Boston University

Electrical & Computer Engineering

EC463 Capstone Senior Design Project

User's Manual

Pollux

by Team 22

Wenhao Cao wenhaoc@bu.edu
Karl Carisme karlc@bu.edu
Louis Cicala cicala@bu.edu
Noah Hathout nhathout@bu.edu
Caelan Wong caewong@bu.edu

Table of Contents

Executive Summary

Maintaining cleanliness in the kitchen is a priority for many, but deep cleaning and sanitizing can be time consuming and involve the use of harmful chemicals that may linger in the environment.

Pollux will be a self-operating countertop cleaning robot, capable of cleaning and disinfecting surfaces without the use of harsh chemicals. Utilizing UV light for sterilization, it will offer a safe and effective alternative to traditional cleaners.

Pollux will autonomously navigate countertops, avoiding obstacles and detecting edges. It will be equipped with UVC diodes capable of killing 99% of germs, and a gyroscopic failsafe mechanism to ensure users can never be exposed to this UV light.

Pollux will provide a convenient, hands-free method to maintain cleanliness, not only in kitchens but on any surface where sanitation is a priority. Ultimately, the Pollux aims to reduce the time and effort users spend on cleaning and absolve users of the usage of harmful chemicals.

1 Introduction

Everyone desires cleanliness in their homes, and nowhere else is it as important than in the kitchen. Kitchen surfaces like tables and countertops regularly come into contact with raw or undercooked food, so sanitation becomes imperative once it is understood that these pathogens can cause serious foodborne illnesses.

However, deep cleaning something like a kitchen countertop is an involved task – cleaning at more than face value requires time and consideration. It is easy to wipe down a countertop so it looks clean, but this does not make it truly clean. Unless germs are killed, they may persist, which is highly undesirable in a kitchen.

There is a large consumer market for cleaning products advertised to kill such germs and to make the cleaning process more efficient and effective. However, users may not be aware of the underlying health risks associated with these chemical cleaners. A June 2024 study, "Cleaning Products…" by Salonen et al., found the following:

> "The use of cleaning products indoors may increase occupants' exposure to a variety of harmful environmental pollutants including volatile and semi volatile organic compounds (VOCs and SVOCs), particulate matter (PM), and nitrogen dioxide…
>
> household cleaning agents are sensitizers or airway irritants, and have been associated with impaired respiratory health among children and adults…
>
> use of products in spray form at home can increase asthma incidence, current asthma, and poorly controlled asthma in adults and wheezing in children" [1]

Considering these issues, there is not only a financial incentive to provide consumers with a time-saving solution to properly clean surfaces, but also legitimate health concerns over how the cleanliness-minded can safely clean surfaces.

Therefore, we built the Pollux: an autonomous countertop cleaning robot that disinfects using UV light. The Pollux saves users both time and effort by cleaning surfaces completely independently, with no input required other than starting the device. Additionally, the Pollux solves the problem of hazardous sanitation products by using a UV light instead of any chemicals and by maintaining separation between the user and the light.

Over the past two semesters, we have developed and implemented these ideas as a team. The result is an easy to use robot that we hope can make a positive impact on household efficiency and safety.

