# APL103

# Temperature measurement using Infra-Red Cameras

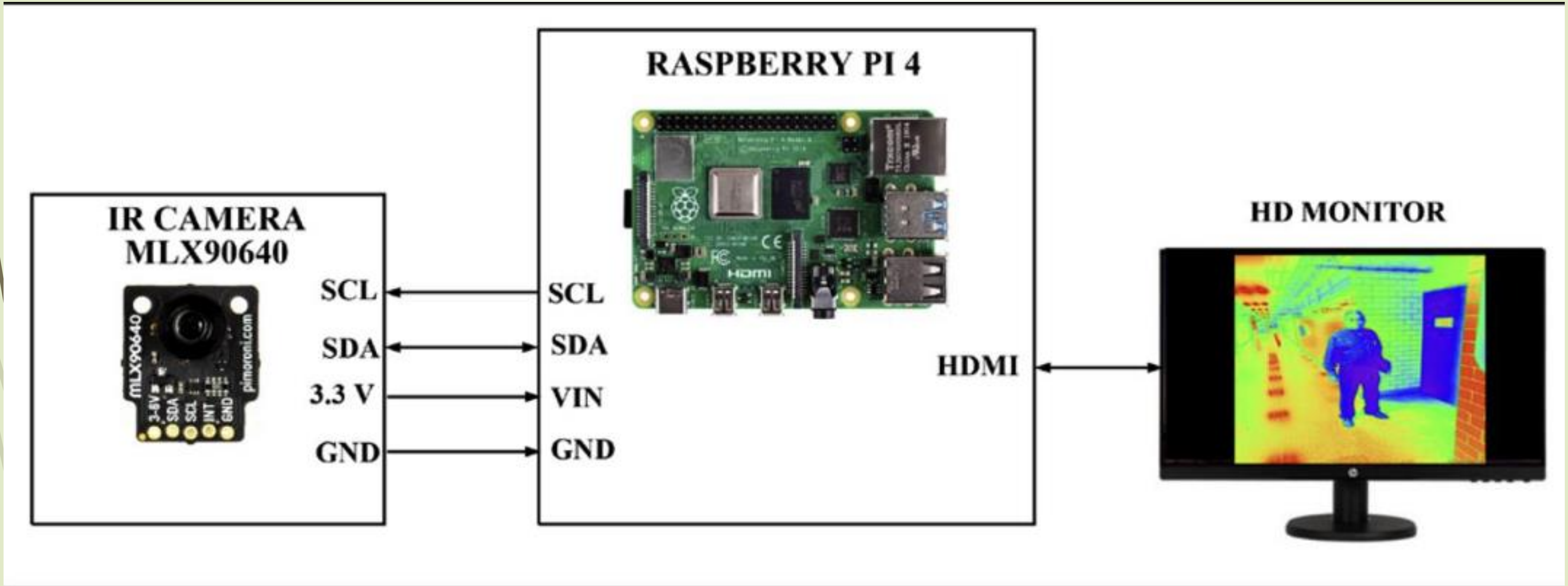
*Presented by :-*
*Ankit Yadav*
*Atharva Anand Soni*
*Hitesh Jain*
*Sparsh Agrawal*
*Shweta Tiwari*
*Tejasraj Mangla*

# Introduction

- IR cameras utilize infrared radiation emitted by objects to map temperatures, revolutionize temperature measurement with their non-contact, real-time capabilities.

- This presentation explores their principles, applications, and recent advancements, highlighting their efficacy and transformative potential in diverse fields.

# Components of an IR camera



# Components of an IR camera

- MLX90640 Sensor: Captures infrared radiation within its field of view, offering detailed thermal information. The IR camera allows to measure temperature values from -40 °C to 300 °C, with a resolution of ±1 °C and can be configured to use a refresh rate of frames from 0.5 Hz to 64 Hz. This means 64 frames, or samples per second, or 64 Hz. It is compact in size which makes it easy to use in almost any type of application that needs infrared vision.
- Integration with Raspberry Pi: This computer module was selected because it is possible to directly connect both the IR camera and the HD monitor to it. Another reason this module was used was that the Raspberry Pi 4 processor is powerful enough to execute the mathematical operations performed by the functions and routines used in programming, which can be done using other embedded modules, but with certain limitations.
- HD Displays: Enable users to visualize thermal images in greater detail, making it easier to interpret and analyze temperature data.

