MKEL 1123
ADVANCED MICROPROCESSOR SYSTEM
Section: 06

MILESTONE 6

LECTURER'S NAME: DR. USMAN ULLAH SHEIKH

Group Members:

KOAY KAH LEONG	MKE201025
TAN PEI HONG	MKE201046

Arm-based Temperature Controlled Lighting System

***Abstract*— LEDs are generally considered to be an important lighting source of the future. In this paper, an ARM-based LED intelligent lighting system is being proposed. Based on the ambient temperature obtained by sensors, the microcontroller would generate PWM signals to the LED to control the color produced automatically. The data received by the temperature sensors are being optimized using Kalman filter as to obtain a more stabilized reading so that the LED color tone will change accordingly.**

Keywords*— *Image color analysis, Lighting, PWM, Temperature control, Arm, Kalman Filter

I. INTRODUCTION

In modern day living, lighting plays a significant part in the everyday life of people. With different weather condition or ambience temperatures, lighting can create different types of atmosphere that would satisfy people's need psychologically. Color's influence on temperature perception has long been well-known of, according to researches [1]. The colors that has a dominant color or frequencies towards the red spectrum are warm while the colors towards blue spectrum appears to be cold [2]. These are based on studies which were involved on temperature judgment on non-contact thermal stimuli.

In regards to this, a smart temperature-controlled lighting system is being proposed where the color tone of the lightning would change in accordance to the ambience temperature. For instance, when the temperature is high, the lighting would change to a cooler color tone where it would give people a false perception of the actual temperature of the surrounding. This is to integrate the expectations based on the visual information of a person together with the sensory inputs in a way where it emphasizes the contrast between the two making the perception opposed to the expectation. This implementation can be applied in a wide variety of field such as landscape lightning, office lighting or automotive interior lightning.

This project is to be implemented mainly using a STM32 Nucleo-64 development board with temperature sensor and LED(RGB) for color mixing. The color mixing on the LED are being done by the pulse width modulation (PWM) technique based on the temperature value it received, to generate the appropriate color through additive mixing. To add on, the temperature data that were received directly from the temperature sensor, will go through a Kalman filter as to reduce the noise of from the signal so that a more accurate reading can be obtained. The main draw of this system is the simple usability, less energy usage, autonomy control and most importantly, it's economical as it uses low power devices to drive the whole system. In the next section, some of the similar systems and topics are being reviewed. Section III will be discussing on the implementation of the project such as the hardware and software required as well as their configuration, and the algorithm used. Section IV are about the results that has been achieved from the implementation.

II. CURRENT WORKS

Over the decade, several similar microcontroller-based systems had emerged with the rise in popularity of automated systems and IoT. Here are some of the relevant papers on the topic.

In this paper by Gurmu M [3], the authors has proposed an Arduino controlled system whereby it uses DHT11 sensor to control the speed of fan accordingly as to cool down the room when it's hot. The speed of the fan is being adjusted by the PWM signals generated from the microcontroller.

This paper [4] discusses the implementation of a Arduino based temperature and humidity control system to help cultivate oyster mushroom. It utilizes DHT11 sensor to measure the temperature as well as humidity and with ESP8266 (Wi-Fi module), it enabled data to be sent online so as farmers could remotely monitor their crops.

The authors of this paper have proposed an ATMega8535 microcontroller controlled, Wi-Fi based temperature monitoring system which would determine the melting point of a material caused by the thermal events within a furnace. It uses a thermocouple sensor to measure and sample the temperature, which then used Kalman Filter algorithm to filter to noise from the sensors.

III. METHODOLOGY

A. Hardware

The microcontroller unit that was used in the system is NUCLEO-F411RE, which is based on high performance ARM Cortex-M4 embedded microprocessor. Couple more of hardware are needed to implement the whole system which includes LM35 temperature sensor, a 2x16 LCD display, RGB LED, potentiometer and resistors. Fig .1 shows a simplified block diagram for the hardware.

