Initial Project Document

Envoy Commander

UNIVERSITY OF
CENTRAL FLORIDA

Department of Electrical Engineering and Computer Science

Samuel M. Richie, Dr. Lei Wei
Senior Design 1

Group 32

Yash Gharat	Computer Engineering	yash.gharat@knights.ucf.edu
Andrew Cuevas	Computer Engineering	cuevas.andrew@knights.ucf.edu
Anthony Soffian	Electrical Engineering	antsoffian@knights.ucf.edu

Project Customer/ Advisor/ Contributor

Dr. Chinwendu Enyioha

https://cenyioha.eecs.ucf.edu/
cenyioha@ucf.edu

Project Narrative

One of the most classic reinforcement learning problems is that of the multi-armed bandit problem. In this problem, we have some learning agent that supposedly has multiple arms and is situated in front of some number of slot machines. Each slot machine is given its own distribution of success that is unknown to the learning agent. When each lever is pulled on a slot machine, the learning agent is given either a reward of 0 or +1 for failure or success, respectively. The learning agent wants to maximize the rewards it receives from the machines, without knowing exactly when each of the machines will result in a success. There are many solutions to the multi-armed bandit problem, but many of them are just simulations. In the Envoy Commander, we want to investigate solving a more complicated version of this problem using a real-world learning agent and a simple game.

The goal of the Envoy Commander is to create an intelligent learning agent that can seamlessly communicate with one or more dummy field agents to optimize a received reward when completing a simple task or playing a game. The learning agent should be able to add more field agents in as though they are plug-and-play and coordinate all of them to either complete the task better (receiving a better reward) or faster. To do this, it will act as a virtual commander to the field agents, using the observations they make through various sensors such as ultrasonic, photoresistors, and bumper switches to decide on the appropriate next action to take while maximizing its rewards. These sensors, as with any, come with some noise and uncertainty in measurements which are usually left unconsidered by using thresholds that will give a binary response (detected/not detected) and minimize the amount of false-positives or false-negatives. However, our learning agent should learn/estimate these uncertainties and use them as factors in its decision making (i.e. the most accurate sensor is taken as a priority). There might also be some uncertainty in the communication between the learning agent and the dummy agent and it should account for this as well.

To further define the aforementioned “learning agent”, we are expecting a microcontroller with a set of sensors that will be able to understand/track the progress of the field agents but only rely on their sensor observations to take actions from state to state. The knowledge of the progress will be used to determine the reward it receives and to know when it has reached the goal state. The learning agent will also need to wirelessly communicate with the field agents to receive state observations and send actions. With this, comes the problem of concurrency with messages, this will have to be handled efficiently such that the learning process and overall progress towards the goal is not hindered. The learning agent should also have enough computing ability to implement a reinforcement learning algorithm, which will be along the lines of Q-Learning, without too much strain as it will still need to continue to monitor/execute the other functions simultaneously. On the other hand, the dummy field agents only need limited capabilities, depending largely on the chosen task/game. For example, if the game were a maze, the dummy agents would only need to move in four directions and make observations with simple sensors. However, if the game were to sort colored blocks, the agents would need

a camera to distinguish the various colors and be able to construct a color mask and send those values to the commander, making it more complicated. In all situations, the field agents require the ability to wirelessly send/receive messages to their commander.

