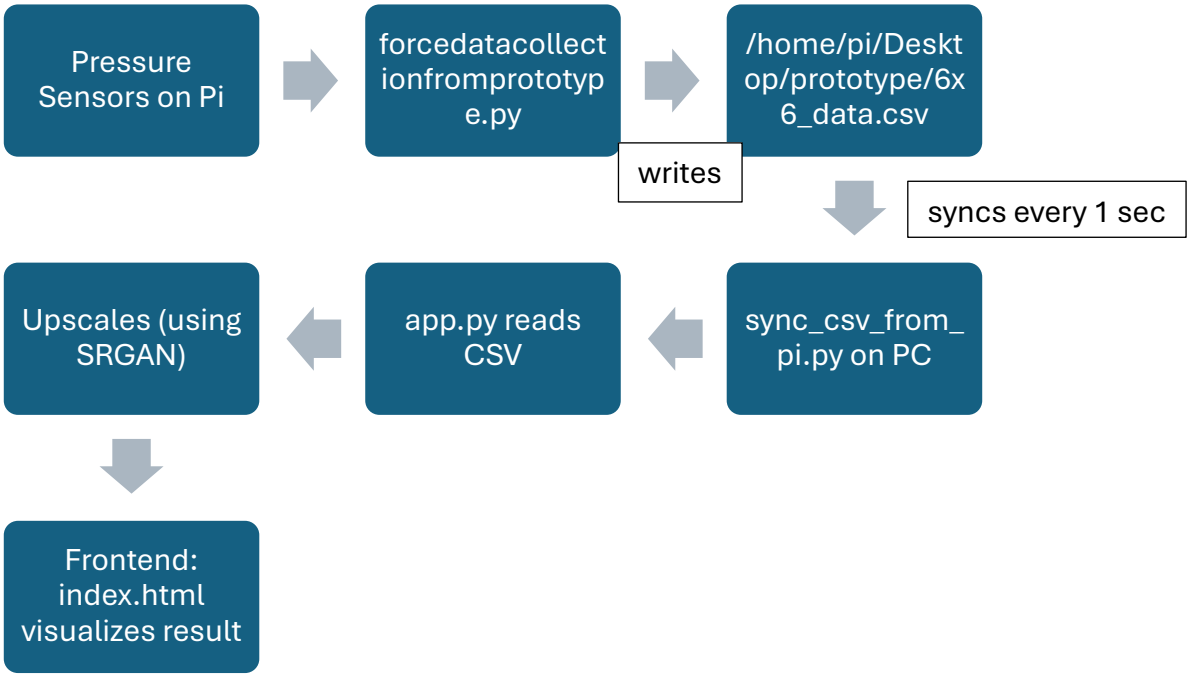
Real-Time Pressure Mapping UI with SRGAN

Introduction:

This project provides a **real-time end-to-end system** that reads analog pressure data from a sensor array (via Raspberry Pi), visualizes it as a heatmap, upscales it is using a **Super-Resolution GAN (SRGAN),** and displays both the raw and enhanced images on a web-based user interface with real-time pressure statistics.

System Overview:



Features:

- Real-time acquisition from 36 pressure sensors
- Heatmap of the raw 6x6 matrix and SRGAN-generated 32x32 map
- Upscaling with PyTorch-based SRGAN
- Displays maximum and total pressure values
- Frontend auto-refreshes data every second

Setup Instructions:

1. Install Python Dependencies

On your PC (Flask UI server):

```
bash
pip install flask flask-cors numpy matplotlib pillow torch torchvision
```

2. On your **Raspberry Pi** (data collection):

```
bash
pip install pandas numpy scipy adafruit-circuitpython-tca9548a adafruit-circuitpython-ads1x15
```

Enable I^2C on Raspberry Pi via sudo raspi-config.

Running the Full System:

Step 1: Collect Pressure Data (on Pi)

Run this on your Raspberry Pi: python forcedatacollectionfromprototype.py

This script:

- Reads values from 36 sensors (ADS1115 over TCA9548A)
- Converts them using calibration from Calibration Data.csv
- Saves a 6x6 force matrix to 6x6_data.csv every 0.1 sec

Step 2: Sync CSV to Your PC

Run this on your **PC**:python sync_csv_from_pi.py

This syncs the Pi's 6x6_data.csv to your local directory every second via SCP.

Update:

```
PI_USER = "pi"
PI_IP = "your_pi_ip_address"
LOCAL_CSV_PATH = "your/local/path"
```

Step 3: Run Flask Backend

python app.py

Access API at: http://127.0.0.1:5000/predict

This:

- Reads the synced CSV
- Generates a heatmap from the 6x6 data
- Upscales, it using SRGAN
- Returns max force, total force, and image URLs

Step 4: Open the Web UI

Open index.html in your browser. It:

- Polls the backend every second
- Updates both the input and SRGAN output images
- Shows the pressure statistics

How the SRGAN Works

In model_inference.py:

- Loads a PyTorch SRGAN generator
- Takes a resized 6×6 RGB heatmap image
- Produces a 32×32 upscaled RGB image
- Inference is done via:

generator.load_state_dict(torch.load("models/generator.pth"))

Upscaled output is returned as a PIL image, then saved and served via Flask.