# Principle of Operation in IR Cameras

- Infrared Emission: Objects emit infrared radiation based on their thermal energy, with hotter objects emitting more intense radiation and cooler ones emitting less.
- Sensor Pixel Detection: The MLX90640 sensor's array of pixels acts as miniature infrared sensors, detecting incoming radiation from objects.
- Radiation Absorption: Thermal sensor material in each pixel absorbs incoming radiation, leading to changes like generating electrical signals or altering material temperature.
- Signal Conversion: Absorbed infrared radiation is converted into electrical signals within pixels, with signal strength corresponding to radiation intensity and object temperature.
- Temperature Calculation: The sensor processes pixel signals, amplifies and converts them to digital data, applying corrections for factors like ambient temperature and emissivity.
- Blackbody Emissivity: MLX90640 considers object emissivity, adjusting temperature calculations based on materials' ability to emit infrared radiation compared to a perfect emitter.
- Calibrated Temperature Output: MLX90640 calculates temperatures for each pixel's area on an object's surface, creating thermal images with different colors representing varying temperatures.

# Accuracy in IR Cameras

- **1.Sensor Calibration and Quality:-**

• Regular Calibration

• Sensor Quality Matters

- **2.Emissivity of the Target Object:-**

• Material Properties Dictate Emissivity

• Knowing Emissivity is Key

- **3.Ambient Temperature Variations:-**

• Impact on Sensor Performance

• Minimize Environmental Influence

# Accuracy in IR Cameras

**4.Field of View and Distance to Object:**

- Field of View (FOV)
- Maintaining Appropriate Distance

Most thermal cameras have a data sheet showing accuracy specifications, like 2% or ±2°C of the temperature reading. This accuracy specification is calculated using a measurement uncertainty analysis technique known as RSS or Root Sum of Squares. The accuracy of standard infrared cameras will have a maximum error margin of 2°C.

# Spot Size

Spot size defines the area of the target each camera pixel covers.

Say a thermal camera with a 25° lens measures the temperature of a lit match placed at 60 feet. Each pixel usually covers a square inch of the entire scene. However, the match is only 1/8th of square inch, much smaller than what a pixel covers. Most of the infrared energy comes from the subject's surroundings, while only 1/64th of the measurement comes from the match. Hence, the camera will show inaccurate temperature readings of the lit match.

Use a telescopic optic or move closer to the subject, so the ratio between the pixel size and the lit match is much closer to a 1:1 ratio.

# Advancing Accuracy in IR Cameras

**<u>Gaps and Potential Improvements in Accuracy:</u>**

- ➢ Advanced Emissivity Measurement Techniques
- ➢ Environmental Compensation Algorithms
- ➢ Sensor Technology Advancements

**<u>Future Directions for Enhanced Accuracy:</u>**

- ➢ Sensor Fusion
- ➢ Machine Learning for Error Correction

# Experimental analysis

- Developed a system for displaying thermal images captured by an IR camera on an HD monitor.
- Utilized the MLX90640 camera array with 24x32 IR sensors, providing temperature values for objects and people.
- Interpolated sensor array values to achieve higher resolution thermal images.
- Processed data using a Raspberry Pi 4 module, achieving 2.8 frames per second under specific configurations.
- Future improvements include exploring advanced interpolation methods for better image quality and incorporating a web server for remote thermal image visualization.

# Data Analysis

**Mean:**

x̄ = Σxi/n

**Standard Deviation:**

σ = √(∑(x−¯x) ( x − x ¯ ) 2 /n)

