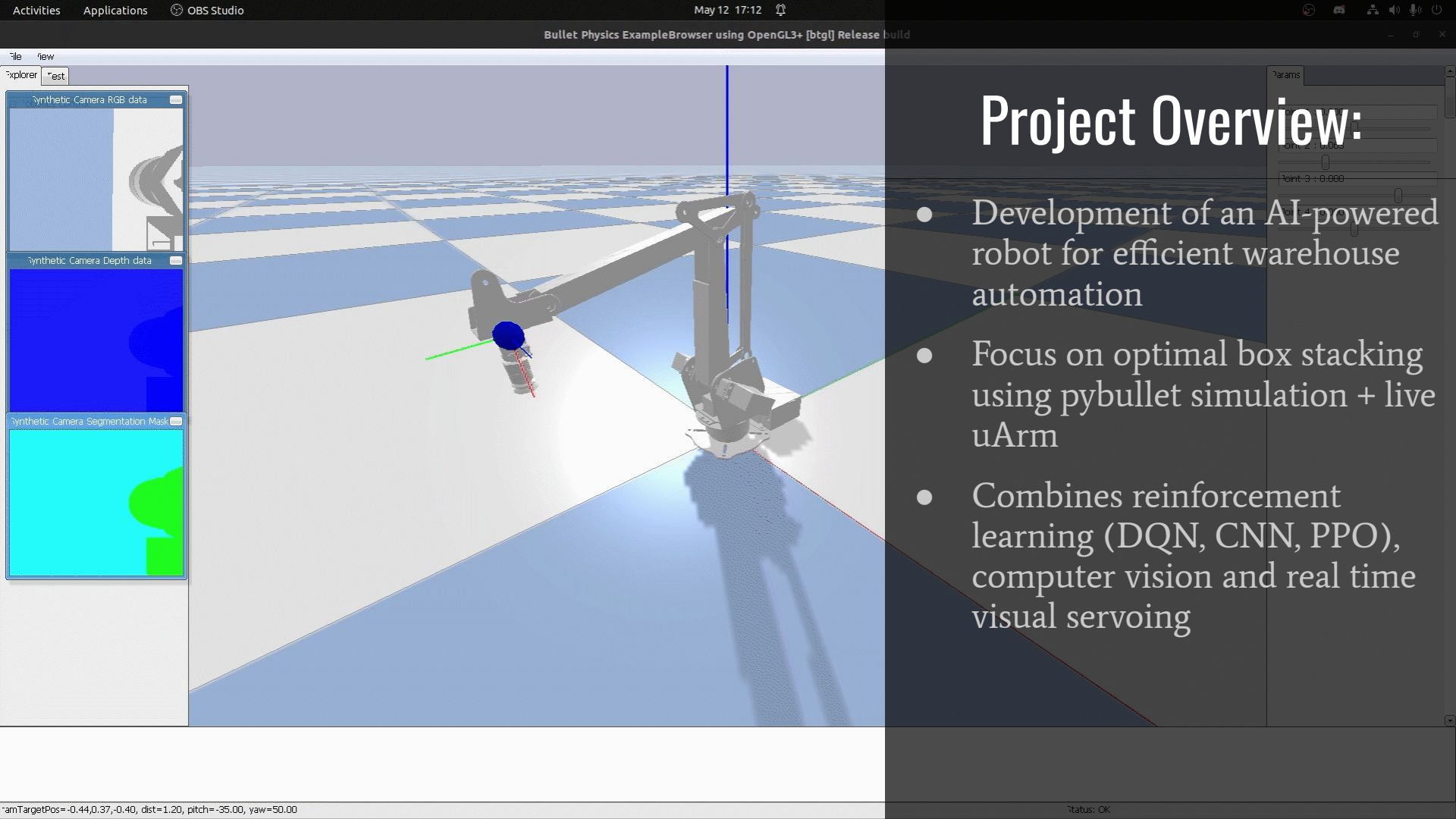
STACKBOT

Progress Update

Lauren S., Luis P., Matthew T., Michele L., & Minh N.



