T.C.
DOKUZ EYLÜL UNIVERSITY
ENGINEERING FACULTY
ELECTRICAL & ELECTRONICS ENGINEERING
DEPARTMENT

Smart Mirror Controller

Final Project

by

Rabia DOĞAN

Advisor

Dr. Özgür TAMER

January,2021

IZMIR

THESIS EVALUATION FORM

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and qualify as an undergraduate thesis, based on the result of the oral examination taken place on __/__/_____

Dr. Özgür TAMER
(ADVISOR)

Prof. Dr. Gülay TOHUMOĞLU
(COMMITTEE MEMBER)

Dr. Abdül BALIKCI
(COMMITTEE MEMBER)

Prof. Dr. Mehmet KUNTALP
(CHAIRPERSON)

ABSTRACT

In the project, general purpose is to give the Smart Mirror controllable features with hand gestures. In this way, the product will be able to appeal to the upper customer segments and increase its market share. The main goal here is to enable users to use most of the smart mirror applications with hand movements, to eliminate the need to touch the mirror, thus to eliminate the contamination in the touched parts of the mirrors and therefore to possible dissatisfaction with the product and to ensure that the product can be used even in cases where the users cannot touch the mirror with their hands for various reasons, such as the bathroom. Although camera-based systems are generally used for gesture recognition, it will not be welcomed by many users to have a camera in a personal use area such as the bathroom.Therefore, passive infrared sensor arrays will be preferred for gesture recognition in our project. They can be preferred in application areas where cameras are relatively weak because they work with the infrared radiation emitted by living creatures. We aim to detect hand gesture and then the movement of the hand gesture. In this way, gesture distinction can be made with a simpler electronic design with less processing load and lower cost than a standard camera. The project is carried out and supported by Vestel Elektronik A.Ş. with the code TEYDEB 3170688.

ÖZET

Projede genel amaç Akıllı Aynaya el jestleri ile kontrol edilebilir özellikler kazandırmaktır.Bu sayede ürün üst müşteri segmentlerine de hitap edebilecek, pazar payını artırabilecektir.Burada temel hedef kullanıcıların akıllı ayna uygulamalarının birçoğunu el hareketleri ile kullanabilmesini sağlayarak, aynaya dokunma ihtiyacını ortadan kaldırmak böylece aynaların dokunulan bölgelerindeki kirlenmeyi ve dolayısıyla da ürün ile ilgili olası memnuniyetsizliği ortadan kaldırmak ve banyo gibi kullanıcıların çeşitli nedenler ile elleri ile aynaya dokunamayacakları durumlarda dahi ürünün kullanılabilmesini sağlamaktır.Jest tanıma için genellikle kamera temelli sistemler kullanılmakla beraber, banyo gibi kişisel kullanıma dönük bir alanda kamera bulunması birçok kullanıcı tarafından hoş karşılanmayacaktır.Bu nedenle projemizde jest tanımlama amacıyla pasif kızılötesi sensör dizileri tercih edilecektir.Bu tip sensörler oldukça düşük çözünürlüklerde görev yapmasına karşın, nesne ya da canlıların yaydığı kızılötesi ışıma ile çalıştıkları için kameraların görece zayıf kaldığı uygulama alanlarında tercih edilebilmektedirler.Pasif kızılötesi sensörler ile öncelikle kullanıcının jestini ve sonrasında da jestin hareketini algılamayı amaçlamaktayız.Bu sayede standart bir kameraya göre daha az işlem yükü ve daha düşük maliyet ve daha basit bir elektronik tasarım ile jest ayrımı yapılabilecektir. Proje Vestel Elektronik A.Ş.taarafından TEYDEB 3170688 kodu ile yürütülmekte ve desteklenmektedir.

Contents

List of Tables

List of Figures

1. INTRODUCTION

The overall goal of the project is to add a motion detection system to the Smart Mirror at the cheapest cost.

One of the most important issues in the project where confidentiality is kept at the forefront is sensor selection. One of the best options for this is the use of a thermopile array sensor. It is a system that measures infrared radiation developed with the thermocouple method dating back 150 years. A thermocouple consists of 4 leads, there are two different materials connected to 2 leads, the other two ends are connected to a voltage meter. If an absorber is attached to the coupling and is exposed to IR radiation from an object, the thermocouple junction becomes hot due to the incident radiation while the absorber collects the incident heat. The thermocouple material also converts the temperature difference voltage indicated by the voltmeter. Therefore, the voltmeter reading is a direct measure of object temperature. This method is simple, does not require any mechanics and can accurately detect static signals [7]. For this, the most suitable sensor selection should be determined by resolution, temperature range, communication method, etc. Considering the circumstances, Heimann HTPA32x32d thermopile infrared array was preferred.

Before Smart Mirror integration, it was decided to use an external microcontroller. The reason for this was that the processor in Smart Mirror did not work very well. It was decided to switch to external card selection in order to extend the process and to deliver the project within the specified time. The sensor PCB to be developed should be designed to connect the sensor to the microcontroller and receive data from the sensor in the most efficient way. The sensor and microcontroller to be selected will show us the way to be followed. At this stage, Vestel decided to manufacture its own processor. In order not to disrupt the project, Raspberry pi 4B was chosen for the continuity of the project.

For the development of the software department, the most suitable software language was chosen for the project. Python was preferred both in terms of libraries and in terms of having a lot of collaborative work. The first step is to get data from the sensor. After successfully obtaining this, the human body and all objects in the background emit infrared radiation in

the resulting image. Therefore, the sensor detects these objects as well. This problem can be solved by threshold methods in the literature. The simple threshold method returns the same threshold value for each pixel.Pixel values less than the threshold value are changed to 0. [14]. Adaptive threshold is the method by which the threshold value is calculated for smaller regions, so different regions will have different thresholds [14]. Another method is the Otsu method. It is also a kind of adaptive blend. Instead of choosing a threshold value, Otsu's method determines it [8] automatically. Apart from all these, a solution method was sought. Right after the smart mirror is turned on, the average temperature of the objects in the environment is subtracted from the main image, allowing the hand to be detected. Thus, an external preprocessing process is also avoided.

The 4 static and 4 dynamic movements are the ones we currently set for a smart mirror. There are some methods to help us understand whether movements are dynamic or static. With the Barycenter method, it is possible to adopt a similar case of planetary motion for the [23] gestures. Another method is MHI (memory History Image) [21]. Images are taken at certain time intervals and compared. The difference between the images shows us that there is movement. But since we have a framerate problem at this stage, this problem could not be solved.

