Final Project Documentation: SMILEY

Sylvie Chen xc455, Maya Shen, Haohua Liu hl766, Tsung-Yin Hsieh th542, Jonathan Tan jmt362

Robot Name: Smiley

Robot Motivation: People often find trouble when they want to take photos of themselves but they don't have a good friend around to take photos for them and it's embarrassing for them to ask a stranger for photos, especially if they want to take a bunch. Or sometimes the aesthetics of the photo your friend helps take of you often isn't satisfying because you just can't see it when you take photos. Thus, we created Smiley, the lively robot to follow you around and help you create photos or videos that you like. Inspired by the tourists on the island, we designed a standing robot that can follow tourists around to help take photos or even Tik Tok video. By emancipating the user's hand to take photos, we offer a fun experience of photo taking. This robot can also be beneficial for taking both individual or group photos. It can also be extra useful for youtubers who produce videos solo.

Description:

Our robot Smiley helps tourists take photos and videos and are able to follow them while doing these activities. There are three main interactions

1. Taking Pictures: This interaction allows users to command Smiley to take a picture. Along with the basic functionality of taking a picture, "Tell me a joke" commands the

robot to tell a joke before taking a picture, which hopefully makes a better picture because the user has a bigger smile.

2. Taking Videos: This interaction allows users to command Smiley to take a video. There's an additional functionality for fancy camera movements for better Tik Tok videos.

3. Robot Movements: This interaction consists of several commands that make the robot move around to find the best shooting angle. "Follow me" triggers the robot to follow the user. The user can also command the robot to move forward, move backward, turn left, and turn left.

Overview of Functionality

Here's an overview of the specific functionalities within each tutorial.

Because there are many commands, we planned the following tutorial: When a user initiates interaction with the robot, the robot will first make an introduction of itself, and say a list of commands for the user to try. When the user triggers a command for the first time, the robot will describe what this command does as well as any additional instructions, then carry out the actual interaction (e.g. taking a picture).

Triggers:

1. if a person stands close to the robot – the Robot starts to say the introduction script.

2. If a person speaks any commands – the Robot starts to teach you what the command can do and let you speak again to start

Routines:

1. During "Take a photo" or "Take a video", the Robot will make a "beep" and a "shutter" sounds to let people know when it starts taking photos/videos
2. During the Robot finish taking photos/videos, it will tell the user to access the photos/video through the QRCode

To help you with ease of reading the following graph:

When user does a **Command/Trigger**, if for the first time, the robot will do the **First Time(Tutorial) instruction,** otherwise the robot

Functionality	**Command/Trigger**	**First Time (Tutorial)**	**After first time**
Introduction	Trigger - User stands in front of Smiley	1. Introduce Smiley 2. Instructs user to try one of the commands	N/A
Repeat commands	Command - "Repeat all commands"	"Here are all the commands: take a photo, take a video, follow me."	
Taking photos	Command - "Take a photo"	1. "Sure, I will take a picture for you" 2. Robot produces a "beep" and a "shutter" sounds to imitate camera	

		3. Take picture 4. “Great photo! You can access it by scanning QR code”	
Tell a joke & take photo	First time - right after “Take a photo” tutorial Other times - “Tell me a joke”	1. Describe the functionality 2. Tell a joke 3. Take picture	1. Tell a joke 2. Take picture
Taking videos	Command - “Take a video”	1. Describe the functionality 2. Introduce “Fancy movements” command, which instructs the robot to make fancy camera movements 3. Robot produces a “beep” and a “shutter” sounds to imitate camera 4. Take video 5. Instruct user to scan QR code	1. “beep” and a “shutter” sounds 2. Take video 3. Instruct user to scan QR code
Follow me	Command - “Follow me”	1. Describe the functionality 2. Instructs user to start moving 3. Follow the user 4. (optional):Stop when getting too close to user or any obstacle and ask user for navigation	1. Follow user 2. Stop when getting too close to user or any obstacle and ask user for navigation