**Best fit Straight Line:**

y = ax+b

**R-square/ Coefficient of Determination**

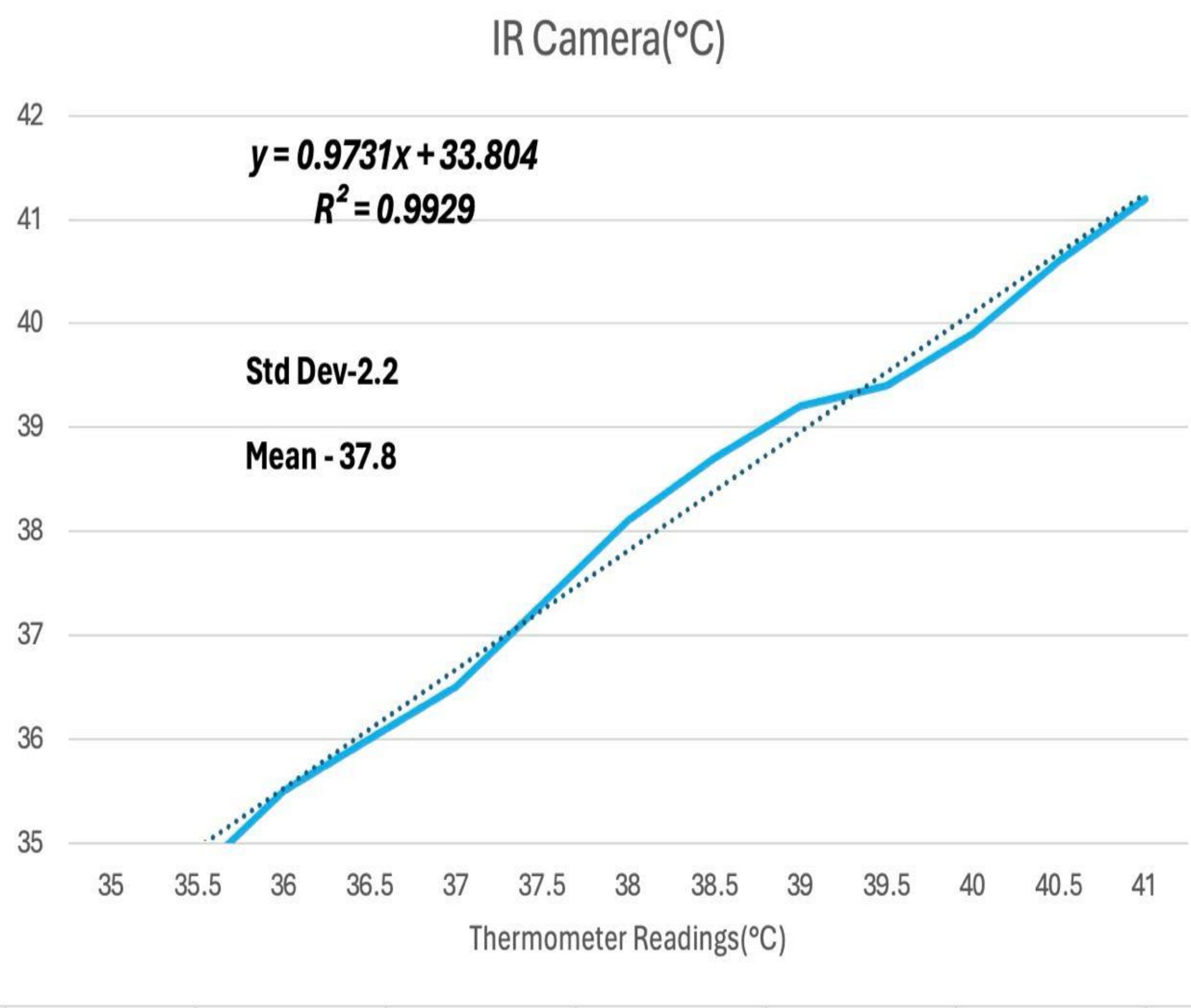
R-square or the Coefficient of Determination is defined by the percent of the variation in the dependent variable explained by the independent variable(s).

$$R^2 = 1 - \frac{\text{sum squared regression (SSR)}}{\text{total sum of squares (SST)}} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}.$$

Group 1 A

| Thermometer (°C) | IR Camera (°C) |
|---|---|
| 41 | 41.2 |
| 40.5 | 40.6 |
| 40 | 39.9 |
| 39.5 | 39.4 |
| 39 | 39.2 |
| 38.5 | 38.7 |
| 38 | 38.1 |
| 37.5 | 37.3 |
| 37 | 36.5 |
| 36.5 | 36 |
| 36 | 35.5 |
| 35.5 | 34.7 |
| 35 | 34.5 |