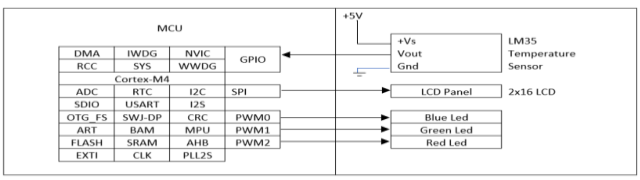
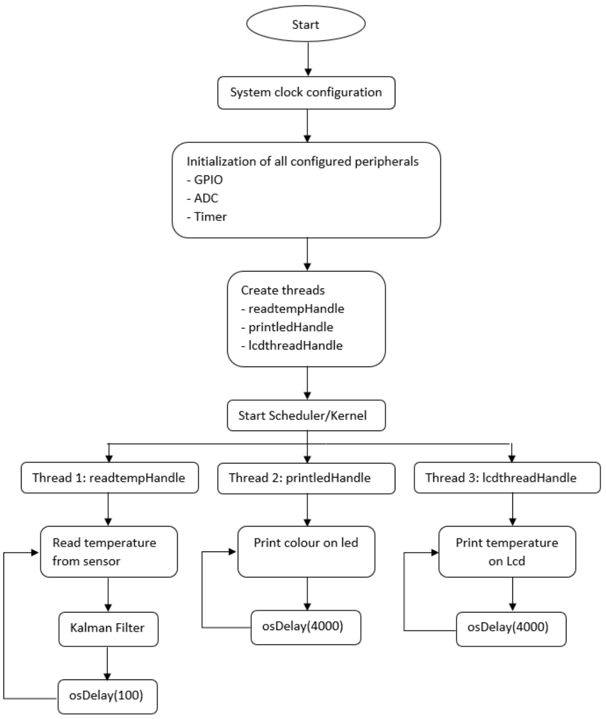


Fig. 1. Hardware block diagram

B. Software

This project is being developed using STM32CubeMx which will do the all the needed configuration for the development board including the pinout, clock, ADC and etc. After the configuration has been done, it will generate the C code for the board, and the main implementation of the program is being done using the Keil uVision 5. STMStudio was also used to interface with the microcontroller and show the real time data from the board.

C. *An overview of the program*



Fi g. 2. Flow Chart

Fig.2 shows the flow chart of the program. The Real-time operating system (RTOS) that run on the microcontroller STM32F411RE allow the execution of multiple tasks concurrently. This enabled the project to be controlled by 3 threads which will run concurrently. Each of the threads perform a different task where readtemp thread is mainly used for extracting the sensor reading from the LM35 temperature sensor. After extracting the reading from the sensor, the reading will then be optimized using Kalman filter to reduce the noise. The printled thread will then perform the PWM signal generation to light up the corresponding color to the RGB LED. The lcdthread thread will be controlling the lcd display where the temperature reading and the color to be changed will be printed.

D. *Algorithms*

In this project, couple of algorithms are being adopted which will be discussed in detail in this section.

Kalman Filter

The Kalman Filter, or also known as Linear quadratic estimation (LQE) is an algorithm which are widely used for data filtering and reducing error in data measurement especially on temperature monitoring, due to the typical thermal process nature which can be modelled as Gaussian noise process [5]. Which is why the Kalman Filter will be applied on this project, as to stabilize the temperature values received from the LM35 temperature sensor.

The Kalman filter estimates the state from its present and previous measurement to acquire an estimation of its current state. In order to implement it, the measured process needs to be described by a linear system into state and output equation as follows:

$$x_{k+1} = Ax_k + Bu_k + w_k \qquad (1)$$

$$z_{k+1} = Hx_k + v_k \qquad (2)$$

The constants A, B and H from the first and second equation represents the matrices. The k is the time index while x represents the state of the system. The u from the equation (1) is the input to the system while z is the process noise and v is the measurement of noise [6]. Equation (1) is usually described as the process model and equation (2) is the measurement model.

The Kalman filter is a feedback control process where it would first estimate the process state and then acquire the feedback from the measurement with noise. Generally, it can be described in 2 segments as being shown in Fig.3. $\hat{x}_k$ is an estimation of $\hat{x}^-_k$. The P represents the error covariance between the estimation and the actual value, K is the Kalman gain, Q and R are the process and measurement noise covariance while H is the observation matrix. The time update equations as shown in Fig. 3 are used to project the current and error covariance which it is useful for obtaining the priori estimation while the equations for the measurement update are used for the feedback control to obtain an improved posteriori estimation [7].

