SD15 | THE TEAM

Maxim Popov
- Robotics + Kinematics
- West Coast Swing Dancer
- Grad Student in 5990

Zach Campbell
- Mediapipe + Realsense Guy
- Guardian of the depth camera
- Teaches Taekwondo in his spare time

Dylan Booth
- ROS Communications
- Really into ISU Athletics
- TA for Cybersecurity 2310

Isaac Lo
- MuJoCo Simulator
- TA for CS 3210
- Engaged to be married

Max Mayer-Mader
- ROS Expert
- Robotics Lab Researcher
- The canonical Max

CLIENT:
DR. BOWEN WENG

PROJECT REAL STEEL

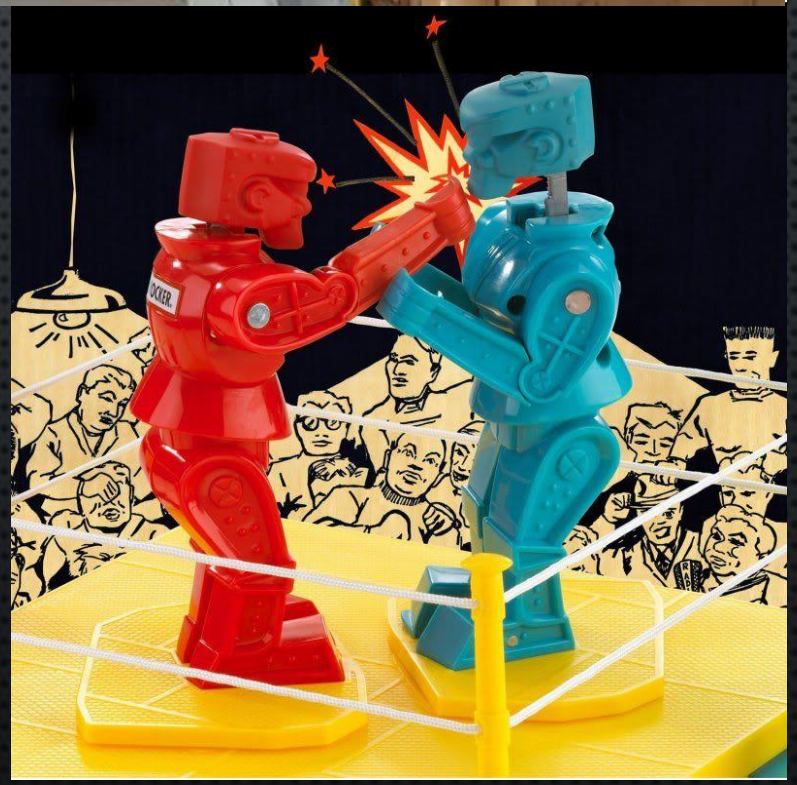
IT'S A BOXING ROBOT

PREMISE: DETAILS

- Mimicking human kinematics to emulate the technology of the movie using:
  - Existing pose detection libraries
  - Depth camera
  - Dr. Bowen's humanoid robot
- Follow-up to a project from last semester involving pose detection and recognition

PREMISE: WHY?

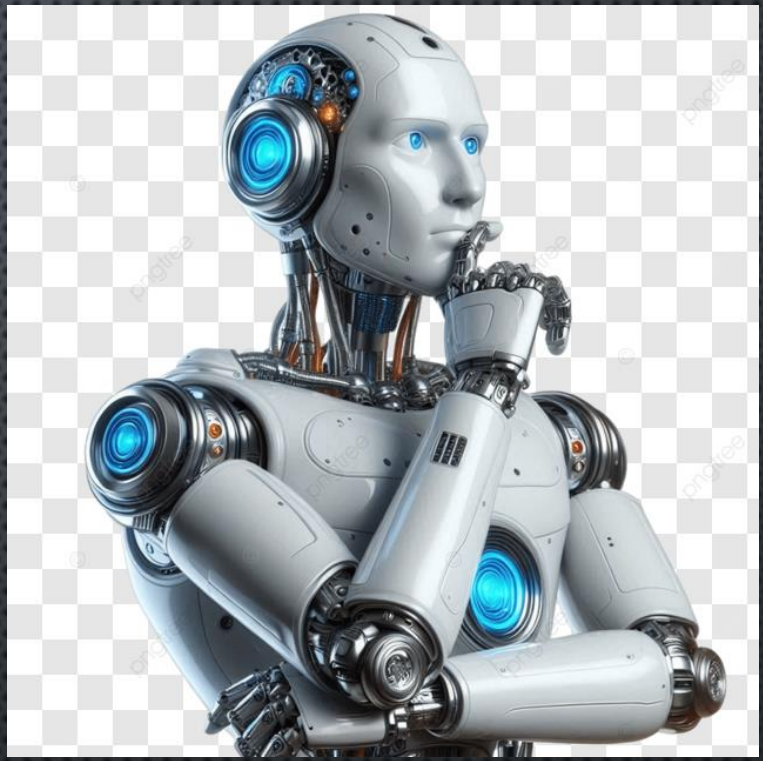
- WHY?
  - **DANGEROUS WORK:** HEAVY LOADS, DANGEROUS ENVIRONMENT, ETC.
  - **MENIAL WORK:** JOBS, CHORES, ETC.
  - **RESEARCH:** STEPPING STONE TO AUTONOMOUS PHYSICAL AGENTS;
    - COPYCAT LEARNING → REINFORCEMENT LEARNING
  - **THE REAL REASON:** BOXING