Project Overview:

- Development of an AI-powered robot for efficient warehouse automation
- Focus on optimal box stacking using pybullet simulation + live uArm
- Combines reinforcement learning (DQN, CNN, PPO), computer vision and real time visual servoing

Key Technologies:

- Robotics:
  - uArm (PyBullet, Physical Arm)
- AI:
  - Deep Q-Network (DQN)
  - Convolutional Neural Networks (CNN)
  - Proximal Policy Optimisation (PPO)
- Simulation: PyBullet for realistic physics and motion control and quick training
  - Gazebo has better physics but takes longer to train
- Software:
  - Python (Simulation and Training)
  - C++ (Arm Control)
  - ROS2
  - Robotics Toolbox

Project Goals:

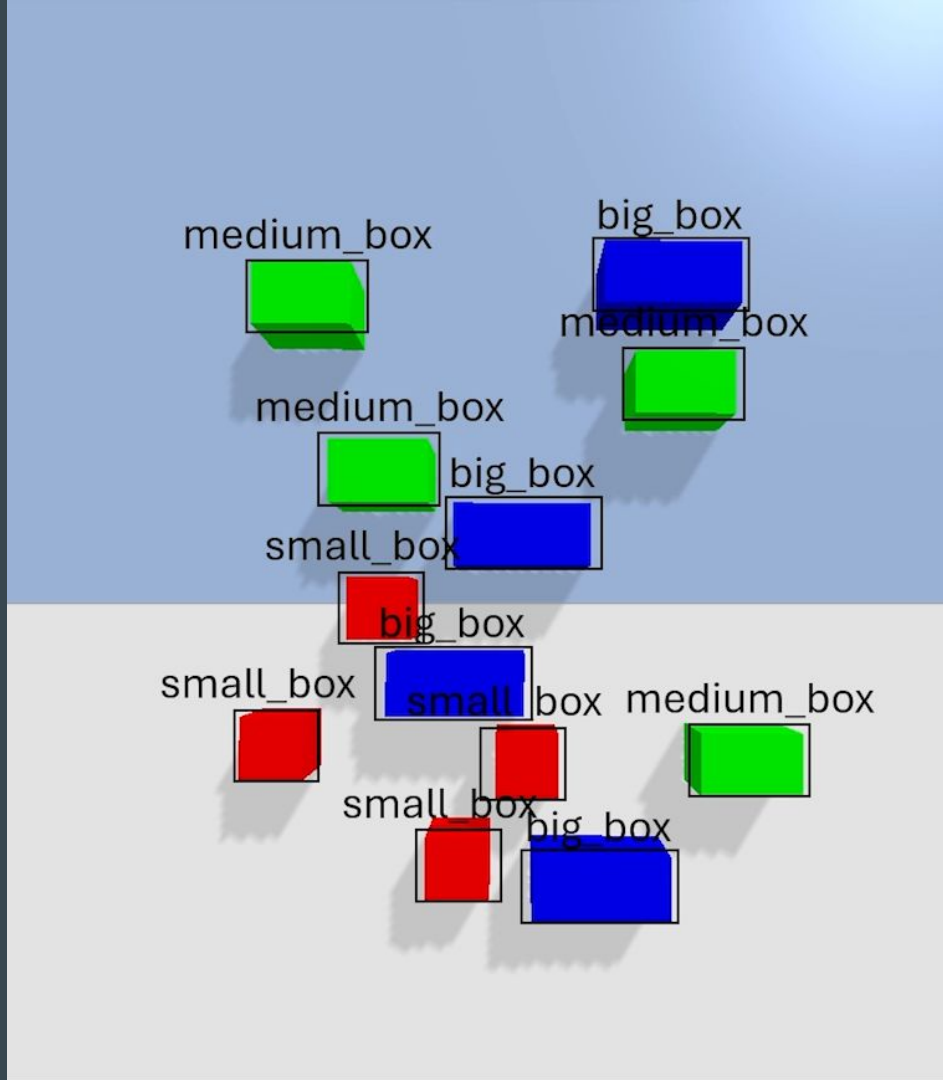
- Efficient, collision-free stacking
- Real-time decision-making
- Scalability to different warehouse environments

Current Status:

- Environment setup completed
- Basic stacking logic implemented
- Ongoing training and tuning for optimal performance

