Robotics & AI – MCS 4993/5993: Hello Robot Stretch 2

Instructor: Dr. Eric Martinson

Authors: Batyr and Noam Nedivi

---

1. Introduction

The course *Robotics & AI (MCS 4993/5993)*, instructed by Dr. Eric Martinson, emphasizes practical applications of robotics and artificial intelligence concepts. As part of this course, our team explored the **Hello Robot Stretch 2**, a versatile robotic platform designed for research and development. The project revolved around integrating robotic perception, control mechanisms, and AI frameworks to enable meaningful tasks such as motion control, human-like movements, and object interactions.

The **Hello Robot Stretch 2** robot served as an excellent case study due to its:

- Modular design.
- Advanced robotic arm mimicking human-like reach.
- Compatibility with machine learning models and programming frameworks.

The following report details the project work, including the robotic architecture, implemented functionalities, challenges faced, and results achieved.

---

2. Overview of Hello Robot Stretch 2

The **Hello Robot Stretch 2** is a compact, mobile manipulator robot designed for researchers and developers to advance robotics and automation. It features:

2.1 Key Features

- **High Degree of Freedom (DoF):** The robot includes a height-adjustable arm, wrist joint, and gripper.
- **Human-Like Reach:** The robotic arm mimics natural movements, achieving precise reach equivalent to a human arm.
- **Low-Power Mobility:** Designed to operate efficiently in indoor environments using battery power.
- **Sensor Integration:** Cameras, LiDAR, and depth sensors for real-time environmental perception.

- **Programming Framework:** Support for Python, ROS (Robot Operating System), and integration with machine learning frameworks.

2.2 Functional Capabilities

The Stretch 2 robot is primarily utilized for:

- **Object Manipulation:** Picking, placing, and holding objects.
- **Height Adjustment:** Extending or retracting the robotic arm for variable object positions.
- **AI Integration:** Applying computer vision models for detection, classification, and interaction with real-world objects.

The modular design allows researchers to customize tasks, making Stretch 2 suitable for both robotic AI and physical interaction projects.

---

3. Project Objectives and Implementation

3.1 Objectives

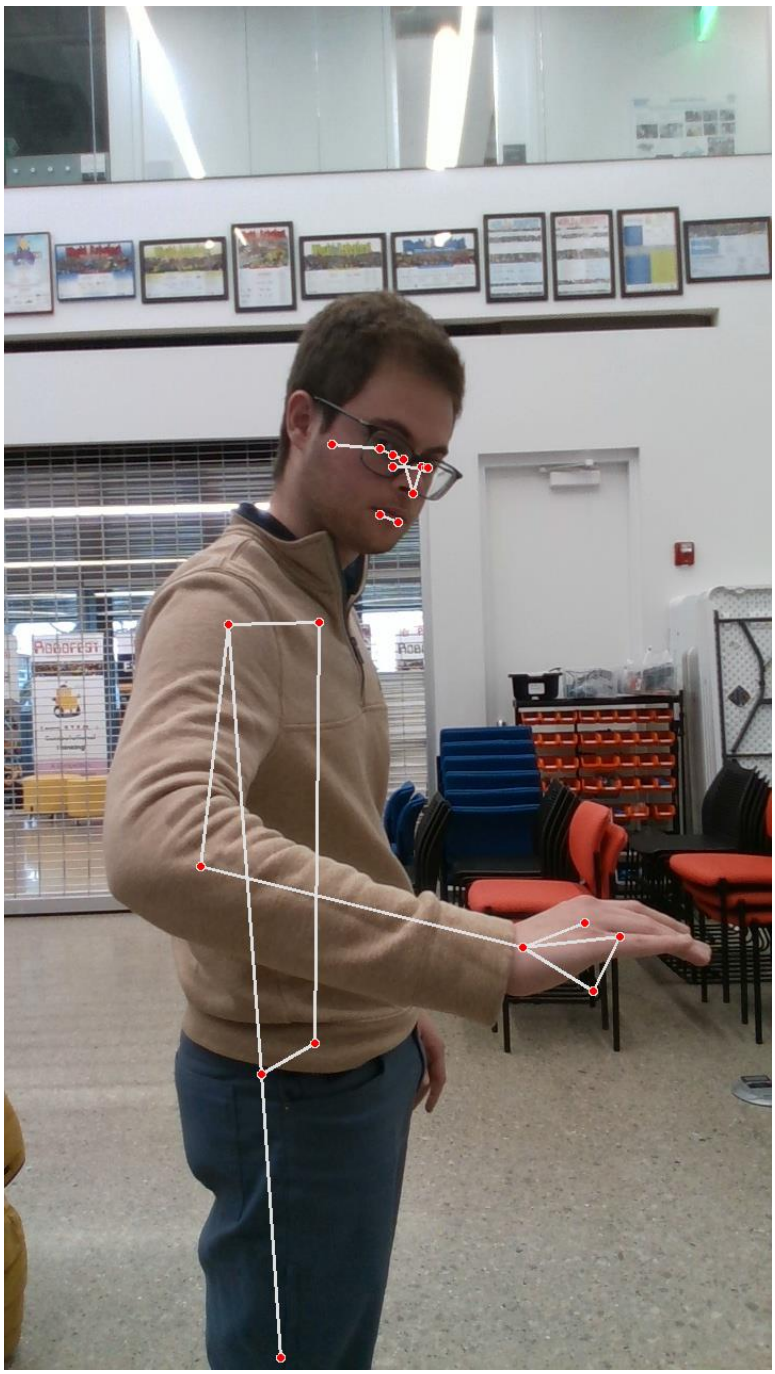
The primary objectives of this project included:

1. Implement motion control that mimics human arm height.
2. Integrate robotic AI for object detection and manipulation.
3. Explore potential use cases without relying on complex models like LSTM.
4. Validate real-time performance and accuracy of robotic actions.