2 System Overview and Installation

2.1 Overview Block Diagram

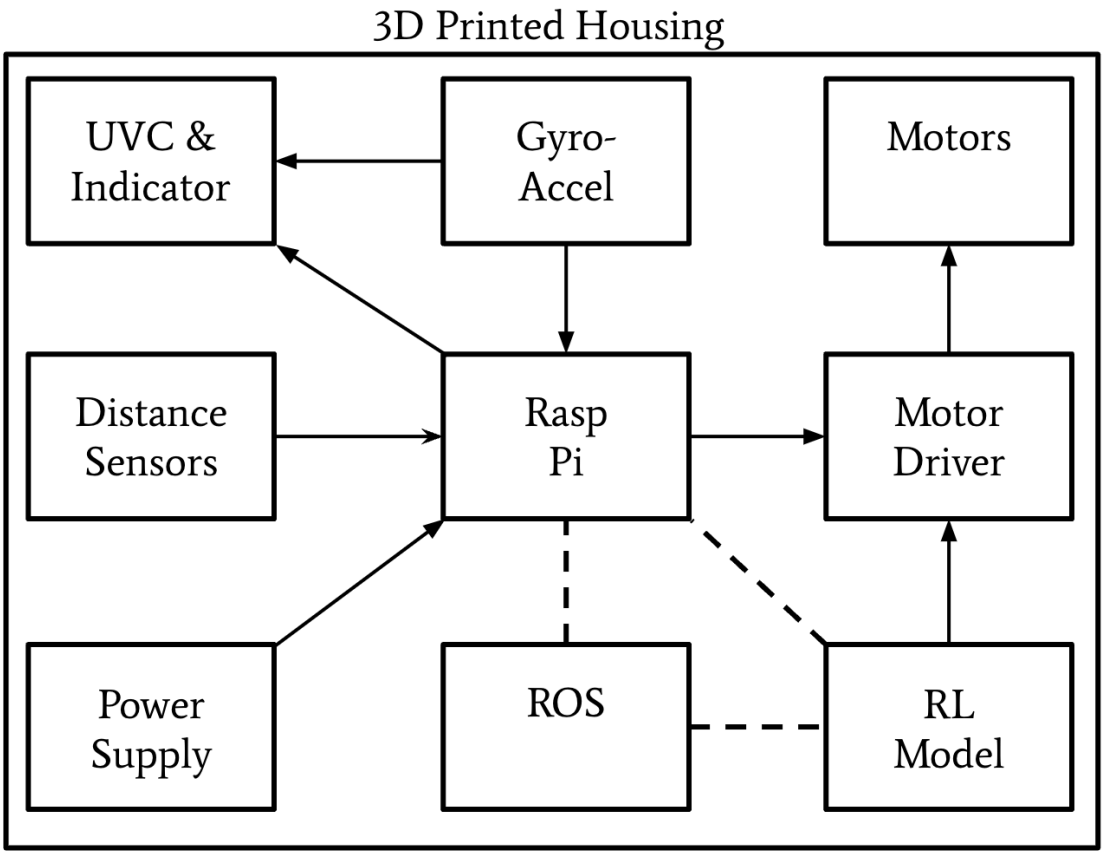

Figure 2.1.1 – Block diagram representing the major components of the Pollux. Arrows represent hardware components, all of which are powered by the main power supply and all of which connect to the central Raspberry Pi. Dashed lines represent software components, which are further elaborated below.