The next step is to perform classification methods to match the resulting image with the commands mentioned earlier. Some methods are as follows;

The Hybrid Method (combination of Haar-Like and HOG features) is a feature extraction method. Then using this method, multiclass SVM(Support Vector Machine) is used for final gesture recognition [9]. HOG (Histogram of Oriented Gradient) features are one of the feature extraction methods. Hand recognition after applying this method is that the matching image is found in the database. This step is performed using nearest neighbor algorithms. [10]. k-NN(k-nearest neighbor); It is one of the most popular machine learning algorithms because it is old, simple, and resistant to noisy training data. However, it also has a downside. It is necessary to get the necessary information up-to-date for the big data we store in the long term. There are database clusters. Then, using the fall detection algorithm, the object in front is distinguished from other objects. Finally, k-NN classification is made using the data in the [12] database. CNN (convolutional neural networks) is one of the most widely used algorithms for image classification. An image classifier takes a photo or video as input and classifies it into one of the possible categories it was trained to identify. [22] [15] [23]. Using the CNN model for this allows

us to reach the most accurate result. Since we have 32x32 type input images, it was decided to use the LENET-5 model. As a result of the method used, very efficient results were obtained.

All that remains is the communication between the selected microcontroller and Smart Mirror. In order to do this, android software must be made on the smart mirror side. This is expected to be written by the android team in the team.

2. TECHNICAL BACKGROUND

2.1 Introduction to Convolutional Neural Networks

In deep learning, CNNs are the most common networks used with image classification. CNNs were inspired by the human visual system proposed by Fukushima [5] and LeCun et al. [6] State-of-the-art approaches to pattern recognition, object detection, and many other image applications. It was a deep CNN solution by Krizhevsky et al. [16] CNNs are very different from other pattern recognition algorithms because CNNs combine both feature extraction and classification [6]. The simple network model consists of five different layers: an input layer, a convolution layer, a pooling layer, a fully connected layer, and an output layer. These layers are divided into two parts: feature extraction and classification. Feature extraction consists of an input layer, a convolution layer, and a pool layer, while classification consists of a fully connected layer and an output layer. The input layer specifies a fixed size for input images, which are resized as needed. The image is then convoluted with multiple learned kernels using the weights shared by the convolution layer. Then, the repository layer reduces the image size while trying to preserve the information it contains. The outputs of feature extraction are known as feature maps. Classification combines extracted features into fully connected layers. Finally, there is one output neuron for each object category in the output layer. The output of the classification section is the classification result.

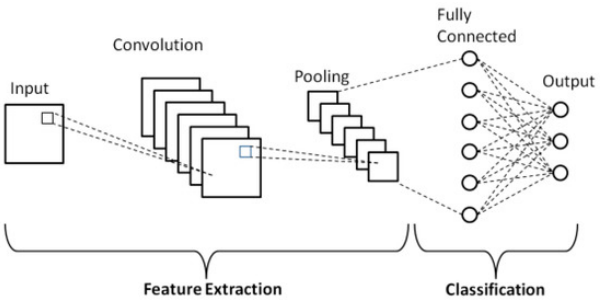


Figure 2.1: Schematic diagram of a basic convolutional neural network (CNN) architecture [20]

Deep learning techniques have emerged recently and advances in convolutional neural networks (CNN) surpass the classical approach to hand gesture recognition as it eliminates the need to derive complex handcrafted features from images [17]. CNN's automate the feature extraction process by learning high-level abstractions in images and capturing the most distinguishing feature values using the hierarchical architecture. Thus, it solves the disadvantage of obtaining inconsistent property descriptors when working with large numbers of motion classes with very small cross-class variations [24].

2.2 Model Architecture for LE-NET5

LeNet-5 was one of the earliest convolutional neural networks to support the deep learning event. After countless years of analysis and numerous compelling iterations, the final result was named LeNet-5 in 1988.

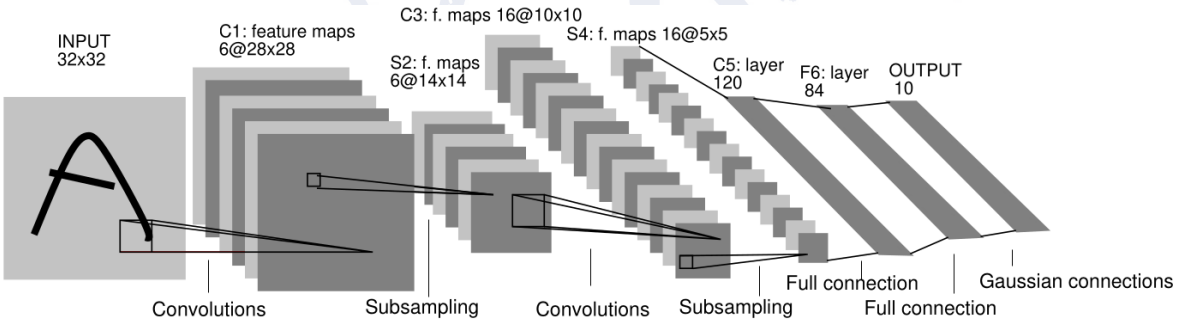


Figure 2.2: Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical. [6]

LeNet-5 A total of seven layers, each with no input with trainable parameters; each layer has multiple FeatureMap, which is a property of each of the FeatureMap inputs extracted via a convolution filter, and then each FeatureMap has multiple neurons.

C1 layer-convolutional layer:

The first convolution operation is performed on the input image (using 6 convolution kernels of size 5*5) to obtain 6 feature maps of C1 (6 feature maps of size 28 28, 32-5 + 1 = 28). The size of the convolution kernel is 5, and there are 6 (5 * 5 + 1) = 156 parameters in total, where +1 indicates that a kernel has a bias. For convolution layer C1, each pixel in C1 is dependent on 5 of 5 pixels and 1 aberration in the input image, so there are 156 28 * 28 = 122304 links in total. [6]

S2 layer-pooling layer (downsampling layer):

The pooling is done immediately after the first convolution. Pooling is done using 2 cores and S2, 14x14 (28/2 = 14) 6 feature maps are obtained. S2's pooling layer is a weighting coefficient plus an offset multiplied by the sum of the pixels in the 2*2 area in C1, and then the result is remapped. So each pooling core has two training parameters i.e. 2x6 = 12 training parameters but there are 5x14x14x6 = 5880 connections. [6]

C3 layer-convolutional layer:

After the first pooling, the second convolution, the output of the second convolution is C3, 16 pieces of 10x10 feature maps, and the size of the convolution kernel is 5. The first 6 feature maps of C3 (corresponding to column 6 of the first). red box) connects to 3 feature maps connected to S2 layer and next 6 feature maps are connected to S2 layer 4 feature maps are connected, next 3 feature maps are connected to 4 feature maps are unconnected in S2 layer and last one is linked to all feature maps in S2 layer. The convolution kernel size is still 5x5, so there are 6 (3*5*5 + 1) + 6 (4*5*5 + 1) + 3 (4*5*5 + 1) +1 (6*5*5 + 1) = 1516 parameters. The image size is 10 10 so there are 151600 connections. [6]