The learning agent will employ Q-Learning to learn from its environment. Q-Learning is a specific reinforcement learning algorithm in which the learning agent maintains a Q-Table with a value for each state-action pair. The Q-value (Quality) is dependent on the reward received for that state-action pair. In other words, it keeps track of the reward received for a specific action taken from a specific state. When the agent encounters the same state again, it will choose the action that previously gave the highest reward and continue on. However, using hyperparameters (parameters that tune the learning process), we can have some chance that the agent instead takes a random action so that it can explore other possibilities and not always be stuck in the same path. Without this probability, the agent is likely to always take the value it chose first, even if that was a random choice. The more states and episodes the learning agent endures through, the more it learns and optimizes the way it completes the task and plays the game. But since the algorithm is so reward-dependent, it is also important to choose a reward scheme that will push the learning agent towards efficiently and intelligently completing the goal instead of just trying to finish. A good example is the multi-armed bandit problem discussed earlier. Which only gave a reward with a successful lever-pull. Since it was trying to optimize the rewards from the slot machine, just choosing the best machine is useless if it did not give a winning pull. The agent is therefore not rewarded for just guessing the right machine. It has to choose the right machine and get a successful result in order to be rewarded, there is no partial credit.

Our supervisor/customer, Dr. Chinwendu Enyioha also has some suggestions regarding the overall implementation of this project. The primary one to be considered is the notion of a dynamic/stochastic environment. Since a real-world application for a learning agent does not necessarily guarantee a uniform, constant environment, it should not be given one. The learning agent should be able to finish the task with a very general set of constraints such that the design of either itself or its field-agent subordinates need not be changed. For example, just moving around/changing the maze would be ok, but adding a button to open a gate would not, since the field agent would need something to press the button with. The frequency of the environment changing and the extent to which it is changed can also vary.

While the Envoy Commander may not seem to have an immediate application, the techniques used in the solution can come into play in many settings. Package-delivery robots, militaristic strategy, and even simple daily tasks such as vacuuming. The key goal is to have a centralized, intelligent, learning agent that will coordinate other, unintelligent, field agents to perform a task in the most efficient way by using a combination of reinforcement learning, wireless communication, and sensor observations while being able to plug-and-play other field agents constrained only by the computing power of the central learning agent. Through episodic tasks and stochastic environments, the main agent should maximize its rewards and continue to learn without much help, if any at all, from a human agent.

Requirements Specifications

Specifications

Specifications are split into three main categories, which will be the main components of the project and are based on a modular but collaborative design. If the goal to be accomplished by the field agent should change, so too would the arena and learning agent(s) modify their respective requirements

Learning Agent Requirements

Table 1: Learning Agent Requirements

1.1	A sufficiently intelligent artificial intelligence able to solve the problem given to the agent
1.2	Sufficient memory able to hold the intelligence contained within the agent's microcontroller
1.3	Sufficient random access memory to support the active decision making of the learning agent (> 3 GB of RAM)
1.4	Sense the environment around it
1.5*	Support short-range (< 50 ft.) communications with external agents*
1.6**	Carry and utilize a low power energy source**
1.7	Consist of less than 50% of the total project budget (< $200)
1.8	Higher level language implementation, that is translatable into lower level for agent operation
1.9	Responsive, and display status outside of the program being modified
1.10	Communication with the dummy (field) agent(s)
1.11	Transformation of external stimuli into usable data for internal action

*The communication medium may eventually become wireless, which would therefore be added on as a requirement descriptor for 1.5
**The power train of the learning agent may or may not be wireless, of which description would be added on to requirement 1.6

Field Agent Requirements

Table 2: Dummy Agent Requirements

2.1	Sensors that are able to observe the environment constructed
2.2	Support a variety of sensors including but not limited to ultrasonic sensors, bumpers, and photoresistors
2.3	Operation in at least a 2-dimensional place of motion
2.4	A reasonable and small size, to scale of the arena
2.5*	Support short range communications
2.6**	The energy consumed by the field agent must be low
2.7	Cost a maximum of 25% of the project budget (< $100)
2.8	Must be able to communicate with the learning agent
2.9	The field agent must be able to communicate with the project supervisors and demonstrators while in use

*The communication medium may or may not become wireless for this specific agent and component of the project. The requirement will change accordingly.
**The power supply may or may not be wireless, which would change the standard for low-power where the former would require a much lesser power supply than a wired one would allow.