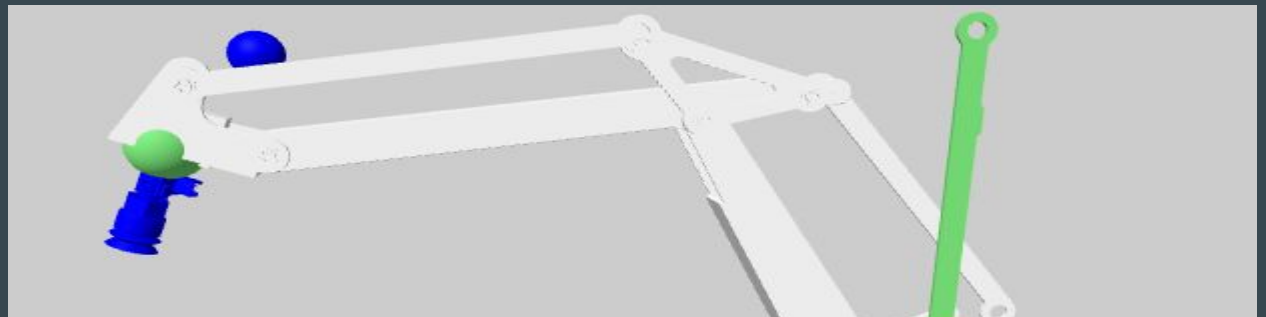
Using CNN for training

- CNN to process the RGB Image and detect boxes
- Boxes will have different colors, which categorises specific dimensions
- Train the CNN to also regress the box's center coordinates (cx, cy) in image pixels
- Feed those outputs into the uArm control node to plan pick-up