Mean → 37.8 °C
Std. Dev → 2.2°C



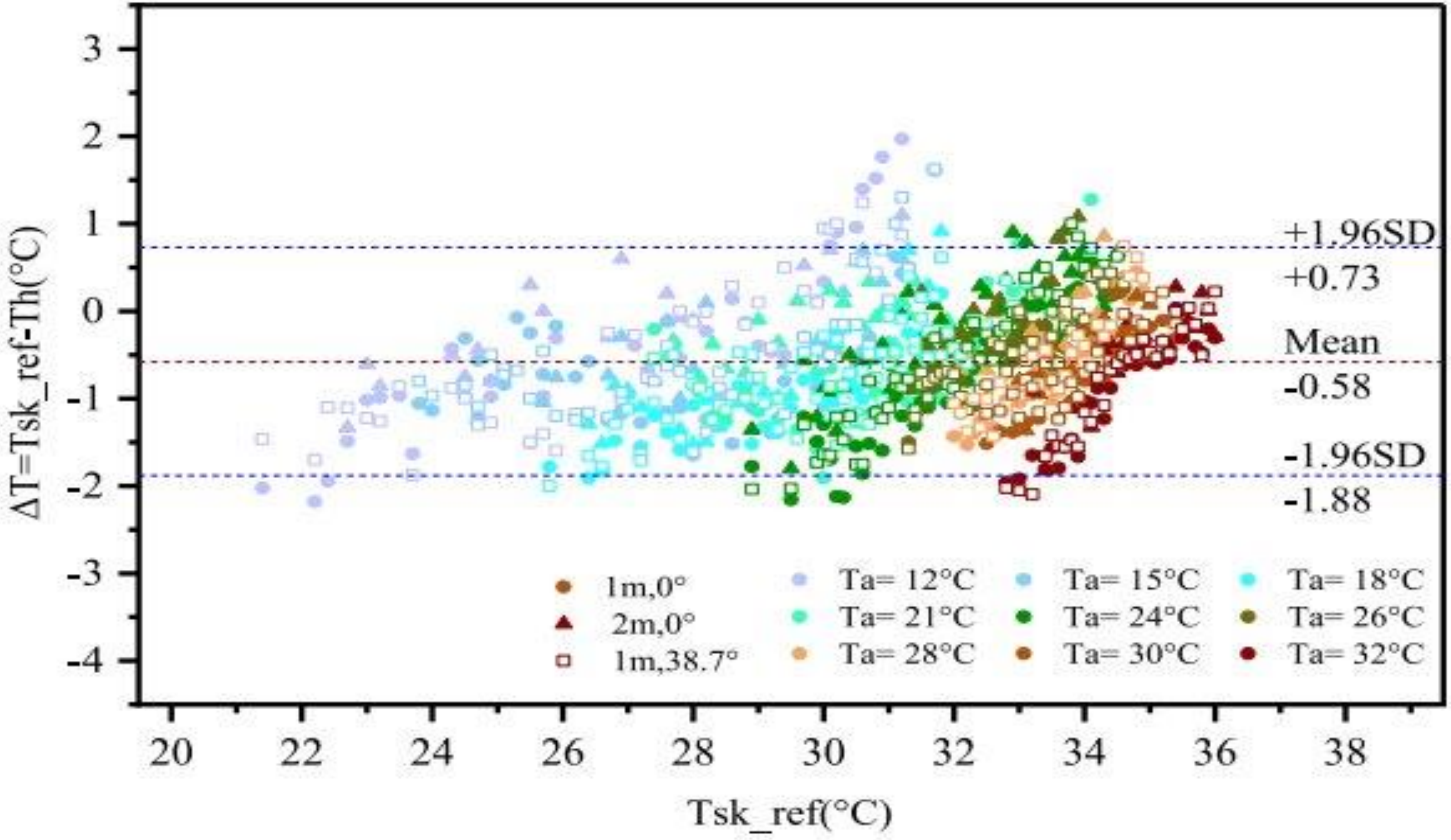
# Dataset-2(Based on Spot Size)

$\Delta T = Th - Tsk\_ref$ where, Tsk_ref is the skin temperature value obtained from calibrated corrected device(assumed to be true value for our analysis).

The measurement difference between the infrared camera and reference skin temperature (mean bias ±95% limits of agreement).

