AGRISORT

AUTOMATED SORTING OF FRUITS AND VEGETABLES

PRAHARSHA K
NATHAN PUSKURI
BIMLENDRA RAY

GROUP 11
UNIVERSITY OF TEXAS AT DALLAS

INTRODUCTION

- ## OVERVIEW:
  - Automated system utilizing a **Fetch manipulator** for **sorting fruits and vegetables into bins.**

- ## TECHNOLOGY USED
  - ROS (Robot Operating System) and Gazebo for implementation and simulation
  - AWS Rekognition

- ## MOTIVATION
  - Modernize agricultural sorting processes
  - Enhance productivity and reduce reliance on manual labor
  - Minimize food waste and operational costs

IMAGE CLASSIFICATION

WHERE DO WE GET THE IMAGES FROM?

PUBLISH-SUBSCRIBE ARCHITECTURE

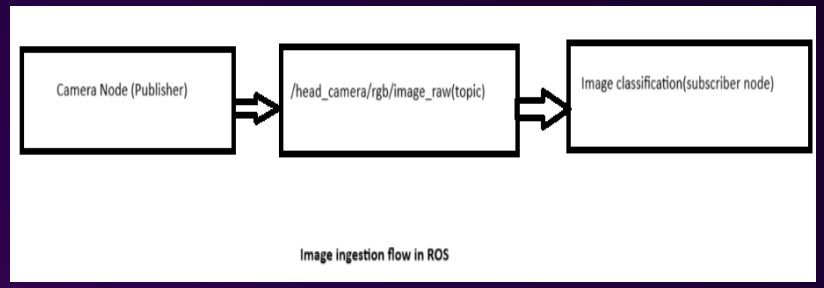


Image ingestion flow in ROS

- A messaging pattern
- Publishers send messages to communication channel (topic)
- subscribers listen to messages from the channel

Topics in ROS:

- **/head_camera/rgb/image_raw** for raw color images.
- /head_camera/depth/image_raw for raw depth data.
- /head_camera/camera_info for camera calibration details.