S4 layer-pooling layer (downsampling layer):

S4 is the pooling layer, the window size is still 2*2, a total of 16 feature maps and 16 10x10 maps of the C3 layer are pooled in units of 2x2 to get 16 5x5 feature maps. This layer has a total of 32 training parameters, 2x16, 5x5x5x16 = 2000 connections. [6]

C5 layer-convolution layer:

The C5 layer is a convolution layer. Since the size of the 16 images of the S4 layer is 5x5, the size of the image formed after convolution is 1x1, which is the same as the size of the convolution kernel. This results in 120 convolution results. Each is linked to 16 maps from the previous level. So there are (5x5x16 + 1) x120 = 48120 parameters and there are also 48120 connections. [6]

F6 layer-fully connected layer:

Layer 6 is a fully connected layer. The F6 layer has 84 nodes corresponding to a 7x12 bitmap, -1 means white, 1 means black, so the black and white of each symbol's bitmap corresponds to a code. The training parameters and number of connections for this layer is (120 + 1) x84 = 10164. [6]

2.3 Python Libraries for Project

2.3.1 Python-Peripherial

Python-periphery is a pure Python library for GPIO, LED, PWM, SPI, I2C, MMIO, and Serial peripheral I/O interface access in userspace Linux. Useful in embedded Linux environments (including Raspberry Pi) for interfacing with external peripherals. [4]

2.3.2 Tensorflow-Keras

TensorFlow 2 is an end-to-end, open-source machine learning platform. [11] It enables it to efficiently execute low-level tensor operations on the CPU, GPU, or TPU. Calculates the gradient of arbitrarily differentiable expressions. [11] Scales computation to many devices, such as clusters of hundreds of GPUs. [11] Streams programs ("graphics") to external runtimes such as servers, browsers, mobile, and embedded devices. [11] It combines these four

key features in one platform. [13] **3. MATERIALS AND METHODS**

3.1 Htpa32x32d Thermopile Infrared Array

The sensor chosen to be used in the project is HTPA32x32d thermopile array sensor. HTPA32x32d thermopile array sensor 32x32 pixel, operates between -10 and 70 degrees, provides I2C communication, has an internal EEPROM and provides an 8-bit data set. EEPROM data contains calibration data for each pixel of the sensor.

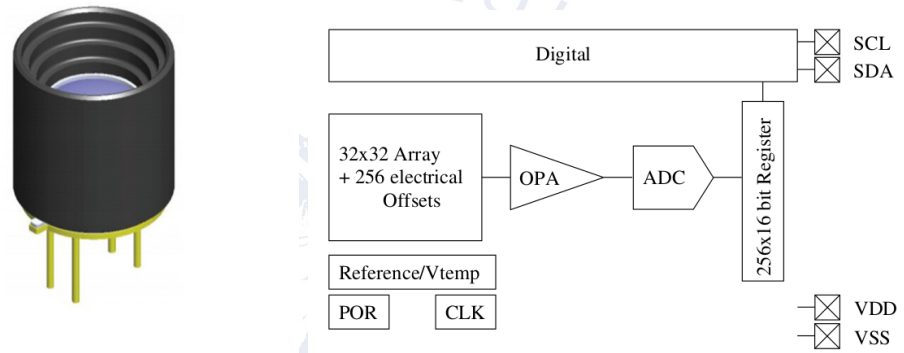


Figure 3.1: Schematic for HTPA32x32d

Table 3.1: Genaral Features HTPA32x32d

Features\{}	Sensitivity	Therm. Pix. Time Const.	Digital Interface	EEPROM Size
	450V/W	$< 4ms$	I2C	64kBit
Features\{}	Max Frame	Field of View	Selectable Clock	Storage Temperature
	60 Hz	33*33 deg	1 to 13 Mhz	-40/85 Deg.C

3.2 Sensor and MCU Communication PCB Design

A mini card has been designed between the sensor and the Raspberry pi card to communicate. However, the card could not be printed. The card that will connect the sensor and the motherboard has been designed completely according to the standards recommended by Heimann company.

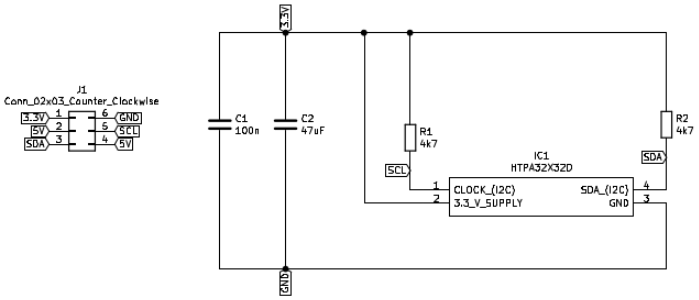


Figure 3.2: Schematic Design of Communication PCB

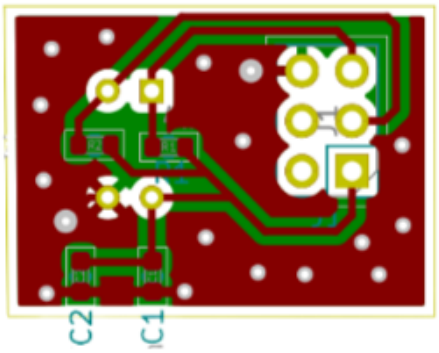


Figure 3.3: PCB Design of Communication PCB

In addition to this, we have produced a product suitable for a schematic on a mini perforated plate for use in communication with the card. In addition, we designed a protective cage from a 3d printer to protect the sensor.

