**Optimising Air Flow for Cooling Indoor and Al Fresco Dining Areas**

Goh Kheng Xi Jevan(A0199806L), *Undergraduate, ESP*, Lock Mei Lin(A0204751M), *Undergraduate, ESP*, Toh Wei Wen(A0204844H), *Undergraduate, ESP*.

***Abstract –* This project aims to optimise the airflow for cooling indoor and Al Fresco dining areas. Using thermal sensors, a prototype of damping and variable fan speed systems – to illustrate cooling indoor and outdoor areas respectively – will be presented in this report.**

I. INTRODUCTION

Being near the equator, Singapore experiences a staggering daily maximum temperature of about 30 to 32 degrees Celsius on average [1]. Hence, ACMV (Air conditioning and Mechanical Ventilation) systems are commonly installed to combat the tremendous tropical heat. On average, an ACMV system accounts for more than 40% of a building’s energy consumption [2]. Therefore, this project proposed the installation of sensors to control centralised ACMV systems automatically and better optimise cost and energy efficiency.

However, as this system can also be integrated in an outdoor environment with decentralised cooling system, this project focuses on environments with both indoor and outdoor elements.

# II. CURRENT PRODUCTS

This section describes some of the products that is currently on the market.

## A. Indoor ACMV systems

Currently, indoor ACMV systems employ different types of VAV (Variable Air Volume) diffusers to optimize cooling efficiency by controlling the air flow into each room through dampers [3].

However, air flow into a room is controlled by a setpoint temperature, that must be manually adjusted, causing energy inefficiency (temperature might not be optimal) and inconvenience. Additionally, rooms with multiple diffusers share the same setpoint temperature, resulting in non-flexible control of each damper and hence, possible cool air wastage.

## B. Outdoor cooling systems

Currently, many outdoor dining areas install fans that must be manually adjusted, generating energy wastage when the area have lesser patrons.

On the market, there are commercial fans with variable fan speeds that adapts to the surrounding temperature (e.g. AirEffect). However, they often rely on a localized thermal sensor that might not be a good representation of environment [4].

# III. SYSTEM DESIGN

This project proposed a decentralized sensor system to achieve a better representation of the environment temperature. Figure 1 shows an illustration of the system design. The green box indicates the Arduino Uno, while the red circles depict the thermal sensor, placed at various locations of a room.

Chart, scatter chart

Description automatically generated  
Figure 1: Illustration of the system design.

Outdoors

Indoors

These sensors will each be responsible for a distribution of temperature cooling systems. For indoor environment, the sensor will be controlling a damper which will regulate the airflow from an external fan (simulating cool air from the AHU – Air Handling Unit). For outdoor environment, the sensor will directly be controlling the speed of an AC fan.

# IV. MECHANICAL DESIGN

This section describes the mechanical design for the system, which includes the dampers for the indoor environment and variable fans for the outdoor environment.

## A. Indoor damping system

A round VCD (Volume Control Damper) will be placed in front of a fan simulating the airflow from the A.H.U. The circular damper will be connected to a stepper motor that is controlled by an Arduino Uno.

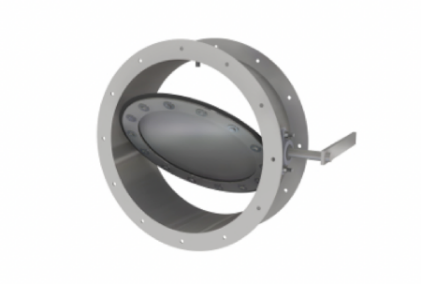
The design of the damping system is illustrated in Figure 2. The entire VCD will be fabricated with acrylic.



Figure 2: Design of the external circular damping system (front view).

## B. Outdoor variable fan speed system

A 3-wire fan with variable fan speed will be used to cool the outdoor environment.

# V. SENSOR DESIGN

Figure 3 depicts a sketch of the thermal sensor design.

**Diagram

Description automatically generated**  
Figure 3: Sketch of the thermal sensor setup.

Each thermal sensor is a 100k NTC thermistor connected in series with a 100kΩ resistor. They are then placed in various locations and connected in parallel with one another. A change in the temperature around a thermistor will induce a change in the thermistor’s resistance, causing a change in the voltage across it. This change in voltage can be recorded by the Arduino Uno’s analog input to trigger a signal to the damper’s or fan’s motors.