REALTIME IMAGES

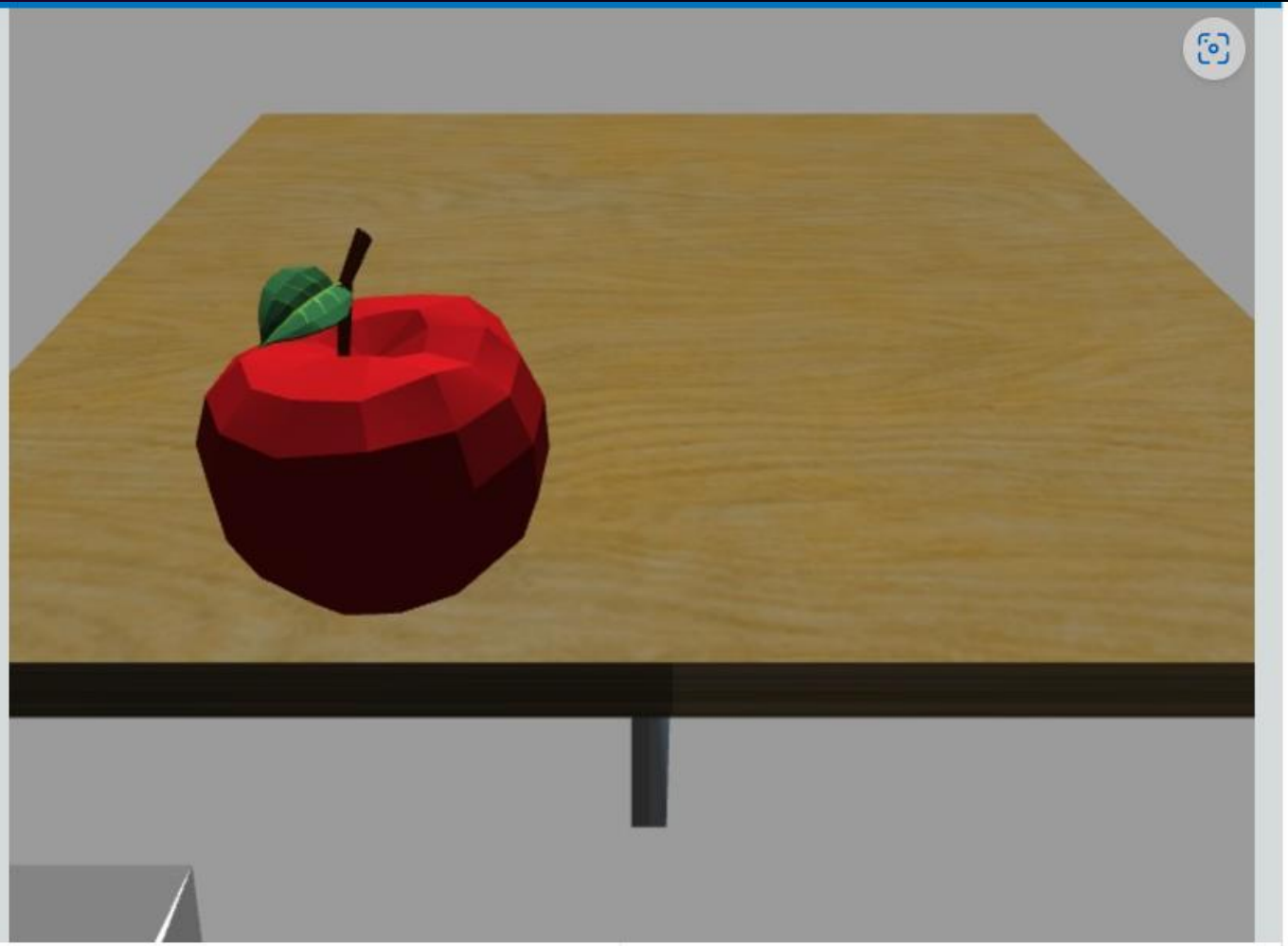
AWS REKOGNITION

1. Cloud-based Image and video analysis service that helps computers understand what's inside an image
2. Can identify objects, people, text, scenes, people and more

Steps:

1. Send an image through API call
2. Image processing
3. Get labels and confidence scores


```
{
  "Labels": [
    {
      "Name": "Apple",
      "Confidence": 98.6
    },
    {
      "Name": "Banana",
      "Confidence": 92.3
    }
  ]
}
```



Choose a sample image

Use your own image

Upload or drag and drop

Image must be .jpeg or .png format and no larger than 5MB. Your image isn't stored

Check whether we support your label

Enter a label name

Results

Apple	99.9 %
Food	99.9 %
Fruit	99.9 %
Plant	99.9 %
Produce	99.9 %
Furniture	99.8 %

Show more

Request

Response

IMAGE CLASSIFICATION

```python
# Initialize the ROS node
rospy.init_node('camera_image_saver', anonymous=True)

# Define the image topic to subscribe to
image_topic = "/head_camera/rgb/image_raw"

# Create a subscriber to the image topic
rospy.Subscriber(image_topic, Image, image_callback)

# Keep the script running to listen for incoming messages
rospy.spin()
```

IMAGE CLASSIFICATION

```
rekognition_client = boto3.client('rekognition',
                                  region_name='us-east-1',
                                  aws_access_key_id=aws_access_key_id,
                                  aws_secret_access_key=aws_secret_access_key)
```

IMAGE CLASSIFICATION

```python
# Call AWS Rekognition to detect labels in the image
response = rekognition_client.detect_labels(
    Image={'Bytes': image_bytes},
    MaxLabels=10,  # Maximum number of labels to return
    MinConfidence=75  # Confidence threshold (optional)
)
labels = [label['Name'].lower() for label in response['Labels']]
```