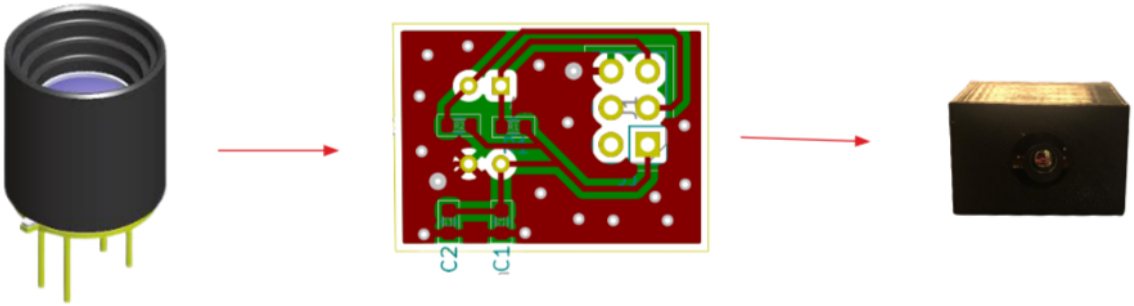


Figure 3.4: Mini Sensor Board

3.3 Dataset

3.3.1 Multi-Modal Hand Gesture Dataset for Hand Gesture Recognition

This dataset was created to validate a hand-gesture recognition system for Human-Machine Interaction (HMI). It is composed of 15 different hand-gestures (4 dynamic and 11 static) that

are split into 16 different hand-poses, acquired by the Leap Motion device. Hand-gestures were performed by 25 different subjects (8 women and 17 men). Every gesture has 20 instances (repetitions) per subject, performed in different locations in the image. [2]
for static and dynamic gestures:
This set contains 16 hand-poses, used for both static and dynamic hand-gestures:
A: L B: fist moved C: index D: ok E: C F: heavy G: hang H: two I: three J: four K: five L: palm M: down N: palm moved O: palm up P: up

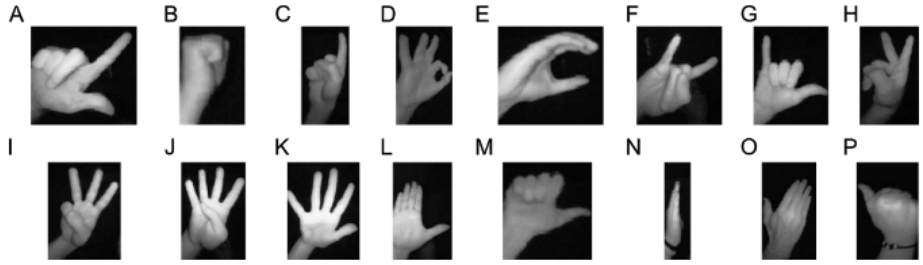


Figure 3.5: Hand Gesture Dataset

3.4 Data Augmentation

Using this dataset, a new train and validation set was created for the static gestures in the project. A total of 36800 and 9200 train and validation sets were created with randomly selected images.However, both resizing and data augmentation were done in order to make the data set suitable for the project. [18]

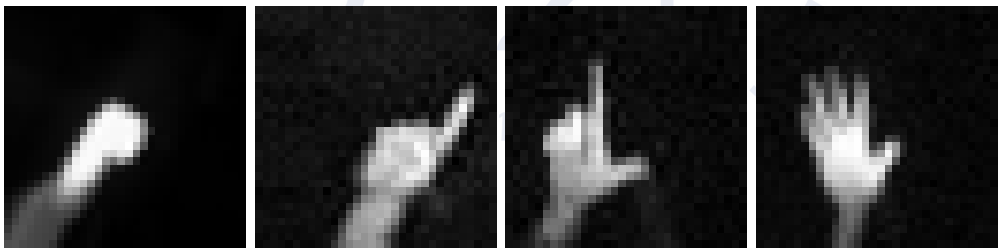


Figure 3.6: Close-Index-Last Page-Open Gestures

4. PROGRESS AND RESULT

In this section, the methods to be applied in the project are included.

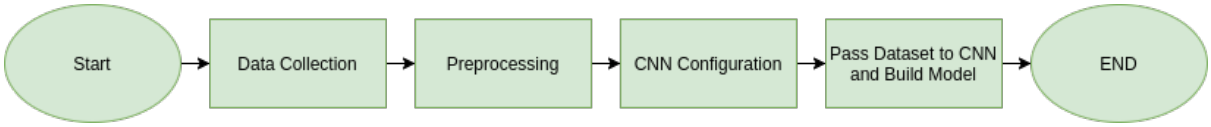


Figure 4.1: System Framework for Static Gestures

4.1 Capture Thermopiles Image

4.1.1 Heimann Thermopile Array Sensor communication

First, I provided the connections between the sensor and the Raspberry Pi in order to receive the image. I provided the communication with the mini card I made in section 4.2 using the I2C protocol.

