IT'S COMPLICATED – GETTING ML INSIDE AAA GAMES

ADRIEN LOGUT – UBISOFT MONTREAL
COLIN GAUDREAU – UBISOFT WINNIPEG

22 MARCH 2022

AGENDA

PART 1

RECAP OF PREVIOUS RESULTS

PART 2

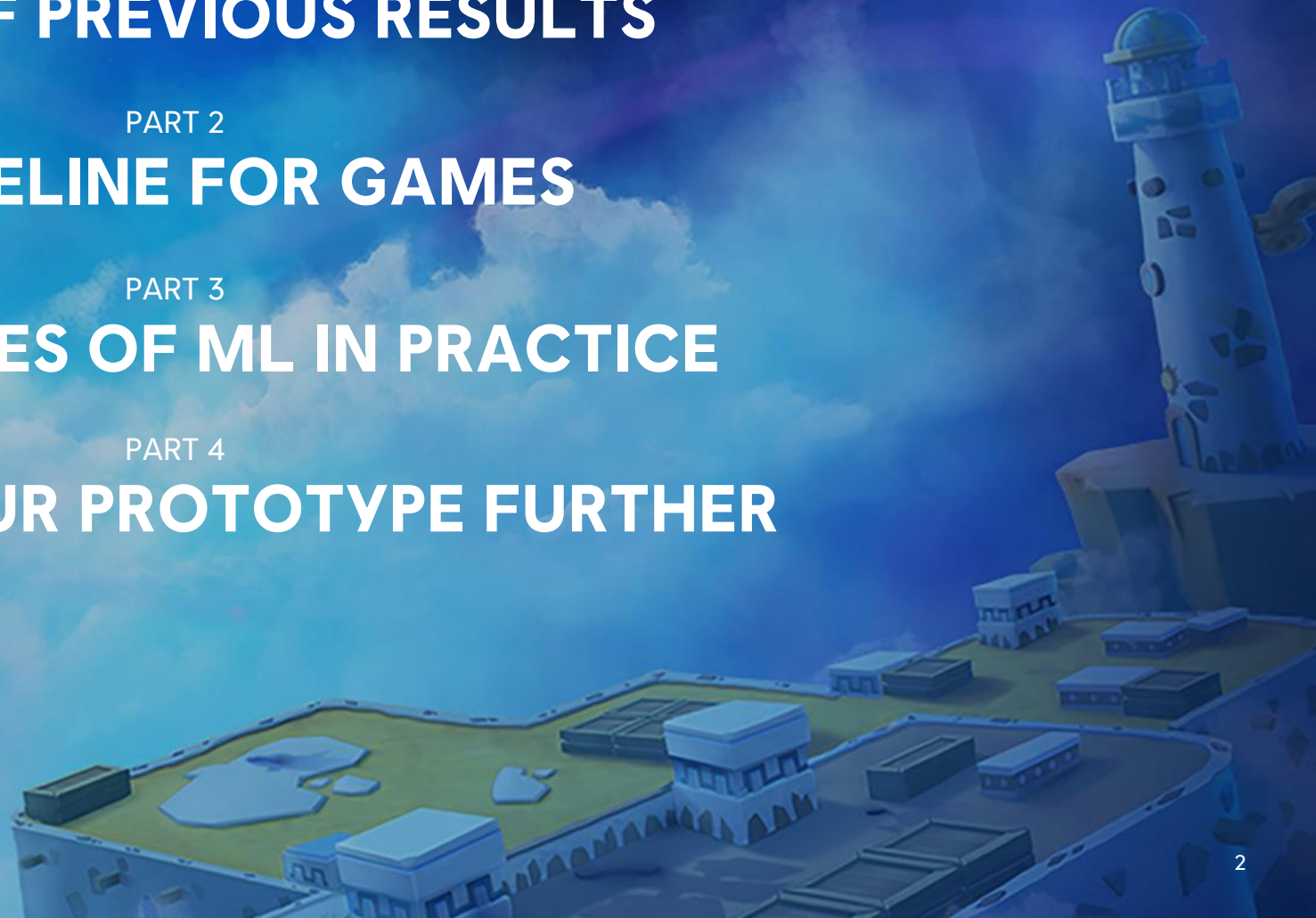
ML PIPELINE FOR GAMES

PART 3

CHALLENGES OF ML IN PRACTICE

PART 4

PUSHING YOUR PROTOTYPE FURTHER



PART 1

RECAP OF PREVIOUS RESULTS

GAMEPLAY BALANCING – FOR HONOR

https://www.gdcvault.com/play/1025891/ML-Tutorial-Day-Smart-Bots

Disclaimer: All videos are prototypes and not game features

SMARTDRIVE – WATCH DOGS 2

https://www.gdcvault.com/play/1025891/ML-Tutorial-Day-Smart-Bots

Disclaimer: All videos are prototypes and not game features

SMARTNAV - HYPERSCAPE

https://www.gdcvault.com/play/1027382/Machine-Learning-Summit-Deep-Reinforcement

