ICP Robotics

[InnovationCentralPerth/intelligent-pick-place-robot: Develop a speech controlled pick and place robot with stereo machine vision using gen AI processing of commands to robot controller]()
[Edge Impulse]()

- Voice Operated Claw to pick up things
  - Check arduino
  - Computer Vision: Object Detection & Localization (W4)
    - Camera: Luxonis OAK-D
    - Object detection model: trained using edge impulse studio
  - Path Planning & Claw Control, Grip (W6)
    - Arduni Braccio Claw
    - Raspberry pi 5: host ROS 2 nodes and control

- Milestone 1.3: Computer vision W4
  - Data collection: capture images and upload to edge impulse studio to train it ✅
  - Model testing: test the vision model ❌
  - Model deployment: deploy the model into raspberry ❌
- Milestone 1.4: Localization W5
  - Build & verify a visual robot model ❌
  - Generate configuration using the MoveIt Setup Assistant 2.0 ❌
  - Arduino Braccio++ Controller Firmware ❌
- Milestone 1.5: Finalization W6
  - Launch ROS 2 Nodes ❌
  - Launch pick_n_place node ❌
- Milestone 2.1: Voice to text ❌
- Milestone 2.2: Text to Action Prompt ❌

===

- Milestone 1.1: Installation of Software W2 ✅
  - Install raspberry pi imager (raspberry OS) ✅
  - ROS 2 Humble (multiple installation steps) ✅
  - MoveIt 2 (kinematics and 3d perception) ✅
- Milestone 1.2: ROS operations W3 ✅
  - DepthAI ROS: use camera and get rgbd ✅
  - micro-ROS: communicate with Arduino Nano ✅

==
ROBOTICS PICK AND PLACE PROJECT - SESSION CONTEXT
==
Date: July 9, 2025
Platform: Raspberry Pi 5 (Debian Bookworm)
User: icp
Working Directory: /home/icp/

==
PROJECT OVERVIEW
==
Developing a robotic pick and place system using:
- Raspberry Pi 5 as main controller
- Arduino Nano RP2040 Connect on Braccio Carrier board
- Braccio robot arm kit for manipulation
- OAK-D Lite camera for computer vision
- ROS2 Humble for system integration
- micro-ROS for Arduino-ROS2 communication

==
HARDWARE SETUP STATUS
==
✅ Raspberry Pi 5
  - OS: Debian Bookworm (Linux 6.12.25+rpt-rpi-2712)
  - ROS2 Humble installed and working
  - Connected to network (IP: 10.130.15.118)

✅ Arduino Nano RP2040 Connect + Braccio Carrier
  - Hardware detected (USB device: Intel Movidius MyriadX)
  - Mounted on Braccio robot arm kit
  - Ready for micro-ROS firmware development

✅ OAK-D Lite Camera
  - Hardware detected (USB 3.0 SUPER speed)
  - Device ID: 1944301041C8EF1200
  - RGB + Depth streaming functional
  - DepthAI library v2.30.0.0 installed

✅ Remote Access
  - TigerVNC server running on port 5901
  - Password: raspberry
  - Connection: 10.130.15.118:5901

==
SOFTWARE ENVIRONMENT STATUS
==

ROS2 Humble Workspace: ~/ros2_humble/
- Core ROS2 installation complete
- All essential packages installed
- Environment sourced via ~/ros2_humble/install/setup.bash

micro-ROS Workspace: ~/microros_ws/
- micro_ros_agent: ✅ Built and functional

- micro_ros_msgs: ✅ Message types available
- micro_ros_setup: ✅ Tools for firmware development
- Agent ready for serial communication with Arduino

OAK-D Workspace: ~/dai_ws/
- depthai_descriptions: ✅ URDF models
- depthai_ros_msgs: ✅ Message interfaces
- vision_msgs: ✅ Vision processing messages
- Test scripts: oak_headless_test.py, oak_simple_test.py
- Test outputs: ~/dai_ws/test_output/ (organized by timestamp)

==
CURRENT CAPABILITIES
==

✅ Computer Vision
- RGB camera streaming (640x480 @ 30fps)
- Depth camera streaming (480p stereo)
- Image capture and processing
- OpenCV integration working

