Prototyping and Testing Embedded Machine Learning in a Robot

Tyler Ingebrand (Project & Machine Learning Manager)
Amy Wieland (Project Manager)
Sean McFadden (Machine Learning Manager)
Nathan Bruck (External Hardware & Arduino Manager)
Nayra Lujano (Research Manager)
Yi Ting Liew (Task Board Manager)
Chris Hazelton (Security Manager)

Advisor: Dr. Rover

Tyler

Project Vision

The goals of this project are:

- To successfully demonstrate embedded machine learning (ML) on an interesting application
- To make recommendations for incorporating embedded ML in a course for the CPR E department

Our project is open-ended!

Tyler

Project Vision

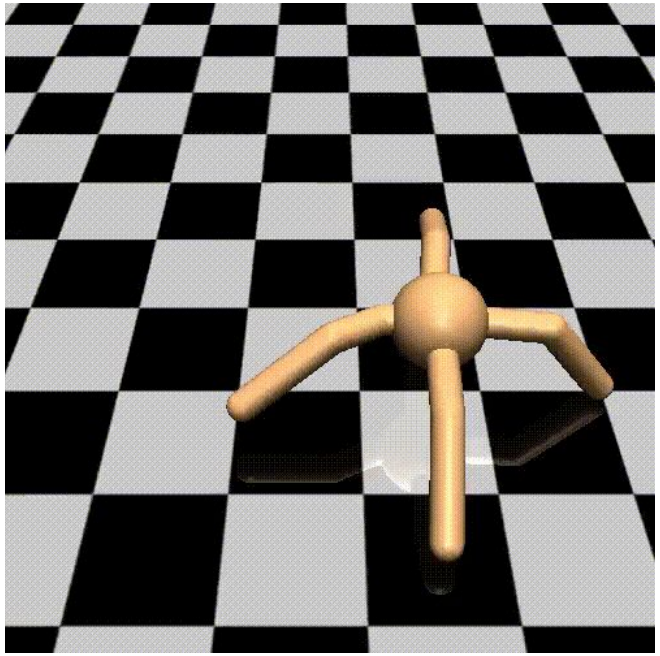
- To train a robot to walk using Reinforcement Learning (RL)
- Why a robot?
  - Because it’s cool
  - Because we could buy 10 or so of these robots to keep in the lab, similar to the Roombas in CPR E 288
- Why RL?
  - RL algorithms are general purpose - a given algorithm works for any robot
  - RL doesn’t require millions of lines of code
  - RL is robust to variance in the environment

Our choice of robot, The Petoi Bittle robot dog

Tyler

Conceptual Sketch

Virtual training in a virtual environment



A popular skeleton model called Ant v2. We will create a similar skeleton model of the Petoi Bittle to train with instead.

Apply neural network in real life (local inference)

An example video of the Petoi Bittle walking

Tyler

Functional Requirements

- Reinforcement training done virtually
  - Training is very computationally expensive and is too slow to do in real life
  - Training in real life also runs the risk of breaking a robot
- Finished NN used locally on the robot
  - The NN should be fast enough to run on the robot at real time speed
- Walk stably based on the NN
  - Travel 3ft in 20 seconds