Disclaimer: All videos are prototypes and not game features

PART 2

ML PIPELINE FOR GAMES

RL 101 – AI DRIVING EXAMPLE

Action

- Acceleration
- Brake
- Steering

Environment

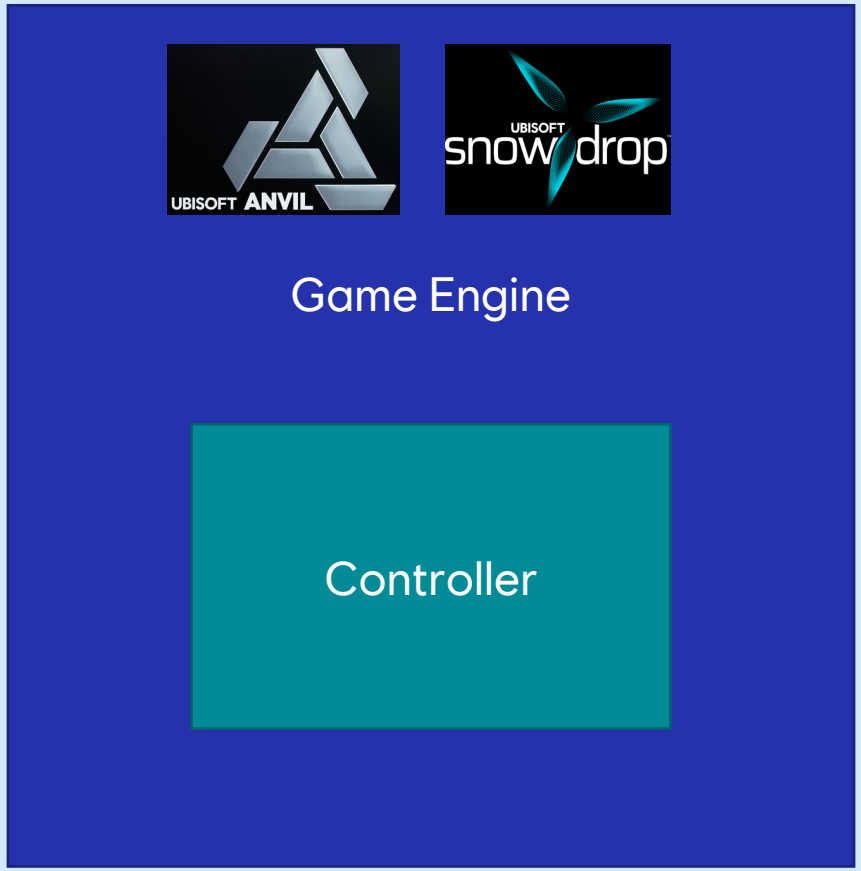
Reward

Agent

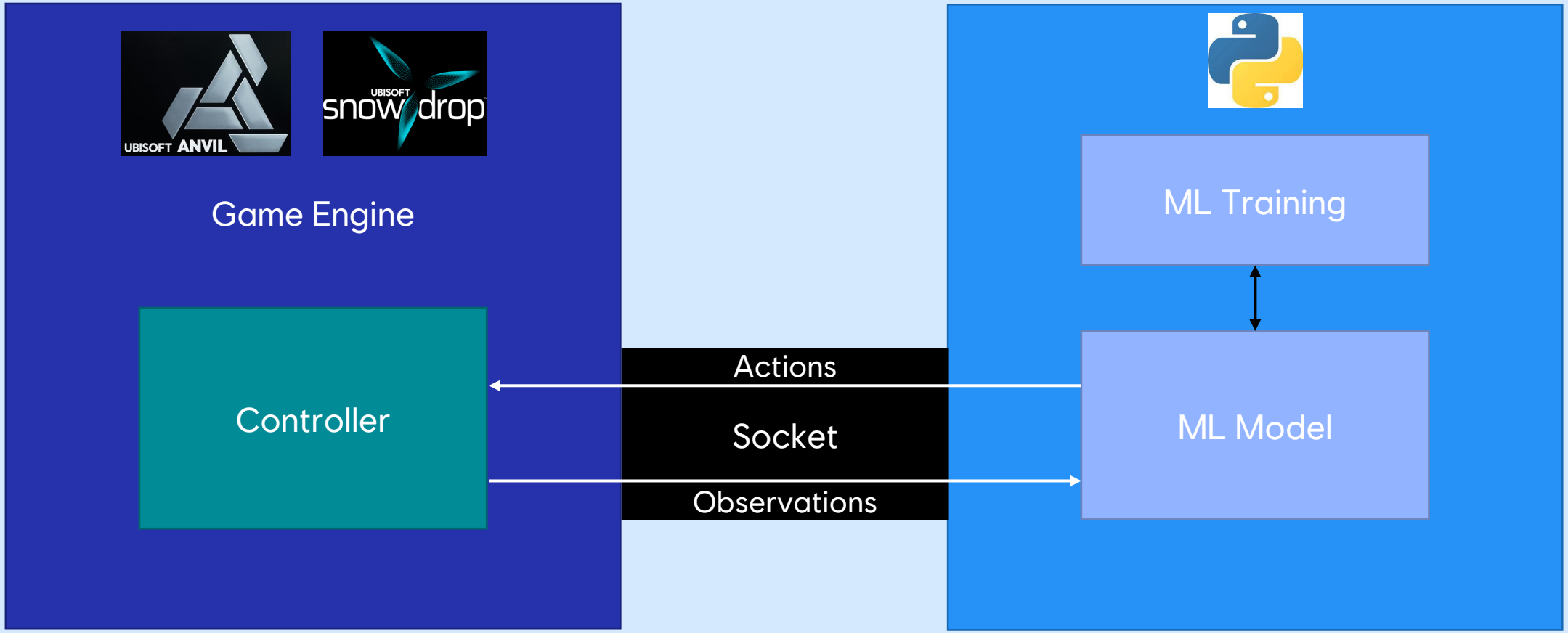
Observation

- Distance from road
- Velocity
- Desired speed

FIRST APPROACH – ML CONTROLLER IN PYTHON



FIRST APPROACH – ML CONTROLLER IN PYTHON



FIRST APPROACH – ML CONTROLLER IN PYTHON

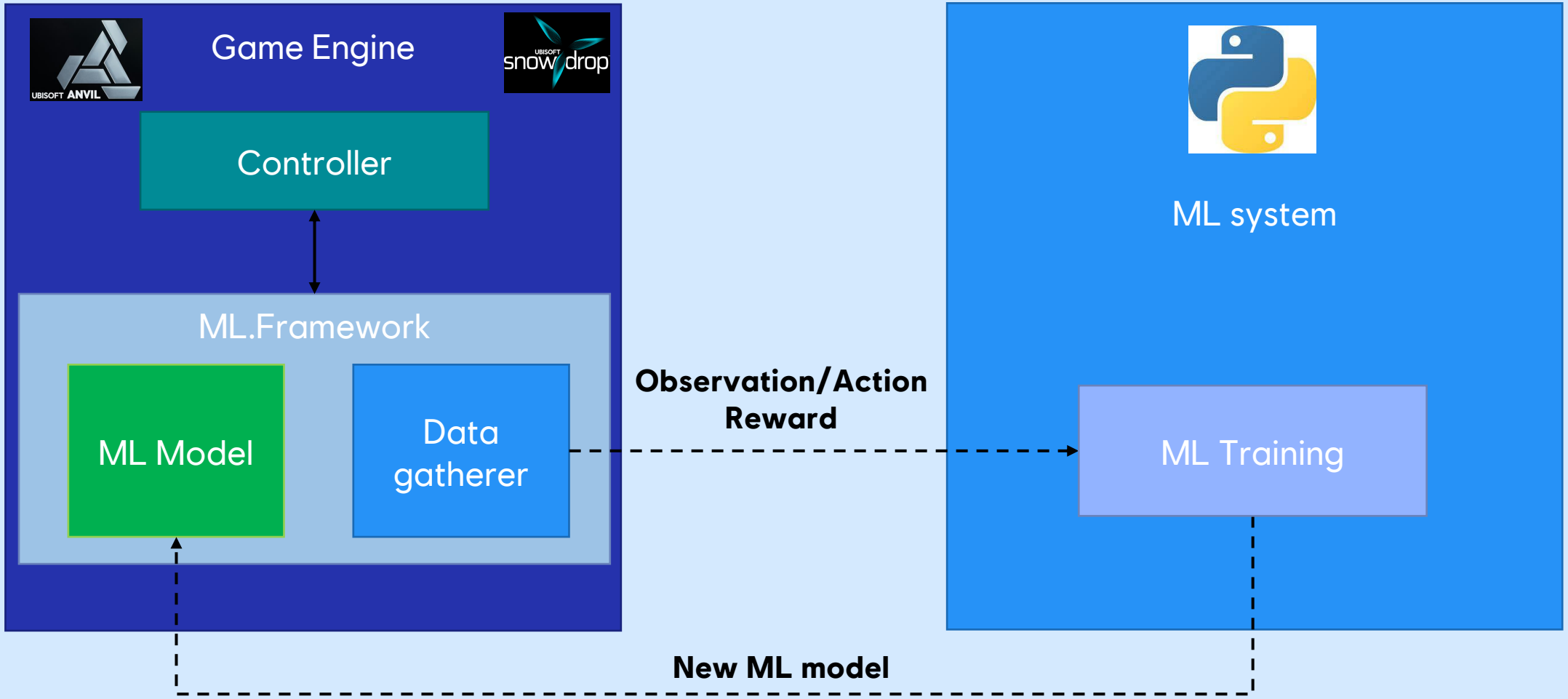
- It can work… for **prototyping**
  - Used on Watch Dogs 2 and For Honor
  - Fast iteration time

FIRST APPROACH – ML CONTROLLER IN PYTHON

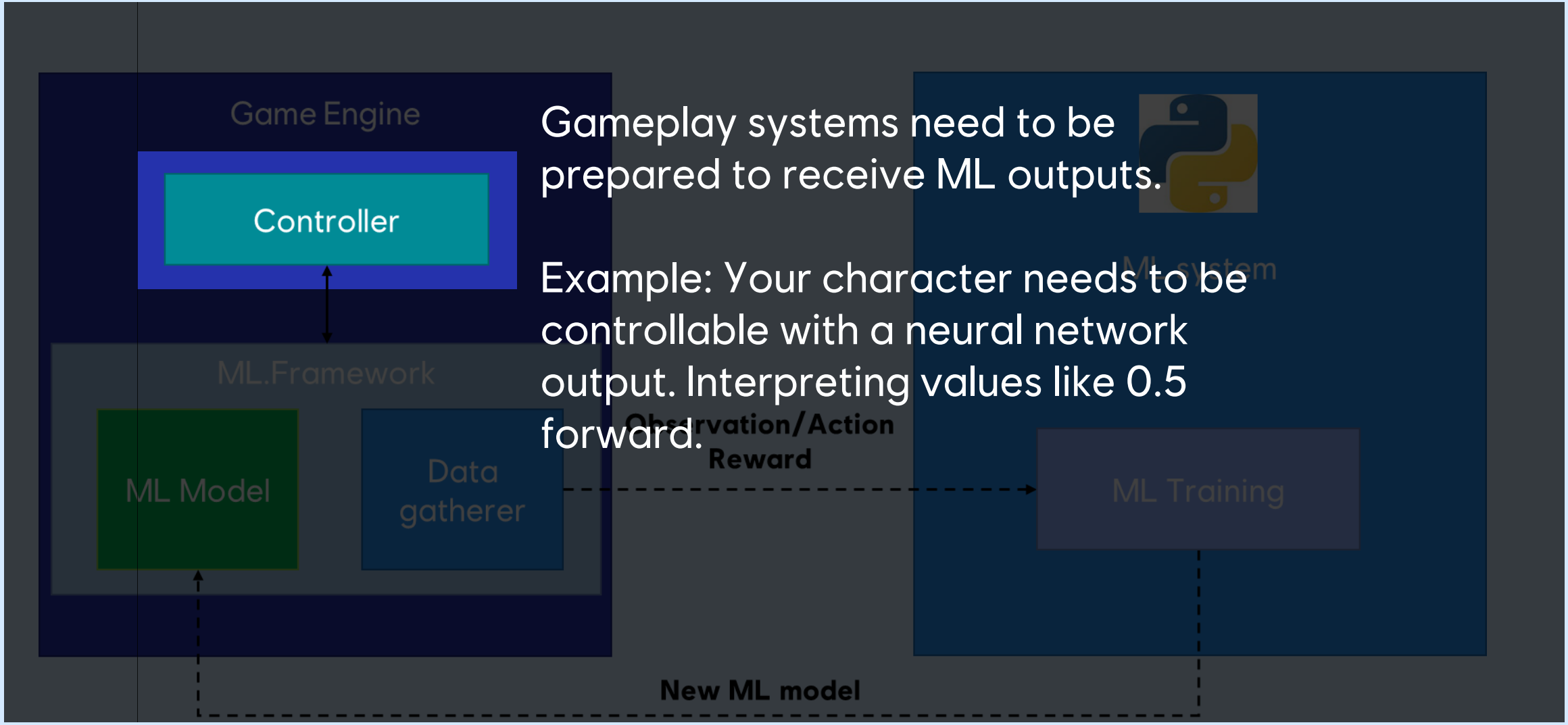
- But it has **drawbacks**
  - Sockets can be slow
  - No easy path to port all in engine