✅ Robot Communication
- micro-ROS agent ready for Arduino communication
- Serial transport configured (/dev/ttyACM0)
- Message types available for joint states and commands

✅ Development Environment
- Python 3.11 with robotics libraries
- colcon build system
- Git repositories cloned and configured
- VNC remote access for GUI applications

==
NEXT DEVELOPMENT PHASES
==

🔄 Phase 1: Arduino Firmware Development
- Install micro-ROS Arduino library
- Develop joint state publisher for Braccio arm
- Implement servo control and safety limits
- Create command subscriber for arm movements

🔄 Phase 2: Computer Vision Integration
- Implement object detection using OAK-D
- Develop spatial coordinate mapping
- Create pick target identification system
- Integrate depth data for 3D positioning

🔄 Phase 3: ROS2 Integration
- Create ROS2 nodes for vision processing
- Implement motion planning for pick/place
- Develop state machine for operation sequence
- Add safety monitoring and error handling

🔄 Phase 4: System Integration
- Connect all components through ROS2
- Implement complete pick and place workflow
- Add user interface and monitoring
- Testing and optimization

==
KEY COMMANDS FOR CONTINUATION
==

Start Development Environment:

```
cd ~/
source ~/ros2_humble/install/setup.bash
```

Start micro-ROS Agent:

```
cd ~/microros_ws
source install/local_setup.bash
ros2 run micro_ros_agent micro_ros_agent serial --dev /dev/ttyACM0
```

Test OAK-D Camera:

```
cd ~/dai_ws
python3 oak_headless_test.py
```

Manage VNC Server:

```
~/vnc_service.sh {start|stop|restart|status}
```

Build ROS2 Packages:

```
colcon build
```

==
IMPORTANT FILE LOCATIONS
==

```
~/ros2_humble/              - ROS2 Humble installation
~/microros_ws/              - micro-ROS workspace
~/dai_ws/                   - OAK-D camera workspace
~/dai_ws/test_output/       - Camera test images
~/.vnc/                     - VNC configuration
~/PROJECT_CONTEXT.txt       - This context file
```

==
CONTACT & COLLABORATION
==
This project is being developed collaboratively with Claude Code.
All development sessions should reference this context document.
Update this file as project progresses and new capabilities are added.

Last Updated: July 9, 2025
Session Status: VNC accessible, all core systems functional
Ready for Phase 1: Arduino firmware development

18072025

🔍 Root Cause Identified: Incomplete Impulse

The Edge Impulse project needs to be properly completed before it can be deployed. Here's what needs to be done:

🛠️ Edge Impulse Project Requirements:

1. Data Collection: Must have training data with camera/image inputs
2. Impulse Design: Must have proper image processing blocks configured
3. Model Training: Must be trained and show good accuracy
4. Model Deployment: Must be properly deployed for Linux ARM64

📋 Next Steps:

Option 1: Complete the Edge Impulse Project
- Log into Edge Impulse Studio
- Ensure your project has:
  - Training data with images
  - Proper impulse design (Image → Processing → Learning → Output)
  - Trained model with good accuracy
  - Deployment configured for Linux ARM64

Option 2: Use a Different Complete Project
- Try with a different, complete Edge Impulse project
- Or use a publicly available example project

Option 3: Test with HTTP Server Mode
Once the project is complete, you can test it with:
edge-impulse-linux-runner --run-http-server 4912

🎯 Current Status:

- ✅ OAK-D camera working perfectly
- ✅ UVC mode active
- ✅ Edge Impulse runner installed
- ❌ Edge Impulse project incomplete (needs training/deployment)

==
SESSION UPDATE: July 15, 2025 - Edge Impulse Integration Complete
==

🔄 COMPLETED IN THIS SESSION:
- ✅ Node.js 22.17.0 installed via NVM
- ✅ Edge Impulse Linux CLI v1.15.6 installed globally
- ✅ Build dependencies installed (gcc, g++, make, sox, gstreamer)
- ✅ depthai-python repository cloned and configured
- ✅ OAK-D camera UVC mode successfully activated
- ✅ Camera exposed as /dev/video0 and /dev/video1 (Luxonis UVC Cam)
- ✅ Edge Impulse runner functional (HTTP server mode working)