Chris

Non-functional Requirements

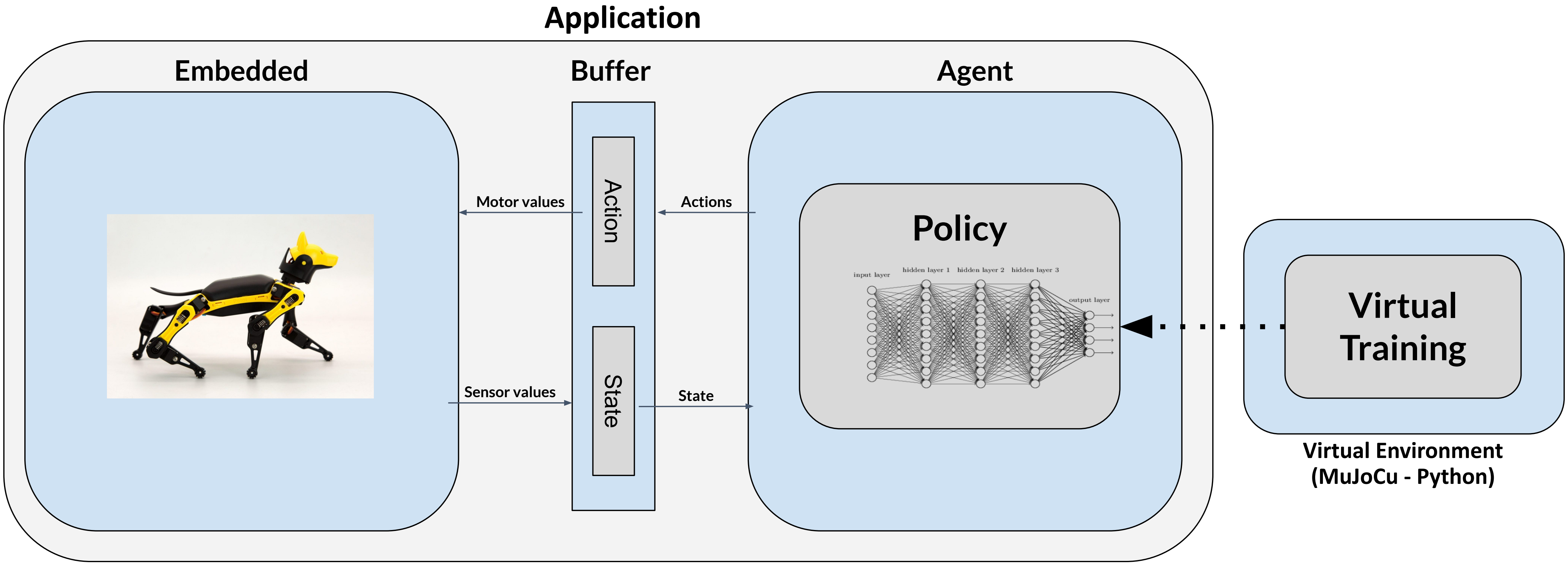
- Compile a list of useful ML resources
- Describe the requirements for implementing RL on the Petoi Bittle
- Modular development - so a future class could use some of our code and focus on the machine learning
- Use common languages (C++ and Python) for reproducibility
- Budget constraint ($600)
- Time constraint (2 Semesters)
- Free tools - OpenAI gym, MuJoCo as a 3D simulator

