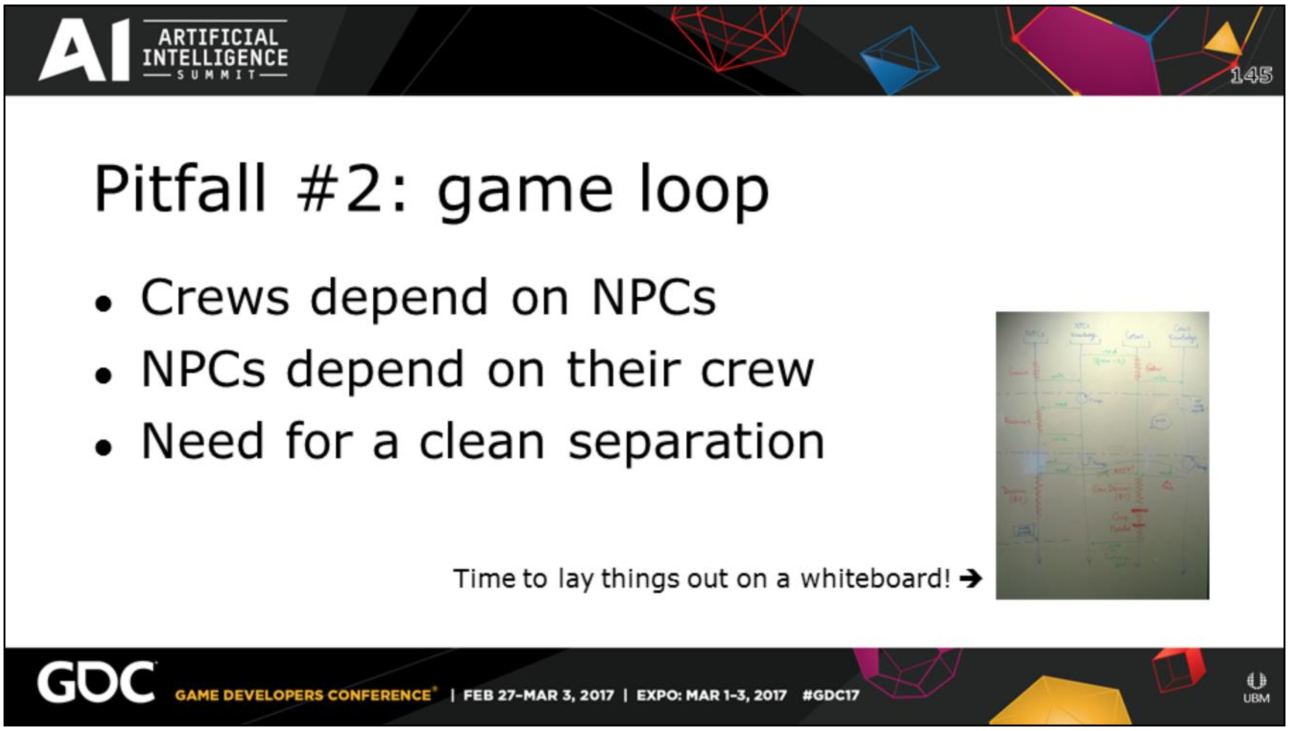
- Individual vs. crew behavior flow control
- Need for "notifications"
- Simple dictionary
  - Readable & writable on both sides
  - Stored in crew knowledge

Before I close this, I just want to want you about two pitfalls of this system. The first one is how to handle behavior tree interactions.

I'm talking about how to control the execution flow between individual and crew behavior trees.

For instance you often want to know if some part of an individual behavior is done executing before you switch to a new stage of group behavior. We needed some kind of "notification" for that.

So we implemented a simple dictionary that is readable and writable from both sides, and stored in crew knowledge. This did the trick.