2.2 Physical Description

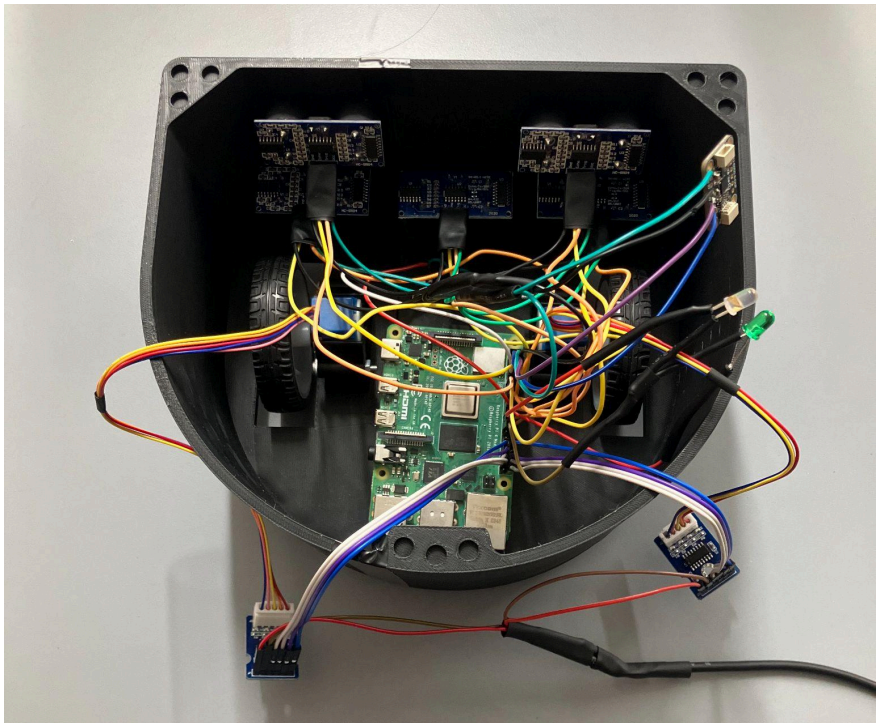

Figure 2.2.1 – The Pollux with the lid off and the battery removed

Figure 2.2.2 - The Pollux fully-assembled and with the lid on

2.3 Installation, Setup, and Support

No installations are required to make full use of the Pollux. On the robot's Raspberry Pi is a startup script that, once powered on, runs all required ROS nodes and begins dictating the robot's movement after a start-up delay of approximately one minute. A simple way to tell when the robot is fully-booted is once the green LED on top of the robot turns on – this LED is an on/off indicator.

To set up and use the Pollux, first ensure that the battery is sufficiently charged. This can be done by removing the lid and pressing the thin button on the side of the battery. This will subsequently display what percentage of the battery remains. Place the Pollux on a surface (countertop, table, etc) and clear off any area that you wish to be sanitized. Then, simply press the power switch on.

The Pollux has been designed for one continuous mode of operation. There is no need to configure anything or interact directly with any software. If you encounter any bugs or strange behavior from the Pollux, first ensure that the battery is sufficiently charged, as the robot may operate slower at low battery. If the battery is not low, turn the Pollux off and turn it back on. This initiates a full reboot – all ROS nodes are terminated and then re-initialized on startup.

3 Operation of the Project

3.1 Operating Mode 1: Standard Operation

<u>Starting the Pollux</u>

1. Place the Pollux on a level surface
2. Press the power switch to 'on'
3. Allow up to two minutes for the Pollux to boot

<u>Sanitization</u>

1. The Pollux will move slowly as it sanitizes the surface underneath it
2. The Pollux will turn whenever it reaches an edge or an obstacle
3. The Pollux may also turn if it has not yet reached a barrier
4. If overturned, the LEDs beneath the Pollux will deactivate

<u>Edge detection</u>

1. When the Pollux reaches an edge, it will halt
2. The Pollux will then turn and continue sanitizing in a different direction

Obstacle detection

1. When the Pollux reaches an obstacle, it will halt
2. The Pollux will then turn and continue sanitizing in a different direction

Stopping the Pollux

1. Press the power switch to ‘off’ – the Pollux can be stopped at any point without risk of UV exposure or interference with software

3.2 Operating Mode 2: Abnormal Operations

If the Pollux is low on battery, it should be recharged until the battery is full or nearly full. The Pollux may halt movement or shut off altogether when the battery runs low enough.

Although it is unlikely, the reinforcement learning model may instruct the robot to enter a movement loop. If this behavior is observed, simply turn the Pollux off and power it back on in order to reboot it.

3.3 Safety Issues

The Pollux is lightweight and moves very slowly, thus the risk of physical injury from the Pollux is slim to none.

Similarly, the Pollux is a low-voltage system, with no component operating at any higher than 5 volts. The internal battery is also short-circuit protected. Hence, the risk of electrical shocks or otherwise electrical injuries is very little.

In its alpha version, the Pollux does not utilize any real UVC diodes – instead, faux LEDs mimic the function of UVC light. In the next iteration of the Pollux, though, real UVC diodes will be implemented, which are hazardous to anyone that contacts the emitted UV light. For this reason, the Pollux contains a gyroscope that turns these LEDs off whenever the robot is overturned, i.e. if someone knocks it over during operation. In the future, quantitative testing will need to be performed to assess how much UV light ‘leaks’ out from underneath the robot – however, our gyroscope system fully prevents direct exposure.