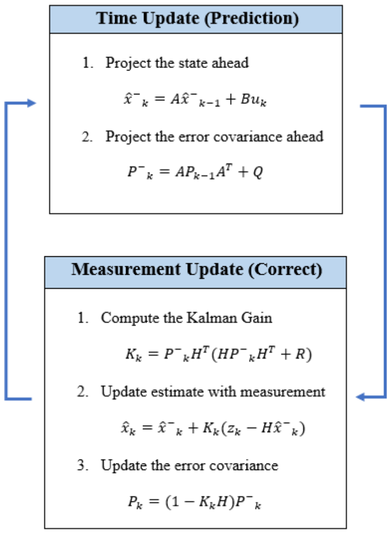


Fig. 3 Flow of Kalman Filter

On this project, the temperature value is obtained directly from the LM35 temperature sensor. Kalman filter are being implemented on the raw temperature data to filter out the noise. Since only single variable are being used here, which is the temperature, thus all matrices equate to 1x1. This results in matric A and H to become 1 because the value is directly being measured. The Matric B will become 0 because there are no input to the system. This mekes the Kalman filter to be operating in only 1 dimension or in other words, this implementation will only work with scalar, not vectors. With these constraints, it will further simplify the equation to as follows:

$$K = \frac{P \times H}{H \times P \times H + R} \qquad (3)$$

$$P = (1 - K \times H)P + Q \qquad (4)$$

$$x = x + K(x - Hx) \qquad (5)$$

Pulse Width Modulation (PWM)

In this project, RGB LEDs is being used to display a wide variation of colors. Basically, RGB LED is made up of three individual primary colors LEDs which are red, green, and blue and these LEDs emits three colors at the same time to generate a wide variation of colors through additive mixing. Each color component can be controlled individually and has a range of intensity from 0 to 255. The combination of these intensities from each color component can produce up to 16 million (256x256x256) of different colors. PWM (pulse width modulation) dimming is being adopted to control the intensity of each color component by modifying the ratio of ON time and OFF time (duty cycle) of each color component. Higher percentage of duty cycle indicates longer ON time of the LED. Longer ON time of LED results in higher voltage provided to respective color pin and high intensity of respective color component. In the formula below, ϕ is the LED luminous flux, ϕ_{on} is the luminous flux when LED is ON, D_{on} is the PWM duty cycle. Luminous flux is defined as the brightness or the intensity of the LEDs.

$$\phi = \phi_{on} * D_{on} \qquad (6)$$

In equation (7), R_{led} is the LED equivalent resistance, I_{on} is the current when LED is ON, η is the efficiency of luminous. Due to the value of R_{led}, I_{on}, and η only can be affected by the material of LED, they are always fixed as the material of LED remain unchanged. This resulted the value of luminous flux (ϕ_{on}) is fixed.

$$\phi_{on} = I_{on}^2 * R_{led} * \eta \qquad (7)$$

Since the value of luminous flux (ϕ_{on}) is fixed, in equation (6), we can conclude that the LED luminous flux (ϕ) is directly proportional to the PWM duty cycle D_{on}. According to International Commission on Illumination (CIE 1931), a chromatic diagram is introduced as a standard of producing color and sufficient to represents all colors.

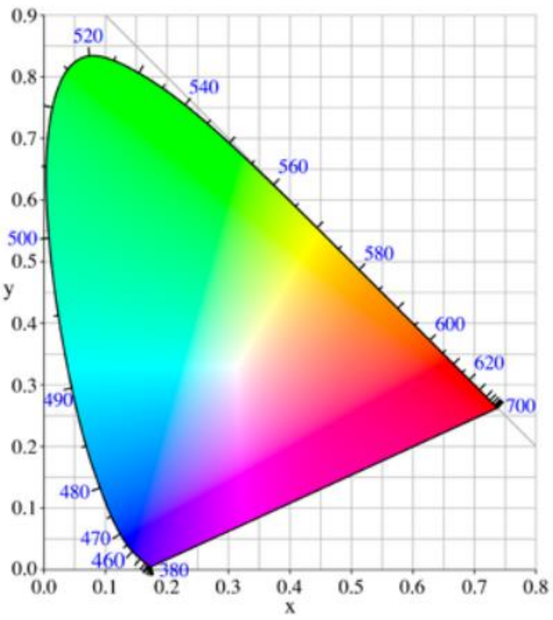


Fig. 4 Chromaticity diagram

X-axis is the ratio of red while Y axis is the ratio of green. The chromaticity coordinates $C(x_c, y_c, z_c)$ can be obtained from chromatic diagram is the combination of RGB colors, where Red is represented by chromaticity coordinate $X'(x'_r, y'_r, z'_r)$, Green by $Y'(x'_g, y'_g, z'_g)$ and Blue by $Z'(x'_b, y'_b, z'_b)$. Hence, according to optical law, the equations can be derived as:

$$X' = (x'_r X, y'_r Y, z'_r Z) \qquad (8)$$
$$Y' = (x'_g X, y'_g Y, z'_g Z) \qquad (9)$$
$$Z' = (x'_b X, y'_b Y, z'_b Z) \qquad (10)$$

X Y Z represents the primary color in chromaticity diagram. It can be written in the form of matrix below:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = A \text{ x } \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad (11)$$

where $A = \begin{bmatrix} x'_r & y'_r & z'_r \\ x'_g & y'_g & z'_g \\ x'_b & y'_b & z'_b \end{bmatrix}$, we transpose it and get

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = A^{-1} \text{ x } \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \qquad (12)$$

we inverse A matrix and get A^{-1}. According to CIE 1931 standard, we get

$$C = [x_c, y_c, z_c] \text{ x } \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} \qquad (13)$$

We substitute 5 to 6 and we get

$$C = [x_c, y_c, z_c] \text{ x } A^{-1} \text{ x} \begin{Bmatrix} X' \\ Y' \\ Z' \end{Bmatrix} = [r, g, b] \text{ x } \begin{Bmatrix} X' \\ Y' \\ Z' \end{Bmatrix} \qquad (14)$$

where r, g and b represent the brightness ratio of each RGB color component in RGB LEDs, and combined to form a mixed color C.

TABLE I. COLOUR MATCHING BY VARYING INTENSITY VALUE

	Duty cycle and intensity value set to each RGB color component		
Color	r	g	b
Red	100% (255)	0% (0)	0% (0)
Yellow	50% (127)	50% (127)	0% (0)
Green	0% (0)	100% (255)	0% (0)
Blue	0% (0)	0% (0)	100% (255)

Due to the intensity range of each RGB color component is varying from 0 to 255, the timer was configured in such a way that 100% of brightness ratio or duty cycle is equivalent to 255. Table I shows the duty cycle and intensity that is set to each RGB color component of the LED to produce 4 colors which are red, yellow green and blue. In this project, the timer on the microcontroller STM32F411RE have a clock frequency of 100Mhz. Prescaler register and auto reload register (ARR) is used to vary the duty cycle. Duty cycle is the percentage of the total time that the signal is in HIGH state per cycle. Since the timer is to be configured in such a way that 100% duty cycle is equivalent to 255 range of intensity, the clock rate divided by 255. Next, the ARR is set to 255 to let the timer to count up to 255 in one cycle. Hence, the prescaler register will be set to the value of 100M/255^2. There is a total of 16 million (256x256x256) possible color can be achieved as the intensity value in each RGB LED components varying from 0 to 255.

IV. Results and Discussion

The temperature data can be plot real time by using the STM Studio which will interface with the stm32 microcontroller and show the data real time as can be observed in Fig 5. As can be seen from the plotted temperature reading, the raw data obtained from the sensor is rather noisy as it has an average constant variation of $\pm 1.5℃$ despite the actual surrounding temperature being fixed. Therefore, Kalman filter is being applied on the raw data in order to flatten out these variations as to obtain a more stable temperature reading. This would also avoid voltage spikes where the temperature value would suddenly jump to a value very far away from the average value.