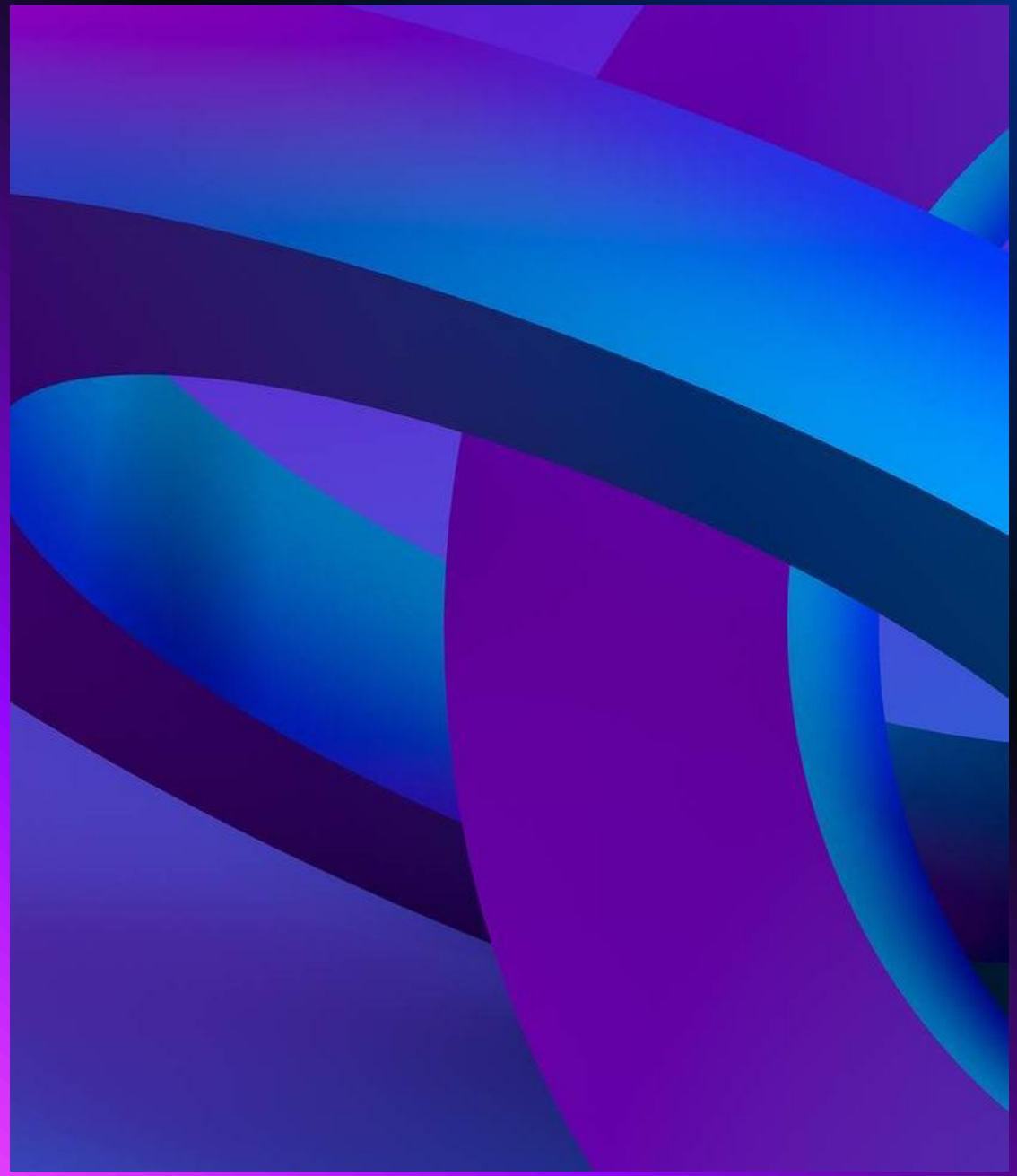


IMAGE CLASSIFICATION

How we achieved classification ?

