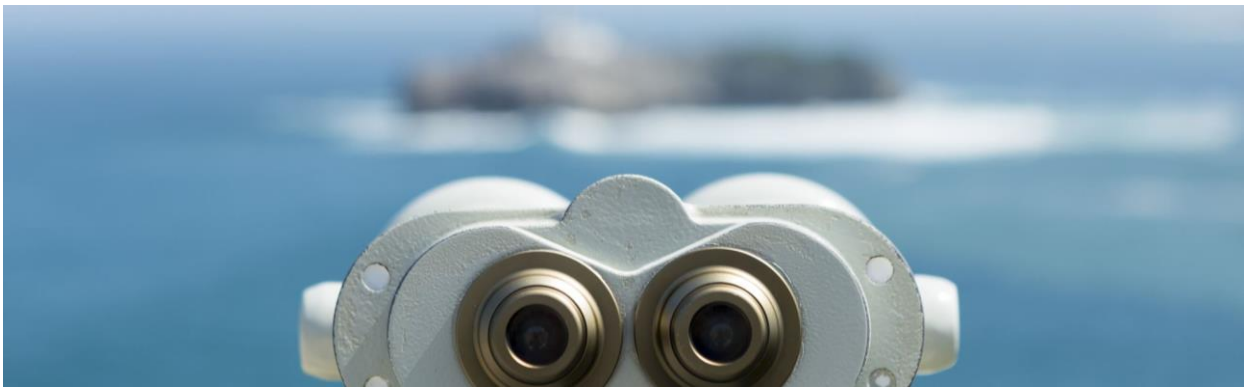

Person Tracking Camera

Demo Video Link

Github Repository

Akash Agrawal, Sudarshan Bandyopadhyay, Swapnil Mane

Aim:
This project revolves around the creation of a person tracking camera system utilizing the Nicla Vision and ESP32-CAM microcontroller alongside the MG995 servo motor. Its core aim is to precisely monitor a person's facial movements in real-time and adjust the camera's position accordingly. Face detection is executed through the MTMN algorithm, facilitating the identification and localization of faces within the camera's view. These identified face coordinates serve as inputs for controlling the servo motors, enabling seamless tracking of the person's movements by the camera. Through the integration of face detection and servo motor control, this project highlights an efficient solution for automated face tracking applicable across diverse domains like surveillance systems and human-computer interaction interfaces. Leveraging Nicla Vision, the project displays camera-level control, incorporating depth-based zoom and Haar Cascade Face Detection functionalities.

Implementation Process:
1. Hardware Setup
2. AI-based Face Detection Algorithm Implementation and Quantization.
3. Servo Motor Control and Integration of Face Detection with Servo Control using Proportional Controller on ESP32
4.Nicla Vision based camera level control (depth-based zoom)

1. Hardware Setup:
We utilized Nicla Vision to showcase camera-level control in our setup. The ESP32-CAM microcontroller acted as the central hub for managing servo control. Powering the ESP32-CAM module was achieved by supplying a 5V power source. Additionally, we connected the MG995 servo motor to the ESP32-CAM. Ensuring adequate power supply, we linked the servo motor's Vcc (5V) output pin to the corresponding power source on the ESP32-CAM. The signal pin of the servo motor was then connected to pin 14 of the ESP32-CAM, designated as the PWM output pin. This configuration enabled precise

control signals from the ESP32-CAM to be transmitted to the servo motor, ensuring accurate movement. Moreover, the integrated camera on the ESP32-CAM module was interfaced using the Serial Camera Control Bus (SCCB) interface.

Architectural Diagram:
Below Shown is the process flow diagram of our system and the hardware setup.