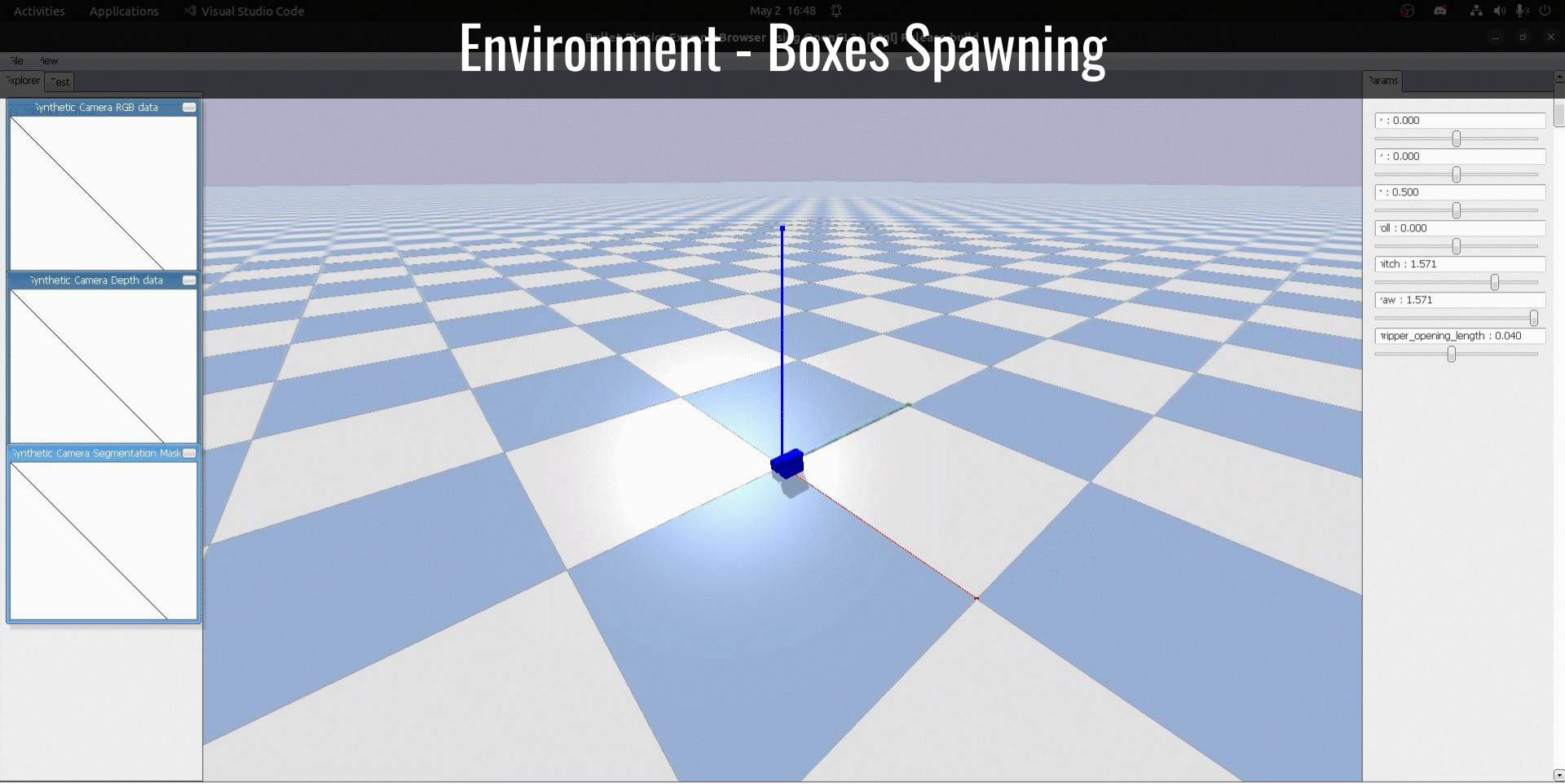


Concerns/Difficulties

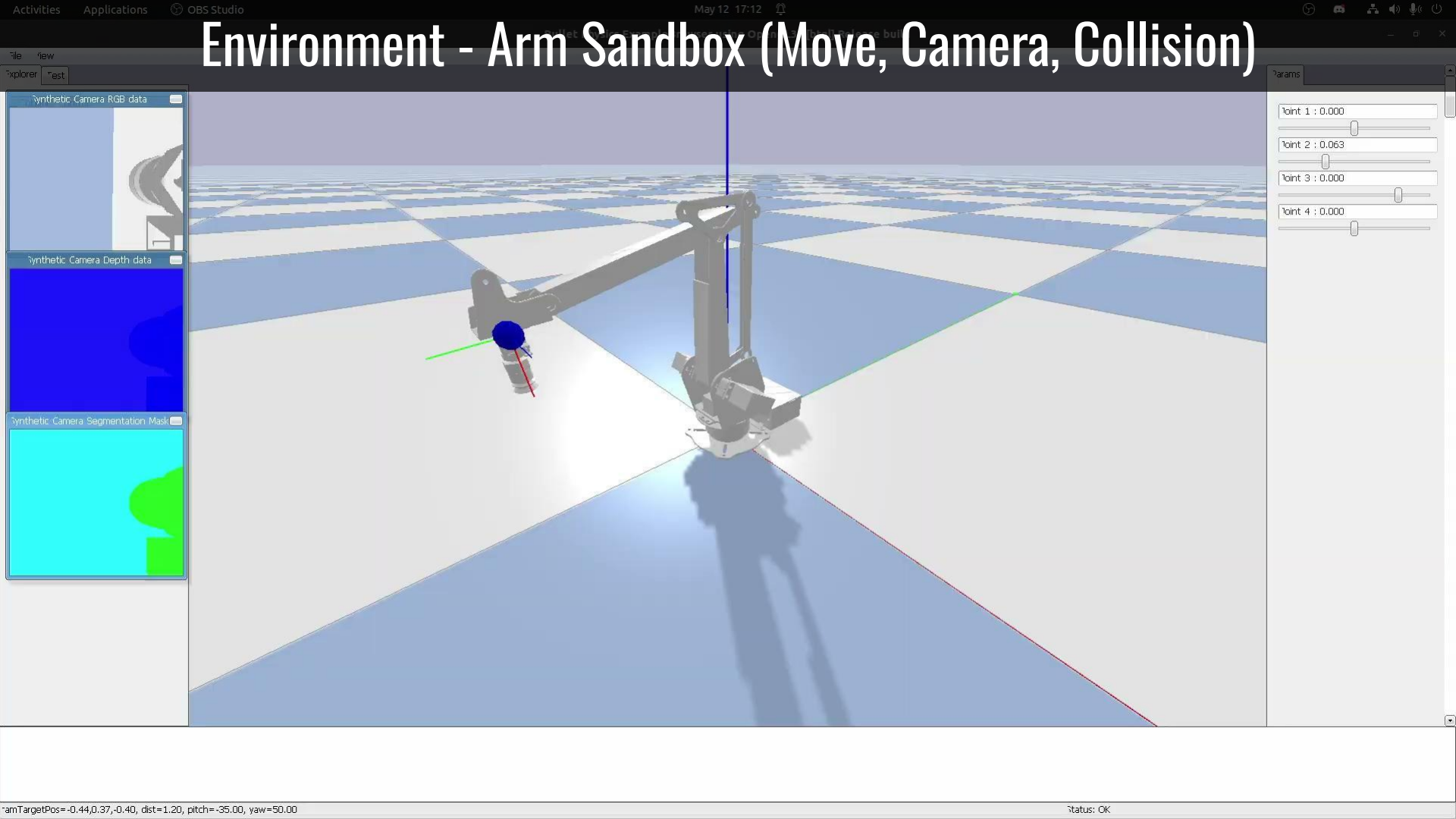
- DATA & LABELS: Get box images and marking its true center, while recognising its dimension
- LIGHTING & COLOUR: Handling shadows on different lighting
- CALIBRATION: Mapping pixels to robot XY with < 5mm error
- MULTI BOX CASES: Isolate and center one box when there are others in camera view.
- CAMERA MOUNTING: Having a stable, unobstructed view as relative to the arm as possible
- SIMULATED ROBOT MODEL: Constructing the correct joints/constraints to match with real robot



Environment - Boxes Spawning



Environment - Arm Sandbox (Move, Camera, Collision)