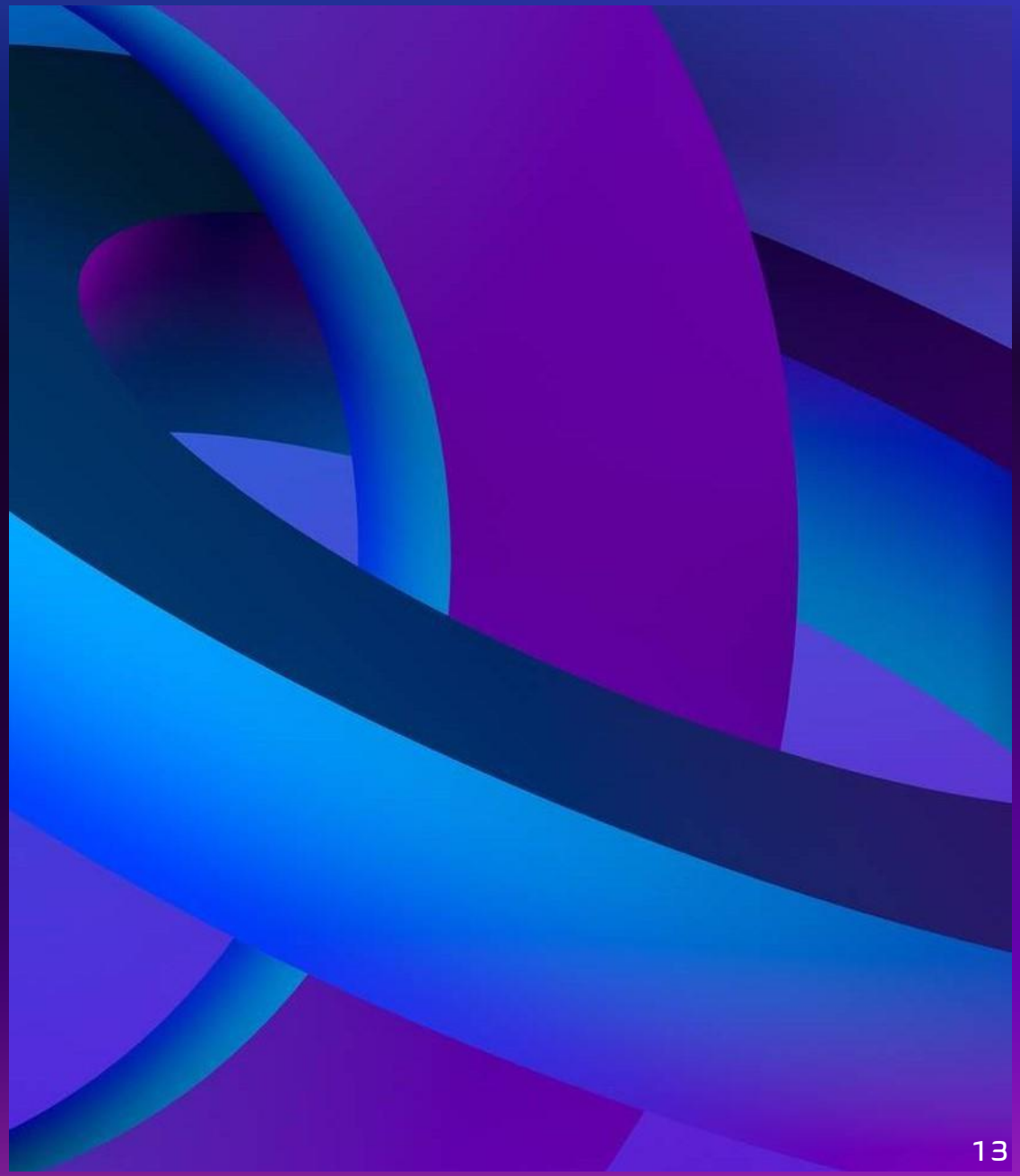
- Captured the environment using the Fetch robot's front camera.
- Used the AWS Rekognition API to classify objects in the captured images.

MOVING THE OBJECTS

- Inverse Kinematics/ MoveIt

DEMO

HTTPS://YOUTU.BE/VV4QKLCOZDG



CHALLENGES

Grasplt

Image classification

- Frozen camera view from ROS topic
- Object detection accuracy depends on orientation of object

THANK YOU