Pitfall #2: game loop

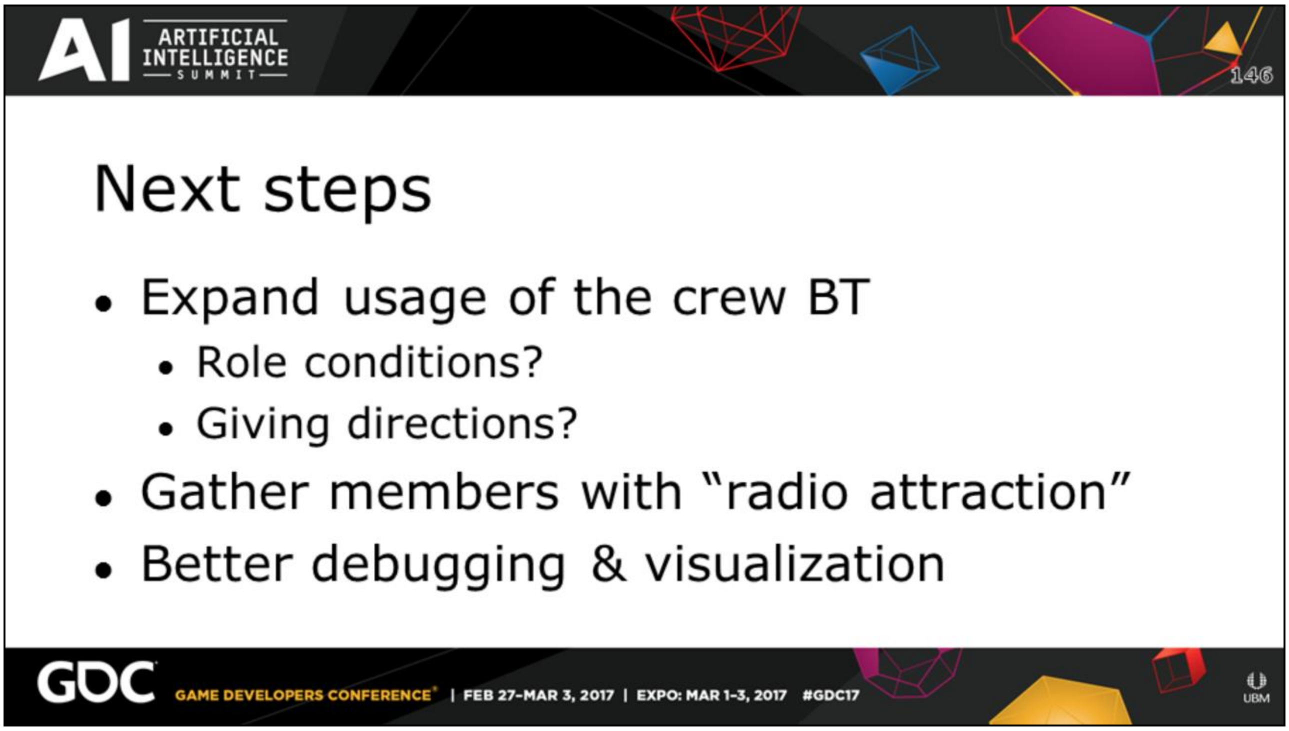
- Crews depend on NPCs
- NPCs depend on their crew
- Need for a clean separation

Time to lay things out on a whiteboard! ➔

The second pitfall is how to organize the updates in the game loop.

Crews depend on NPCs, and NPCs depend on their crew, so there's a bit of a chicken and egg problem. So it's important to keep a clean separation between the individual level and the collective level.

Our solution isn't very interesting since it's implementation-dependent and changed a lot over time, but please note that is something that requires some thoughts to get right.


ARTIFICIAL INTELLIGENCE SUMMIT

146

Next steps

- Expand usage of the crew BT
  - Role conditions?
  - Giving directions?
- Gather members with “radio attraction”
- Better debugging & visualization