Chris

Use Cases for Walking Robot

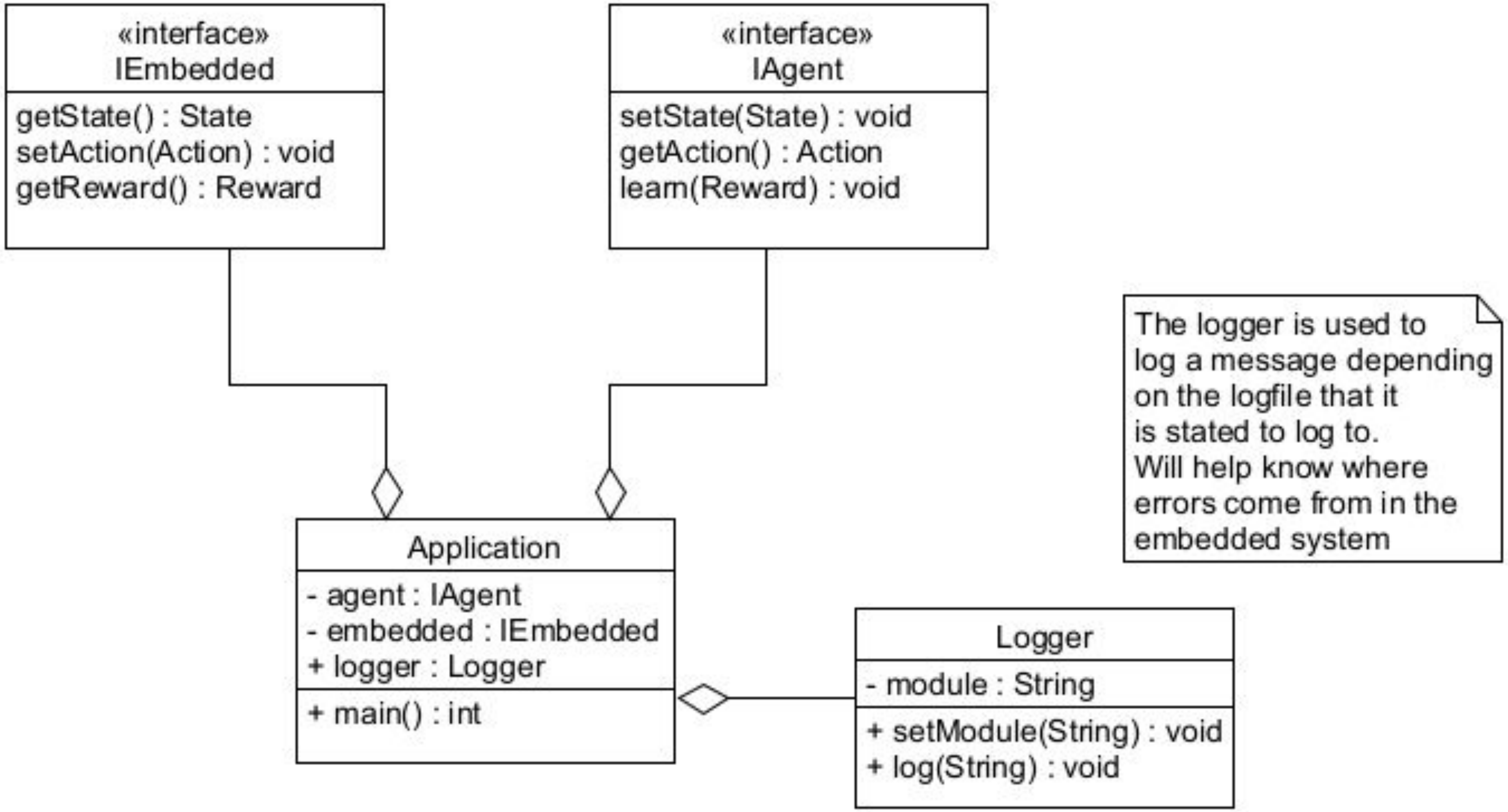
- Teachers and Students
- Search and Rescue
- Delivery Service