With the device connected to a Raspberry Pi, and with the Pi configured. [1] correctly for I2C, I was able to see the devices connected with the i2cdetect command.

```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- 1a -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Figure 4.2: Thermopile Infrared Array Device and EEPROM Addresses

In order to be able to read the data properly, the python-periphery [4] library was used. The sensor is divided into two parts (Top and Bottom Half), which are also divided into 4 blocks. The reading order is shown below for different blocks. When a conversion is initiated, the X Block of the upper and lower half are measured simultaneously. Each block consists of 128 Pixels sampled entirely in parallel. The reading order in the lower half is mirrored compared to the upper half so the center lines are always read last.

Table 4.1: Read Data 1 Command (Top Half of Array)

Addr/CMD	0x1A (7 Bit!) / 0x0A							
Read Data	7	6	5	4	3	2	1	0
1. Byte / 2. Byte	PTAT 1 MSB / LSB or Vdd 1 MSB / LSB							
3. Byte / 4. Byte	Pixel (0+BLOCK*128) MSB / LSB							
5. Byte / 6. Byte	Pixel (1+BLOCK*128) MSB / LSB							
...								
257. Byte / 258. Byte	Pixel (127+BLOCK*128) MSB / LSB							

Table 4.2: Read Data 2 Command (Bottom Half of Array)

Addr/CMD	0x1A (7 Bit!) / 0x0B							
Read Data	7	6	5	4	3	2	1	0
1. Byte / 2. Byte	PTAT 2 MSB / LSB or Vdd 2 MSB / LSB							
3. Byte / 4. Byte	Pixel (992-BLOCK*128) MSB / LSB							
5. Byte / 6. Byte	Pixel (993-BLOCK*128) MSB / LSB							
...								
65. Byte / 66. Byte	Pixel (1023-BLOCK*128) MSB / LSB							
65. Byte / 66. Byte	Pixel (1023-BLOCK*128) MSB / LSB							
67. Byte / 68. Byte	Pixel (960-BLOCK*128) MSB / LSB							
69. Byte / 70. Byte	Pixel (961-BLOCK*128) MSB / LSB							
...								
129. Byte / 130. Byte	Pixel (991-BLOCK*128) MSB / LSB							
131. Byte / 132. Byte	Pixel (928-BLOCK*128) MSB / LSB							
...								
257. Byte / 258. Bytes	Pixel (927-BLOCK*128) MSB / LSB							

Each block is checked before it is read. The python-opencv [3] library was used to visualize the obtained result.

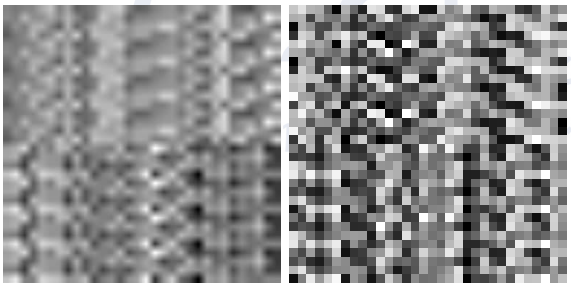


Figure 4.3: Thermal Image

Each pixel (or each analog-to-digital converter, given the repeating structure corresponding to each "block" of the sensor) has its own offset and sensitivity to incident light. Without calibrating it, this constant "noise" suppresses the signal from changing IR/temperature conditions. By subtracting the two frames in quick succession, this common noise signal is removed.

However, it is still quite noisy, as this frame subtraction increases random noise (since we now have contributions from two frames) and does not correct pixel-dependent sensitivity. Only fabrication calibration will be done with EEPROM data in the next step.

4.1.2 Calibrating images from Heimann Thermopile Array Sensor

After reading an image off a Heimann thermopile array, the pixel values can be converted to temperature readings through the use of calibration parameters stored on the device. To extract the calibration parameters, it is easiest to first read off the entire EEPROM on the thermopile array.

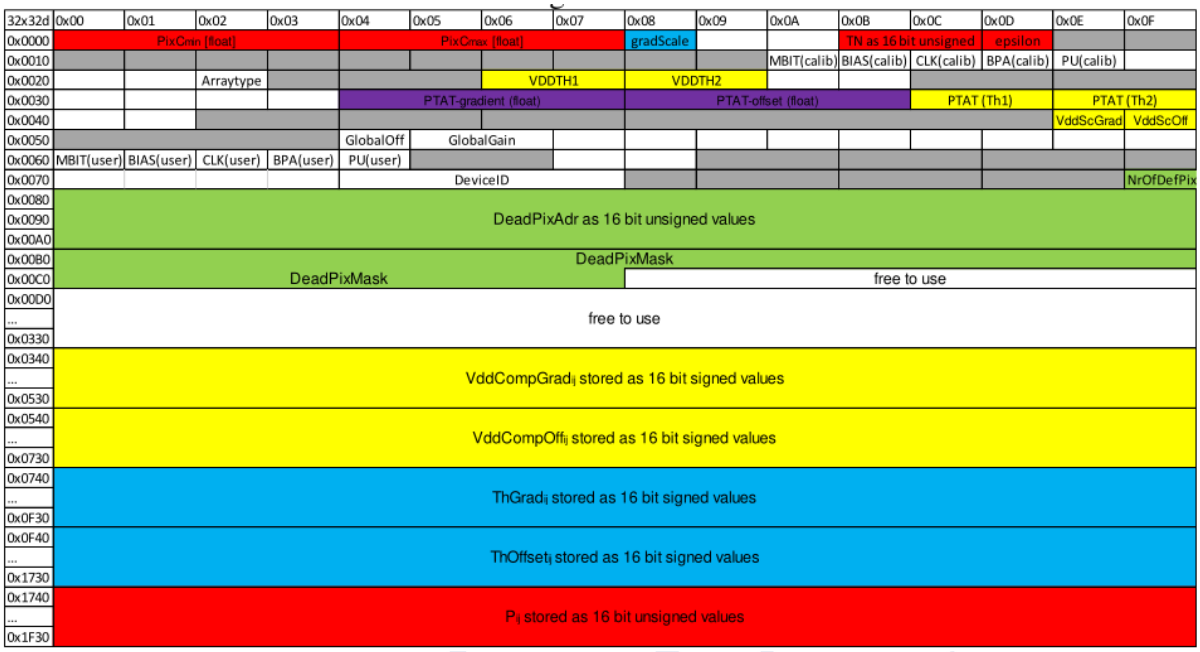


Figure 4.4: EEPROM overview 32x32d [19]

Then, parameters and calibration values can be extracted from this array, as described in the Heimann datasheet. [19]

Calibration for only one pixel is done as follows.

$PTAT_{av} = \frac{\sum_{i=0}^{7} PTAT_i}{8} = 38152 Digits$

$PTAT_{gradient} = 0.0211 dK/Digit$ and $PTAT_{offset} = 2195.0 dK$

$V_{00} = 34435 Digits$

$elOffset[0] = 34240$

$gradScale = 24$

$ThGrad_{00} = 11137$

$ThOffset_{00} = 65506$

$VDD_{av} = 35000$

$VDD_{TH1} = 33942$

$VDD_{TH2} = 36942$

$PTAT_{TH1} = 30000$

$PTAT_{TH_2} = 42000$

$VddCompGrad[0] = 10356$

$VddCompOff[0] = 51390$

$VddScGrad = 16$

$VddScOff = 23$

$PixC_{00} = 1 \cdot 087 \cdot 10^8$

$PCSCALEVAL = 1 \cdot 10^8$

Calculation of ambient temperature:

$T_a = PTAT_{av} \cdot PTAT_{gradient} + PTAT_{offset} = 38152 \cdot 0.0211 + 2195.0dK = 3000dK$

Compensation of thermal offset:

$V_{00\_Comp} = V_{00} - \frac{Th_{Grad00} \cdot T_a}{2^{gradScale}} - Th_{Offset_{00}} = 34439$

Compensation of electrical offset:

$V^*_{00\_Comp} = V_{00\_Comp} - elOffset[0] = 199$