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

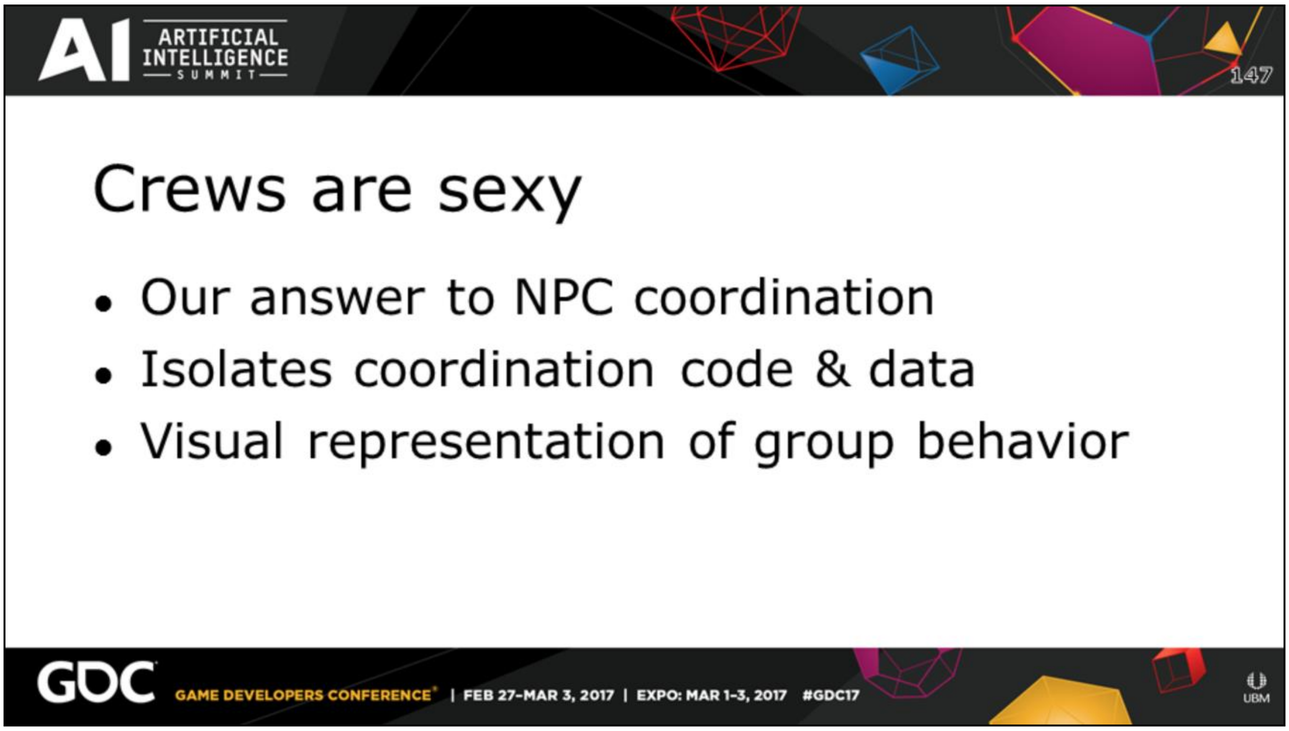
UBM

So clearly this is just the beginning, and we need to improve our implementation.

I wish we exposed more things in the crew behavior tree. For instance, the conditions for assigning a role to a given NPC could be defined in it, but for now they’re all written directly in C++ code. Same for the execution of roles.

Something else we didn’t implement is the gathering of crew members using some kind of psychic/radio attraction. It could be used notably for coordinated distractions in patrol, but this remained at the idea level.

And last but not least, this almost goes without saying: NPC coordination is something very abstract, and requires good tools for debugging. Ours were pretty rough, so that’s something we’ll have to improve in the future.



Crews are sexy

- Our answer to NPC coordination
- Isolates coordination code & data
- Visual representation of group behavior

GDC GAME DEVELOPERS CONFERENCE® | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

So here are my last words. Crews are sexy!