HAVING A STANDARDIZE WAY IN-ENGINE IS CRITICAL TO PUSH IN PRODUCTION

OUR APPROACH – ML IN ENGINE WITH PYTHON TRAINING



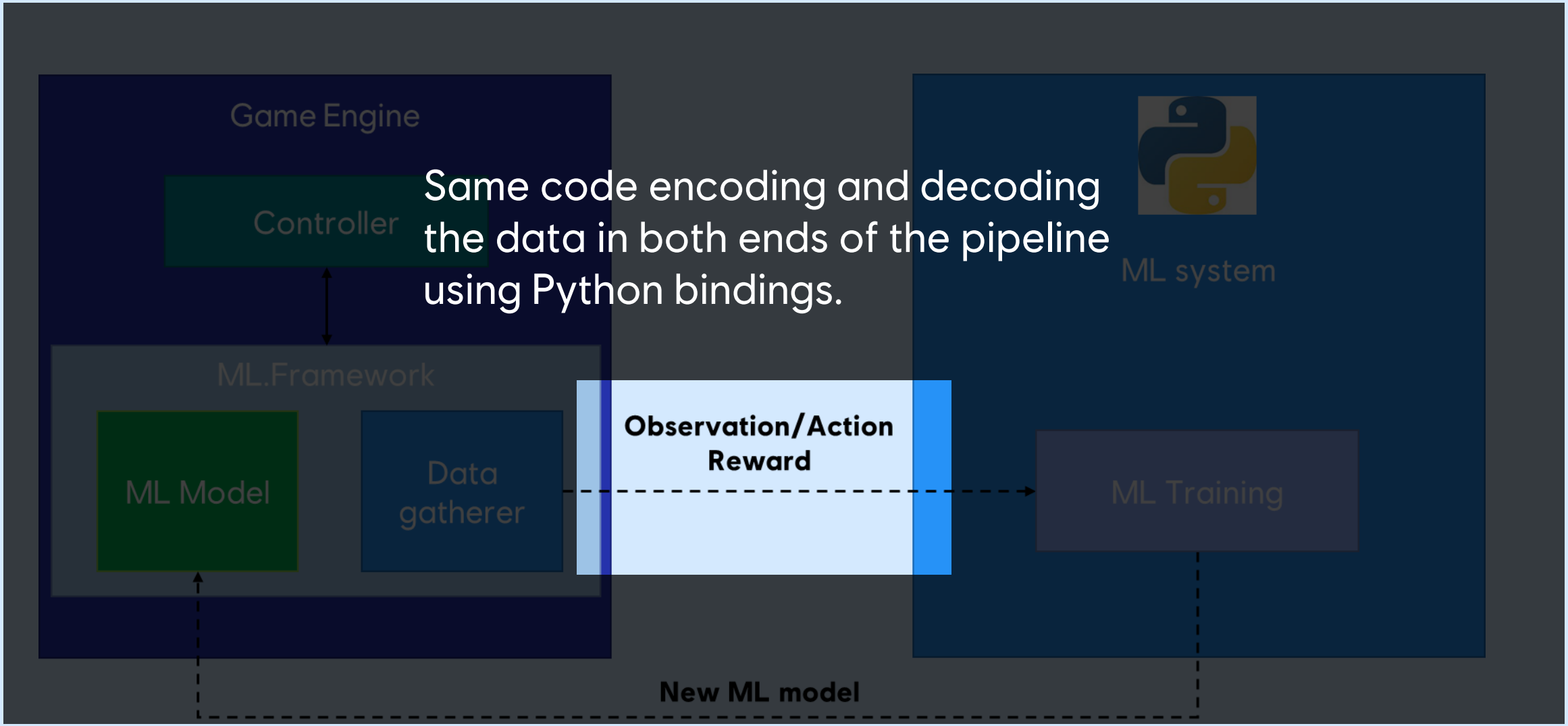
PREPARING YOUR GAMEPLAY SYSTEMS



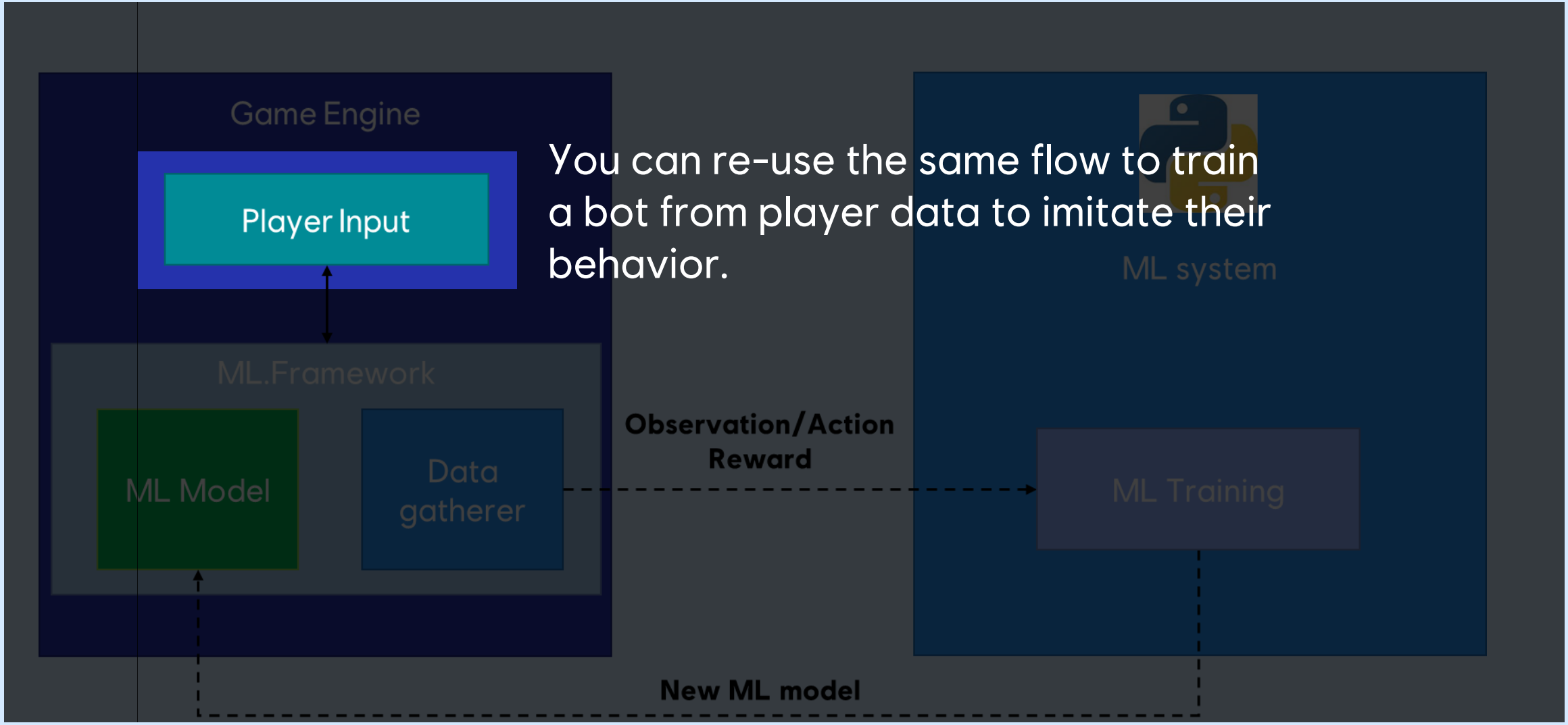
Gameplay systems need to be prepared to receive ML outputs.

Example: Your character needs to be controllable with a neural network output. Interpreting values like 0.5 forward.

UNIFYING THE ECOSYSTEM



IMITATION LEARNING



You can re-use the same flow to train a bot from player data to imitate their behavior.