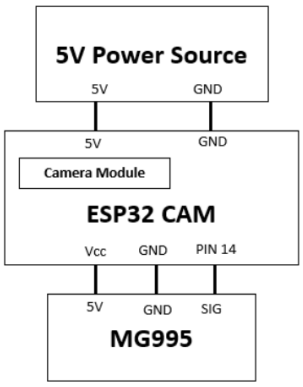

Fig 1: Pin Map of Face Tracking System

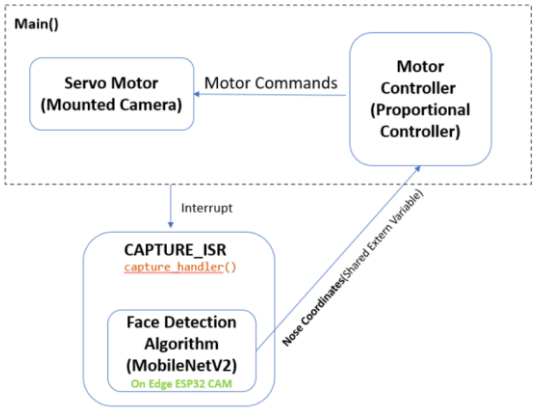

Illustration 1: System Flow

2. Face Detection: We have used MTMN face detection in ESP 32 CAM and Haar Cascade Face detection in Nicla Vision.

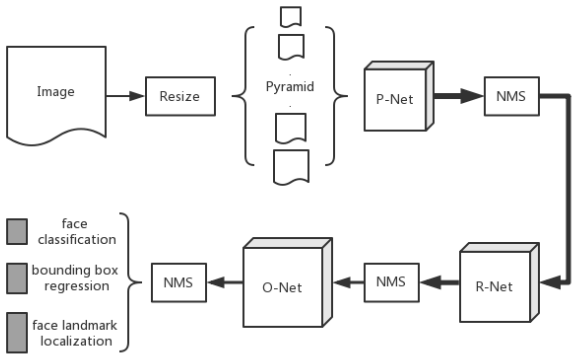

Fig: MTMN is a lightweight **Human Face Detection Model**, which is built around a new mobile architecture called MobileNetV2 and Multi-task Cascaded Convolutional Networks, and is specially designed for embedded devices.

Quantisation Configuration :

Float Configuration (CONFIG_MTMN_LITE_FLOAT):
This is the configuration for the floating-point version of the neural networks.
Functions like pnet_lite_f, rnet_lite_f, and onet_lite_f are called when this configuration is active.
Floating-point weights and activations are used during the forward pass.

Quantized Configuration (CONFIG_MTMN_LITE_QUANT):
Functions pnet_lite_q, rnet_lite_q, and onet_lite_q are called when quantization is active.
These functions involve quantized weights and activations. Quantized operations are more hardware-friendly.
The DL_XTENSA_IMPL argument - quantized operations are implemented for the Xtensa architecture, which is used in ESPRESSIF chips.

Average Time Consumption (ms)
MTMN lite in quantization: 56.8
MTMN lite in float: 73.84
Speedup Achieved: x1.30

3. Servo Motor Control:

In the camera system designed for face tracking, a servo motor played a crucial role in controlling the camera's movement. The servo motor received commands directing it to specific angles, enabling it to adjust its position in response to the movement of a human face within its field of view. Continuous monitoring of detected face coordinates allowed the servo motor to realign the camera accordingly.

To address the challenge posed by a camera in constant motion, a PID Controller was introduced. This controller, comprising proportional (P), integral (I), and derivative (D) components, enhanced the system's ability to control the servo motor dynamically and responsively. Integration of the P-Controller ensured precise face tracking, even when the camera was moving, thus overcoming the complexities associated with a moving camera scenario.

Implementation to provide servo angles:

1. Employing a proportional controller, the system tracks facial movements by continually comparing the detected face's position, specifically focusing on the x-coordinate of the nose, with the desired position.
2. To enable smooth and uninterrupted servo motion, the angle sent to the servo motor is computed by adding the change in angle (delta angle) to the previous angle state.
3. Error computation relies on the formula: Error = (X_res/2) – X_nose, where X_res denotes the frame width (e.g., 240 pixels for a frame size of 240×176).
4. The delta angle, crucial for servo movement, results from multiplying the proportional gain (Kp) by the error: Delta Angle = Kp * error. The proportional gain governs system responsiveness and sensitivity.
5. Ensuring absolute angle transmission to the servo motor involves adding the obtained delta angle to the previous angle state.

4. Nicla Vision based camera level control (depth-based zoom):
A predefined Haar Cascade model is employed for human face detection, which effectively creates a bounding box around the detected face. The output of this model includes the positions of the top-left and bottom-right vertices of the bounding box, providing precise localization of the detected face.

In the process of focus adjustment on the frame, the total pixel area occupied by the face frame is compared with the camera frame resolution. If the pixel area occupied by the face falls below a

defined threshold, a zoom-in operation is triggered to enhance focus. Conversely, if the occupied area exceeds the threshold, a zoom-out operation is initiated to adjust the view accordingly. This dynamic approach ensures optimal framing.

```
if (area_face) > referenceFaceArea:
    sensor.set_framesize(sensor.QVGA)
else:
    sensor.set_framesize(sensor.HQVGA)
```

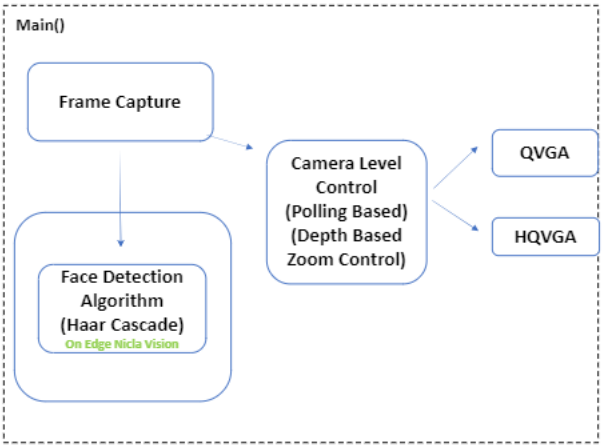

Fig : Process Flow Diagram for camera level control (depth zoom) on Nicla Vision

Conclusion:

We set out to demonstrate two critical technologies, face tracking using servo motor control and camera level control to achieve zoom in / zoom out tasks on embedded devices. We were able to perform both tasks, albeit on two different microcontrollers. As future work we plan to make the system more robust and integrate the two capabilities together in one ecosystem.