Movement commands	First time Command - Immediately after “Follow me” tutorial Command - “Turn left/right” “Move front/back”	1. Introduce the functionality and commands 2. Do as user says 3. (optional):Stop when getting too close to user or any obstacle and ask user for navigation	1. Move as user instructs 2. Stop when getting too close to user or any obstacle and ask user for navigation

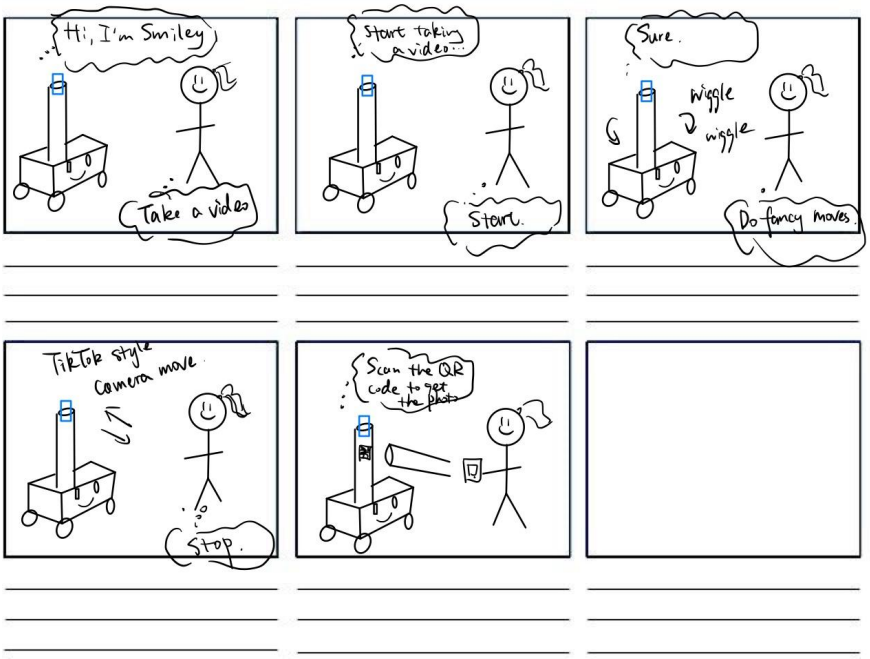
2. Storyboarding Process

We started out our interaction design with drawing out the storyboards for each interaction.

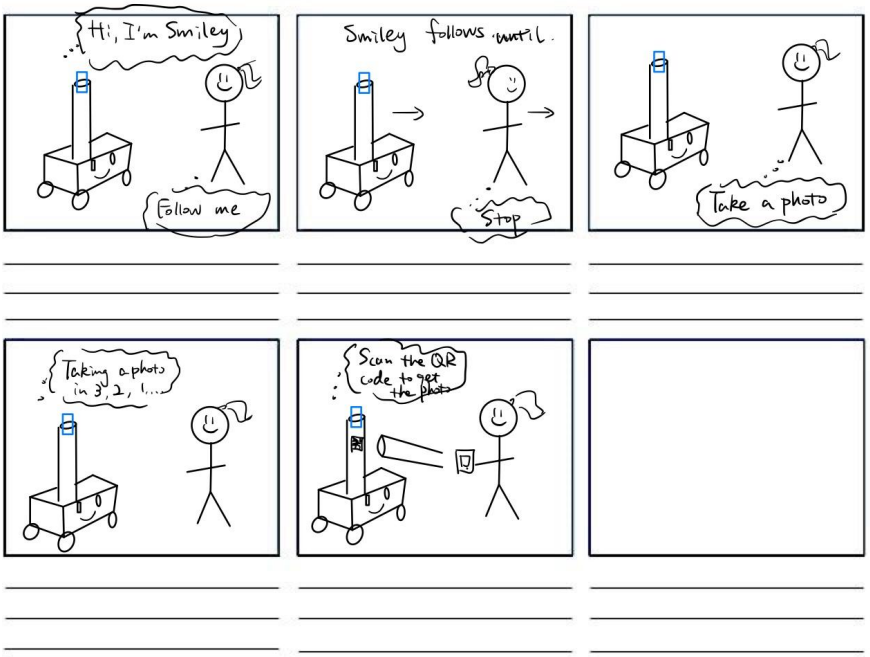
[Original Sketch]

Here're 2 rough sketches.