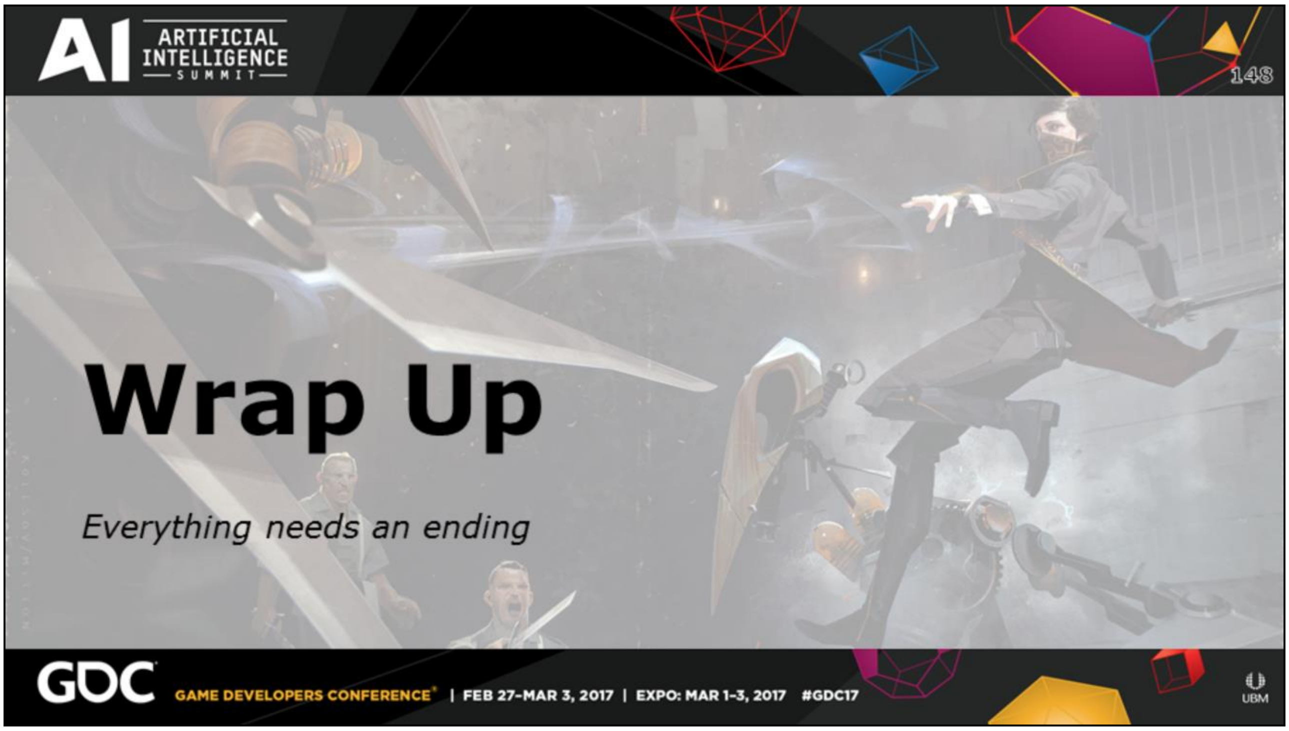
We us them as a machete to clear the uncharted territory of NPC coordination.

They allow us to separate the coordination logic from the individual behavior.

Using behavior trees gives a visual representation of the flow of group behaviors, which is something very precious during development.

I wish we will find more use cases for crews so we can refine them even more.

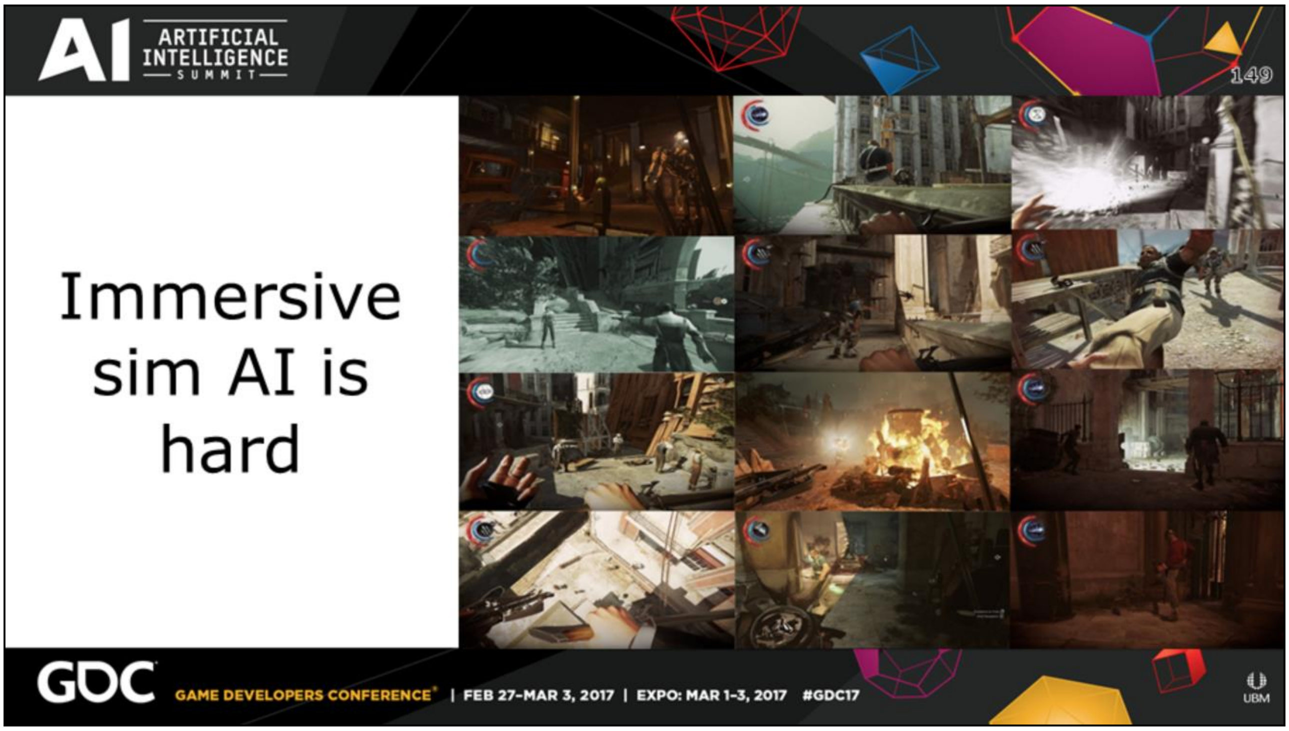
So that was the last of our systems. Time to wrap up, Xavier!