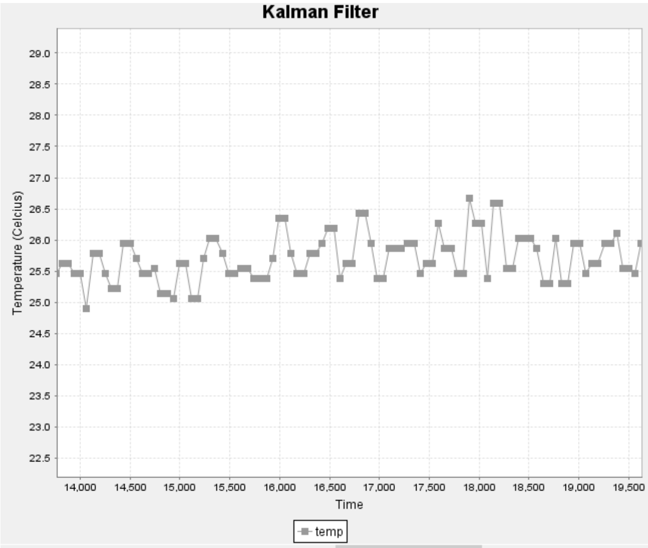


Fig. 5 Real time temperature reading

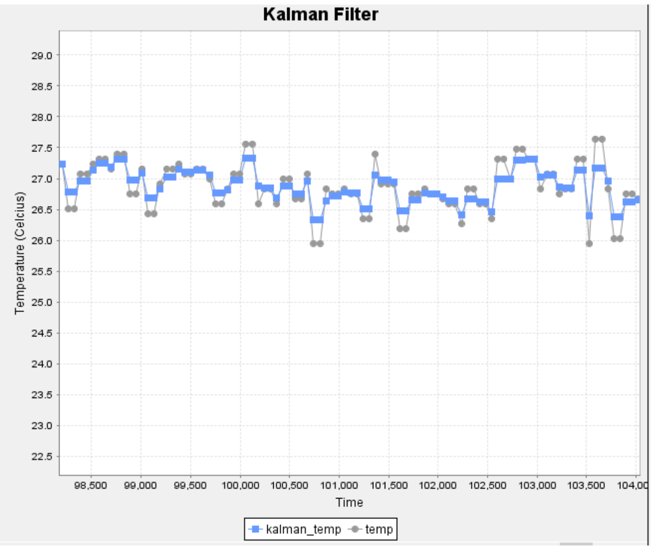


Fig. 6 Raw temperature reading vs optimized Kalman temperature reading

Fig. 6 shows the plot with together with the filtered readings from the Kalman Filter. As can be observed by the results, the voltage variation is slightly reduced as compared to the raw temperature reading, this shows that the filter work as being intended to. However, as can be observed from the figure, the variation still look as big and close to the original temperature reading, this is because this was running with a lower noice covariance. The noise covariance, R, has a direct impact on the Kalman gain as can be seen from equation (3). As the noise covariance increases, it reduces the Kalman gain which would filter more noise. This can be verified from the simulated graph shown on Fig.7.

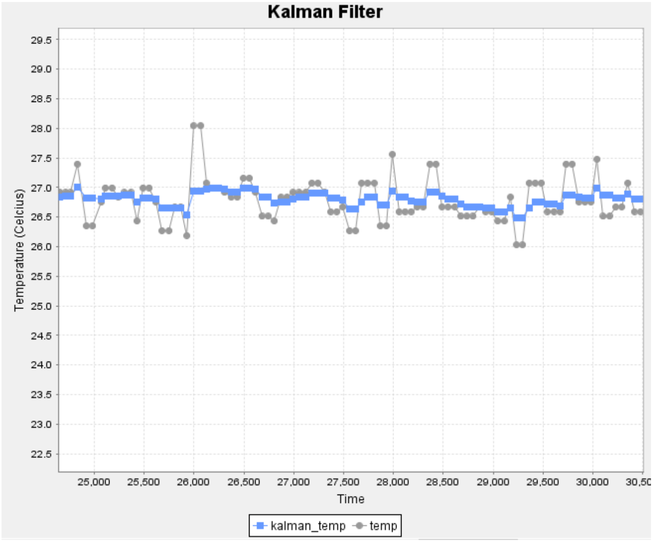


Fig. 7 Results on temperature reading with higher noise covariance

As can be observed from Fig.7, the filtered reading has shown a much more stable temperature reading when a higher noise covariance is used. The reading was improved tremendously as it doesn't vary as much over a short period of time. However, there is a tradeoff of having a higher noise covariance, which is it would slow down the speed of the filter. The more temperature reading is being optimized would result in a delayed temperature result. A fine line needs to be drawn in between these 2 aspects as to get an optimal performance of the filter.