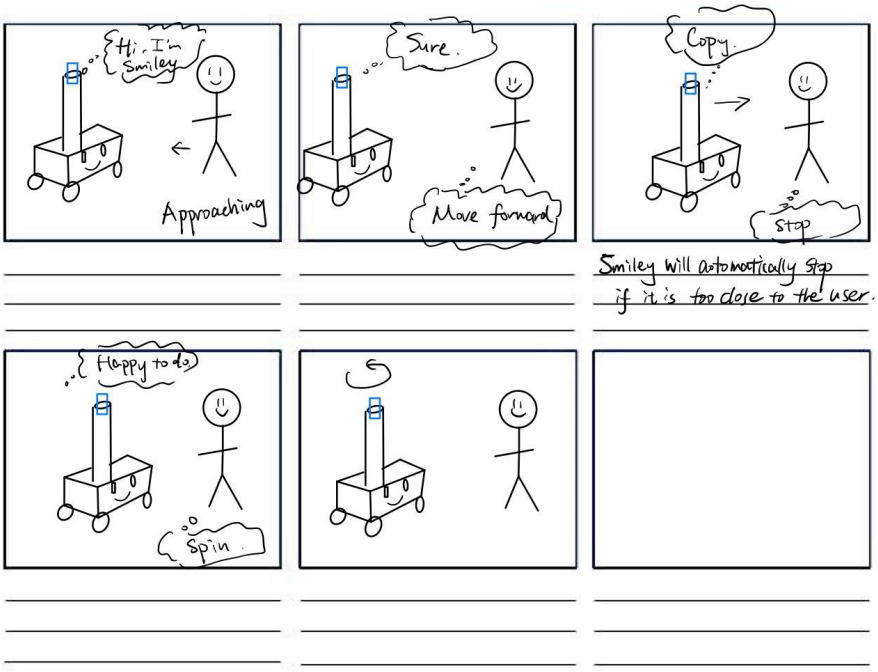
[Revised Sketch : Video-Taking]



[Revised Sketch : Photo-Taking]

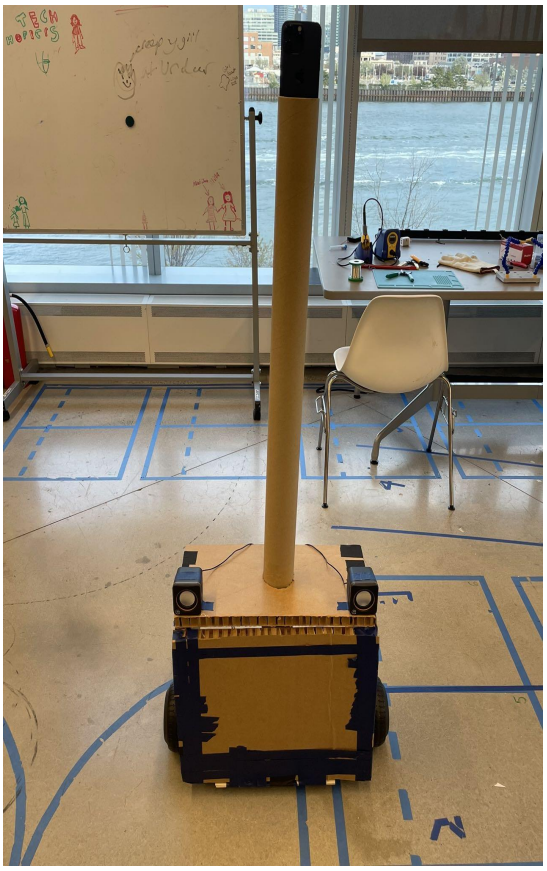


[Revised Sketch : Movement]