PRIORITIZATION

- Hand position: ignoring all else
- Upper body: joints and limb segment orientations
- Fine hand tracking (stretch goal): fists
- Lower body (stretch goal): locomotion

CONSTRAINTS



- SINGLE UNIT
  - NO WEARABLE TRACKERS
  - SOFTWARE RUN ON ROBOT
- ROBOT POSITIONING
  - HUNG FROM HARNESS
  - CONNECTED TO LAPTOP AND CAMERA VIA CORD IN THE BACK OF THE HEAD

THE TECH STACK

TECH STACK - HARDWARE

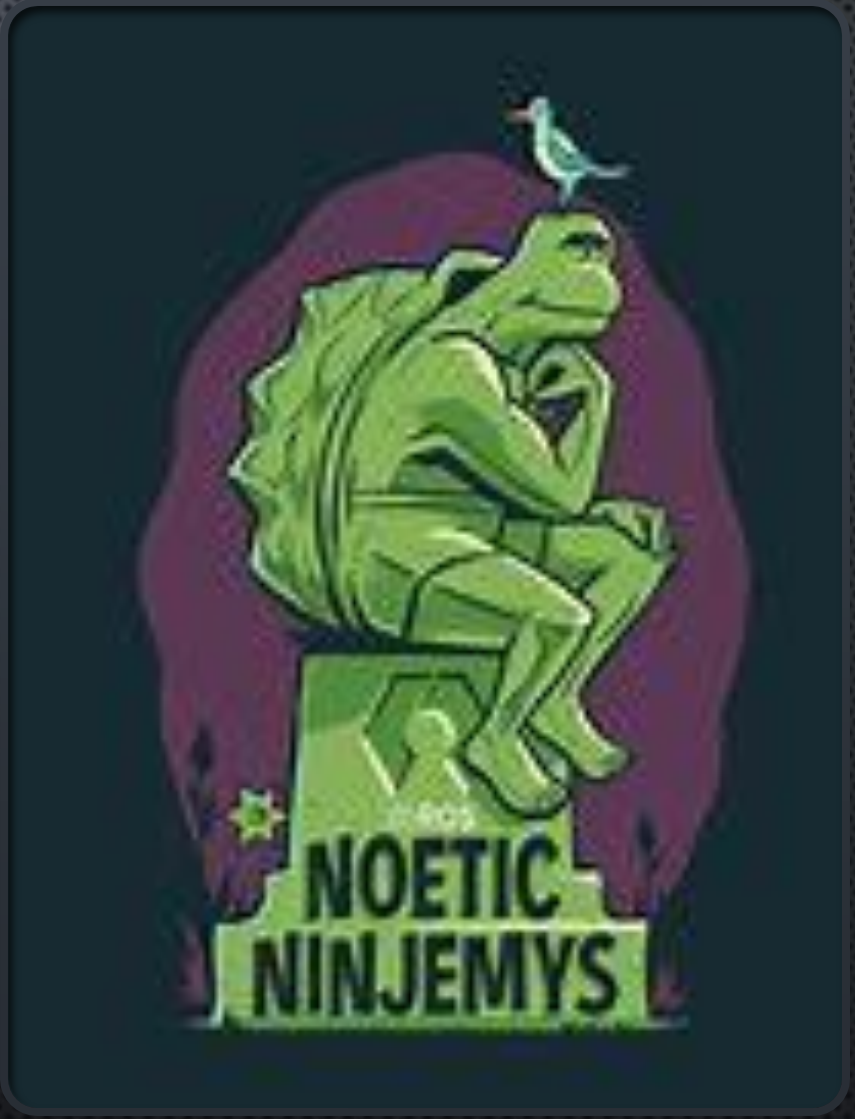
- Intel RealSense 3D Depth Camera 435i
  - RGB Camera
  - Depth Camera
- Unitree G1 – Humanoid Robot