MAKE ML ACCESSIBLE TO ALL PRODUCTIONS

- Manage all our models.
- Collect data
- Communicate data to Python.
- Neural network inference

- Read experiences from engine.
- Dump new models to engine.
- Do the ML training

MAKE ML ACCESSIBLE TO ALL PRODUCTIONS

- Share all the tedious glue code as middleware
  - Dedicated team for support and consulting
- Same flow for prototyping and production
  - Training can be easily switched off
- Currently used in unannounced projects

PART 3

CHALLENGES OF APPLYING ML IN GAMES

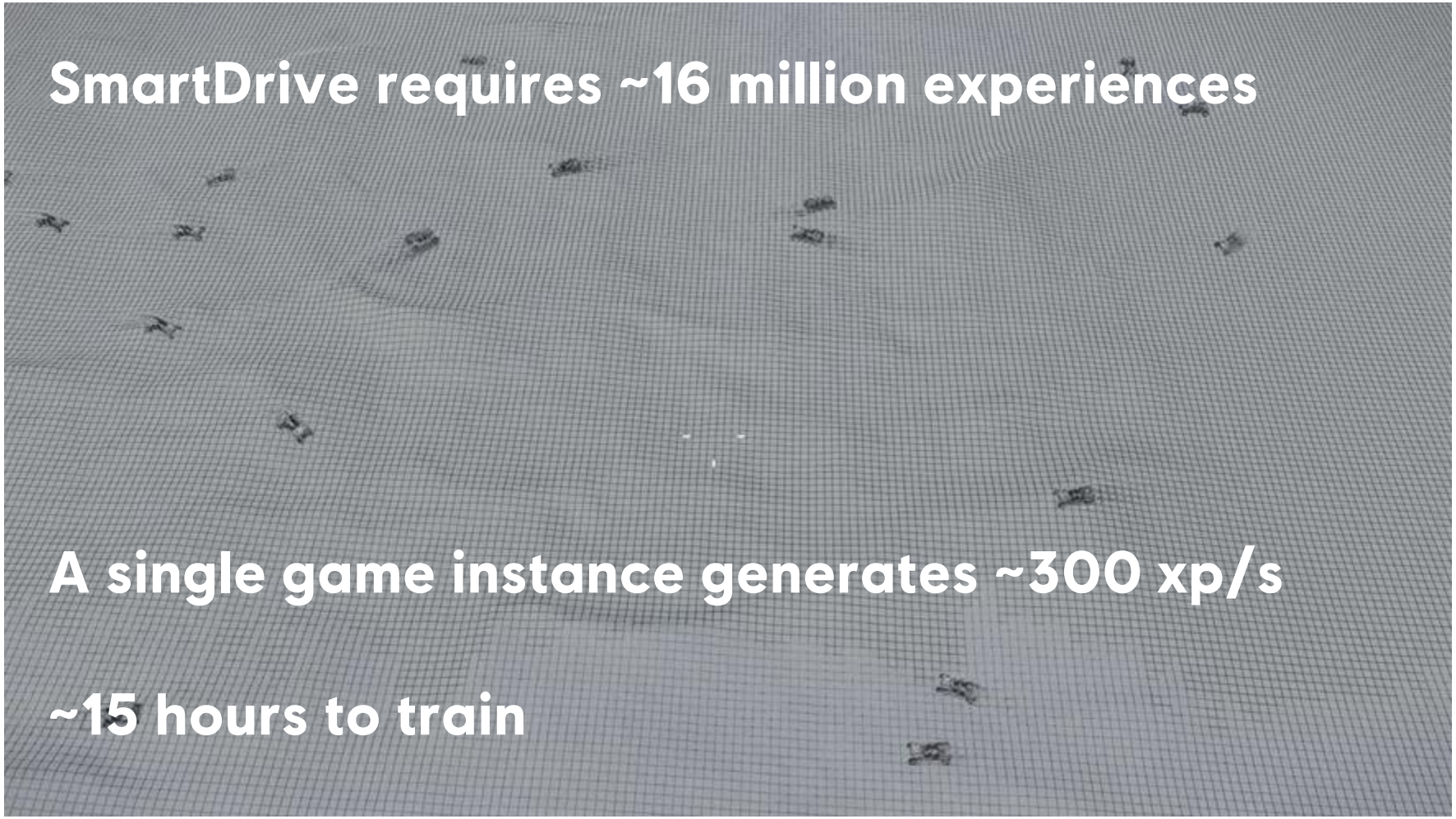
WHAT MAKES A GOOD ML PROBLEM?

WHAT MAKES A GOOD ML PROBLEM?

Classical solutions fails

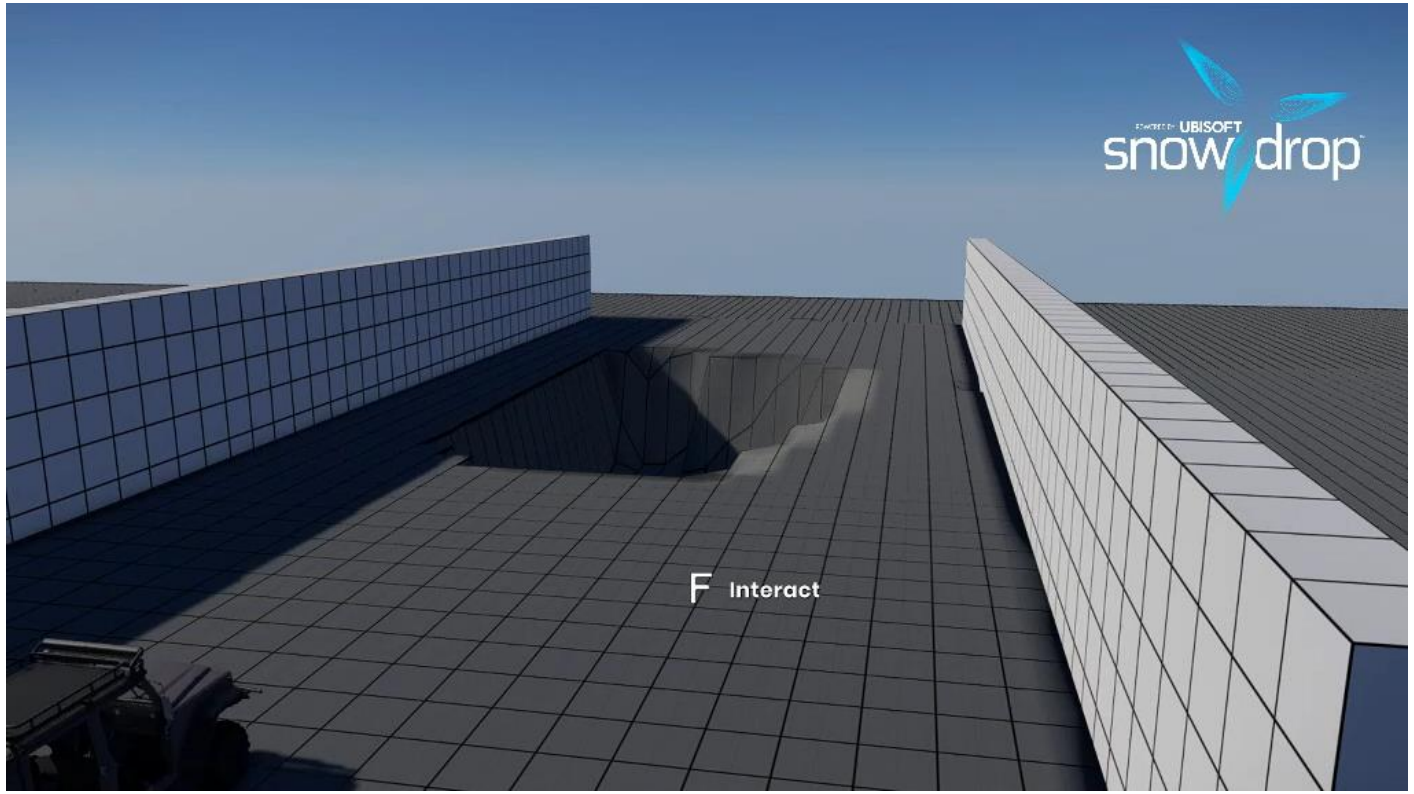
WHAT MAKES A GOOD ML PROBLEM?