3. Prototype Design Process:

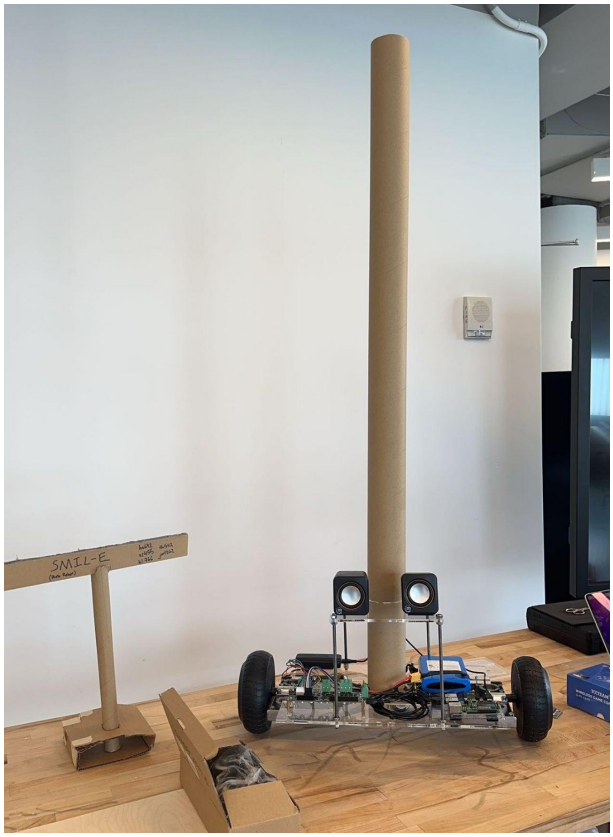
[Prototype 0]

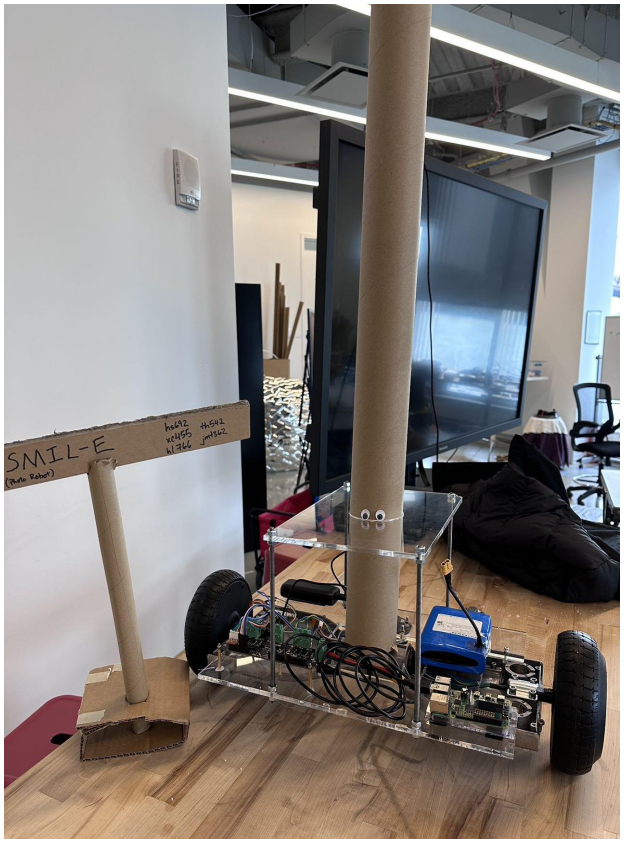


The hoverboard we are given already has two wheels that move around. We created our first iteration of prototype out of thick paper boxes material with a box covering the nitty-gritty details of our hoverboard and the wires and batteries. We taped the box up and positioned the box on top of the wheels to make it stable.