Compensation of supply voltage:

$V_{00-VDD_{Comp}} = V^*_{00\_Comp} - \frac{\frac{Vdd_{CompGrad}[0] \cdot PTAT_{av}}{2^{Vdd_{ScGrad}}} + V_{Vdd_{Compoff}}[0]}{2^{Vdd_{ScOff}}}$

$\cdot (VDD_{av} - VDD_{TH1} - (\frac{VDD_{TH2} - VDD_{TH1}}{PTAT_{TH2} - PTAT_{TH1}}) \cdot (PTAT_{av} - PTAT_{TH1})) = 199 - 1 = 198$ The sensitivity coefficients (PixC ij) are calculated:

$PixC_{00} = (\frac{P_{00} \cdot (PixC_{Max} - PixC_{Min})}{65535} + PixC_{Min}) \cdot \frac{epsilon}{100} \cdot \frac{GlobalGain}{100000} = 1 \cdot 087 \cdot 10^8$

Leading to a compensation of the pixel voltage:

$V_{00PixC} = \frac{V_{00-VDD_{Comp}} \cdot PCSCALEVAL}{PixC_0} = 182$

All operations are applied for 1024 pixels. Application result images are as in figure 4.4.

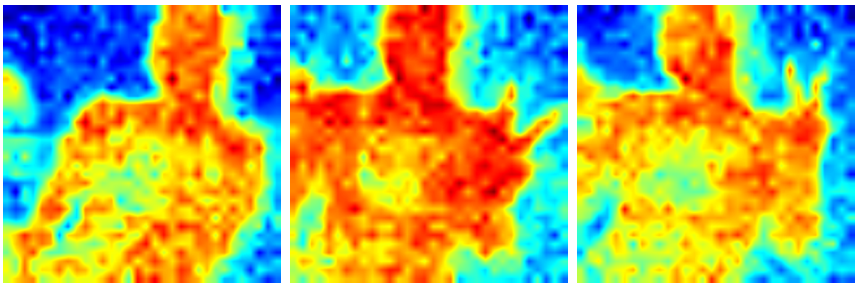


Figure 4.5: Thermal Images with EEPROM Calibration Data

NOTE:All steps to acquired the image are made with reference to the datasheet [19].

4.2 Hand Thermal Image Isolation

The hand was isolated from the background without using any image processing method. For this, it has been arranged in a way that can remove the ambient temperature of the device from the image before giving a command. First, the average of 10 images was taken and given to all images. Thus, the background temperature was isolated.

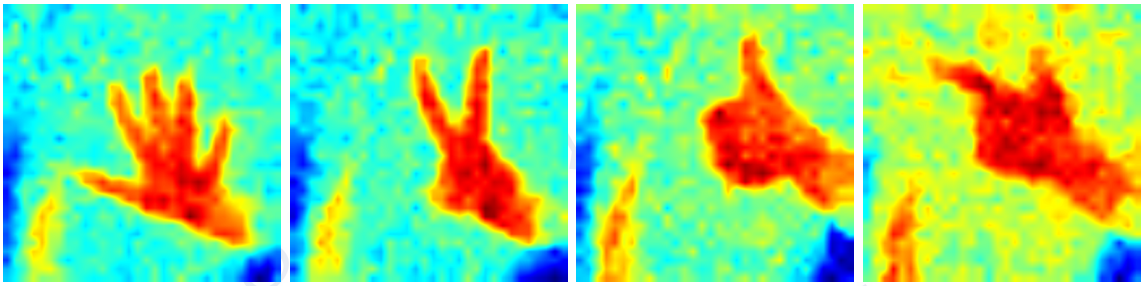


Figure 4.6: Thermal Images with Background

4.3 Hand Gesture Recognition

4.3.1 Static Gesture Recognition

A number of scenarios have been prepared for command matching for the smart mirror. The scenarios prepared are as follows;

For Static Movements:

There are 4 fixed movements. It is the ability to Open, Close, Tap and Return to Home.

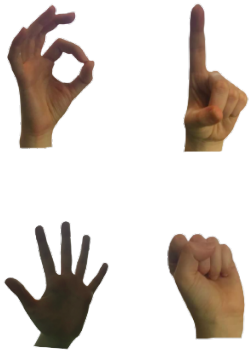


Figure 4.7: Gestures of Open,Close,Touch,Return to the Home Page

We tried to create our own data set using the data set we mentioned in Section 3.3. We used images that are similar to each of the four identified gestures. It was found correct to use K: Five set for Open, B: First Moved set for Close, C: Index set for Touch, A:L set for ,Return to the Home Page. [2] First, the images were cropped and resized that are chosen mixed. After these processes, the data reproduced by data augmentation were divided into two as validation and training data. The number of trains defined for each movement is approximately 9200, and the number of validation images is around 2300.

Model Architecture for LE-NET5

In 1989, Yann LeCun presented a convolutional neural network called LeNet. Generally, LeNet refers to LeNet-5 and is a simple convolutional neural network. [6]

The fact that our Input Image sizes are 32X32 was the biggest factor pushing us to use this model. Since we could not use a ready dataset, the change in the number of layers in order to train our model well, resulted in good results.

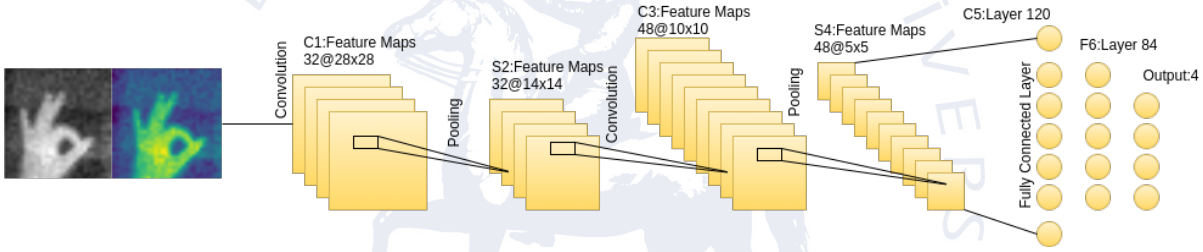


Figure 4.8: A Structural Diagram of the Proposed LENET5 for Static Gesture

The general architecture for LeNet-5 is as given in Figure 3. The input layer C1 acts like the retina, which receives centered and size-normalized gesture images (otherwise, some images may not fit in the input layer). The next layer, S2, consists of several feature maps that have the same role to gestures as their simple cells. In practice, a feature map is a square. The weights in a feature map need to be the same so they can detect the same local feature in the input image. The weights between feature maps are different so they can detect different local features. Each unit in a feature map has a receiver field.

Table 4.3: The structural layers and number of parameters of the LENET5.

Model:"Static_Gesture_Model		
Layer(type)	Output Shape	Param#
conv2d_8 (Conv2D)	(None, 28, 28, 32)	832
max_pooling2d_8 (MaxPooling2)	(None, 14, 14, 32)	0
conv2d_9 (Conv2D)	(None, 10, 10, 48)	38448
max_pooling2d_9 (MaxPooling2)	(None, 5, 5, 48)	0
flatten_4 (Flatten)	(None, 1200)	0
dense_16 (Dense)	(None,120)	307456
dense_17 (Dense)	(None, 84)	21588
dense_18 (Dense)	(None, 10)	850
dense_19 (Dense)	(None, 4)	44
Total params: 369,218 Trainable params: 369,218 Non-trainable params: 0		

If to summarize the model we outlined in the table, we use two convolutional layers, then twice a pooling layer (32-filters and 48-filters, respectively), and finally three fully connected layers with 4-class softmax units.