4 Technical Background

4.1 Robot

The Pollux robot is built around a Raspberry Pi 4B. Connected to the Pi's GPIO pins are ultrasonic sensors, motor drivers, LEDs, and an accelerometer/gyroscope. The robot essentially serves to take input from the surrounding environment, and output in the form of movement and LED control. The robot moves by making use of two adjacent stepper motors, controlled by the aforementioned motor drivers. Also attached to the underside of the robot is a central caster wheel – when combined with the two motorized wheels, this allows the Pollux to rotate in place and maneuver more effectively in tighter spaces. On top of the robot are two LEDs – one indicates on/off, and the other indicates whether or not the UVC LEDs underneath the robot are on. Then, on the underside of the robot is a strip of three LEDs that stay on whenever the robot is upright, mimicking UV sanitization.

4.2 Housing

The housing for the Pollux was custom-designed to securely and efficiently accommodate all the internal components. It was modeled in Onshape and is fully 3D printed. The overall structure features a semi-circular shape, which allows the robot's edges to better fit into corners and along walls during operation The cured rear of the housing adds a sleek, modern appearance to the device.

To ensure secure assembly, there are 18 embedded magnets between the body and lid. These provide a firm hold, even if the robot is turned upside down. The lid is engraved with the product's name and includes two LED holes: one as an on/off indicator and the second as a UV light indicator.

Internally, the housing contains a spot for every component. There are precisely modeled compartments for each of the five ultrasonic sensors. There are three located along the bottom front, and two mounted on the front walls. The two front sensors are angled 15 degrees downward for improved object detection. The housing also supports two motor mounts and wheels, a dedicated ledge mount for the IMU (to keep it securely fixed to prevent unnecessary movement), a raspberry pi mounting platform, and a shelf-like battery mount. The battery mount was designed to position the larger battery above other components for optimal space usage.

4.3 Robotic Operating System (ROS)

Pollux relies on ROS Noetic as the software backbone, running headlessly on a 64-bit Ubuntu 20.04 image on a Raspberry Pi 4 Model B. All on-board software is divided into numerous ROS nodes that communicate through the standard publisher-subscriber fashion. This allows the sensing, actuation, and decision-making subsystems run concurrently while not depending on each other too much. This modular design keeps the coding, debugging and testing manageable, and allows any one component to be replaced, edited, or added without changing or damaging the rest of the system.

On boot up, a dedicated file on the Pi named "pollux-robot.service" is invoked, where the environment is created for PyTorch to successfully launch. Next, the main ROS launch file, roscore, is ran, which spawns the two ultrasonic-sensor-publishing nodes, the IMU-based LED failsafe, the motor driver, and finally the reinforcement learning controller ("rl_brain_node.py"). There is a few minute delay before the robot launches due to the launching of all the separate nodes, motors, and sensors, but soon after the robot launches user's are also able to SSH into the robot to access the Pi and kill/run whichever nodes they desire (more detail about this on the project README.md: click here).

Sensor data originates from the two "hw_publisher" nodes. The first node publishes the data from three bottom-facing ultrasonic sensors, while the second node publishes two forward-facing ultrasonic sensors. These streams, together with the IMU readings coming from the accelerometer and gyroscope, feed into the reinforcement learning controller's ROS node. Inside this node ("rl_brain_node.py") is a Proximal-Policy-Optimization (PPO) network that operates at two hertz and chooses movement commands (forward, backward, left-spin, right-spin, or stop). Its reward function penalizes cliff readings, close front obstacles, excessive horizontal/vertical acceleration that suggest falls, unnecessary backward movement, and idling. On the other hand, the reward function rewards exploratory motion and successful coverage of new ground.

While the brain node controls mobility, the two LED nodes run in parallel. "led_control_node.py" takes in high-level LED commands from the "led_gyro_node.py", which checks if the IMU reports a tilt beyond a certain threshold, which in turn should turn off the UV LEDs as well as the UV-indicator LED on the top of the robot. Because this safeguard is implemented in a separate ROS node, it can remain on and working even if the learning controller is paused, crashed, or replaced.