Arena Requirements

Table 3: Environment Requirements

3.1	Modularity to allow for a dynamic and stochastic arena
3.2	Reasonable weight, which may scale with the size of the agents in question. (< 50 lbs for 5x5 ft^2)
3.3	Stimuli to interact with the respective agents
3.4*	Structurally sound, and withstand reasonable impacts from the moving robot*
3.5	Implement a game in conjunction with the field agents
3.6	Sufficient information for the learning agent to accomplish its goal
3.7	The arena must cost within 25% of the final project budget

Potential Project Constraints

The three main constraints we are facing are technological, budget-based, and time, with the latter acting as the most significant in our case as a majority of our project software-oriented requires more detailed effort into its execution as well as the testing of the minor electrical components that we would utilize. Since we are shooting for a fairly low budget with our minimized group this will end on the lower section of constraints, but this also dampens our potential technological uses as we strive to build the simplest robot to carry the bulk of our software elements.

Project Block Diagrams

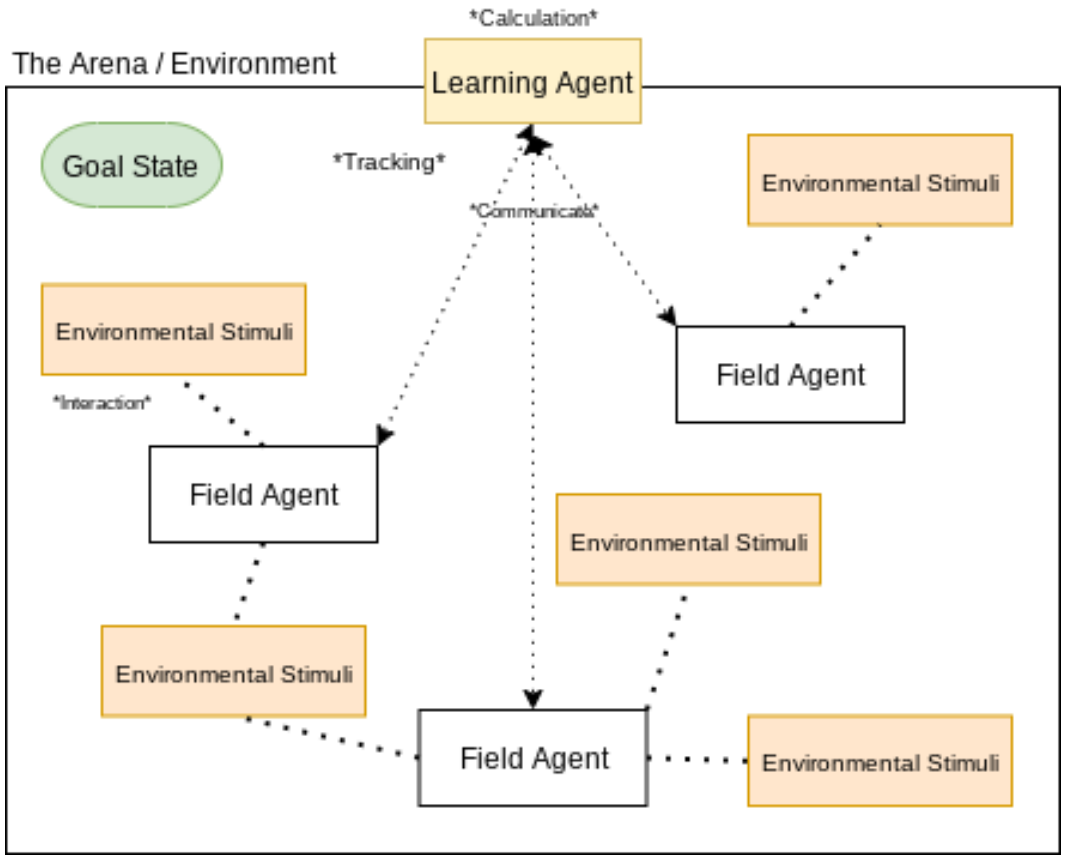


Figure 1: Conceptual Block Diagram (With Multiple Field Agents)