You can generate or have access to a lot of data



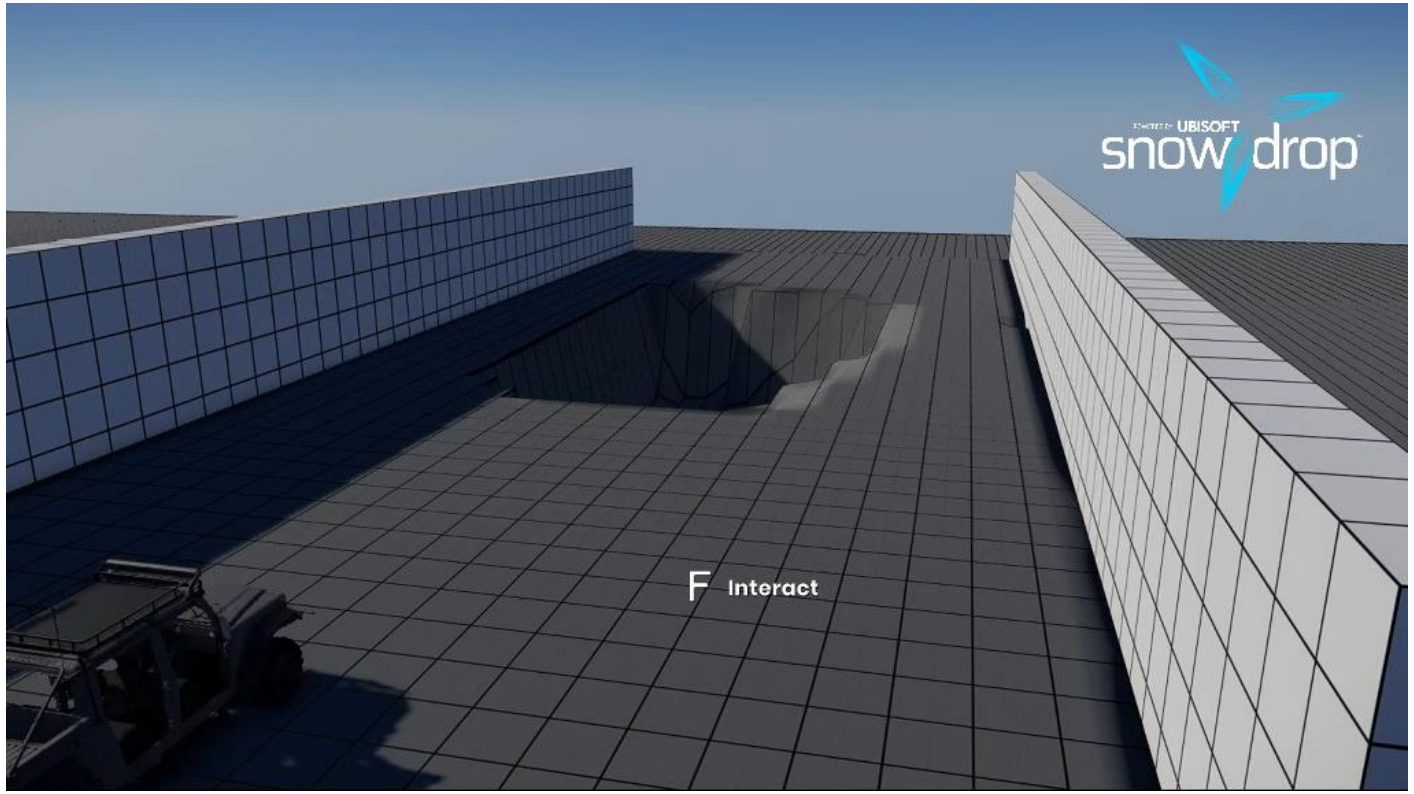
WHAT MAKES A GOOD ML PROBLEM?

For RL, your actions should strongly influence whether you achieve the goal.



WHAT MAKES A GOOD ML PROBLEM?

For RL, your actions should strongly influence whether you achieve the goal.




HOW COMPLEX SHOULD
MY SOLUTION BE?

CONSIDER YOUR REQUIREMENTS

What is the scope of my problem?

How is my solution deployed?

How often is my model taking an action?

What are the properties of my game world?

Where is my model running?

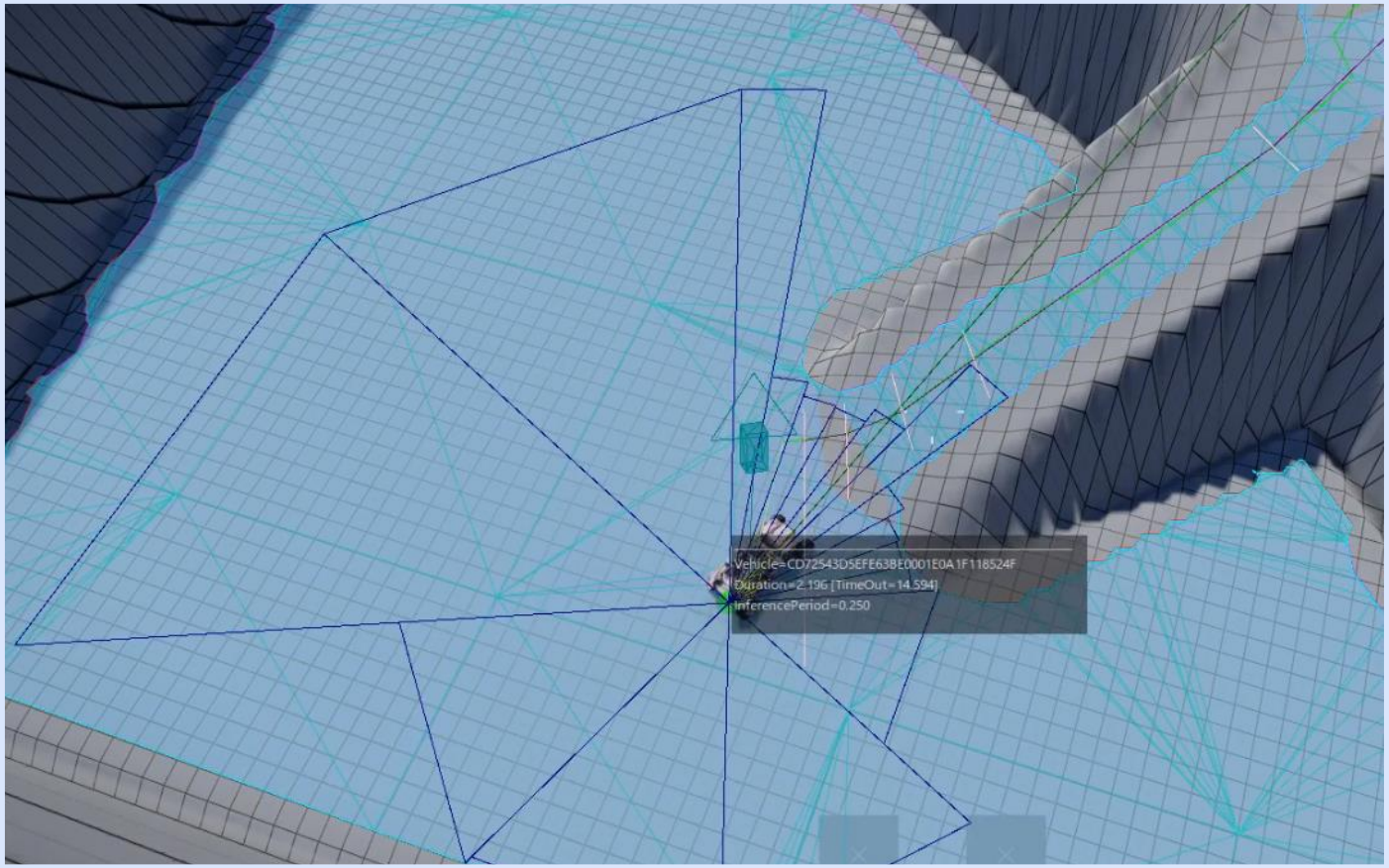
EXAMPLE

SMARTDRIVE

SMARTDRIVE REQUIREMENTS

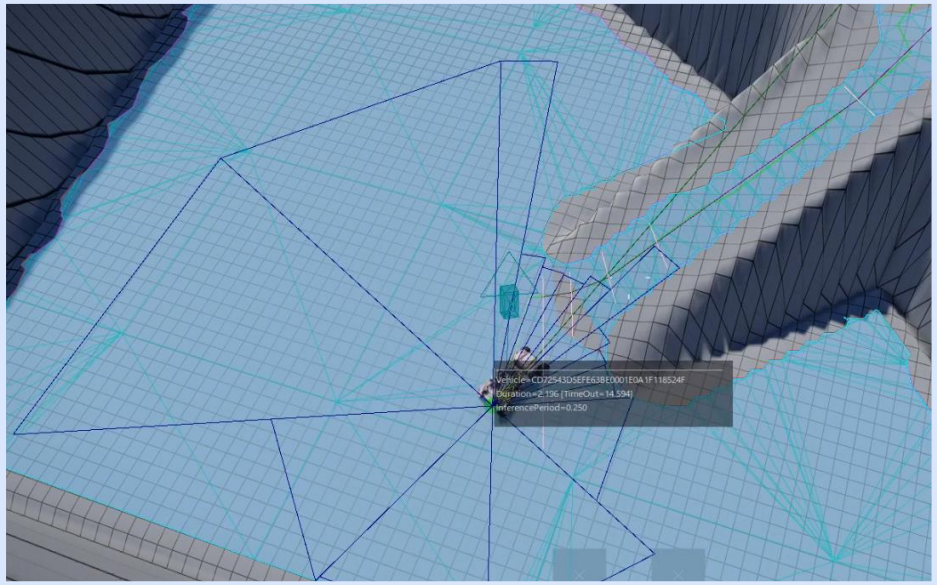
What is the scope of my problem?	• Vehicle pathfollowing and obstacle avoidance
How is my solution deployed?	• Auto-test and player-facing
What are the properties of my game world?	• Static
How often is my model taking an action?	• Several times a second

REPRESENTING OBSTACLES IN SMARTDRIVE



- Split the world around it into “quadrants”.
- Use the distance to the closest point on the nav-mesh for each quadrant.
- Also use the position and next-frame position of the 2 closest obstacles.