Embedding ROS in this way allows for a very straightforward path to upgrades and enhancements. Additional sensors, such as depth cameras or LiDAR, can be added

merely by introducing a new publisher node and subscribing to it with a new and improved brain node. Similarly, replacing the PPO policy with another policy would require only editing the reinforcement learning controller's ROS node, leaving the rest of the robot untouched. Ultimately, this allows Pollux with increased robustness and adaptability to change and enhancements.

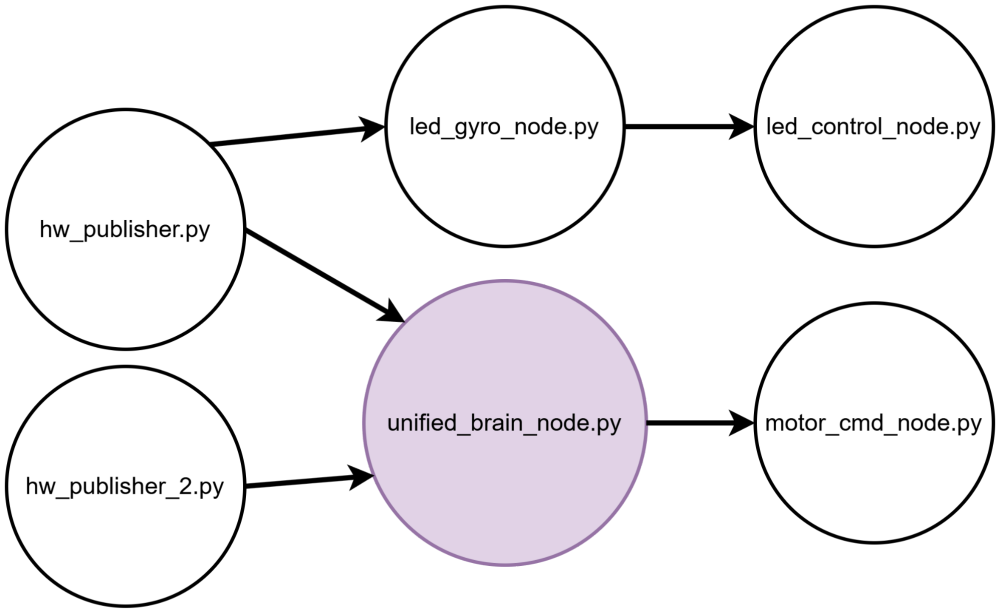

Figure 4.3.1 – ROS node topology for the classical controller.

The diagram shows every ROS node deployed on Pollux when the legacy deterministic brain ("unified_brain_node.py") is in use. Arrows leaving a node indicate topics that the node ***publishes****; while arrows entering a node indicate topics that the node* ***subscribes*** *to. The hardware publisher nodes provide raw sensor streams, the unified brain node converts those streams into high-level motor and LED commands, and the remaining driver nodes act on those commands real-time.*