Conceptual Design Diagram



Amy

System Design - UML Diagram

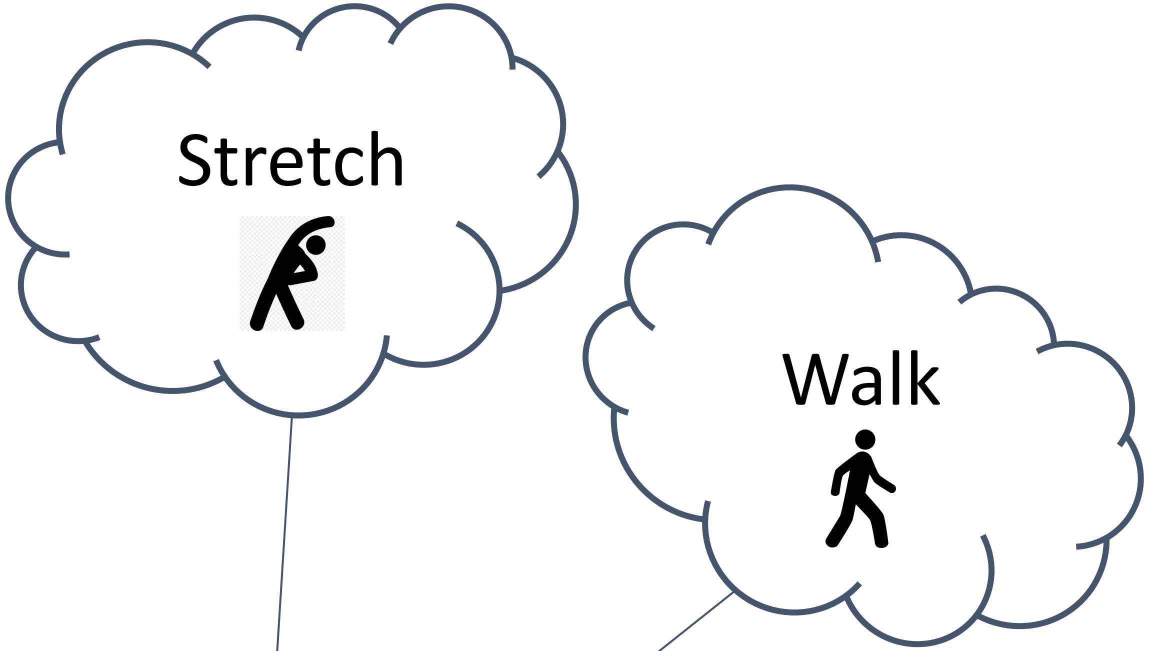


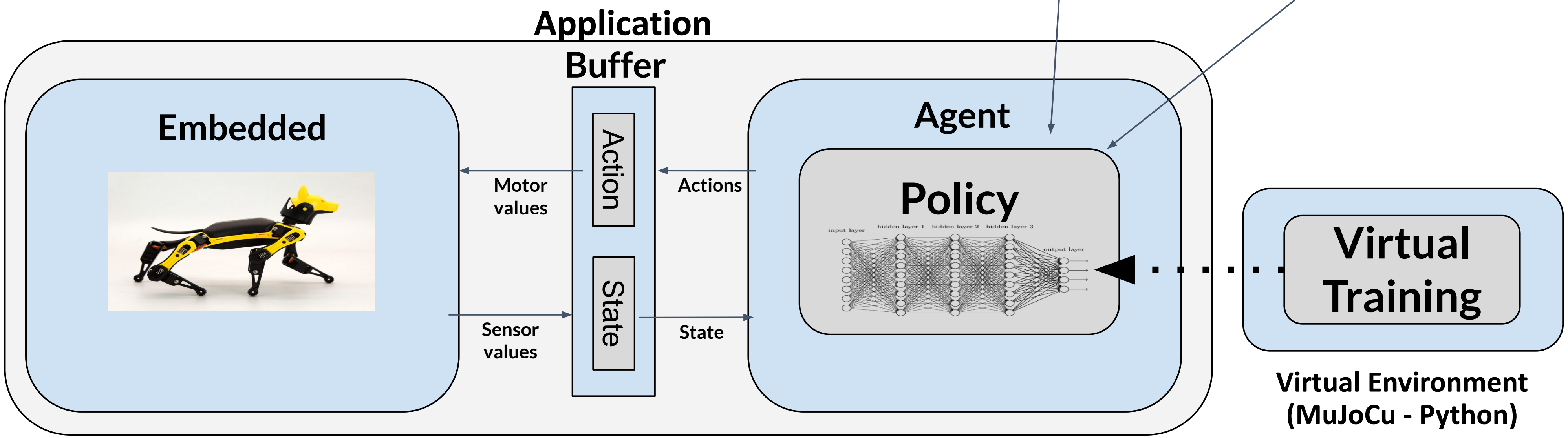
Amy

System Design Cont.

Benefit of using interfaces…

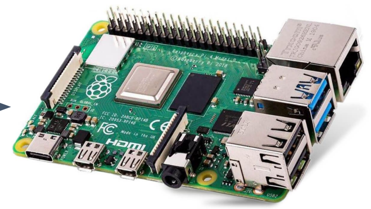
- Allows us to replace the agent with a different implementation and still use the same main application code





Amy

HW Platforms



Positional data of limbs and shell is generated by the MPU6050

All components communicate through an I2Cbus with Raspberry Pi as the master

Servos driven by the PCA9685 which provides 16 PWM 12-bit channels

System Design Cont.

SW Platforms

OpenAI

TensorFlow

MuJoCo

.predict

++

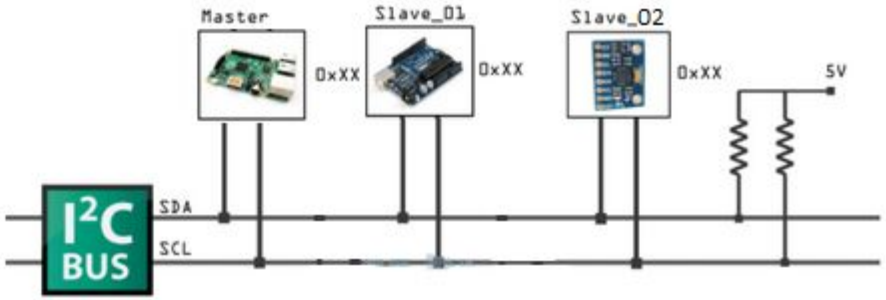
frugally-deep

++

python™

System Design Cont.