REPRESENTING OBSTACLES IN SMARTDRIVE



Pros

- Fast to compute.
- Compact representation.
- Small neural network is sufficient.

Cons

- Requires nav-mesh.
- Limited number of dynamic obstacles.

EXAMPLE

SMARTNAV

SMARTNAV ~~SMARTDRIVE~~ REQUIREMENTS

What is the scope of my problem?

- Pathfollowing and obstacle avoidance
  - ✚ Local pathfinding
- Auto-test and player-facing

How is my solution deployed?

What are the properties of my game world?

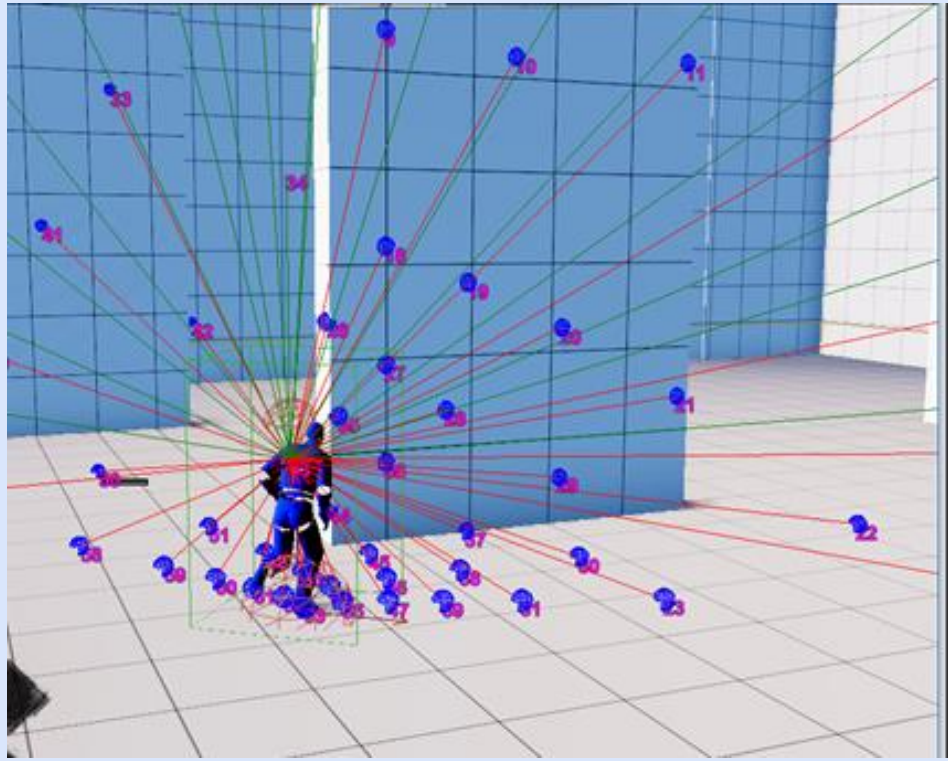
- ~~Static~~
  - Almost any world and navigation abilities

How often is my model taking an action?

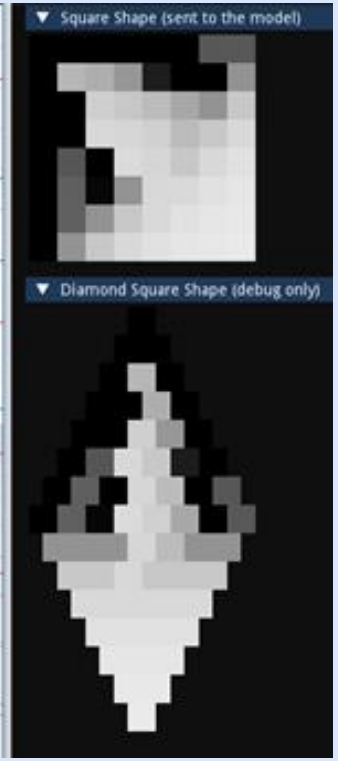
- Several times a second

REPRESENTING OBSTACLES IN SMARTNAV

- Cast rays at the physics world to generate a 2D map of it's surrounding.
- Can have multiple layers for different obstacles types.
  - Think water, destructible/non-destructible.