TECH STACK - SOFTWARE

- ROS1 Noetic Ninjemys
- Google MediaPipe
- OpenCV
- MuJoCo
- Python

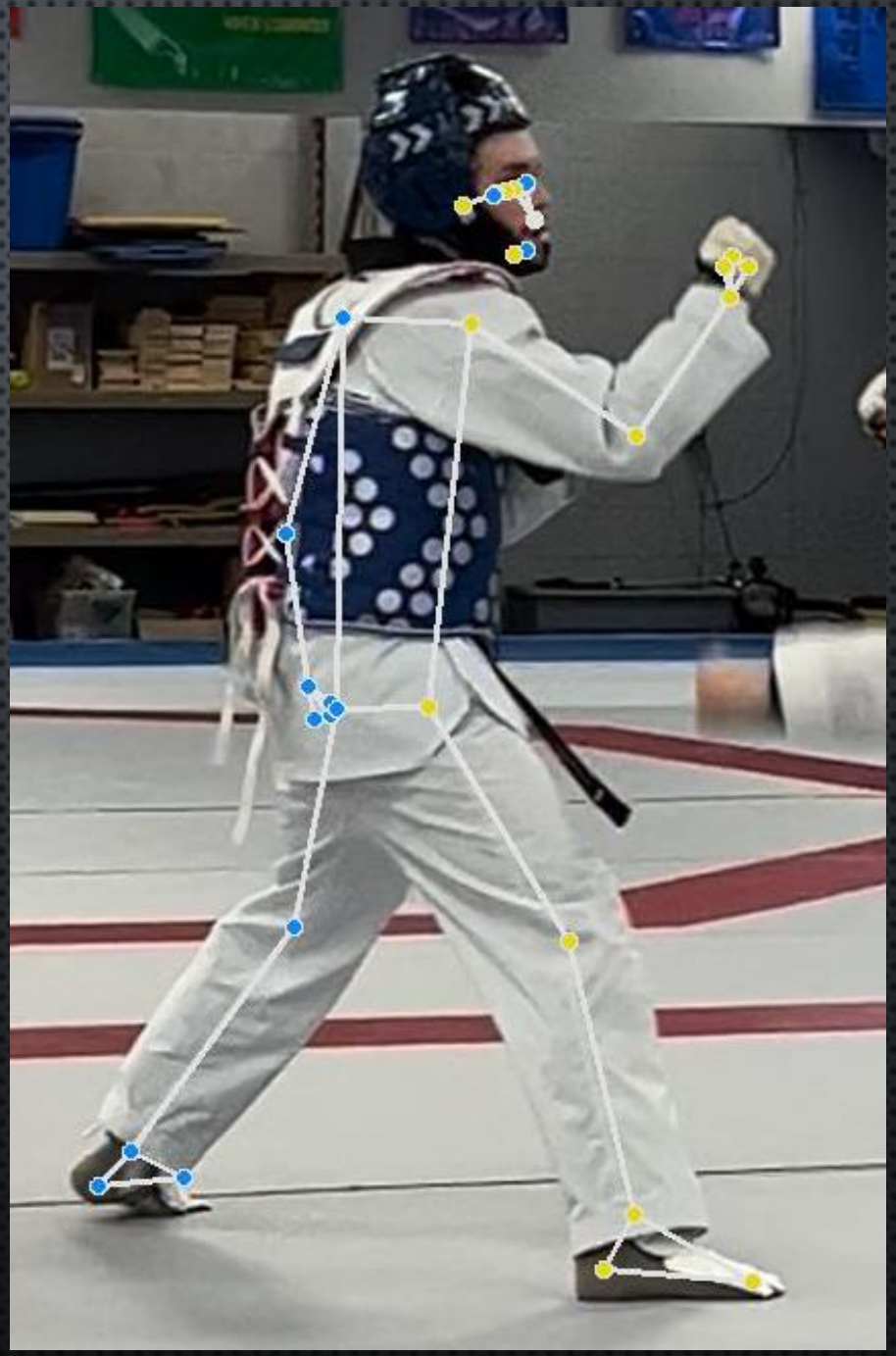
ROS1 NOETIC NINJEMYS

- Robotic Operating System (ROS)
- Standard Application used in Robotic Systems
- Communication system
  - Nodes for each subsystem
  - Messages to communicate between subsystems
- Allows For Multi-Threading



GOOGLE MEDIAPIPE POSE

- Machine-learning Pose Detection Library
- Can detect for still images and real-time video
- Detects pose on 1 person
- Built by Google over the last 5 years
- Provides 33 body landmarks, including hands and face



OPENCV

- Open-source python library for working with images
- Development dependency / indirect dependency
  - Used by Mediapipe and depth camera libraries
- Manages transforms, BGR vs. RGB colors, display, etc.

