Weekly Progress Report
Week 10

Background

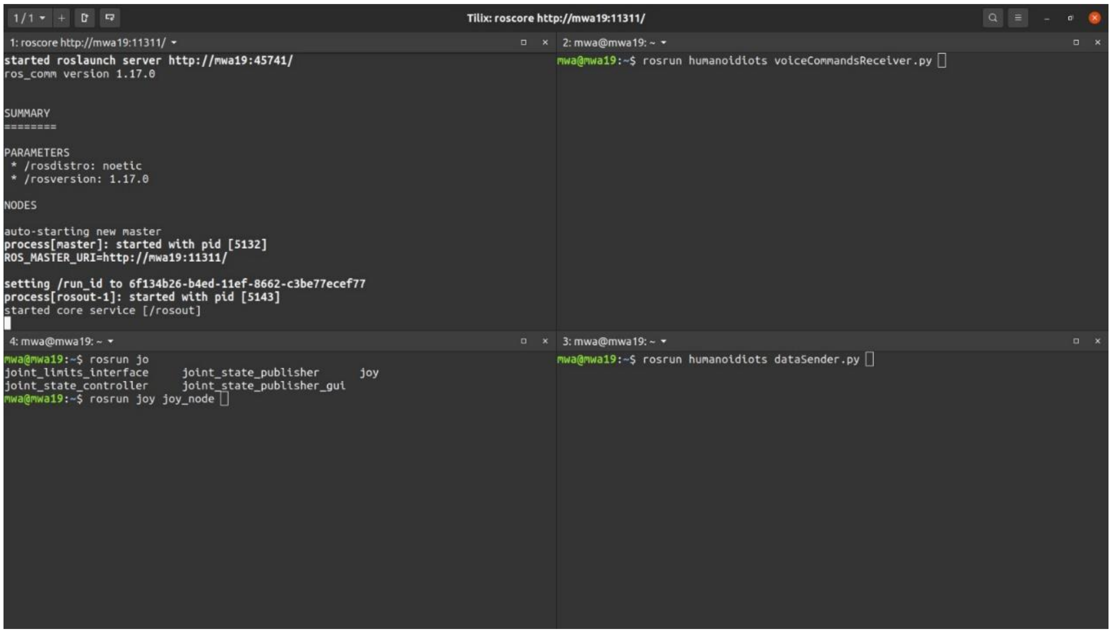
This marks the second week of hands-on work on our humanoid robot project. Following Week 9's foundational setup and initial feature integration, the focus has shifted towards expanding communication capabilities, implementing advanced processing systems, and facilitating our project workflow by categorizing tasks into distinct stages.

This Week's Progress

Work Classification by Stages

1. **Face Recognition**
    - Implemented face detection with live streaming functionality.
    - Developed a system for capturing faces effectively.
    - Addressed challenges related to streaming video rate and lags caused by SD card usage. Optimizations are ongoing to improve performance.
2. **Voice Recognition**
    - Completed the development of the voice recognition system.
    - Processed through the ROS system, allowing the system to receive and handle voice commands effectively. The commands are then processed and transmitted for further action via Bluetooth communication to Arduino.
3. **ROS System Implementation**

- Built a complete Robot Operating System (ROS) for processing joystick input data.
- The system now receives voice recognition data via Wi-Fi, processes it, and transmits numerical commands through Bluetooth to Arduino.


1/1
Tilix: roscore http://mwa19:11311/
1: roscore http://mwa19:11311/
2: mwa@mwa19: ~
4: mwa@mwa19: ~
3: mwa@mwa19: ~

```
started roslaunch server http://mwa19:45741/
ros_comm version 1.17.0


SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.17.0

NODES

auto-starting new master
process[master]: started with pid [5132]
ROS_MASTER_URI=http://mwa19:11311/

setting /run_id to 6f134b26-b4ed-11ef-8662-c3be77ecef77
process[rosout-1]: started with pid [5143]
started core service [/rosout]
```

```
mwa@mwa19:~$ rosrun humanoidiots voiceCommandsReceiver.py
```

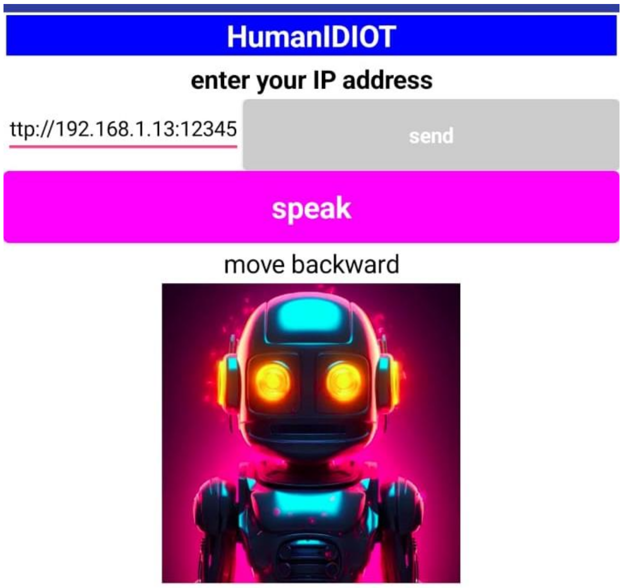
```
mwa@mwa19:~$ rosrun jo
joint_limits_interface     joint_state_publisher      joy
joint_state_controller     joint_state_publisher_gui
mwa@mwa19:~$ rosrun joy joy_node
```

```
mwa@mwa19:~$ rosrun humanoidiots dataSender.py
```

Example:

- Moving the joystick forward or giving a voice command ("Move Forward") sends data to the laptop for processing.
- The processed data is converted into numerical commands, which are then transmitted via Bluetooth to the Arduino to actuate motor movements.

4. **Application and Controller Development**
    - Finalized the application GUI for streamlined user interaction.
    - Ensured the controller integration allows smooth joystick operation, with input processing and response.



ttp://192.168.1.13:12345

move backward

5. **System Integration**
    - Established a reliable communication channel for receiving data from Android devices via Wi-Fi.
    - Tested and confirmed accurate data transmission to the Arduino for action execution.
6. **Bluetooth Testing**
    - Conducted a Bluetooth distance test, confirming stable communication over extended ranges.

Challenges Encountered

- **Streaming Performance:**
  - Video streaming faced significant lags due to SD card read/write operations, which affected the real-time nature of the face recognition process. Efforts are underway to optimize storage operations and video buffer management.
- **Data Synchronization:**
  - Managing smooth transitions between multiple input sources, including joystick commands and voice recognition, required careful tuning of the ROS system.

What's Next

- **Optimize Face Recognition Performance:**
  - Continue optimizing the streaming process to minimize lags and ensure smooth video capture and storage.
- **Enhance System Efficiency:**
  - Optimize ROS communication protocols to reduce latency.
  - Address any residual issues in data synchronization and processing.
- **Finalize Mechanical Design:**
  - Conclude discussions on the robot's mechanical structure and initiate the assembly phase.