ARTIFICIAL INTELLIGENCE SUMMIT
148
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

Wrap Up

Everything needs an ending

Thanks Laurent, here are some parting thoughts before we dissolve our crew.



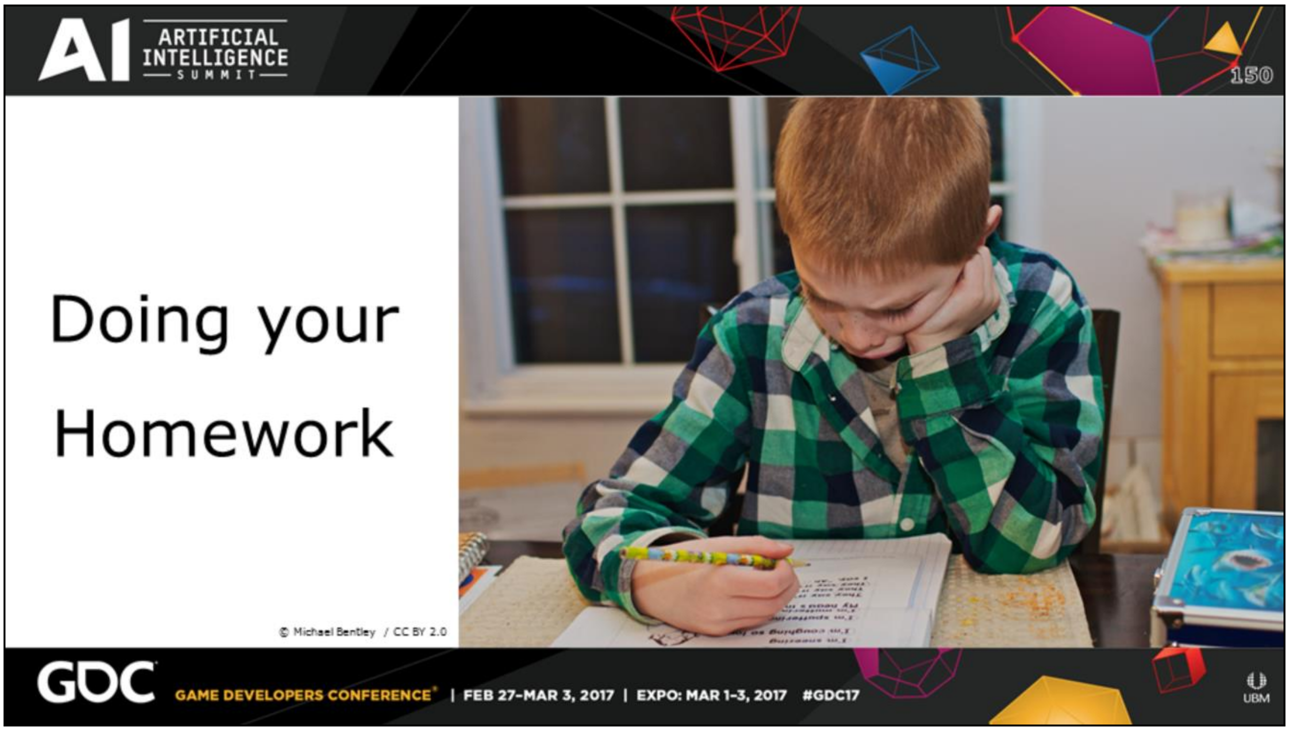
ARTIFICIAL INTELLIGENCE SUMMIT

149

Immersive sim AI is hard

GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Implementing Dishonored 2 AI was quite the ride. Even while being very conservative and prudent, sometimes the good old “keep it simple” principle just does not cut it in front of the hoard of systems interacting together. That’s why we had to commit to put muscular systems behind simulation.