🎯 CURRENT SYSTEM STATUS:
Hardware Integration:
- OAK-D Lite Camera: ✅ Working (Serial: 1944301041C8EF1200)
- UVC Mode: ✅ Active (PID 41778: python3 examples/UVC/uvc_rgb.py)
- Video Devices: ✅ /dev/video0, /dev/video1 available
- Camera Resolution: 1920x1080 @ 60fps NV12 format

Software Stack:
- Node.js: v22.17.0 (via NVM)
- Edge Impulse CLI: v1.15.6
- DepthAI Python: v2.30.0.0
- TigerVNC: Running on port 5901
- ROS2 Humble: Ready for integration

Edge Impulse Integration:
- CLI Tools: ✅ Installed and working
- Camera Detection: ✅ "Luxonis Device" recognized
- Model Download: ✅ ARM64 models downloading successfully
- Issue Identified: ❌ Edge Impulse project incomplete (needs proper training)

📁 NEW FILE LOCATIONS:

```
~/dai_ws/depthai-python/         - UVC camera integration
~/dai_ws/depthai-python/bin/      - Python venv for UVC mode
~/.ei-linux-runner/models/        - Downloaded Edge Impulse models
~/.nvm/versions/node/v22.17.0/bin/ - Edge Impulse CLI tools
```

⚡ ACTIVE PROCESSES:
- PID 41778: python3 examples/UVC/uvc_rgb.py (UVC camera server)
- VNC Server: port 5901 (remote access)
- micro-ROS agent: Ready for Arduino communication

🔄 NEXT PHASE READY:
1. Complete Edge Impulse model training in Studio
2. Deploy trained model for Linux ARM64
3. Test real-time object detection with OAK-D
4. Integrate with ROS2 for pick-and-place operations

🛠️ RESUME COMMANDS:

```
# Check UVC camera status
ps aux | grep uvc_rgb
v4l2-ctl --list-devices

# Start Edge Impulse runner
edge-impulse-linux-runner

# Test HTTP server mode
edge-impulse-linux-runner --run-http-server 4912

# Access system remotely
VNC: 10.130.15.118:5901 (password: raspberry)
```

📋 KNOWN ISSUES:
- Edge Impulse project needs completion (training + deployment)
- CLI tools update available (can upgrade via npm)
- ROS2 repository GPG key warning (non-critical)

Last Updated: July 15, 2025
Session Status: Edge Impulse integration complete, ready for model deployment
Next Session: Focus on completing Edge Impulse model training

==
SESSION UPDATE: Aug 4, 2025 - Checking installations and packages
==
🔄 COMLPETED IN SESSION:
- ✅ Packages checked: ROS 2 Humble, MoveIt 2, DepthAI ROS, micro-ROS Agent
- ✅ oak_simple_test.py does not save, so made a new python file to work (press enter to update frame)
saves in folder: /home/icp/dai_ws/test_output

```
cd ~/dai_ws
python3 oak_pic.py
```

- ✅ Depth in greyscale, best object in grey with skincolor tone, dont use black
- ✅ use claud code

==
SESSION UPDATE: Aug 5-8, 2025 - Checking braccio++
==
🔄 COMLPETED IN SESSION:
- ✅ learning arduino braccio carrier datasheet Braccio Carrier | Arduino Documentation
- ✅ Installed Arduino Mbed OS Nano Boards, Arduino_Braccio_plusplus (1.3.2) and micro_ros_arduino
- ✅ braccio examples dont work its braccio++
- ✅ Using this or codes that will loopingly move the claw instead
- ✅ checking voltages on outputs and servos (7.4V is transferred but signal is not transferred), make blink with serial monitor
- ✅ testing voltage output of arduino, D8 7 0 1 from arduino schematic, continuity of wires from arduino to servo, 4,6,7,8,11,12 all are 0 others are 3.2V, 5V is correct
- ✅ removed 6 pin- middle GND, near arduino +7.4V, other side 1.3V, both same voltage

==
SESSION UPDATE: Aug 11-13, 2025 - Edge Impulse
==
🔄 COMLPETED IN SESSION:
- ✅ Data Collection 250 pictures
- ✅ Input to edge impulse studio and place bounding boxes
- ✅ Train data and test
- ✅ Collect, Train, Test more