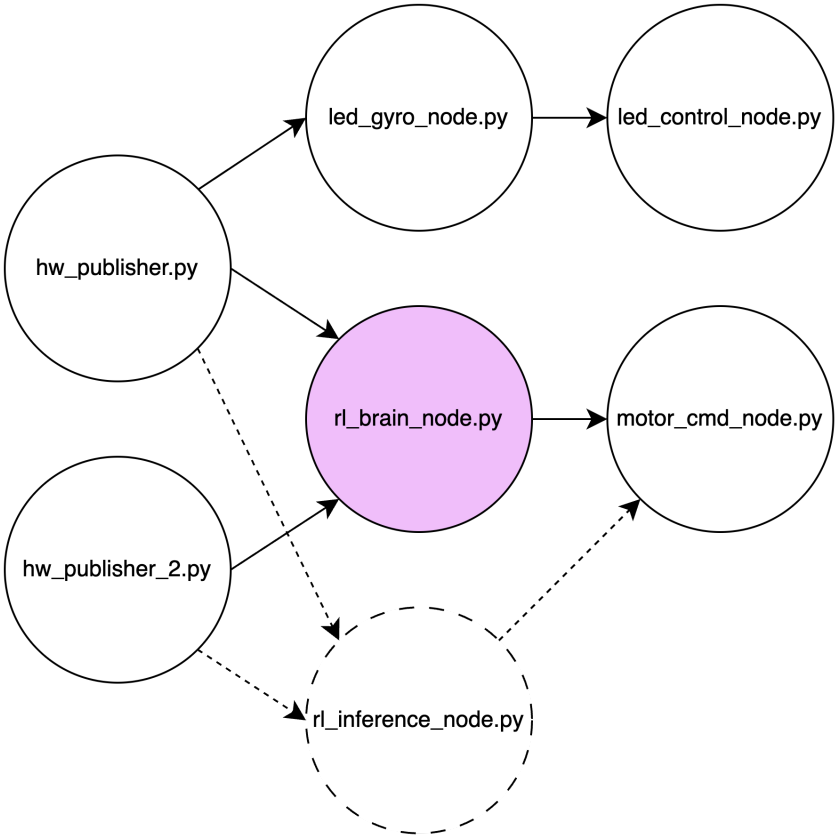

Figure 4.3.2 – ROS node topology for the reinforcement learning controller.
The diagram is identical to Figure 4.3.1 except that the deterministic "unified_brain_node.py" has been replaced by "rl_brain_node.py" or "rl_inference_node.py", which can operate in training mode, or (once trained), in inference mode. The dashed lines indicate that the node is only run after the model is fully trained by running the RL brain node.

4.4 Reinforcement Learning

The autonomous behavior of Pollux is governed by a reinforcement learning (RL) controller that functions as the robot's "brain." Its general purpose is to interpret live sensor data, select from a given set of motion commands, and maximize the surface area covered while guaranteeing that obstacles and cliff edges are avoided. The controller uses the Proximal-Policy-Optimization (PPO) algorithm, chosen for its stability on resource-constrained hardware.

When the robot is running, the bottom and front-facing ultrasonic sensors stream five distance measurements every half a second. Concurrently, the on-board IMU supplies linear acceleration in the horizontal plane, as well as angular velocity. These eight values from the continuous observation are given to the agent/robot, where a single pass of these

values yields one of the five motor commands: forward, backward, spin-left, spin-right, or stop. Motor commands are then published to the motor command node two times a second.

The reward function is built solely to reward good surface coverage patterns and penalize harmful or inefficient behavior. To begin, there is a small positive reward added every time-step to encourage continuous movement, with an added bonus when data seems to vary, which is an indication that the robot is reaching unexplored territory. Large negative penalties are given to the agent whenever any bottom sensor detects a cliff, whenever the front sensors detect possible collision danger, and whenever the IMU registers acceleration above a certain threshold, indicating a fall or slip. It was necessary to ensure the robot would not move backwards too often, since it could cause the robot to fall off cliffs due to a lack of cliff-detecting sensors located at the back of the robot. To do this, backward motion also penalized the agent unless it happened after detecting an obstacle or cliff. After a lot of training, the shaping of this reward function leads to an agent that learns a sweeping pattern of forward spins interrupted by carefully measured retreats away from cliffs or walls.

Training was done in two stages. The first stage was off-board on a desktop PC, running Ubuntu 20.04, Python 3.10, Stable-Baselines3, and Gymnasium. This stage trained and successfully worked in a simulated environment using MatPlotLib. This model is then taken and improved on by “rl_brain_node.py,” which continues the training on-board, with real-life data.

Because the RL controller is done in a single ROS node, it can be replaced, altered, improved without touching the rest of the software stack. In the future, this feature allows the adaptability to possibly incorporate a depth sensing camera or a LiDAR sensor.