ARTIFICIAL INTELLIGENCE SUMMIT
150
© Michael Bentley / CC BY 2.0
GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17
UBM

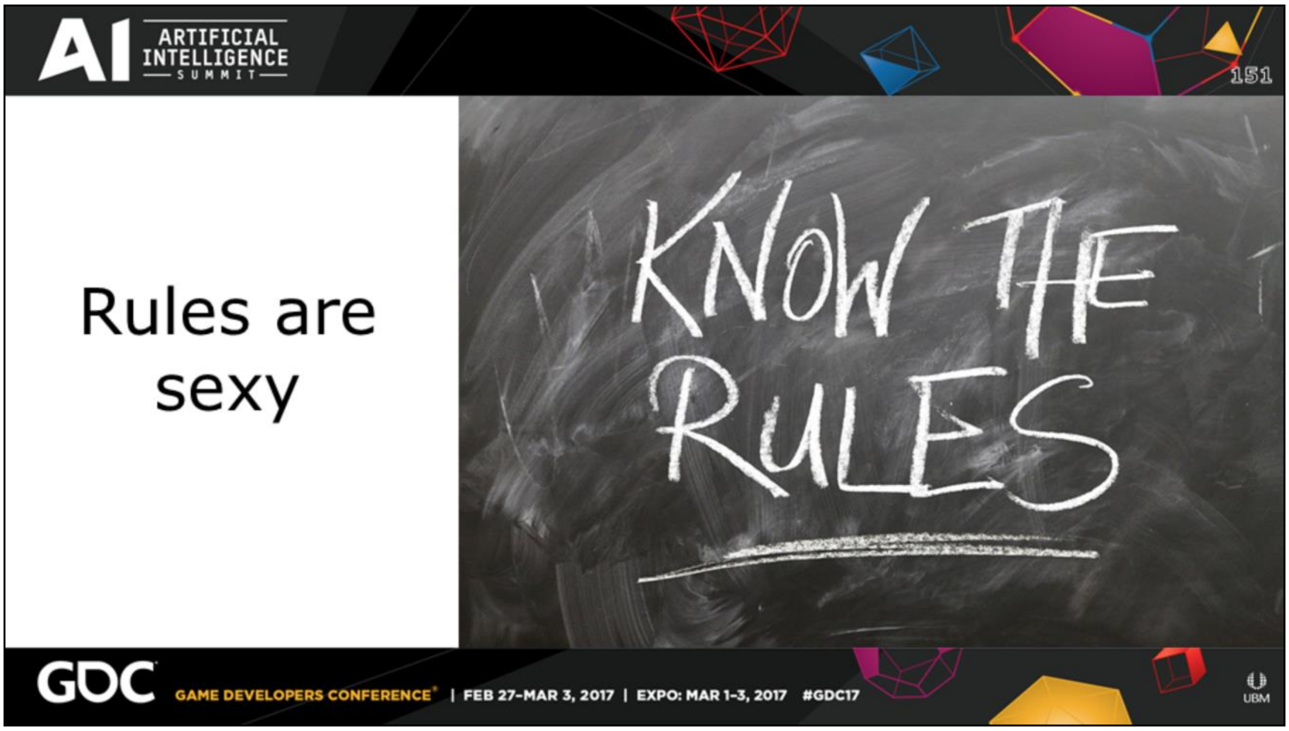
Doing your Homework

It is interesting to observe that the systems we did the most research on were the ones that needed fewer changes over the course of the production. Whereas systems that grew more organically tend to be the ones that really need refactoring.

Obviously it is not possible to write technical document for everything, sometimes you just don't have enough visibility upfront.

But when you get the chance, take the opportunity to stop coding and start writing.