MUJOCO

- Multi-Joint dynamics with Contact
- Robotic Simulation Environment
- Open Source with Python integration
- Compatible with ROS
- Used to simulate code without access to G1 robot

PROJECT DESIGN

D435i Depth Camera

RGB Image
Depth Frame

camera.py

MediaPipe

PYREALSENSE

ROS

Body Landmarks

kinematics.py

Inverse Kinematics
Forward Kinematics

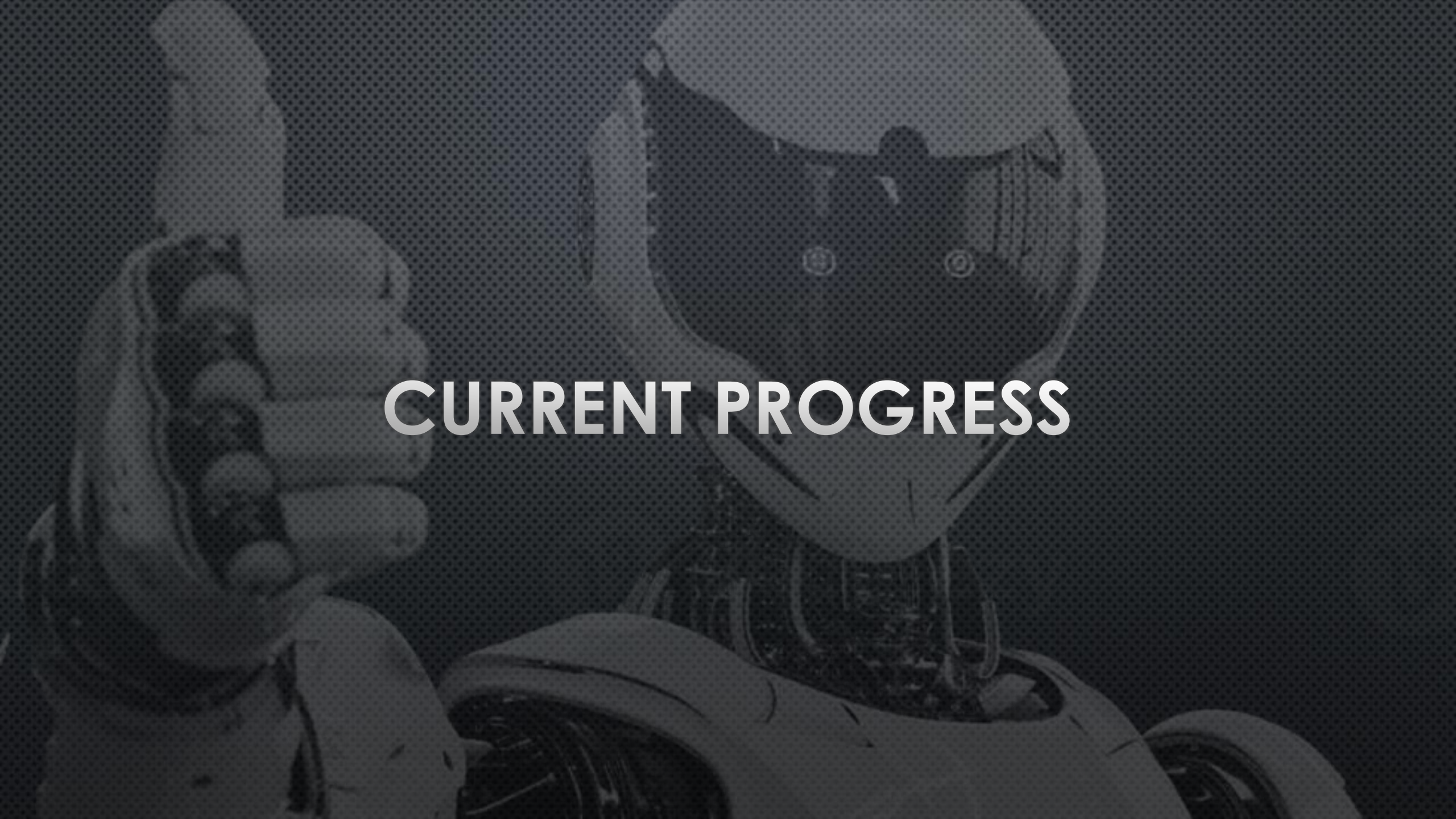
CSV of Motor Positions

Robot & Simulator

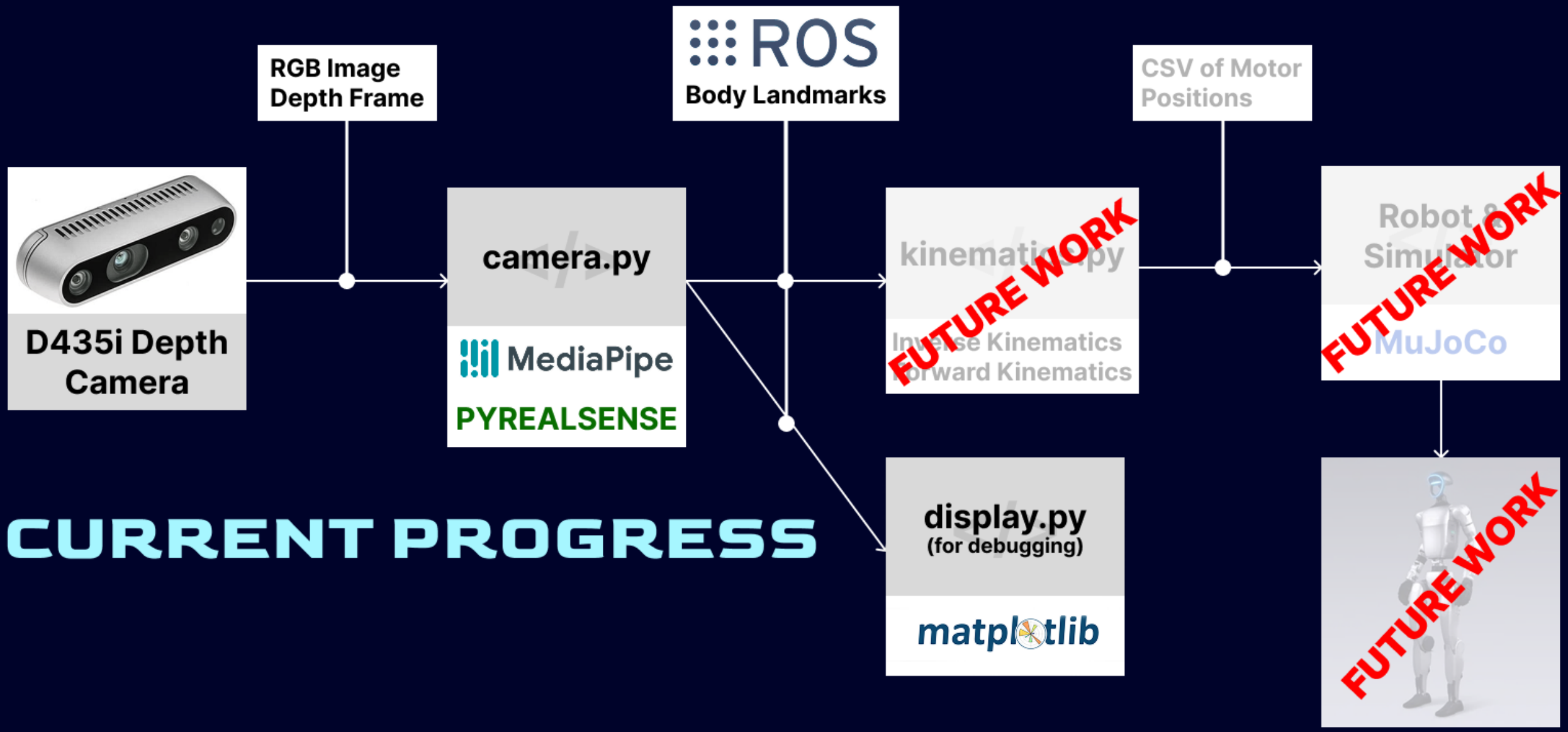
MuJoCo

display.py
(for debugging)