Casts from agent into world



Resulting Map

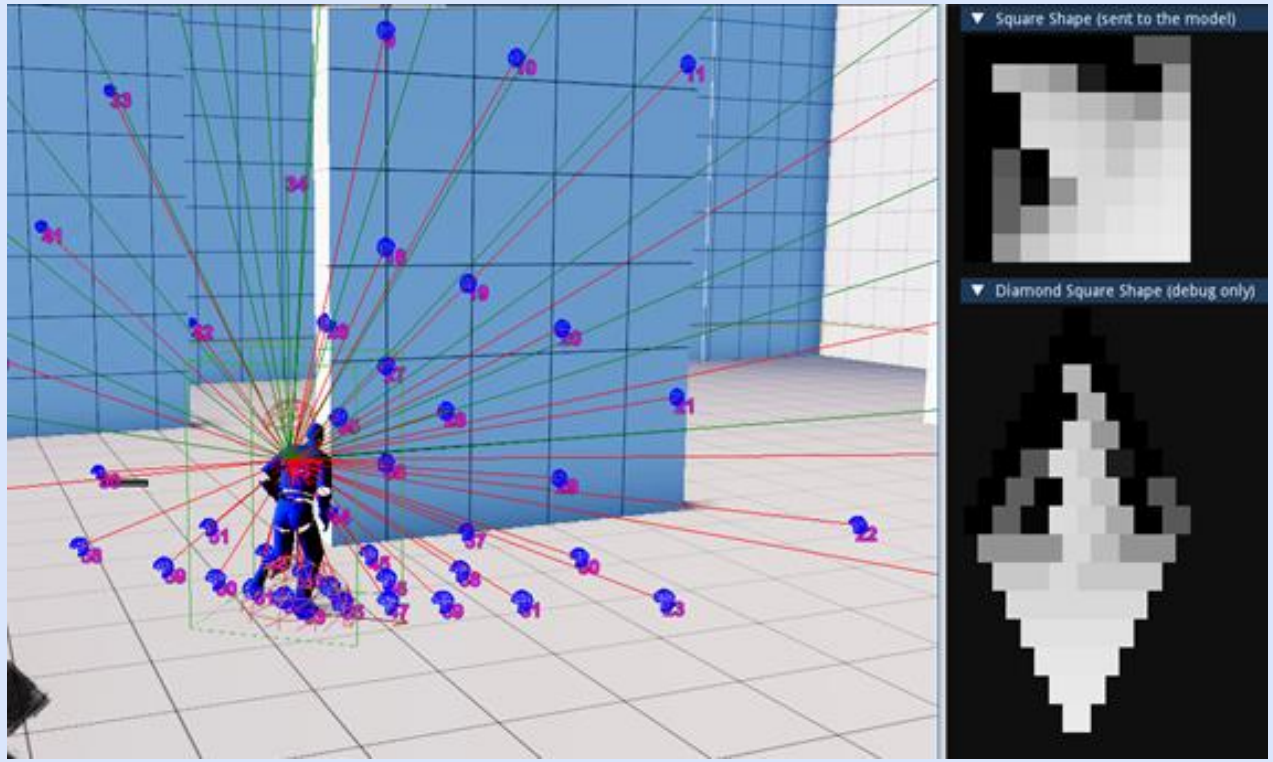
REPRESENTING OBSTACLES IN SMARTNAV

Pros

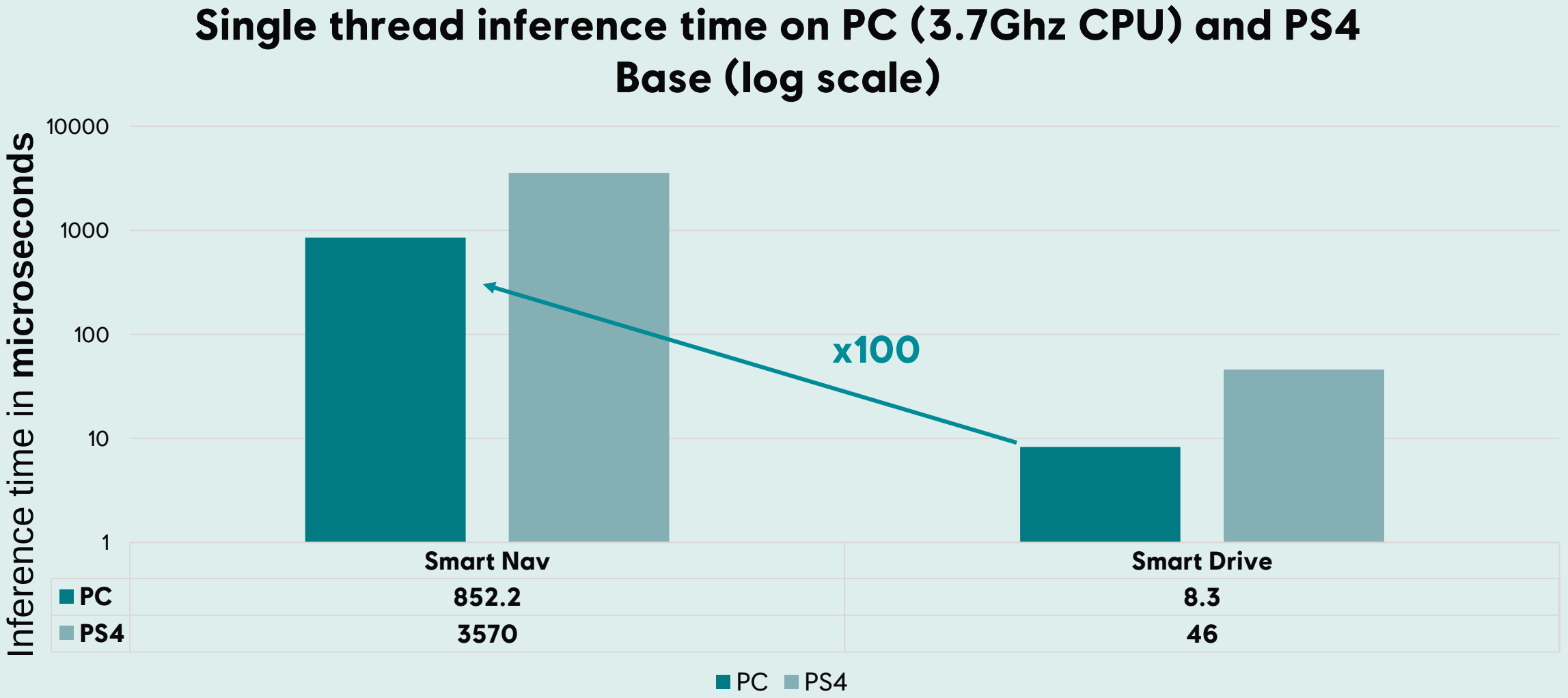
- Doesn’t require nav-mesh.
- Works in 3D.
- Only depends on physics raycasts.

Cons

- Long to compute.
- Increases model complexity and size.
- Increases learning time.



DIFFERENCE IN PERFORMANCE



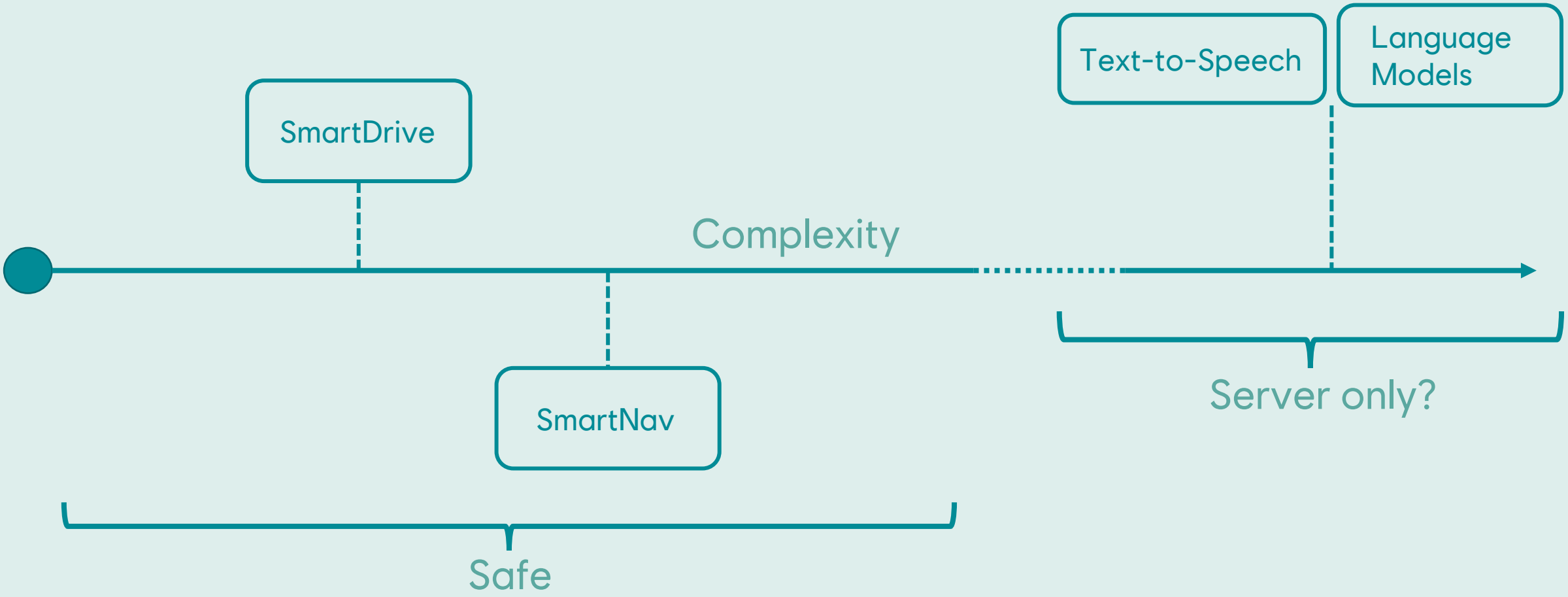
WHAT ABOUT MEMORY?

- Model loaded once for (possibly) many agents.
- Buffer for each agent.
  - Holds the inputs, intermediate calculations, and final results.

Model	Size of the model (kB)	Buffer needed per agent (kB)
Smart Nav	43 000	32
Smart Drive	357	2

Memory footprint during inference

IS MEMORY A PROBLEM?



PART 4

PUSHING YOUR PROTOTYPE FURTHER


Prototype
Production

INVEST IN YOUR ML PIPELINE

IMPROVING PERFORMANCE

- Increase time between decisions.
  - Careful, you must re-train!
  - **SmartDrive** takes decisions every 250 ms.

- Simplify the model.

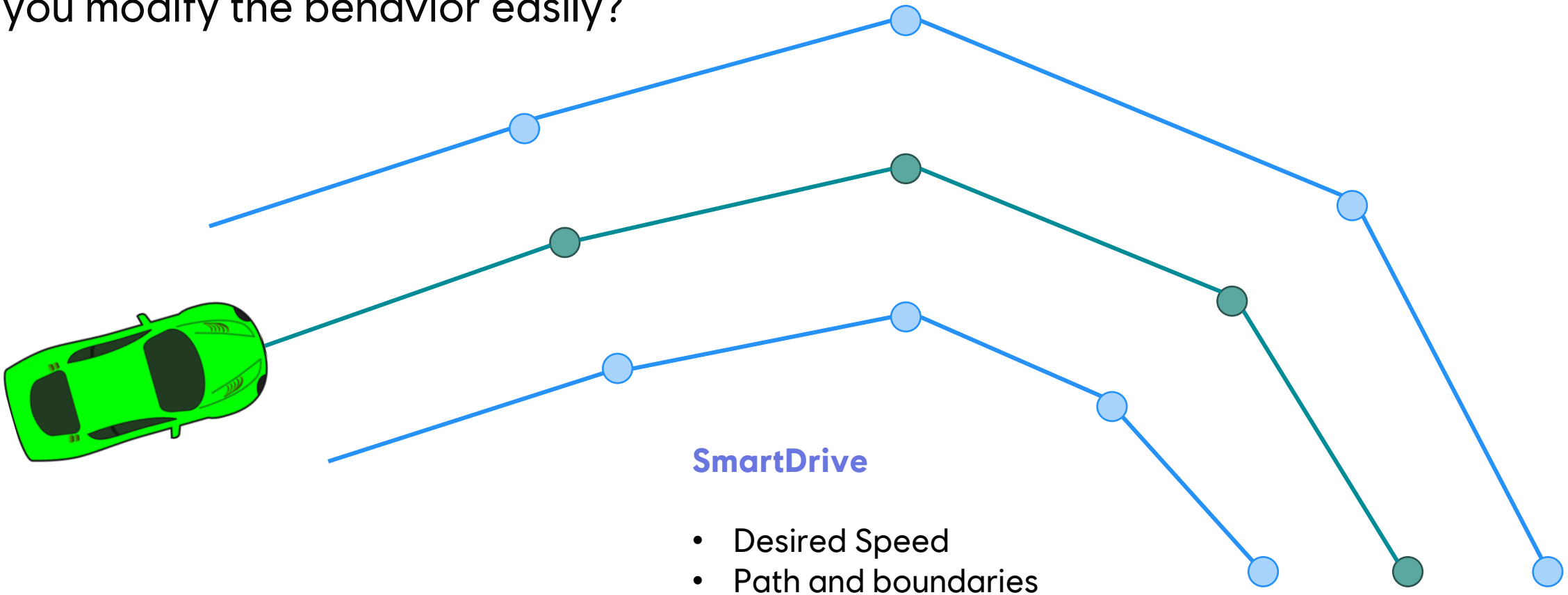
  SmartNav
  - Reduced inference time by 40% by completely removing the LSTM layer.
  - No noticeable difference in quality.