Photo © Michael Bentley / CC BY 2.0
https://www.flickr.com/photos/donhomer/12862432364


ARTIFICIAL INTELLIGENCE SUMMIT

Rules are sexy

KNOW THE RULES

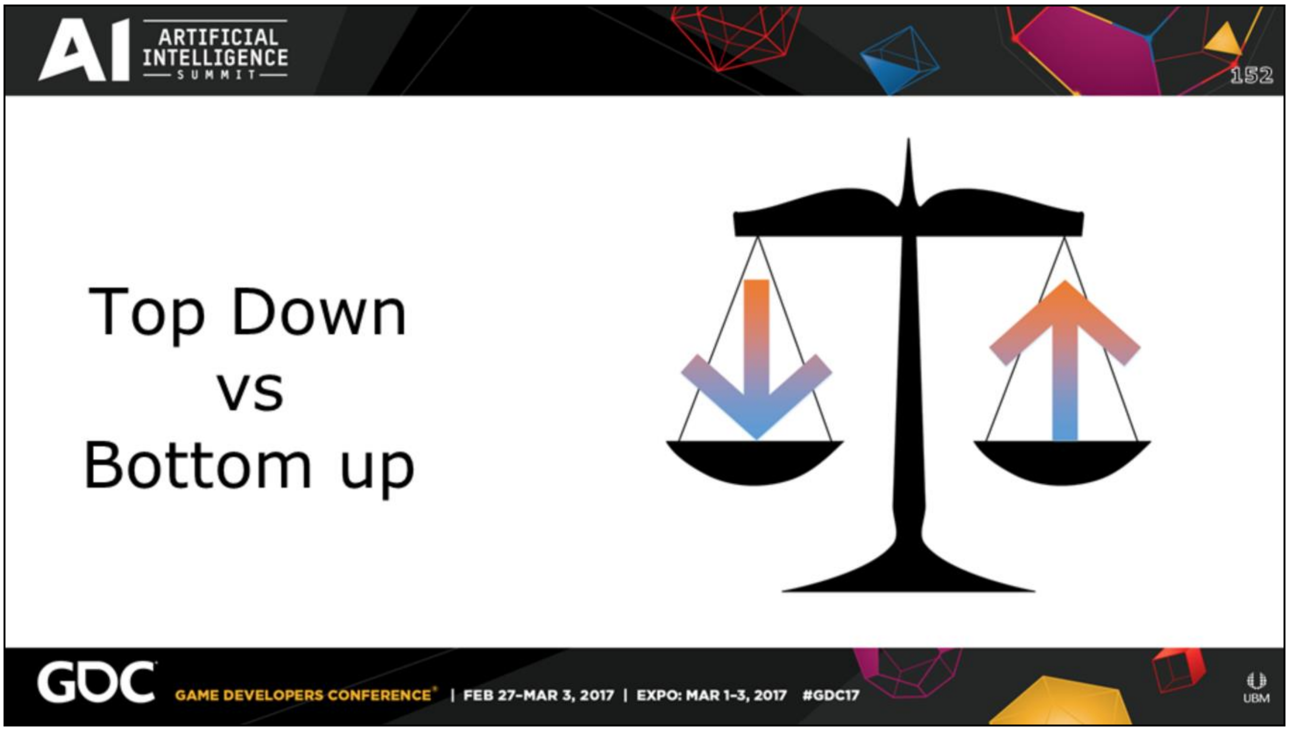
GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

UBM

AI is really about pushing proper content toward the player when it makes the most sense.

Given the amount of assets modern games have to manage, a data driven way to select the moments and the content is absolutely crucial to maintain a decent iteration rate.

That’s exactly what the rule system does for us.



ARTIFICIAL INTELLIGENCE SUMMIT

152

Top Down vs Bottom up

GDC GAME DEVELOPERS CONFERENCE | FEB 27-MAR 3, 2017 | EXPO: MAR 1-3, 2017 #GDC17

Coordination is a hard topic. Hard to design and implement, hard to debug and maintain, so more often that not games choose to fake it.

Since the Left 4 Dead’s AI Director, there has been a general trend of using some kind of master AI to drive something else that just individual NPCs behavior.

Our experience with crews convinced us that a top level AI making suggestions to individual NPCs strikes the correct balance in order to achieve elaborate coordinated behaviors. This looks like an exciting topic for the games to come.

Take back what's yours !