- Standards Applicable to our Project...
- IEEE P2940
  - Standard for measuring robot agility
- IEEE P1872.2
  - Standard for autonomous robots ontology
- IEEE 1725-2021
  - Standard for rechargeable batteries
- IEEE 802.11
  - Standard for wireless LANs
- I2C Bus Protocol



Nathan

Prototype Implementation

Objective: to evaluate Machine Learning approach

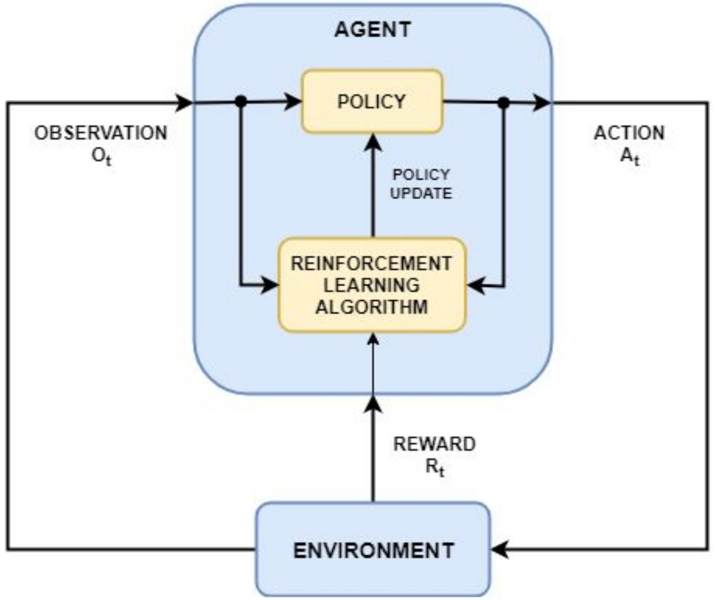
- resources for ML
- future use of robot dog
- proof-of-concept demonstration
- in-class demonstration

Nayra

Prototype Implementation (cont.)

Reinforcement Learning (Actor-Critic Method)

- NN knows nothing
- Random action
- Actor Action
  - The actor improves based on the feedback of the critic
- Critic learns
  - The critic learns by analyzing its predictions compared to the experienced results
- Actor increases accuracy



Prototype Implementation (cont.)

Things to Note:

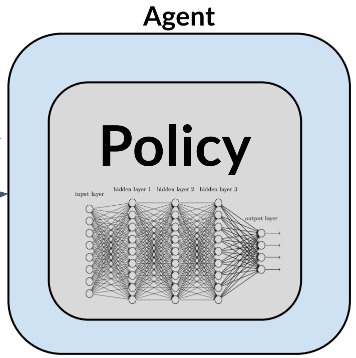
- The critic predicts next state
- The critic learns based on the reward
- Next state calculated
  - Simplified: $V_0 = V_1 + r$
- The critic adjusts
- Critic improves
- Actor improves

Design Complexity

- Unable to communicate Raspberry Pi and NyBoard on robot
    - Further investigate documentation and internet tutorials
    - Seek guidance from Dr. Rover or other faculty with I2C experience
- Virtual training environment fail to simulate
    - Adjust virtual environment
    - Train on physical robot