IMPROVING PERFORMANCE

Make sure all your inputs are necessary

IMPROVING FLEXIBILITY

Can you modify the behavior easily?

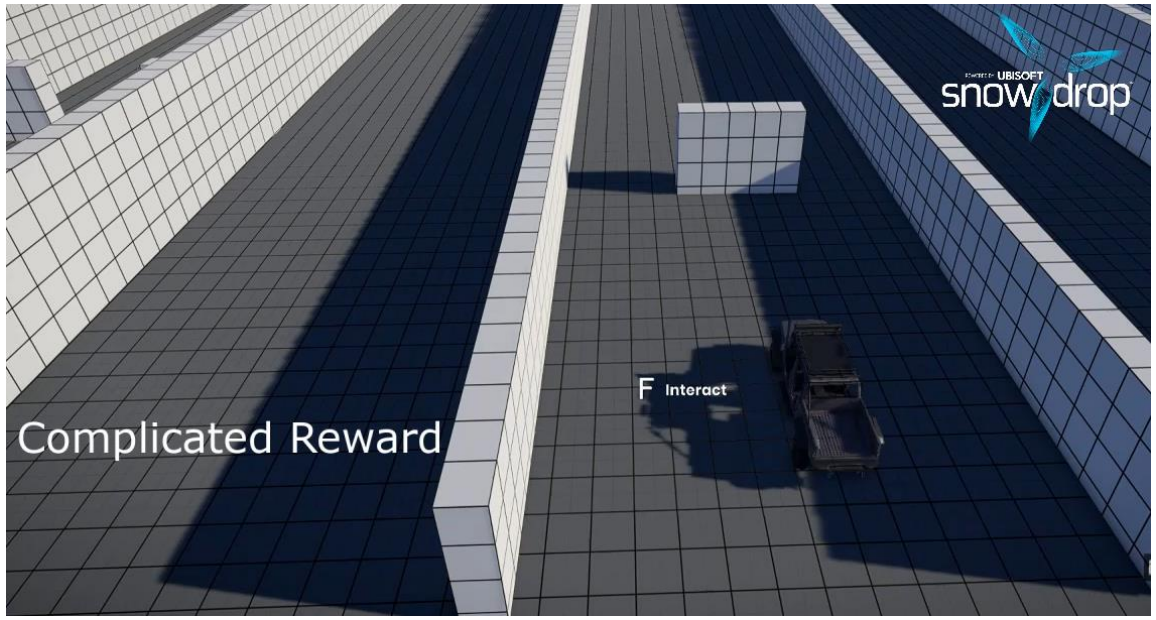


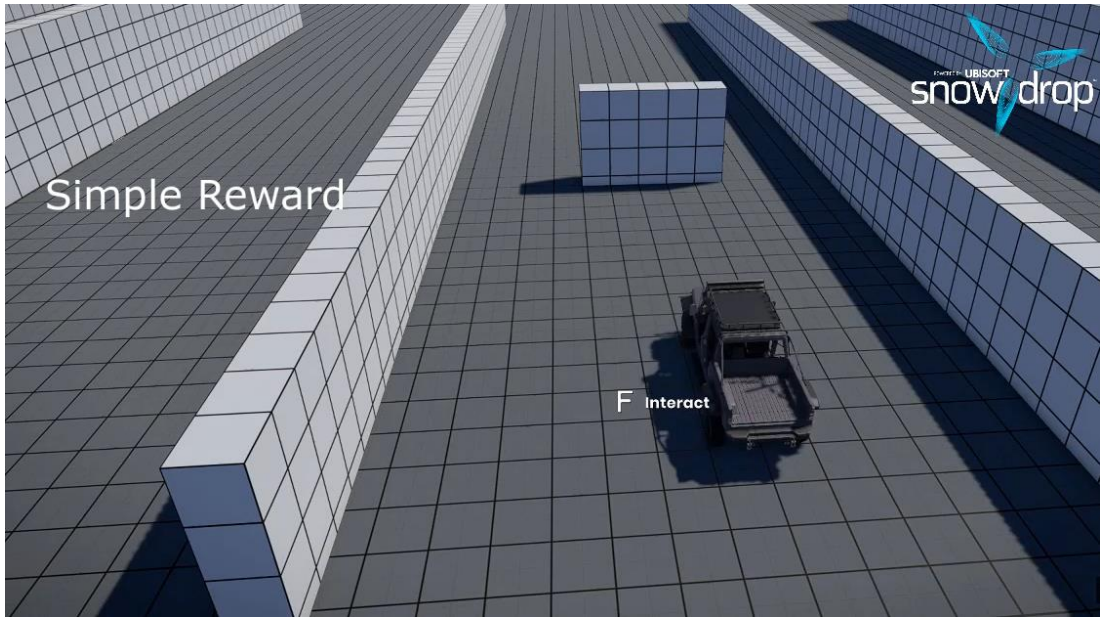
SmartDrive

- Desired Speed
- Path and boundaries

IMPROVING FLEXIBILITY

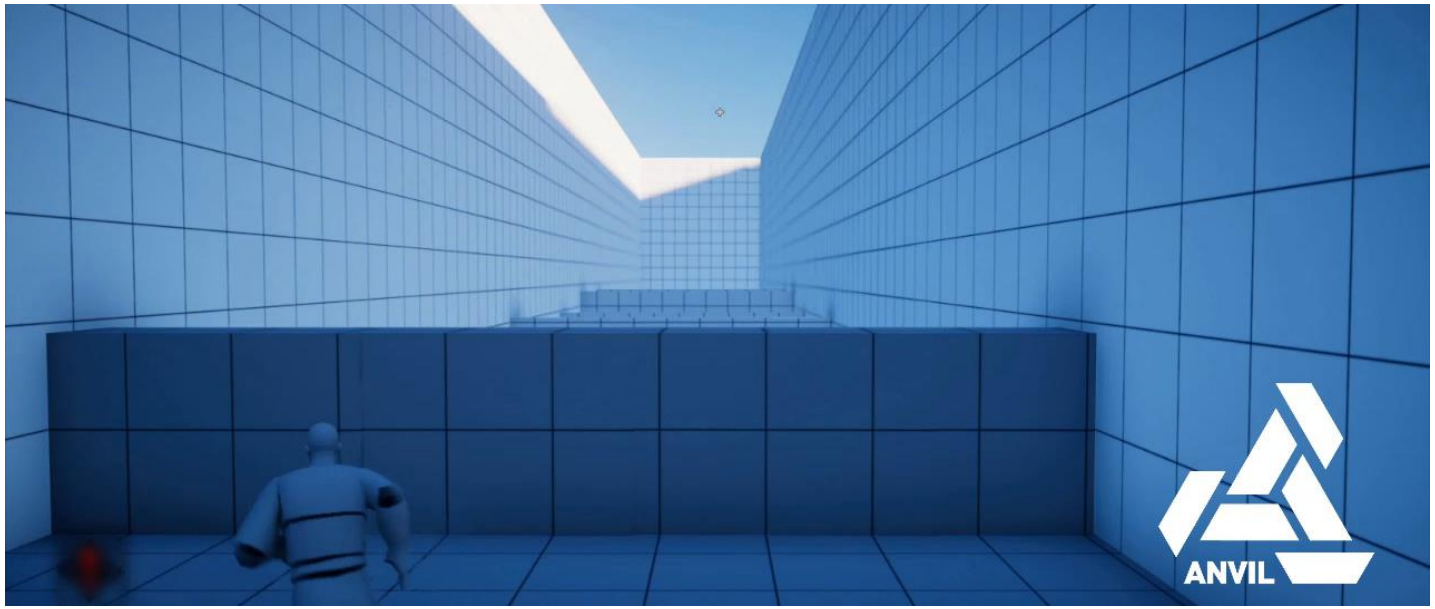
For RL applications, it is critical to have the simplest reward function as possible.





IMPROVING FLEXIBILITY

Can you easily re-adapt your solution to similar problem?



SmartNav

IMPROVING TRAINING TIME

- Run many instances of the game on a single machine.
- Run on multiple machines.
- Train on dedicated machine(s).

SmartNav Training Time

• Single machine for game and training	15 – 18h
• 5 machines for game, 3 instances each • 1 machine for training	4 – 7h


CONCLUSION

CONCLUSION

Investing in ML engineering is worth it.

Pushing to production needs both game expertise and ML expertise

Paradigm shift, change the way we design NPCs

“You tell a ML bot what to do, not how to do it”


Technology Group
UBISOFT MONTRÉAL
UBISOFT WINNIPEG
IT'S COMPLICATED
BUT WORTH IT
ANVIL
POWERED BY UBISOFT
snowdrop