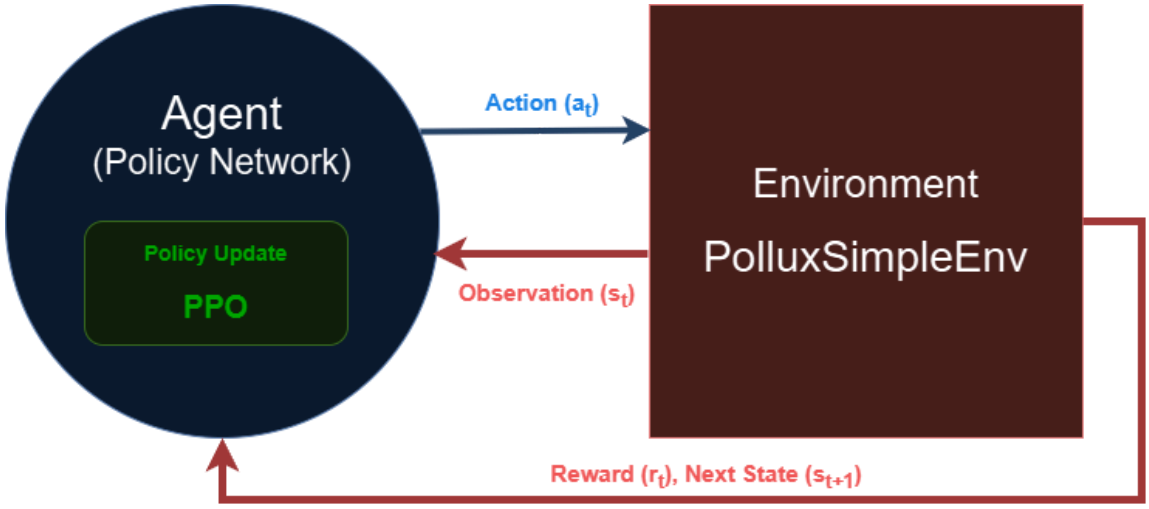

Figure 4.4.1 – the internal agent-environment interaction, including state representations, rewards, action commands, and other observations

5 Relevant Engineering Standards

The most relevant standard for the Pollux is the speed at which it moves. Because the Pollux sanitizes using UV light, it is crucial to ensure that it not only covers as much area as possible, but also that each section of the area it covers receives enough UV light to have a proper germicidal effect.

According to a study from the National Institute of Health, a minimum dose of 27 mJ per cm^2 of UVC light (222nm) was required for a 99% germicidal effect [2]. We can convert this figure to 27mW*s per cm^2, meaning that a 27 mW UVC diode at a wavelength of 222nm requires a full second to kill 95% of germs on an area of one square centimeter. Alternatively, we say that any wattage is sufficient to kill 95% of germs, so long as the time for which it is exposed results in a sum of 27mW exposed to each square centimeter.

Sourcing safe and effective UVC diodes would be an extensive process, and it is not one that was covered during the development of the Pollux, since it is highly likely that these LEDs would need to be specially designed and ordered. However, “A critical review of ultra-violet light emitting diodes…” by Rauch et al provides insight into the strength of UVC LEDs that could be expected. Rauch et al note that the average price point of UVC LEDs between 265 and 280 nm is $1466 USD per W^2, the same as $1.466 USD per mW^2 [3].

Assume that the Pollux Beta Version will contain a strip of five LEDs on the bottom. In order to maintain a cost of around $200, there is a surplus of roughly $60 for these LEDs.

This equates to $12 per LED, from which an LED strength of about 8mW each can easily be calculated. Hence, five UVC LEDs at 8mW each is the figure that will be used to determine how fast the Pollux should move.

Recall the figure of 27mW*s per cm^2 – if we assume that the five LEDs are arranged in a strip such that the length of the underside of the Pollux receives 8mW in a one-centimeter thick band, then the width component of of this figure can be ignored since the Pollux's navigation should ensure that all area is covered length-wise. Hence, 27mW*s per cm^2 becomes 8mW*3.375s per cm, or 3.375s per cm based on our circumstances.

One last issue remains – the aforementioned price point applies to UVC LEDs between 265nm and 280nm (assume 270nm for simplicity), whereas an LED of wavelength 222nm is required for the above sanitization speed. A solution to this issue is to simply expose each centimeter of surface to 8mW*3.375s per cm scaled by an appropriate factor, such that each cm receives the same amount of energy as if it were done using 222nm LEDs. Utilizing the formula $E = hc / \lambda$, it can be calculated that each photon of 222nm contains 22% more energy than a photon of 270nm. Thus, scaling 3.375s per cm by 1.22 yields 4.1175s per cm. To account for any discrepancies, this number will be bumped up to an even 5 seconds per cm.