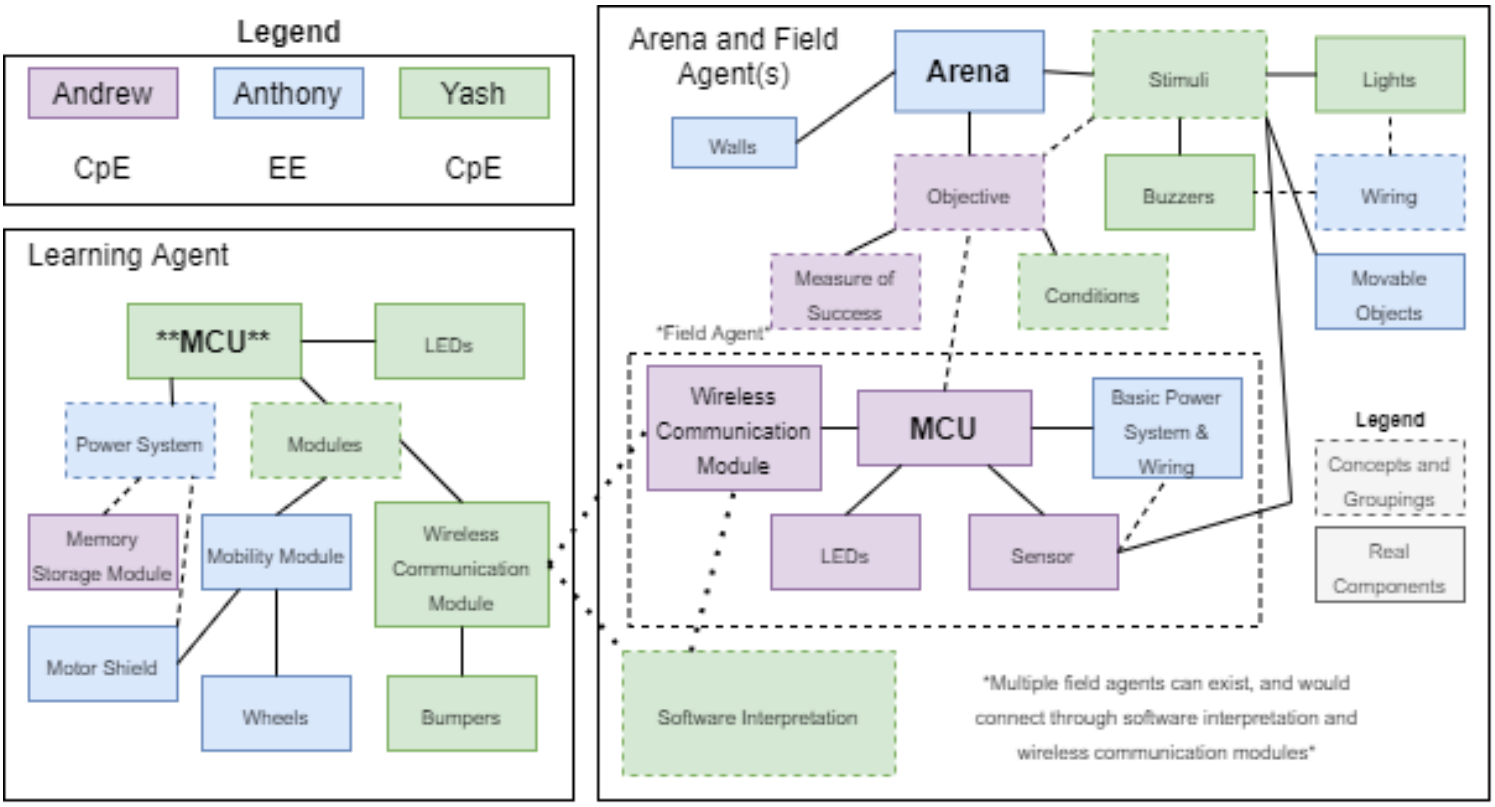
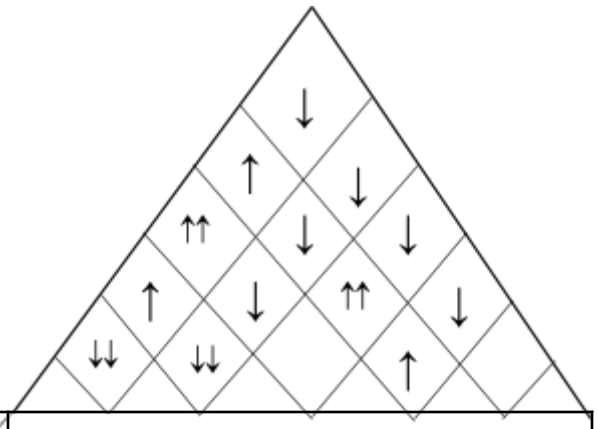