In the middle of the box, there's a cylindrical paper roll that we measured to be the perfect size that fits through an iphone with a perfect eye-level height so people can use the iphone we put on top of the roll to take photos. We also placed two speaker on top of the box so we can simulate Smiley talking from its own body.

[First Iteration : Prototype 1]





We made our first iteration due to the instability of the paper box as Smiley moves around. With the wheels turning, the box sometimes shifted and made sounds. It sometimes tips over that we decided to replace the paper box with a plastic structure that holds our robot stable.

To achieve this process, we utilized clear acrylic as a foundation board and screwed it on top of the hover board to put batteries and wires. We downsized the bigger box to a smaller one to hold our speaker around the cylindrical roll, rendering a more stable and combat structure. We utilized four long screws to support the speaker like a small box. That's also when we started to dress up our Smiley and put two cute little eyes on the cylindrical roll.

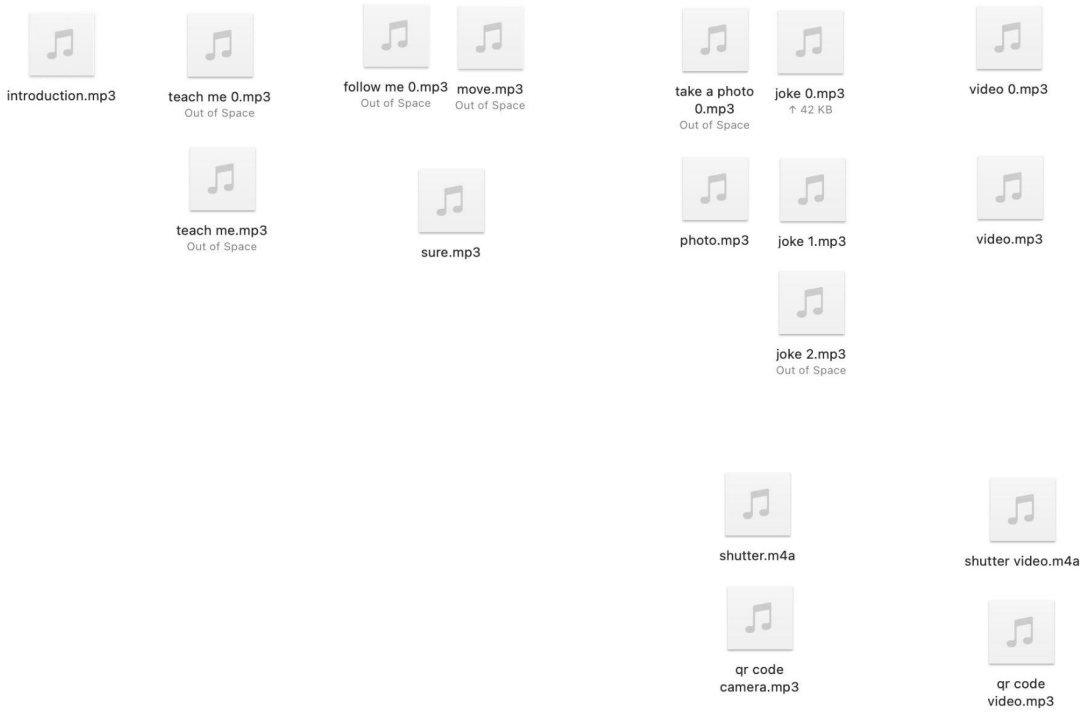
[Second Iteration : Prototype 2]

For our last iteration, to make Smiley more interactive and vivid, we decided to dress up our robot. We first moved his eyes up to the top of the roll so it matches people's eye level which makes it easier for the users to notice. We also put the QR code sign on top of the eyes for users to scan easily. We created a pink chiffon lace dress and tied it up using a black elastic on the pole, rendering Smiley a vivid robot that feels more like a person.