matplotlib



CURRENT PROGRESS



CURRENT PROGRESS

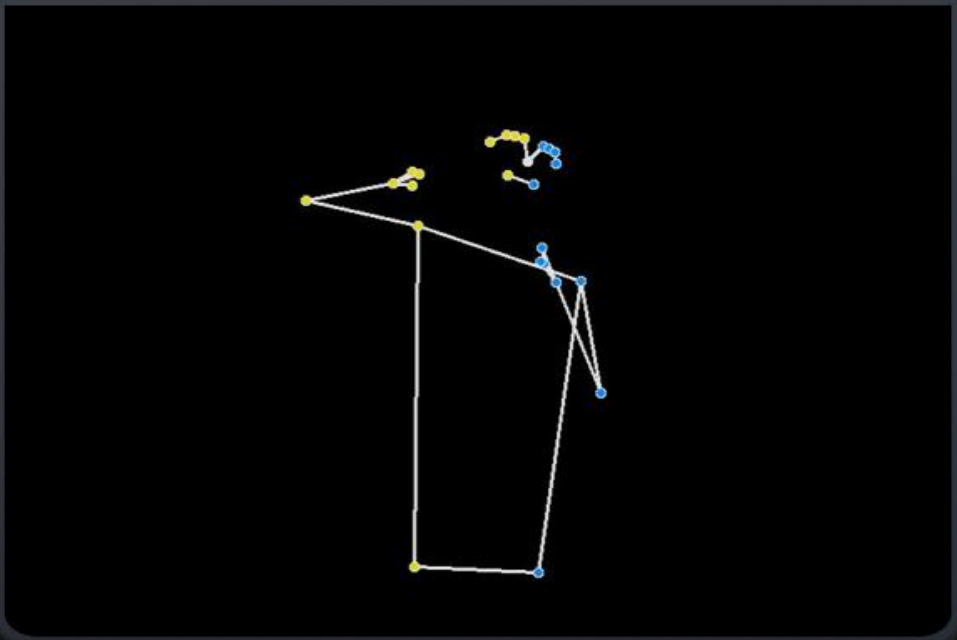
CURRENT PROGRESS – LATENCY

- Achieved 100ms end-to-end latency in initial tests
- Identified key bottlenecks:
  - MediaPipe Pose processing time (~60-80ms/frame)
  - ROS message latency
    - MediaPipe processes frames faster than ROS can read them
  - Computation/Matplotlib draw time
- Implemented asynchronous processing pipeline (prior drawing)

```
landmarker.detect_async(mp_image, int(time.time() * 1000))
```

CURRENT PROGRESS – VISUALIZATION

- Printing coordinate outputs / saving identified points to file
- 3D positional plotting with Matplotlib
- Implemented ROS-based coordinate display node
- Depth based filter visualization for RealSense camera
- **Demos at the end of the presentation!**