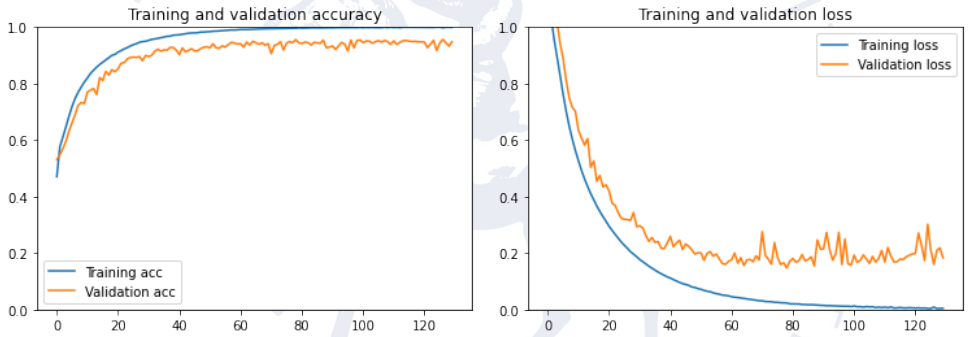


Figure 4.9: Loss and Accuracy Curves (Training and Validation Set)

And the result is close to perfect. Data augmentations worked. The learning algorithm, which reached 95% validation accuracy, gave very successful results. In the next step, images will be given to the model again for testing and the results will be observed.

Test Sets and Results

4.3.2 Dynamic Gesture Recognition

Table 4.4: Trim Register 1 (write only) [19]

Addr / CMD	0x1A (7 Bit!) / 0x03							
Trim Reg 1	7	6	5	4	3	2	1	0
Name	RFU		REF_CAL		MBIT TRIM			

REF_CAL: selectable amplification

MBIT_TRIM: m = 4 to 12 (m+4) bit as ADC resolution

In order to get the maximum efficiency from the sensor, to set the ADC Resolution to 16 bits, m of 12 was determined according to the data in the table. Since the framerate was too low, it was decided to speed up by sacrificing quality, but the quality was too low, revealing that ADC resolution should not be compromised. According to the resolutions, the images are as in figure 2.

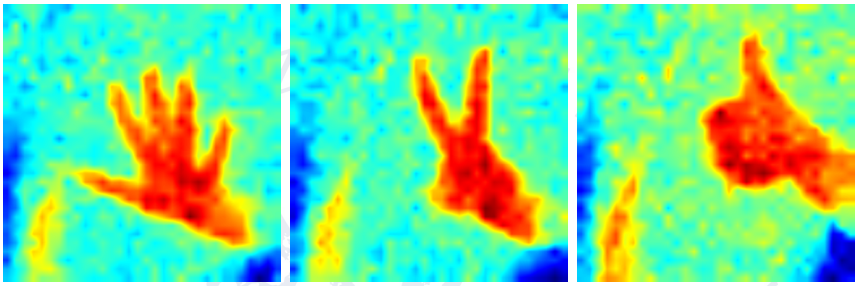


Figure 4.10: Thermal Images 8Bit-12Bit-16Bit ADC Resolution)

Table 4.5: Trim Register 2 (write only) [19]

Addr / CMD	0x1A (7 Bit!) / 0x04							
Trim Reg 1	7	6	5	4	3	2	1	0
Name	RFU			BIAS TRIM TOP				

BIAS_TRIM_TOP:0 to 31 $\longrightarrow$ 1μA to 13μA

Table 4.6: Trim Register 3 (write only) [19]

Addr / CMD	0x1A (7 Bit!) / 0x05							
Trim Reg 1	7	6	5	4	3	2	1	0
Name	RFU			BIAS TRIM BOT				

BIAS_TRIM_BOT: 0 to 31 $\longrightarrow$ 1μA to 13μA

This setting is used to adjust the bias current of the ADC. A faster clock frequency requires a higher bias current setting. [19]

Having ADC resolution 16 made us think that it did not affect us much. Afterward, it was said that if the BIAS current value is set to the maximum, its speed may increase. However, as a result of this experiment, almost no change was observed in the framerate.

Table 4.7: Trim Register 4 (write only) [19]

Addr / CMD	0x1A (7 Bit!) / 0x06							
Trim Reg 1	7	6	5	4	3	2	1	0
Name	RFU		CLK TRIM					

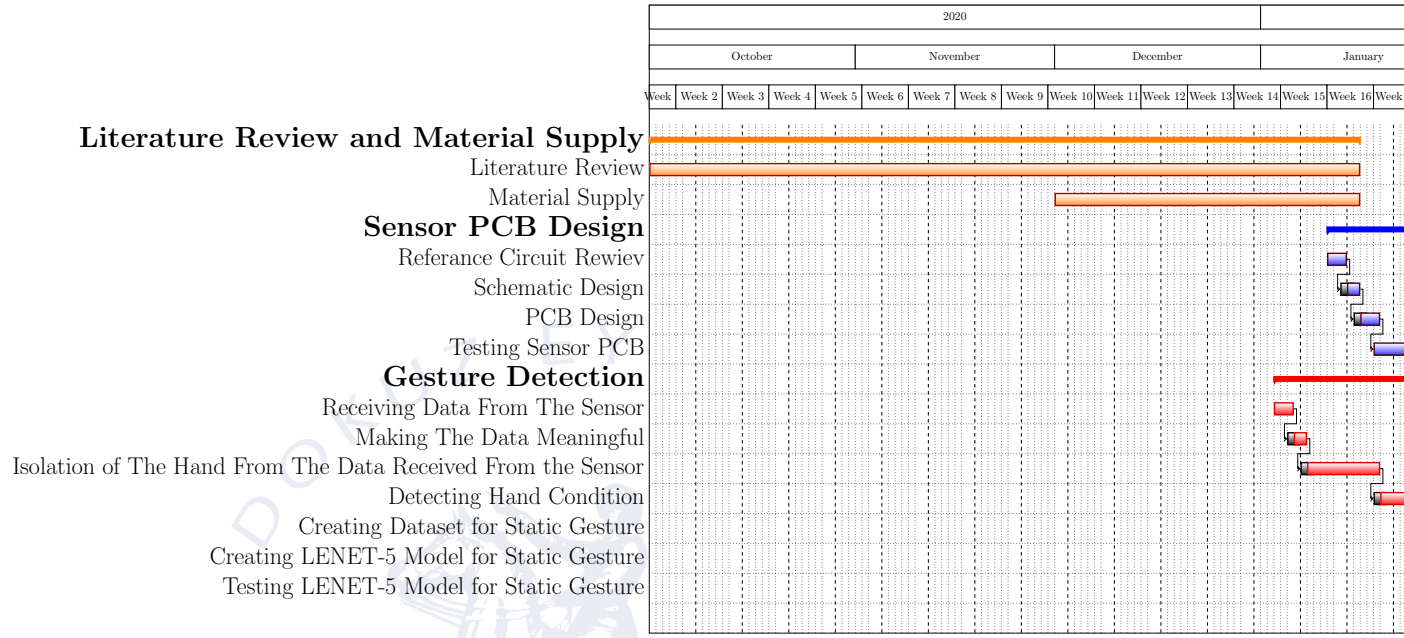
CLK_TRIM:0 to 63 $\longrightarrow$ $1MHz$ to $13MHz$

The point where we could most easily observe the increase in Frame Rate was the Clock Trim set. Although I set the maximum value, I could not get the desired result here.

4.4 Matching Gesture to Commands of Smart Mirror

In the project carried out by Vestel, friends who are interested in the smart mirror part are expected to define the gestures for the android side of the smart mirror.

5. Work Plan and Work Packages



6. COST ANALYSIS

6.1 Economical Costs

In this project, the economic cost is only in the hardware part. A Raspberry Pi 4B and Thermopil array sensor are used in our project. The market price of Raspberry Pi 4B is $55. The thermopil array sensor, on the other hand, has been chosen the most suitable for our project and has been researched for the market. The price of the HEIMANN HTPA32x32d Thermopil array sensor is $87. There is currently no separate cost in the software part. Because using opensource software.

6.2 Environmental,Political and Social Costs

Infrared sensor arrays have emerged in recent years and are products that allow the application of thermal imaging technologies to consumer electronics. With the introduction of Covid-19 into our lives, it is frequently encountered in the field of thermal imaging. Since a consumer electronics product using these products has not been produced in Turkey yet, it is considered to contribute to the national knowledge. In the project where privacy is prioritized, the privacy of the person will be ensured by using a thermopile array sensor. Also, the project is executed by Vestel A.Ş. so the project is expected to inspire different projects within Vestel. The fact that these sensors are newly introduced to the market will pave the way for innovative applications in many fields and thus method changes that can turn into patents. For example, although gesture recognition is performed with cameras conventionally, it offers a significant innovation in methodology, using passive infrared arrays. Smart mirrors that can be controlled with gesture recognition can have a positive effect on the preference of the products in question by the consumers. In doing so, using a technology that does not impair personal privacy will ensure that the users put the product into their lives safely and the applications related to the product become widespread.