Another alternative apart from the Kalman filter to optimize the sensor reading that can be looked into is the Moving Average filter. The Moving Average filter is as a good candidate which can get the job done as well, and would be rather accurate on systems which are expecting a signal to stay close to constant over time. However, for applications like temperature sensing systems where the temperature value would be constantly changing over time, Kalman filter would outperform the Moving Average filter as it is a recursive and adaptive filter which corrects its weighting each iteration. However, that being said, Moving Average filter would be a good fit too for this system if the temperature doesn't vary that much.

TABLE II. RESULTS ON THE HARDWARE

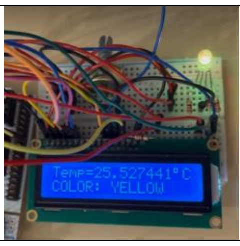
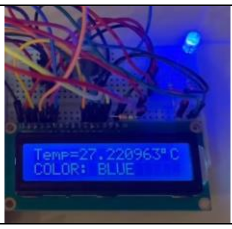
Red	Yellow
	
Green	Blue
	

Table II shows the results that has been obtained on the hardware which is controlled by the NUCLEO-F411RE development board. As can be observed from the results, at a lower(cooler) temperature, the RGB would be having a much warmer color like red. As the temperature gradually rises, the color would slowly go to a much cooler color spectrum which is blue. The color change would occur gradually as to avoid a sudden change in the color tone of the lighting so that it's more natural and comfortable. Table III shows the gradual color change from green to blue.

TABLE III. GRADUAL COLOR CHANGE ON THE RGB LED

One thing that was learned from the experiment was, a perfect yellow usually is difficult to be obtained on an RGB LED where the color would appear close to lime instead when the color code is set to yellow as can be seen on Table II. This is because RGB LED works on the additive principle. In additive mixing of light, the intensity and color are the two main components to produce a color. Two lighting sources focused at the same place will converge to amplify the total intensity. In order to get yellow, the blue LED must be turned off leaving only red and green on. However, by removing the blue source from the LED would cause the overall intensity from the RGB LED to become lower. From RGB **LED** datasheets, red **LED** contributed 800 millicandela (mcd) of light intensity, green **LED** contributed 900 mcd and blue **LED** contributed 4000 mcd. Thus, turning off the blue LED completely would lower the overall intensity of the RGB LED which makes yellow hardly attainable[8]. Hence, a yellow can only strained to be achieved by merely varying the intensity of red and green LED.

V. CONCLUSION

In conclusion, this project was able to be implemented successfully with achieving its main goal of obtaining an accurate desired colour of lightning depending on the temperature from the sensor. Multiple threads function of the CMSIS RTOS were being utilized to enable the concurrent execution of task like getting data from sensor and generating PWM signals to the RGB LED for color changing without affecting one another. Kalman filter was also being adopted in this project as to filter out the noise and get a more stabilized data. The project was being implemented using a high processing speed ARM microprocessor for the system where it is cheap and consume much lesser power. This project has demonstrated how a rather cheap and accessible microcontroller like the stm32 series can be integrated in to the everyday life of people.

REFERENCES

[1] Hsin-Ni Ho, "Color-temperature correspondence and its impact on object temperature perception," *Int. Symp. Found. Vis. Inf.*, 2015.

[2] C. A. Bennett and P. Rey, "What's so hot about red?," *Hum. Factors*, vol. 14, no. 2, pp. 149–154, Jul. 1972.

[3] G. M. Debele and X. Qian, "Automatic Room Temperature Control System Using Arduino UNO R3 and DHT11 Sensor," in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2020*, 2020, pp. 428–432.

[4] A. Najmurrokhman, Kusnandar, A. Daelami, E. Nurlina, U. Komarudin, and H. Ridhatama, "Development of Temperature and Humidity Control System in Internet-of-Things based Oyster Mushroom Cultivation," in *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2020*, 2020, pp. 551–555.

[5] Y. Chen, J. Xu, K. Luo, and S. Xu, "Multi-Temperature and Humidity Data Fusion Algorithm Based on Kalman Filter," *Res. J. Appl. Sci. Eng. Technol.*, vol. 5, no. 6, pp. 2127–2132, 2013.

[6] D. Simon, "Embedded Systems Programming," 2001.

[7] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," 2006.

[8] "Can RGB LED make yellow? - Quora." [Online]. Available: https://www.quora.com/Can-RGB-LED-make-yellow. [Accessed: 20-Jun-2021].