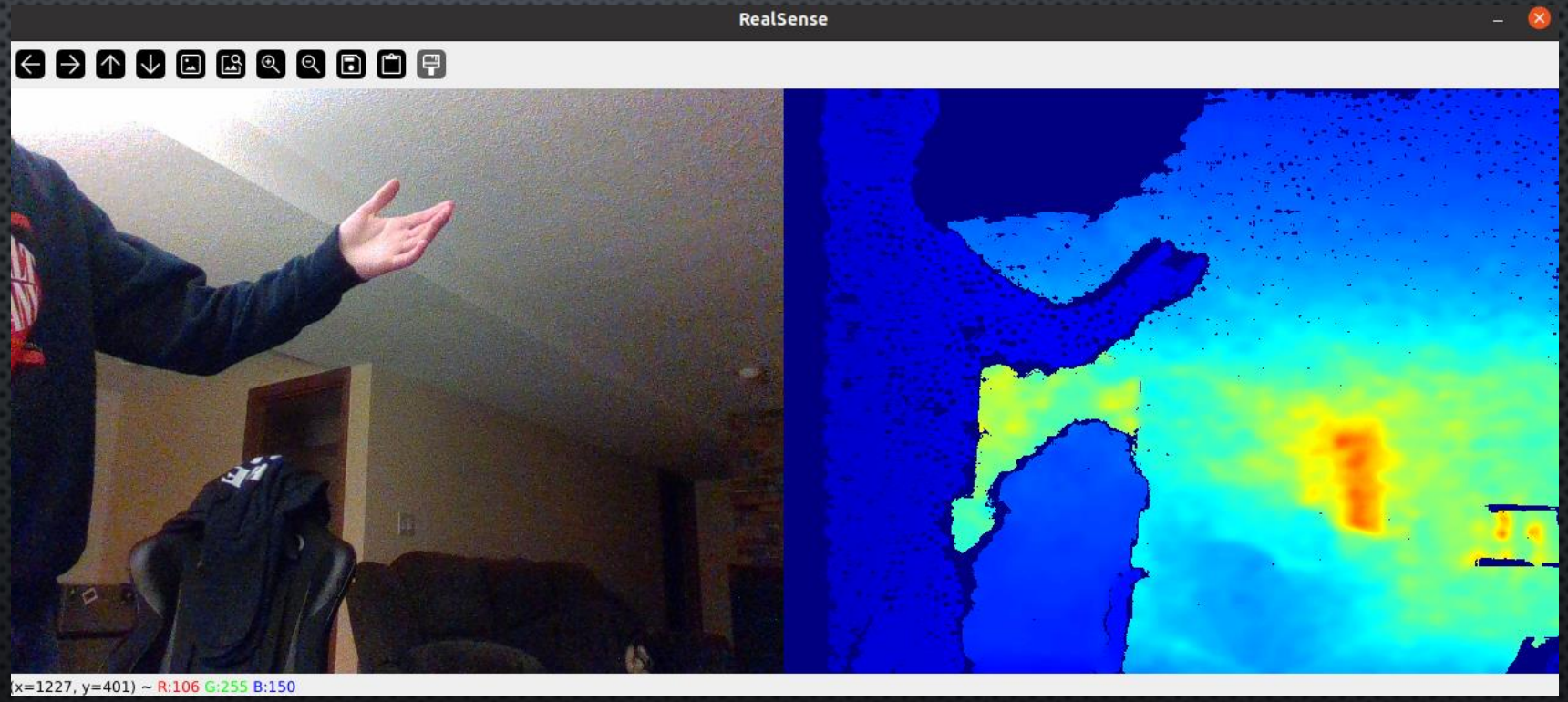
CURRENT PROGRESS – CONVERTING POINTS

- SUCCESSFULLY CONVERTED 2D MEDIAPIPE POSE LANDMARKS TO 3D WORLD COORDINATES USING INTEL REALSENSE DEPTH CAMERA DATA
- ESTABLISHED ROS MESSAGE PUBLISHING FOR BODY COORDINATES (HIPS, SHOULDERS, ELBOWS, WRISTS, HANDS)
- MATRIX TRANSFORM PIPELINE IS FUTURE WORK

```
float32[3] nose
float32[3] left_hip
float32[3] left_shoulder
float32[3] left_elbow
float32[3] left_wrist
float32[3] left_pinky
float32[3] left_index
float32[3] left_thumb
float32[3] right_hip
float32[3] right_shoulder
float32[3] right_elbow
float32[3] right_wrist
float32[3] right_pinky
float32[3] right_index
float32[3] right_thumb
```

CURRENT PROGRESS – DEPTH CAMERA

- D435I DEPTH CAMERA
- PLUG & PLAY
- READ ALIGNED DEPTH FRAME DATA BASED ON MEDIAPIPE LANDMARKS
- REASONABLY CONSISTENT



Challenges & Issues

- Meeting with client: Dr. Weng is very busy
- Depth camera alignment
- Delay in getting access to the MuJoCo simulator
- ROS Versioning
- Callback issues:
  - GUI vs. Main thread for Mediapipe and ROS
  - High order function to form closure on depth data

```python
def make_callback(depth_frame):
    def callback(result: PoseLandmarkerResult, output_image : mp.Image, timestamp_ms: int):
        if len(result.pose_landmarks) == 0:
            return

        relevant_landmarks = get_relevant_landmarks(result)

        landmarks_with_depth = []
        for point in relevant_landmarks:
            x_coord = round(point.x * 640)
            y_coord = round(point.y * 480)
            if x_coord >= CAMERA_WIDTH or x_coord < 0 or y_coord < 0 or y_coord >= CAMERA_HEIGHT:
                print(f"OUT OF BOUNDS ({x_coord}, {y_coord})")
                landmarks_with_depth.append((x_coord, y_coord, 0))
                return

            z_coord = depth_frame.get_distance(x_coord, y_coord)
            landmarks_with_depth.append((x_coord, y_coord, z_coord))
            print(f"READ ({int(time()*1000) - timestamp_ms}ms): {landmarks_with_depth}")

        msg = landmarks()
        msg.left_hip = landmarks_with_depth[0]
        msg.right_hip = landmarks_with_depth[1]
        # ... set remaining landmarks manually here, omitted for screenshot
        pub.publish(msg)
```

FUTURE WORK

- Forward and Inverse Kinematics
- Simulation (Unitree + MuJoCo + ROS)
- Proportion Transforms
- Implementing it on the Robot
- Safe Zones