API Sample Response

GET /predict

Returns:

{

"input_image": "http://127.0.0.1:5000/static/input_6x6.png",
"max_force": 28.83,
"output_image": "http://127.0.0.1:5000/static/output_srgan_32x32.png",
"total_force": 348.47
}

forcedatacollectionfromprototype.py (Pi)

- Reads from sensors via I2C (TCA + ADS1115)
- Converts ADC values to force using calibration
- Saves as 6x6 matrix in 6x6_data.csv

sync_csv_from_pi.py (PC)

- Uses scp to fetch 6x6_data.csv from Pi every 1 sec

app.py (PC)

- Flask server that loads CSV, generates heatmaps, runs SRGAN

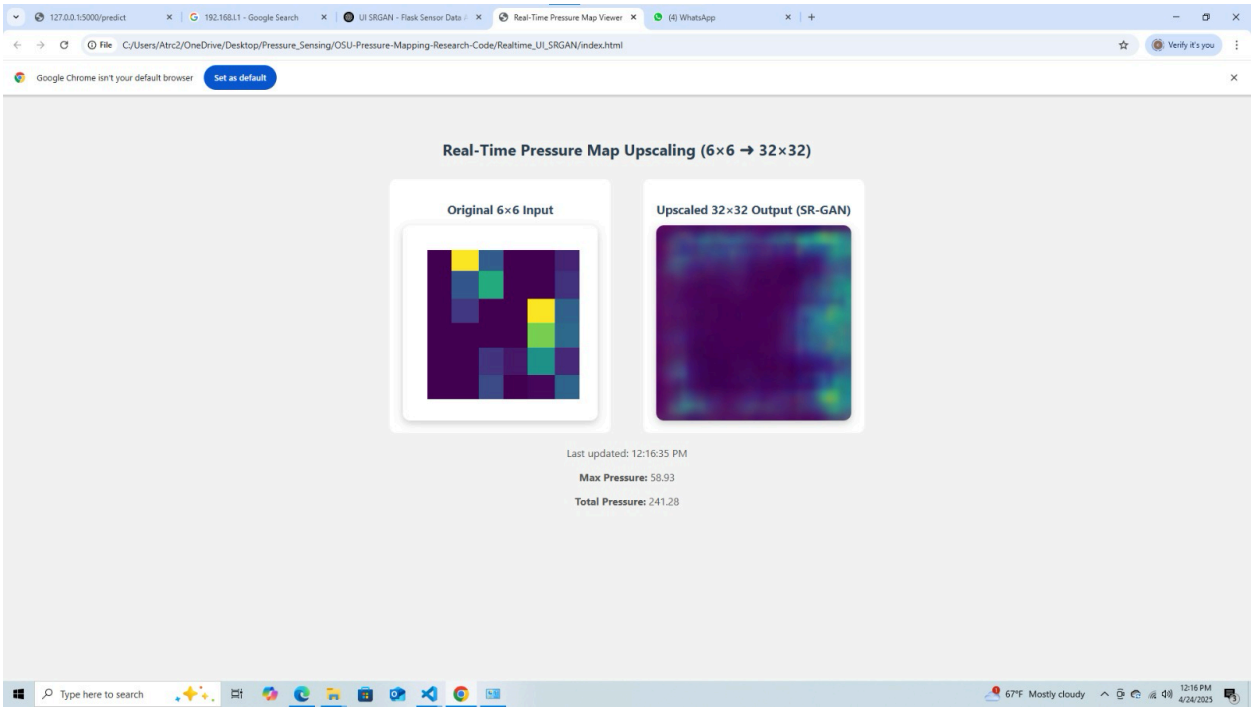
model_inference.py

- Defines and loads SRGAN generator
- Applies upscale to input image

utils.py

- Validates and reads CSV

Sample Output:



Super-Resolution GAN (SRGAN) for Pressure Map Upscaling

This project implements a **Super-Resolution Generative Adversarial Network (SRGAN)** for upscaling low-resolution pressure maps (e.g., 6×6) into high-resolution versions (e.g., 32×32). It is designed specifically for enhancing sensor-based pressure maps for clearer visualization and downstream analysis.

Project Objective

Given a low-resolution pressure matrix (6×6), generate a perceptually accurate high-resolution image (32×32) using deep learning — specifically, SRGAN with perceptual loss.

SRGAN file is 6x6_SRGAN.ipynb and the upscaled images saved in upscaled_250_32x32 folder and input images are images_6x6 folder

The raw reading are in soren.csv file