[Voice Manuals]

In addition to the physical prototype, we also have our specific voice manuals that we operate Smiley with. The purpose of the voice manuals was designed to wizard test Smiley remotely and the text to voice library constantly breaks down on our end so we recorded it and saved them locally for ease of navigation during testing.

Here's a screenshot of the voice manuals we created according to the functionalities within each interaction. The specific command recordings are attached below.



[Voice Manual Work Flow]

Trigger/command	**Teaching(first time play this)**	**Action**
(Introduction) Person standing in front of robot	Robot: Hi! I'm Smile-ly. I'm a loyal robot who can take amazing photos and videos for you. First, let me teach you how to boss me around. Say the command: "teach me".	

Teach me	Robot: Try one of my commands: 1. Follow me, 2. Take a photo, 3. Take a video. Of course, you can always say “repeat commands” and I’ll repeat the commands again. User: [command]	Robot: Try one of my commands: 1. Follow me, 2. Take a photo, 3. Take a really good photo, 4. Tell me a joke. Of course, you can always say “teach me” and I’ll repeat the commands again. User: [command]
Follow me	Robot: This command makes me follow you as you go. Say “follow me” to start and say “Stop” when you’re done. User: Follow me	Robot: “Sure” Moves with the user.
Move Left/Right/Front/Back	Robot: You can also control me by saying “Move Left, Right, Front, or Back”	Robot: “Sure” Moves as the user says
Take a photo	Robot: When you say “Take a photo”, I will take a photo for you in 3 seconds. Pose, ready, start! *Play camera shutter sound.*	Robot: “Sure, I’ll take a photo for you! Say cheese!” *Play camera shutter sound.*
Tell me a joke	Robot: (Right after taking a photo). Want a photo with a bigger smile? Say “Tell me a joke”, I will tell you a joke to make you laugh and take a photo right afterwards.	Robot: (Tell a joke) Are you laughing already? *Play camera shutter sound.*
Take a video	Robot: This command asks me to take a video for you. During the recording, you can also command me to create fancy camera movements. Say “I’m ready” to start and “Stop” to stop recording.	Open the video page

	User :**I'm ready**	
Take a TikTok	Robot: If you say "Take a TikTok", I can help you take a TikTok. When you're ready, say "I'm ready", and I will start taking a TikTok video in 3 seconds. Say "stop" and I will stop recording. Say "Cool camera movement" whenever to try my secret sauce of making a good TikTok video.	Open the video page
		Robot: **3, 2, 1. Say Cheese** (Shutter sound.) Robot: **3, 2, 1. Start** (Shutter sound.)
How do I get the photos?	Robot: you can scan the QR code on me.	
Teach me how to make poses	Robot: If you say "Teach me how to make poses", I can give you some suggestions.	
Move the angle Left/Right/Up/Down	Robot: You can control my camera angle by saying "Move the angle Left/Right/Up/Down"	

[Code]

The robot is wizard by the controller and the sound from the robot is played by a Bluetooth speaker. We derived our code mainly from our class Mobile HRI in Cornell Tech taught by Dr. Wendy Ju. Since the original moving speed in the code is pretty fast, we lower the moving speed of the robot so it would not shock the users while moving around.

https://github.com/JMortonTan/SMIL-E

The code we add to deal with the Odrive watchdog error issues:

```
class odrive_command(Node):
    def __init__(self):
        super().__init__("command_lisener")
        self.odrv = odrive.find_any()
        self.axis0=self.odrv.axis0
        self.axis1=self.odrv.axis1
        self.axis0.requested_state = AXIS_STATE_CLOSED_LOOP_CONTROL
        self.axis1.requested_state = AXIS_STATE_CLOSED_LOOP_CONTROL


        # The Odrive often got watchdog timeout problem, so we make the config.enable_watchdog = False first to prevent the error happens
        self.axis0.config.enable_watchdog = False
        self.axis1.config.enable_watchdog = False

        self.odrv.clear_errors()

        self.axis0.config.enable_watchdog = True
        self.axis1.config.enable_watchdog = True

        self.axis0.config.watchdog_timeout = 4
        self.axis1.config.watchdog_timeout = 4

        self.axis0.watchdog_feed()
        self.axis1.watchdog_feed()

        self.odrv.clear_errors()

        self.get_logger().info("odrv initialization done")
```