FUTURE WORK

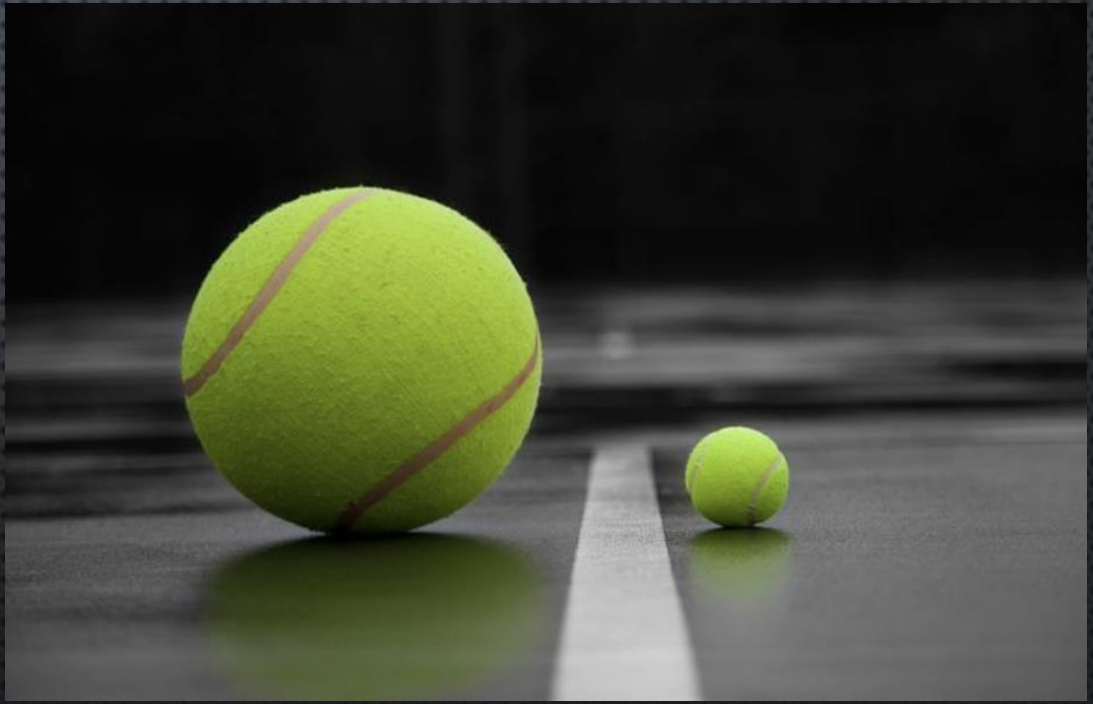
- Forward and Inverse Kinematics

FUTURE WORK

- SIMULATION (UNITREE + MUJOCO + ROS)

FUTURE WORK

- Proportion Transforms

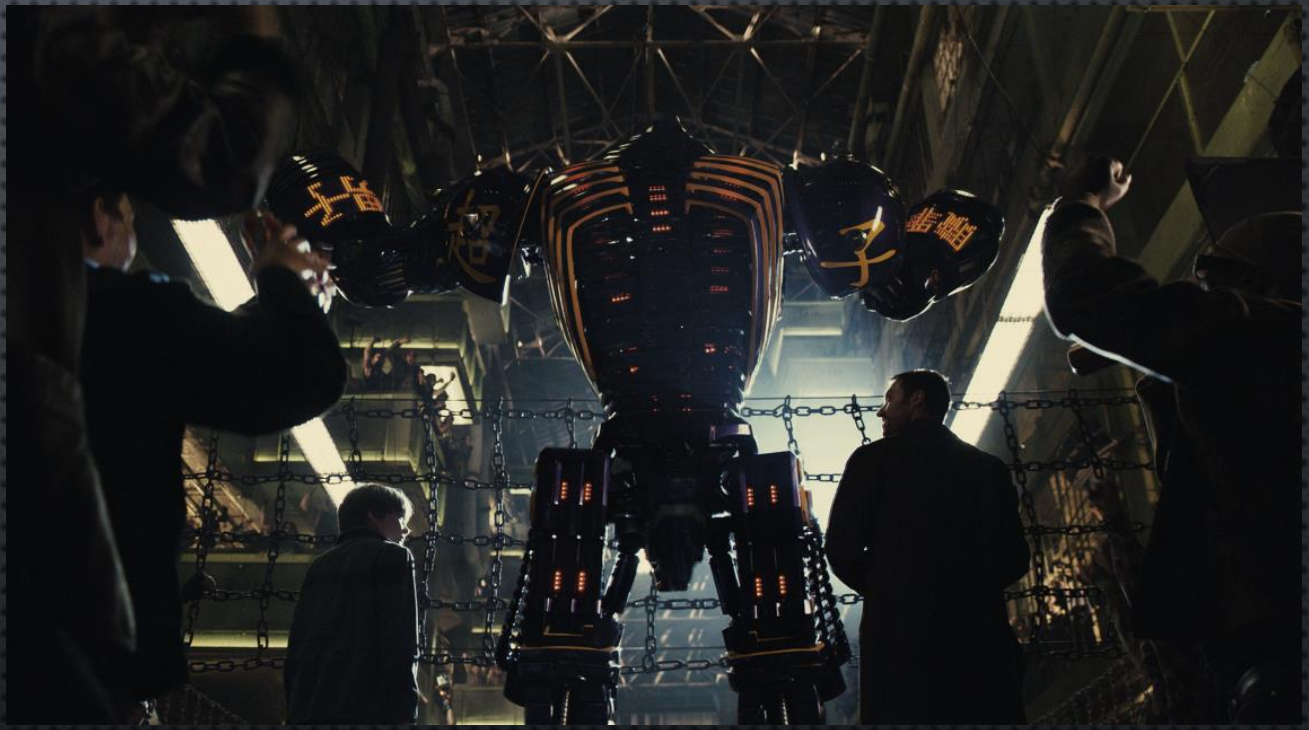


FUTURE WORK

- IMPLEMENTING IT ON THE ROBOT

FUTURE WORK

- SAFE ZONES



REMAINING WORK TIMELINE (I WILL ADD IMAGE)

- Weeks 5-7 inverse kinematics and csv
- Weeks 8-10 live simulation testing
- Weeks 11-14 working w/ robot and stretch goals

DEMO

1. Depth Camera Grays out distances of more than 3 meters
2. Depth Camera + Pose Detection + ROS Communication + Matplotlib