Project Plan

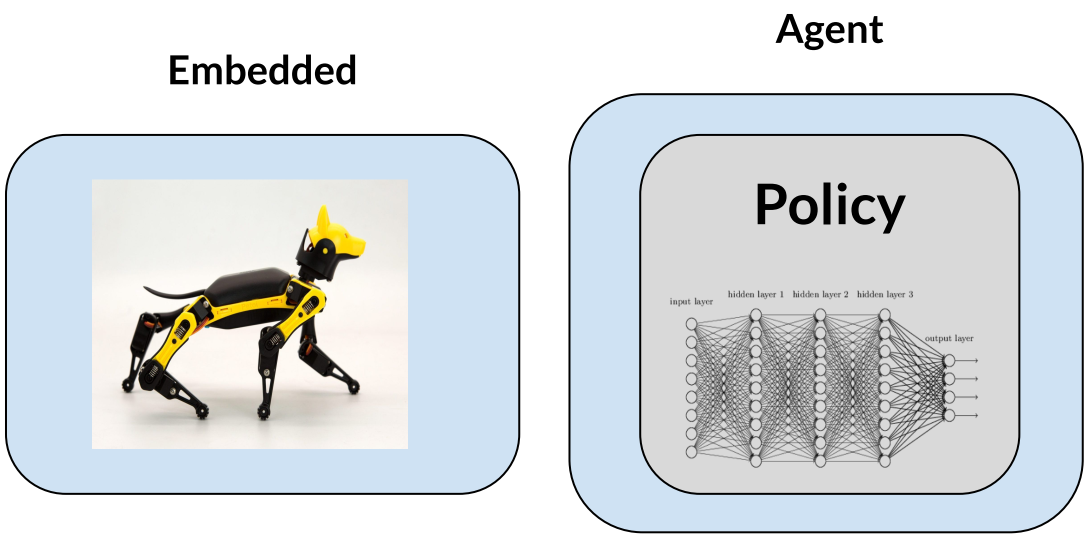
- High risk tasks:
  - Set up environment for training
    - Risk: Environment hard to set up
  - Refine virtual environment to improve model
    - Risk: More refinement is needed
  - Refine NN agent
    - Risk: More refinement is needed
  - Use model in NN agent class
    - Risk: Model setup does not align well



yt

Project Plan (cont.)

- Task: Implement embedded ML walking robot
    - Create interfaces for both the embedded and agent
    - Create general use logger
    - Implement embedded side
    - Implement Stretching agent
    - Implement NN agent