Figure 2: Design and Responsibility Block Diagram

House of Quality



			Engineering Requirements					
			(1) Memory	(2) Cost	(3) Communications	(4) Dimensions	(5) Sensor Quality	(6) Power Use
			+	-	+	-	+	-
Marketing Requirements	(1) Cost	-	↓↓	↓↓	↓↓	↓	↓	↓
	(2) AI Ability	+	↑↑	↓↓	↑↑	↓	↑↑	↓
	(3) Game	+	↑↑	↓	↑↑	↑	↑	
	(4) Power Eff	+	↑	↓↓	↓			↑
	(5) Integrity	+		↓		↑		
Engineering Requirements Parameters			(1) >3 GB RAM	(2) $300 - $500	(3) <50ft Range	(4) 5x5 ft^2 to Scale		

Project Budget

Parts List	
Description	**Price Estimate**
Raspberry Pi Model 4	$35 - $50
Various minor printed circuit board components (communication and power)	$20
Raspberry Pi Compatible Wheels	$10 - $20
Raspberry Pi Compatible Sensors (Ultrasound, Lidar, Photoresistors, Bumpers and Buttons)	$50 - $60
Raspberry Pi Compatible Power Source (Battery casing and motor shield if necessary)	$20 - $30
Learning Agent Chassi (PCB compatible)	$5 - $15
Wireless Modules	$20 - $60
Storage Modules	$10 - $20
Various Basic Microcontrollers*	$10 - $20 ea.
Various Power Casings and Batteries	$10 - $30
3D Print material (PLA)	$15 - $20
Various LED and circuit response components	$10 - $20
Arena materials (Structural)	$20 - $30
Arena Components (Lights, power supplies, stands, etc.)	$30 - $40
Arena communications (buzzers, lights)*	$20
Estimated Cost	$295 - $505

*Depends on field agent functionality

Project Milestones

Senior Design 1			
Milestone	**Dates**	**Duration**	**Status**
Project Selection	8/26 - 9/9	2 weeks	Completed
Divide and Conquer	9/9 - 9/16	1 week	In-Progress
Divide and Conquer 2.0	9/16 - 10/1	2 weeks	In-Progress
Table of Contents	10/22	3 weeks	Not Started
60 Page Draft	11/5	4 weeks	Not Started
100 Page Draft	11/19	6 weeks	Not Started
Finalized Document	12/7	9 weeks	Not Started
Order Parts	TBD	TBD	In-Progress
Senior Design 2			
First Prototype	TBD	TBD	Not Started
First Critical Design Review	TBD	TBD	Not Started
Peer-Reviewed Presentation	TBD	TBD	Not Started
Conference Paper	TBD	TBD	Not Started
First Demo	TBD	TBD	Not Started
FInal Demo	TBD	TBD	Not Started
Final Presentation	TBD	TBD	Not Started