Hence, the Pollux will move at 5 seconds per cm to mimic >95% sanitization, given this scenario.

6 Cost Breakdown

Project Costs for Production of Beta Version (Next Unit after Prototype)				
Item	Quantity	Description	Unit Cost	Extended Cost
1	5	8mW UV-C Light	$12	$60
2	5	Ultrasonic sensors	$2	$10
3	1	Raspberry Pi 4 Model B	$45	$45
4	1	Lithium Battery	$40	$40
5	2	Stepper Motors	$3	$6
6	1	IMU	$15	$15
7	2	Wheels	$1	$2

8	1	Caster wheel	$2	$2
9	18	Magnets	$0.20	$3.60
10	1	Filament for Housing	$15	$15
			Beta Version Total Cost	$198.60

The most expensive component of our cost estimate is the UV-C light, which is essential for the Pollux to perform its intended sanitation function. While this part was excluded from the alpha version for safety reasons during testing, it remains a necessary expense for the beta version.

The second highest cost is the battery. This larger battery was required to support the combined power demands of the Raspberry Pi running the reinforcement learning model, the UV-C light, and the motors.

Given the original budget and the advanced functionality of the Pollux, the total cost of $198.60 is relatively inexpensive, especially for a custom built, autonomous sanitizing robot with smart navigation capabilities. The use of affordable ultrasonic sensors instead of expensive LiDAR systems was a deliberate design choice to keep costs low.

7 Appendices

7.1 Appendix A — Specifications

Requirement	**Value, range, tolerance, units**
Area coverage	50-60% of total surface area covered **at least**
Edge avoidance	100%
Germicidal efficiency	>95% at 5 seconds per cm
User-independence	Minimum 8 hours runtime without need for user interaction.
Gyroscopic sensitivity	Trigger at a tilt greater than 20 degrees.
Function indication	Sanitizing / Non-sanitizing
Durability	Withstand fall from 4 feet (typical countertop height)

7.2 Appendix B — Team Information

Wenhao Cao
Worked on Web control with Karl. Graduating in May 2025

Karl Carisme

Louis Cicala
Worked on electronics and documentation. Graduating in December 2025.

Noah Hathout
Worked on ROS, the reinforcement learning model, simulations, testing, debugging, 3D printing. Graduating in May 2025.

Caelan Wong
Worked on the housing design, all the 3D modeling and documentation. Graduating in May 2025.

7.3 Appendix C — Text Citations

[1] H. Salonen, T. Salthammer, E. Castagnoli, M. Täubel, and L. Morawska, "Cleaning products: Their chemistry, effects on indoor air quality, and implications for human health," *Environment International*, vol. 190, 2024, Art. no. 108836. [Online]. Available: https://doi.org/10.1016/j.envint.2024.108836.

[2] Song BM, Lee GH, Han HJ, Yang JH, Lee EG, Gu H, Park HK, Ryu K, Kim J, Kang SM, Tark D. Ultraviolet-C light at 222 nm has a high disinfecting spectrum in environments contaminated by infectious pathogens, including SARS-CoV-2. PLoS One. 2023 Nov 28;18(11):e0294427. doi: 10.1371/journal.pone.0294427. PMID: 38015931; PMCID: PMC10684113.

[3] Kyle D. Rauch, Sean A. MacIsaac, Bailey Reid, Toni J. Mullin, Ariel J Atkinson, Anthony L Pimentel, Amina K. Stoddart, Karl G. Linden, Graham A. Gagnon, A critical review of ultra-violet light emitting diodes as a one water disinfection technology, Water Research X, Volume 25, 2024, 100271, ISSN 2589-9147, https://doi.org/10.1016/j.wroa.2024.100271. (https://www.sciencedirect.com/science/article/pii/S2589914724000616)