© Blake Patterson / CC BY 2.0

And lastly, we used what would seems an astronomic amount on memory in DSM. Likewise creating a domain specific language like the rules system can be scary, in term of hardware resources management. However costs that were prohibitive on last generation consoles can now be totally reasonable. *But* people habits might change slower than hardware.

Which means that if you are working on current gen, you really should try and have your lead allow you to use those extra resources.

Please, use those megs, use those CPU cycles… Just use them so we can play even cooler games with even cooler AI in it.

Photo © Blake Patterson / CC BY 2.0
https://www.flickr.com/photos/blakespot/6173837649

ARTIFICIAL INTELLIGENCE SUMMIT

154

Thank you,

We're hiring!

job@arkane-studios.com

https://jobs.zenimax.com/locations/view/48

GDC GAME DEVELOPERS CONFERENCE® | FEB 27–MAR 3, 2017 | EXPO: MAR 1–3, 2017 #GDC17

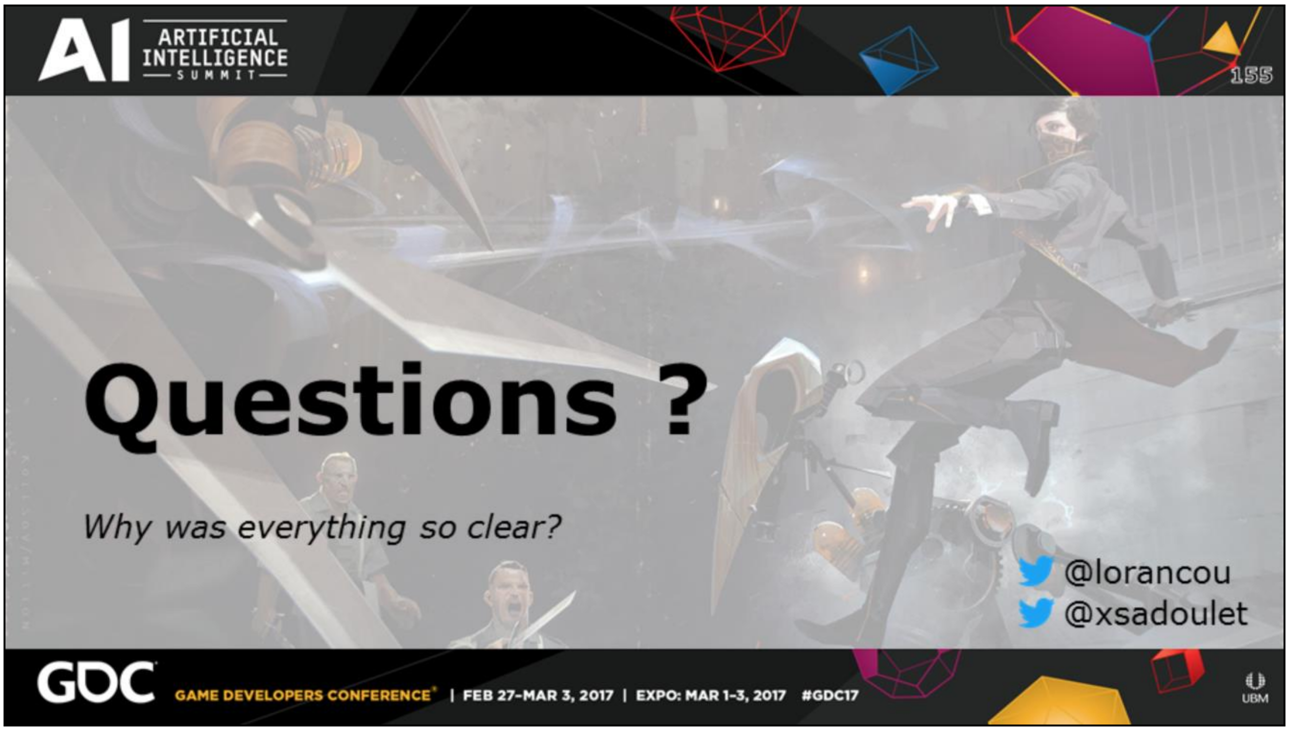
UBM

Thank you for listening.

By the way, we are recruiting*!

---

[*] job@arkane-studios.com
https://jobs.zenimax.com/locations/view/48



Questions ?

Why was everything so clear?

@lorancou
@xsadoulet