==
SESSION UPDATE: Aug 18-21, 2025 - Braccio and Edge Impulse Downloading
==
🔄 COMLPETED IN SESSION:
- ✅ servo working with another arduino [Arduino Code] → [Nano RP2040] → [SP335 Transceiver] → [RS-485 Bus Communication Standard] → [Smart Servos M1-M6]
- ✅ testing braccio claw working using remote control (braccio_plus_plus 01Controlling_Manually_Braccio), everything works except top claw
- ✅ Data Collection 450 pictures
- ✅ Model deployment installing edge impulse model: error in `curl -sL https://deb.nodesource.com/setup_18.x | sudo bash –`So fixed node.js repository
Running vision camera:

```
cd ~/depthai-python
source bin/activate
python3 examples/UVC/uvc_rgb.py
```

Another terminal:

```
edge-impulse-linux-runner
```

Running the depth test

```
cd ~/dai_ws
python3 oak_rgb_depth_test.py
```

Click link to see the bounding box visually
- ✅ Downloading testing model from edge impulse, problem with no ONNX model required by OAK-D, used C++ library instead and converted it into ONNX, all files are in /home/icp/edge_download
- ✅Installed virtualenv and created OpenVINO, installed blobconverter, pruned the model

Problem with next step generating IR files
https://grok.com/chat/3a8d4a64-da69-4e3b-8954-5710d373ad19

==
SESSION UPDATE: Animal Game software
==
One terminal:

```
cd ~/depthai-python
source bin/activate
python3 examples/UVC/uvc_rgb.py
```

Another terminal:
Raspberry to Arduino Reply

```
cd ~/animal_game
python3 animal_ard.py
```

cd ~/animal_game
Speech to text

```
speech_text.py
```

Raspberry to Arduino Reply

```
python3 ard.py
```

Game animal script:

```
python3 sounds2.py
```

Game animal script:

```
python3 sounds_animal.py
```

Detecting animal script:

```
python3 animal_game.py
```

https://www.jaycar.com.au/arduino-compatible-9g-micro-servo-motor/p/YM2758

Learn Tutorials — ROS 2 Documentation: Humble documentation Planning Scene ROS API — MoveIt Documentation: Humble documentation
Make claw working, deploy computer vision model
To do:
🔄 Phase 1: Arduino Firmware Development
- ✅ Develop joint state publisher for Braccio arm
- ✅ Implement servo control and safety limits
- ✅ Create command subscriber for arm movements

🔄 Phase 2: Computer Vision Integration
- ✅ Implement object detection using OAK-Dedge
- ✅ Develop spatial coordinate mapping

- ✅ Create pick target identification system
- ✅ Integrate depth data for 3D positioning

🔄 Phase 3: ROS2 Integration
- ✅ Create ROS2 nodes for vision processing
- ✅ Implement motion planning for pick/place
- ✅ Develop state machine for operation sequence
- ✅ Add safety monitoring and error handling

🔄 Phase 4: System Integration
- ✅ Connect all components through ROS2
- ✅ Implement complete pick and place workflow
- ✅ Add user interface and monitoring

//A go to position A (A-F)
//80,120,125,50,10 Move to positon
//A,C Go to position A to C (A-F)
//X Go to Top
//Y Close gripper, stay in position
//Z Open gripper, stay in position
// {'F':'A','L':'B','E':'C'} Move Animals

# Variation 1
{'F':'B','L':'C','E':'D'}

VNC
Change locations to L1,L2,F,R1,R2,B
Publish final pos

```
mosquitto_sub -h localhost -t "stacker/positions"
mosquitto_sub -h localhost -t "stacker/command"
mosquitto_sub -h localhost -t "stacker/status"
mosquitto_pub -h localhost -p 1883 -t "stacker/command" -m '{"E": "L2", "L": "R2", "F": "L1"}'
cd animal_game/MQTT4

mosquitto_pub -h localhost -t "stacker/positions" -m "{'A': 'frog', 'B': 'bear', 'C': None, 'D': None, 'E': 'elephant', 'F': None}"
mosquitto_pub -h localhost -t "stacker/status" -m "DONE"
```