Milestones

M1. Assemble robots and run factory code

M2. Complete “Mountain Car” problem

M3. Move servos on robot through Pi

M4. Train virtual robot to walk 3ft within 20s

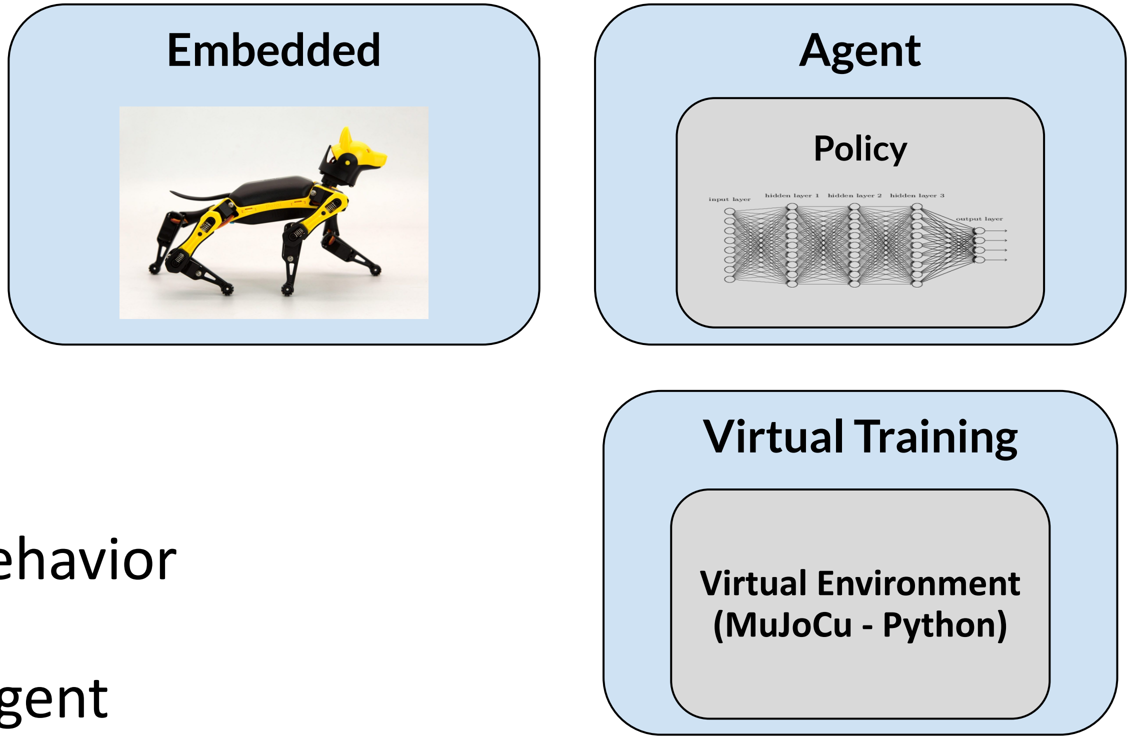
M5. Deploy model onto physical robot

Schedule

TASK	Semester 1												Semester 2																	
	October				November				December				January				February				March				April				May	
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Linux running on pi			X																											
Define the Action and State structs			X																											
Create interfaces for both embedded and agent				X	X		M1																							
Assemble robots and run factory code						X	X		M2																					
Complete "Mountain Car" problem						X	X	X	X																					
Create general use logger									X	X																				
Create virtual dog robot in MuJoCo									X	X	X	X					M3													
Set up environment for training															X	X	X													
Implement embedded side															X	X	X	X												
Implement stretching agent																		X	X				M4							
Create/train a model to walk using our robot in virtual environment																		X	X	X	X	X	X							
Implement neural network agent																							X	X	X		M5			
Use model in NN agent class																									X	X	X			
Refine virtual environment to improve model																												X	X	X
Refine NN agent to improve performance																												X	X	X

Sean

Test Plan (1/2)



- Unit tests
  - Bittle robot
    - Scripts to test all servos
  - Python environment
    - Render environment and inspect behavior
  - NN agent
    - Create test NN and apply it in the agent
- Integration tests
  - Communication between Pi and microcontroller on robot
    - Script on Pi to move servos by sending commands to microcontroller
  - Loading model onto Pi
    - Compare output of each model when given the same inputs

Sean

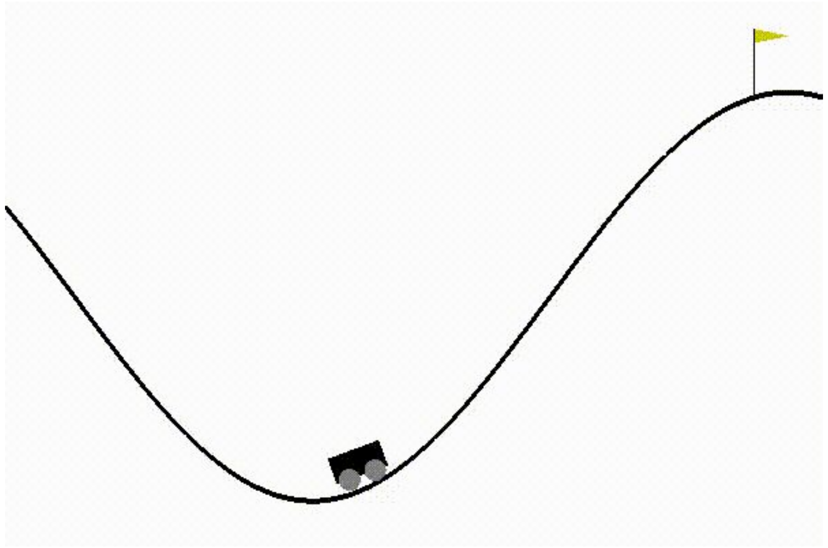
Test Plan (2/2)

- Regression tests
  - Run unit tests and integration tests after new features are added
  - Critical features
    - Movement of robot’s legs
    - NN agent
- Acceptance tests
  - Present modules that mirror each stage in development process for a classroom setting
  - Robot travels 3ft within 20s

Sean

Current Progress

- Research
  - Reinforcement Learning and embedded ML
  - Agent Implementation
- Robots assembled and functioning
  - Raspberry Pi 3B mounting standoffs
- Mountain Car training
  - Deep Q Network (DQN)
  - Deep Deterministic Policy Gradient (DDPG)