| Air temperature | $\Delta T_1$(1 m, 0°) | $\Delta T_2$(2 m, 0°) | $\Delta T_3$(1 m,38.7°) | Total | Diff1 = Th1–Th2 | Diff2 = Th1–Th3 |
|---|---|---|---|---|---|---|
| 12 | −0.46 ± 1.98 | −0.46 ± 1.68 | −0.14 ± 1.28 | −0.40 ± 1.76 | −0.06 ± 0.99 | −0.51 ± 1.26 |
| 15 | −0.63 ± 1.28 | −0.53 ± 1.48 | −0.36 ± 1.14 | −0.54 ± 1.35 | −0.10 ± 0.91 | −0.39 ± 1.14 |
| 18 | −0.98 ± 0.96 | −0.75 ± 1.21 | −0.7 ± 1.28 | −0.82 ± 1.16 | −0.22 ± 0.54 | −0.35 ± 0.83 |
| 21 | −0.70 ± 1.27 | −0.70 ± 1.10 | −0.17 ± 0.71 | −0.57 ± 1.17 | −0.03 ± 0.5 | −0.75 ± 0.99 |
| 24 | −0.78 ± 1.41 | −0.58 ± 1.58 | −0.21 ± 1.30 | −0.53 ± 1.5 | −0.22 ± 0.54 | −0.60 ± 0.71 |
| 26 | −0.58 ± 0.82 | −0.46 ± 1.00 | −0.09 ± 0.94 | −0.41 ± 0.99 | −0.12 ± 0.42 | −0.59 ± 0.78 |
| 28 | −0.70 ± 1.06 | −0.6 ± 1.13 | −0.33 ± 0.92 | −0.56 ± 1.08 | −0.10 ± 0.27 | −0.49 ± 0.70 |
| 30 | −0.78 ± 0.82 | −0.64 ± 0.84 | −0.44 ± 0.82 | −0.64 ± 0.86 | −0.14 ± 0.29 | −0.32 ± 0.51 |
| 32 | −0.90 ± 1.19 | −0.84 ± 1.33 | −0.48 ± 0.96 | −0.76 ± 1.22 | −0.06 ± 0.34 | −0.22 ± 0.54 |
| Total | −0.72 ± 1.30 | −0.61 ± 1.33 | −0.31 ± 1.12 | −0.58 ± 1.31 | −0.12 ± 0.62 | −0.49 ± 0.89 |

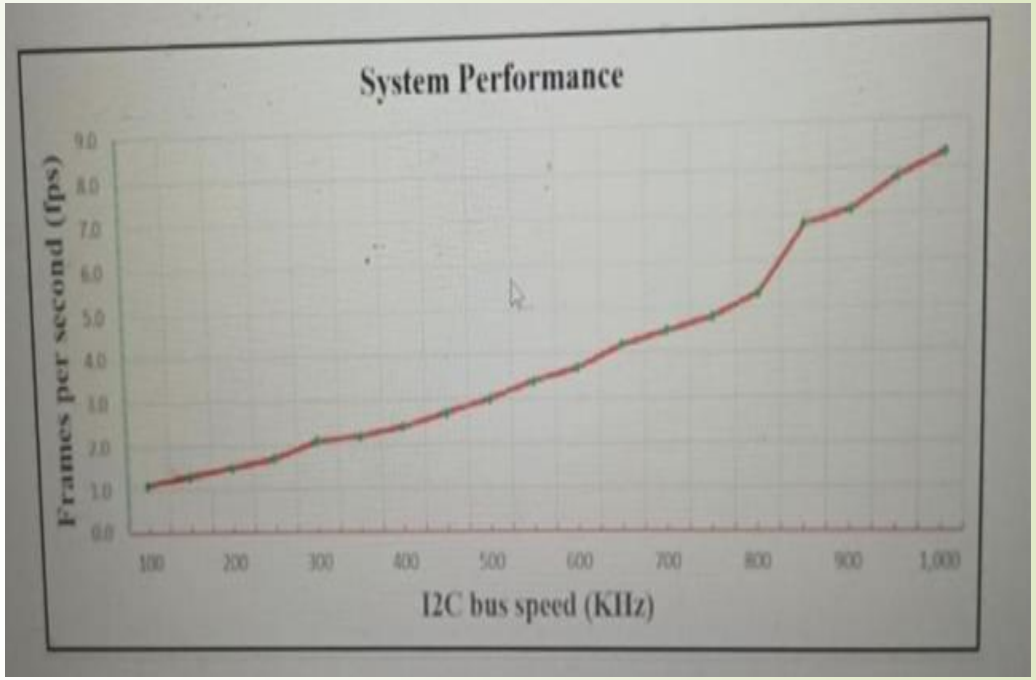
$\Delta T_{1,2,3}(d,\theta)$ is the difference between the infrared camera and the reference skin temperature. $(d,\theta)$ are the position between the infrared camera and the measured human face. d means the distance, $\theta$ represents the angle. Diff is the difference measured through the thermal infrared camera between different positions.