The code we adjust to lower the moving speed of the robot so it would not shock the users while moving around.

```python
def diff_drive_callback(self, msg):
    v = msg.linear.x
    w = msg.angular.z
    # TODO: Your Code Here ------------------------------------------
    # Lower down the speed by divide the speed by 10
    Vl = (v - self.wheel_track * w / 2) / 10
    Vr = (v + self.wheel_track * w / 2) / 10
    # raise NotImplementedError("Differential Drive Controller, remove this line once implemented.")
    # ---------------------------------------------------------------

    # convert to turns per sec (ODrive Unit)
    Vl = Vl / 2*math.pi
    Vr = Vr / 2*math.pi

    self.feed_watchdog()
```

5.Testing Process

The testing process for our robot was carefully designed, taking into consideration the voice manuals provided. To provide you with a clear understanding, we have recorded footage of our testing procedures. The links of our recorded videos are shown below.

At the start of the testing, the robot initiates by giving an introduction on how users can control its functions. We wanted to ensure that users had a clear understanding of how to interact with the robot. Once users give commands, we take control of the robot's movements while also playing corresponding voice manuals that align with the users' reactions.

Throughout the testing phase, we identified an issue with the interaction between users and the robot. The robot was not responding as quickly as the users expected, leading to a less smooth and clear communication process. To address this, we made modifications by introducing additional commands that allowed the robot to slow down the interaction process.

By incorporating these adjustments, we aimed to improve the overall user experience by ensuring that our robot's responses align more closely with the users' commands. This refinement will result in a smoother and more seamless interaction, enhancing the usability and effectiveness of the robot in various scenarios.

Videos of testing interactions:

https://drive.google.com/file/d/1j8ZdjjXTao9yz8G3zcZhiZbl6liC4Qbk/view?usp=drivesdk

https://drive.google.com/file/d/1aYXH_KNU3HkRre2wyRc4cjg9nj7ADFbc/view?usp=drivesdk

https://drive.google.com/file/d/1hAvpKYOZgKroU7bV9i_f9bYNhicl4M1i/view?usp=drivesdk

https://drive.google.com/file/d/1ZH7jCjnetJBWDmm2ut1kJ4wpwV2Zjh76/view?usp=drivesdk

6.Reflection

Throughout the testing phase, we found some issues with the interaction between users and the robot. Smiley robots should be beneficial for taking photos and videos for users. To make the robot meet the requirement, these are our reflections and adjustments for improving our robots.

1. We find that the robots might move too fast toward users, which might shock or scare them. Thus, we slow down the moving speed of the robots and made the movement smoother.
2. We find out that During “follow me”/”Move left/right/front/back” commands, the Robot might get too close to obstacles. Thus, we decide to add a new script for the robot. If the robot gets too close to any obstacle(a person or anything), the robot will stop and say “I’m getting too close! I will stop here.”
3. After the user finishes a command, the space sinks into silence if users doesn’t have any future commands. We can try saying “Anything else? Say ‘Repeat commands’ if you don’t remember”.
4. Users might give a lot of commands simultaneously sometimes, which could be pretty hard for the robots to respond and take action. We add some commands like “Could you repeat the commands slower again” to make the interaction process slower.
5. During movement interaction “follow me” or ”Move left/right/front/back”, we realize that ‘turn left/right/front/back’ is different from “move left/right/front/back’ so we should add this.