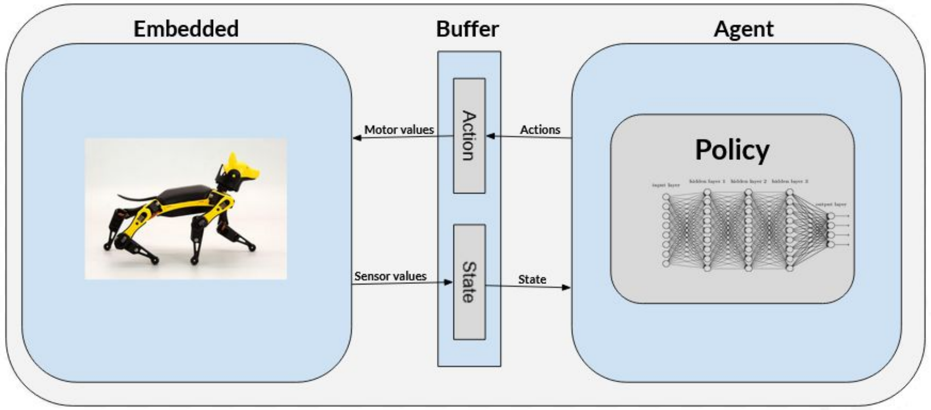
A learned solution to a (simple) mountain car game

Nathan

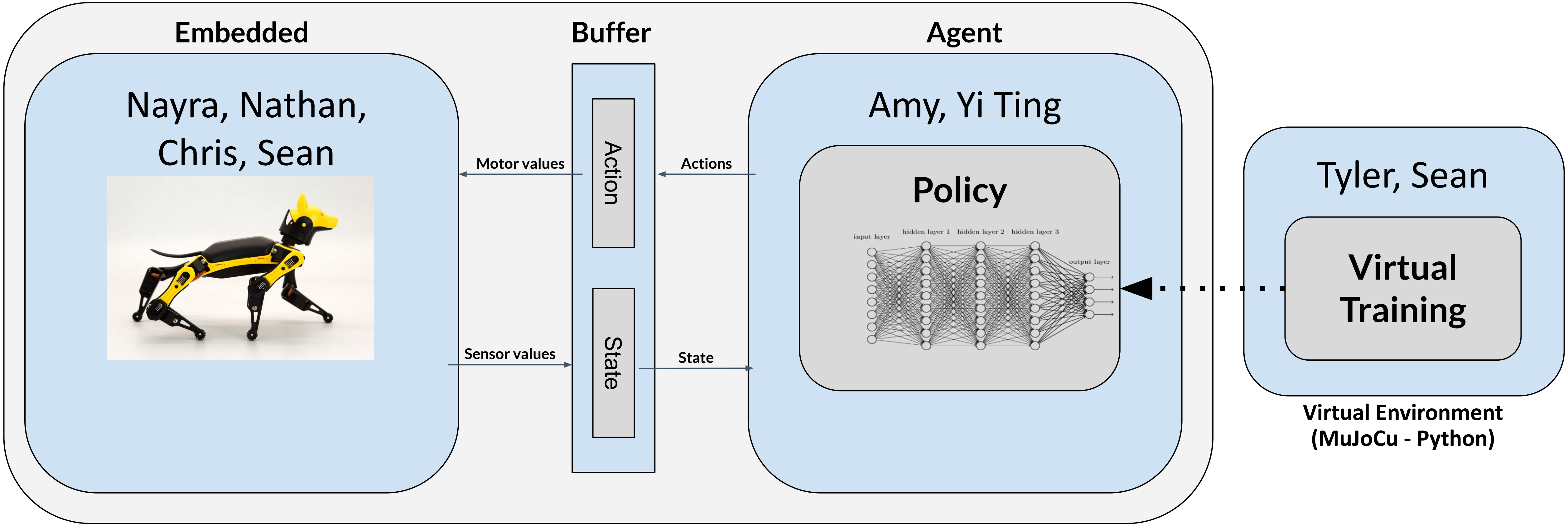
Bittle Demo

Next Semester's Plan

- Embedded systems
  - Establish communication Pi → microcontroller
  - Create unit and integration tests
- Reinforcement learning
  - Apply research to our robot application
  - Finish Mujoco model of robot
  - Train virtual robot in virtual environment using DDPG
- Final steps
  - Deploy model and agent onto robot
  - Fine tune virtual environment



Individual Contributions



Nathan

Thank you for your time.
Questions?