ΔT=Tsk_ref-Th(°C)
3
2
1
0
-1
-2
-3
-4
+1.96SD
+0.73
Mean
-0.58
-1.96SD
-1.88
1m,0°
2m,0°
1m,38.7°
Ta= 12°C
Ta= 15°C
Ta= 18°C
Ta= 21°C
Ta= 24°C
Ta= 26°C
Ta= 28°C
Ta= 30°C
Ta= 32°C
20
22
24
26
28
30
32
34
36
38
Tsk_ref(°C)

# Result and Conclusion

Three test groups were conducted to assess the system. The first aimed to evaluate both performance and accuracy by measuring frames per second (fps) on an HD monitor, resulting in 2.4 fps. Two hundred groups of five individuals were sampled, with pixel values from the camera read and displayed through blitting. The system's accuracy, averaging at 99.34%, was verified by comparing body temperature values to a reference thermometer. The I2C communication frequency stood at 440 KHz. The second group analyzed the impact of I2C speed on fps by configuring speeds from 100 kHz to 1 MHz, with 8 samples per second (sps) in the camera. As expected, higher I2C speeds correlated with increased fps, though it precipitated accelerated heating of the Raspberry Pi 4 processor. No adjustments were made to the camera's fps. Future optimizations might explore modifications in this aspect.

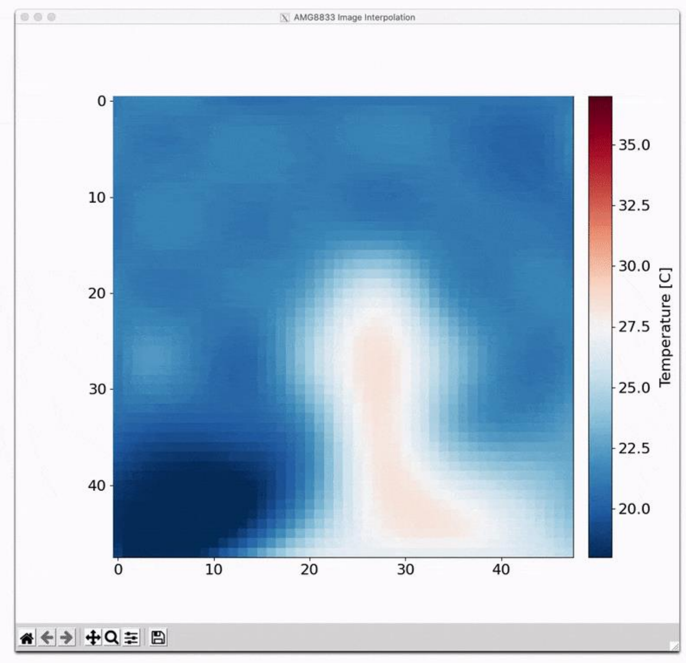


# Result and Conclusion

In the domain of technological inquiry, three distinct sets of experiments were conducted to scrutinize the efficiency and accuracy of the system. Initial trials focused on operational efficiency and precision, yielding a display rate of 2.4 frames per second (fps) and an impressive 99.34% accuracy in temperature readings from 200 groups of five subjects.

With an I2C communication frequency of 440 kHz, attention was turned to the relationship between fps and I2C speed, revealing a direct correlation. However, this came at the cost of accelerated thermal escalation in the Raspberry Pi 4 processor, with no alterations made to the camera's fps. Subsequent trials explored fps without temperature data interpolation, resulting in 15 fps at 440 kHz I2C speed, albeit with reduced resolution.

Future endeavours will involve leveraging OpenCV for improved imaging and potentially integrating a web server for remote thermal image visualization, promising transformative advancements in remote sensing and monitoring technologies.




```python
#######################################################
# Thermal camera Plotter with AMG8833 Infrared Array
#
# by Joshua Hrisko
#    Copyright 2021 | Maker Portal LLC
#
#######################################################
#
import time,sys
sys.path.append('../')
# load AMG8833 module
import amg8833_i2c
import numpy as np
import matplotlib.pyplot as plt
#
#####################################
# Initialization of Sensor
#####################################
#
t0 = time.time()
sensor = []
while (time.time()-t0)<1: # wait 1sec for sensor to start
    try:
        # AD0 = GND, addr = 0x68 | AD0 = 5V, addr = 0x69
        sensor = amg8833_i2c.AMG8833(addr=0x69) # start AMG8833
    except:
        sensor = amg8833_i2c.AMG8833(addr=0x68)
    finally:
        pass
time.sleep(0.1) # wait for sensor to settle

# If no device is found, exit the script
if sensor==[]:
    print("No AMG8833 Found - Check Your Wiring")
```