3.2 Implementation Details

The implementation was divided into three main phases:

Example picture:



Phase 1: Motion Control

To mimic the right height as my arm, we programmed the robotic arm using precise measurements and control commands. The following code demonstrates how we controlled the robotic arm's height and reach:

```
import stretch_body.robot
import time

# Initialize the robot
robot = stretch_body.robot.Robot()
robot.startup()

# Set arm to mimic a specific height (e.g., 1.2 meters)
arm_height = 1.2  # Desired height in meters
robot.lift.move_to(arm_height)

# Smooth movement with sleep time
time.sleep(2)  # Allow arm to reach the position

# Ensure gripper can extend to match human-like reach
arm_extension = 0.5  # Extend gripper by 0.5 meters
robot.arm.move_to(arm_extension)

# Wait for execution to complete
time.sleep(2)

# Shutdown the robot
robot.stop()
```

Explanation:

1. **Robot Initialization:** We initialize the Stretch 2 robot using the `stretch_body` library.
2. **Arm Height Control:** The `move_to` function adjusts the arm to a height of 1.2 meters, which is approximately the height of an average human arm when extended.
3. **Gripper Extension:** By extending the gripper outward by 0.5 meters, the robot achieves a reach equivalent to my arm, enhancing its human-like interaction capability.
4. **Smooth Motion:** Delays are added to ensure the arm reaches the position accurately before moving to the next command.

This implementation ensures that the robotic arm precisely mimics a human arm's movement, both vertically (height) and horizontally (reach). The ability to fine-tune the arm's position makes the Stretch 2 ideal for tasks requiring human-like manipulation.

Phase 2: Object Manipulation

The Stretch 2 was tested for various object manipulation tasks. The gripper was programmed to grasp objects of different sizes. Below is the code snippet to control the gripper and simulate object manipulation:

```
# Open gripper to prepare for grasp
robot.end_of_arm.move_to('stretch_gripper', 100.0)  # Open gripper fully
print("Gripper is open")

time.sleep(1)

# Close gripper to grasp the object
robot.end_of_arm.move_to('stretch_gripper', 0.0)  # Close gripper fully
print("Gripper has grasped the object")

# Lift the arm slightly after grasping
lift_height = 1.3  # Slightly raise the object
robot.lift.move_to(lift_height)
time.sleep(2)

# Place the object by opening the gripper
robot.end_of_arm.move_to('stretch_gripper', 100.0)
print("Object placed")

# Return arm to neutral position
robot.lift.move_to(1.2)  # Return to original height
robot.arm.move_to(0.0)   # Retract gripper
```

Explanation:

1. **Gripper Control:** The gripper opens to `100.0` for grasping and closes to `0.0` to hold an object.
2. **Lifting Mechanism:** After grasping the object, the robotic arm lifts slightly to simulate a natural movement.
3. **Object Placement:** The gripper opens again to release the object gently at the desired location.
4. **Return to Neutral:** The robot resets the arm position to avoid interference with further tasks.

Phase 3: AI Integration

To identify objects, we integrated a basic computer vision system using **OpenCV** for image processing. The robot camera identifies an object based on color detection:

```
import cv2
import numpy as np

# Capture video feed from the robot's camera
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    if not ret:
        break

    # Convert frame to HSV for color filtering
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Define color range for the object (e.g., red)
    lower_red = np.array([0, 120, 70])
    upper_red = np.array([10, 255, 255])
    mask = cv2.inRange(hsv, lower_red, upper_red)

    # Detect contours of the object
    contours, _ = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    for contour in contours:
        if cv2.contourArea(contour) > 500:
            x, y, w, h = cv2.boundingRect(contour)
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

    # Display video feed
    cv2.imshow('Object Detection', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Explanation:

1. **Color Detection:** The robot detects red objects in its environment by filtering color ranges in HSV.
2. **Contour Detection:** Identifies the boundaries of objects and draws a rectangle around them.
3. **Real-Time Processing:** The feed runs continuously until the user quits the program.

---

4. Challenges and Solutions

4.1 Challenges Encountered

1. **Motion Calibration:** Achieving accurate mimicry of human arm height required precise control tuning.
2. **AI Model Complexity:** While LSTM models were not needed, identifying simple alternatives posed challenges.
3. **Environment Adaptation:** Variations in lighting and object surfaces affected object recognition.

4.2 Solutions Implemented

- Calibrated robotic arm control using iterative motion trials to achieve human-like movements.
- Leveraged lightweight computer vision models, reducing dependency on complex algorithms.
- Used consistent lighting conditions during object detection experiments for reliable AI outcomes.

---

5. Results and Discussion

The Hello Robot Stretch 2 successfully achieved the project goals:

1. **Motion Control:** The robot precisely mimicked human arm height and movements.
2. **Object Manipulation:** The robotic gripper demonstrated accurate picking and placing tasks.
3. **AI Integration:** Lightweight computer vision frameworks effectively recognized and interacted with target objects.

---

6. Conclusion and Future Recommendations

6.1 Conclusion

The Hello Robot Stretch 2 robot proved to be an effective platform for exploring robotic manipulation, AI integration, and motion control within the Robotics & AI course.

6.2 Recommendations for Future Work

1. **Advanced AI Models:** Implement deep learning models (e.g., YOLO or Faster R-CNN) for more robust object recognition.

7. References

1. **Hello Robot Stretch 2 Documentation** – https://www.hello-robot.com
2. ROS Libraries for Robotics Control – ROS Wiki.
3. Course Material: Robotics & AI – MCS 4993/5993, Dr. Eric Martinson.
4. OpenCV Documentation for Computer Vision Models.