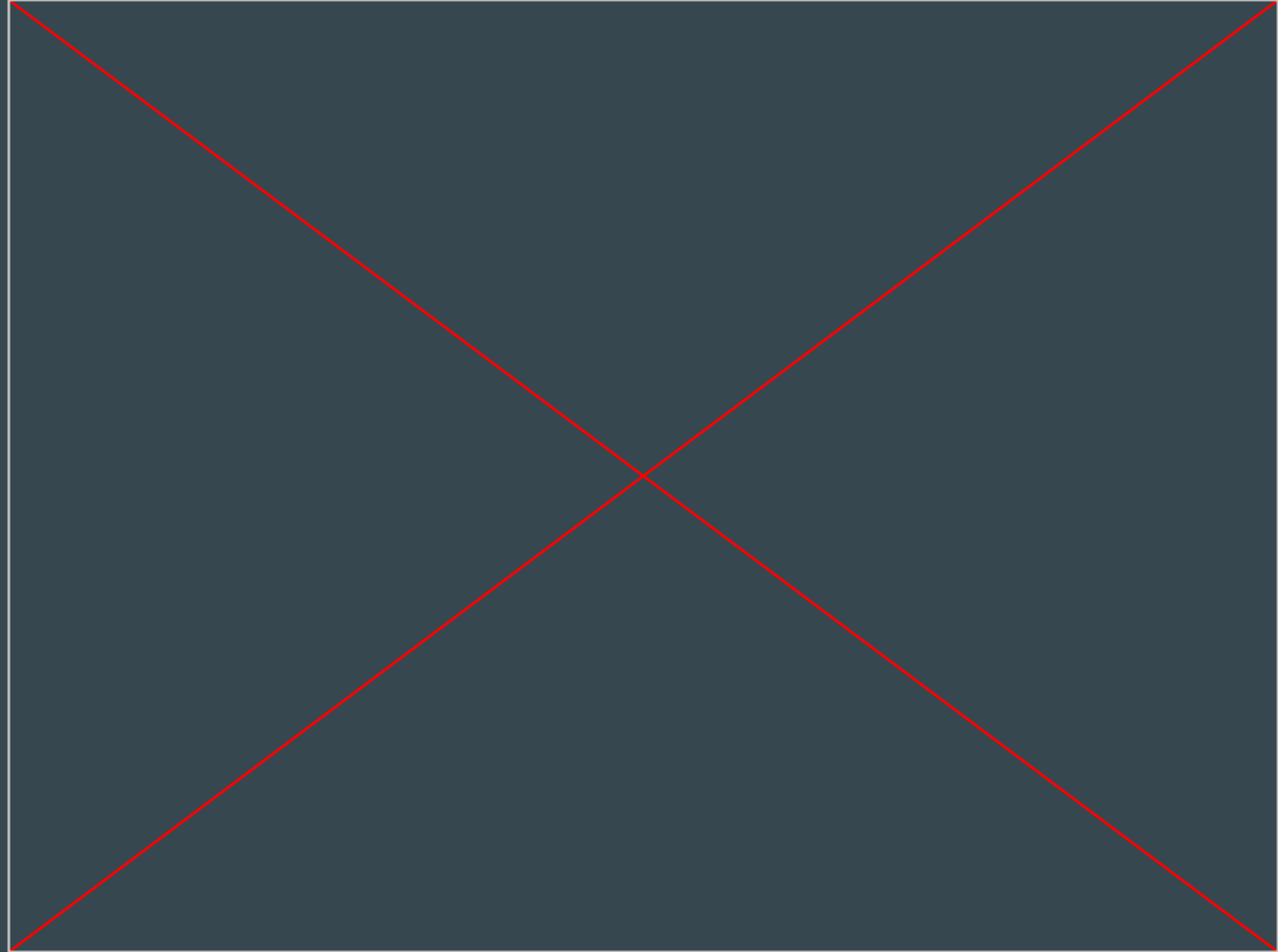
Challenges and Solutions:

- End-Effector Control: Adjusted RPY angles for accurate orientation
- Camera: RPY angles setup, shaking,
- Camera Calibration: Improved depth estimation with fine-tuned focal lengths
- Collision Detection: Enhanced reward functions to prioritise safety

Current Status:

- Environment setup completed
- Basic stacking logic implemented
- Ongoing training and tuning for optimal performance

Environment - RealBot Stacking



Next Steps:

- Fine-tune + Train DQN model for more efficient stacking
- Train CNN Model
- Integrate PPO model into the project
- Integrates the different models into one big boy
- Complete Accurate linked Cobot Model

Then:

- Implement real-world calibration with uArm hardware