7. CONCLUSION

Bibliography

[1] Adafruit's Raspberry Pi Lesson 4. GPIO Setup.
URL https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i

[2] MultiModalHandGesture_dataset.
URL $http://www.gti.ssr.upm.es/data/MultiModalHandGesture_dataset$

[3] OpenCV: OpenCV-Python Tutorials.
URL $https://docs.opencv.org/4.5.2/d6/d00/tutorial_py_root.html$

[4] vsergeev. python-periphery: A pure Python 2/3 library for peripheral I/O (GPIO, LED, PWM, SPI, I2C
URL https://github.com/vsergeev/python-periphery

[5] Fukushima, K. and Miyake, S. "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position". Pattern Recognition, volume 15, no. 6, pages 455–469, 1982.
URL https://linkinghub.elsevier.com/retrieve/pii/0031320382900243

[6] Lecun, Y., Bottou, L., et al. "Gradient-based learning applied to document recognition". Proceedings of the IEEE, volume 86, no. 11, pages 2278–2324, 1998.
URL http://ieeexplore.ieee.org/document/726791/

[7] Schilz, J. "thermophysica minima thermoelectric infrared sensors (thermopiles) for remote temperature measurements ; pyrometry", 2001.

[8] Gonzalez, R. Digital image processing. Prentice Hall, Upper Saddle River, N.J, 2008.

[9] Ghafouri, S. and Seyedarabi, H. "Hybrid method for hand gesture recognition based on combination of haar-like and HOG features". In 2013 21st Iranian Conference on Electrical Engineering (ICEE). IEEE, 2013.

[10] Prasuhn, L., Oyamada, Y., et al. "A HOG-based hand gesture recognition system on a mobile device". In 2014 IEEE International Conference on Image Processing (ICIP). IEEE, 2014.

[11] Abadi, M., Agarwal, A., et al. "TensorFlow: Large-scale machine learning on heterogeneous systems", 2015. Software available from tensorflow.org.
URL https://www.tensorflow.org/

[12] Chen, W.-H. and Ma, H.-P. "A fall detection system based on infrared array sensors with tracking capability for the elderly at home". In 17th. IEEE, 2015.

[13] Chollet, F. et al. "Keras", 2015.
URL https://github.com/fchollet/keras

[14] OpenCV. "Open source computer vision library". 2015opencv, 2015.

[15] Kawashima, T., Kawanishi, Y., et al. "Action recognition from extremely low-resolution thermal image sequence". pages 1–6, 2017.

[16] Krizhevsky, A., Sutskever, I., and Hinton, G. E. "ImageNet classification with deep convolutional neural networks". Communications of the ACM, volume 60, no. 6, pages 84–90, 2017.
URL https://dl.acm.org/doi/10.1145/3065386

[17] Li, G., Tang, H., et al. "Hand gesture recognition based on convolution neural network". Cluster Computing, volume 22, no. S2, pages 2719–2729, 2017. Publisher: Springer Science and Business Media LLC.

[18] Himblot, T. Data augmentation : boost your image dataset with few lines of Python, 2018. Publication Title: Medium.
URL https://medium.com/@thimblot/data-augmentation-boost-your-image-dataset/

[19] Lupp, S. "HTPA32x32dR2L5.0/0.85F7.7eHiC Thermopile Array With Lens Optics Rev3.0". Technical report, Heimann, 2018.

[20] Phung, V. H. and Rhee, E. J. "A Deep Learning Approach for Classification of Cloud Image Patches on Small Datasets". Journal of Information and Communication Convergence Engineering, volume 16, no. 3, pages 173–178, 2018.
URL https://doi.org/10.6109/JICCE.2018.16.3.173

[21] Ba, N. L., Oh, S., et al. "A 256 pixel, 21.6 uw infrared gesture recognition processor for smart devices". Microelectronics Journal, volume 86, pages 49–56, 2019.

[22] Pinto, R. F., Borges, C. D. B., et al. "Static hand gesture recognition based on convolutional neural networks". Journal of Electrical and Computer Engineering, volume 2019, pages 1–12, 2019.

[23] Tateno, S., Zhu, Y., and Meng, F. "Hand gesture recognition system for in-car device control based on infrared array sensor". pages 701–706, 2019.

[24] V, A. and R, R. “A Deep Convolutional Neural Network Approach for Static Hand Gesture Recognition”. Procedia Computer Science, volume 171, pages 2353–2361, 2020. Publisher: Elsevier BV.