# VI. CURRENT RESULTS

The graph in Figure 4 displays the positive linear relationship between thermistor 2’s temperature and its voltage.

Figure 4: Thermistor 2 temperature vs. voltage plot.

Calibration of the thermal sensors was performed by varying a heat source and measuring the temperature around the thermistors. Then the voltage values across each thermistor was obtained and recorded every 10 ms and averaged over a period of 5 seconds. Refer to Annex B for results of all the other thermistors.

# VII. CONSIDERATIONS

## A. Result disparities

There is a disparity between the thermistor’s expected voltage-temperature graph and the actual graph. In theory, the voltage is expected to drop due to the increased resistance of the thermistors with increasing temperature, but instead, an upward sloping graph was obtained.

A key contributing factor to this error is the reading from 4 analog pins without any delay between each analog pin reading, which might have caused some signal interference. Hence, moving forward, the thermal sensors must be recalibrated.

## B. Other limitations

The speed of the fans is limited as it is unable to decrease beneath a required minimum value for overcoming inertia, potentially wasting energy when there are no patrons in that area.

# VIII. ANNEX

## A. References

|  |  |
| --- | --- |
| [1] | Meteorological Service Singapore, "Climate of Singapore," Meteorological Service Singapore, [Online]. Available: http://www.weather.gov.sg/climate-climate-of-singapore/. |
| [2] | D. Zhai and Y. C. Soh, "Balancing indoor thermal comfort and energy consumption of air-conditioning and mechanical ventilation systems via sparse Firefly algorithm optimization," 3 July 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7966028. |
| [3] | Price Industries, "Engineering Guide VAV Diffusers," 2011. [Online]. Available: https://www.priceindustries.com/content/uploads/assets/literature/engineering-guides/vav-diffusers-engineering-guide.pdf. |
| [4] | MacroAir, "AirEffect," MacroAir, [Online]. Available: https://macroairfans.com/controls-automation/aireffect/. |
| [5] | D. Marshall-George, "Direct and indirect evaporative cooling strategies," Condair, [Online]. Available: https://www.condair.sg/knowledge-hub/direct-and-indirect-evaporative-cooling-strategies. |
| [6] | "Indirect Evaporative Technology," Seeley International, [Online]. Available: https://www.seeleyinternational.com/ap/commercial/products/cooling/indirect-evaporative-air-conditioning-commercial-emea/. |

# 

## B. Results (calibration)

Figure 5: Thermistor 1 temperature vs. voltage plot.

Figure 6: Thermistor 2 temperature vs. voltage plot.

Figure 7: Thermistor 3 temperature vs. voltage plot.

Figure 8: Thermistor 4 temperature vs. voltage plot.

## C. Arduino codes (calibration)

//\*\*\*\*\*\*\*\*\*\*\*\*\* LAYOUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 1. DESCRIPTION #T\_DESCRIPTION

// 2. GLOBAL CONSTANTS

// 3. GLOBAL VARIABLES

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// ======== GLOBAL CONSTANTS==========

const int BAUD\_RATE = 9600;

const int NUMBER\_OF\_THERMISTORS = 4;

const int THERMISTOR\_PINS[] = {A0, A1, A2, A3};

const float READING\_PERIOD = 5; // period of each average reading in SECONDS

const int NUMBER\_OF\_SAMPLES = READING\_PERIOD / 0.01; // number of readings for average reading ##each sample is 0.01s

// ====== GLOBAL CONSTANTS END =======

float average\_voltage\_readings[4];

void setup() {

Serial.begin(BAUD\_RATE);

Serial.print("=========== Reading Start ==============");

}

void loop() {

// ------------ getting all readings in period --------------------

float total\_voltage\_readings[4] = {0, 0, 0, 0};

for (int i = 0; i <

NUMBER\_OF\_SAMPLES; ++i) {

for (int thermistor = 0; thermistor < NUMBER\_OF\_THERMISTORS; ++thermistor) {

total\_voltage\_readings[thermistor] += analogRead(THERMISTOR\_PINS[thermistor]);

}

delay(10);

}

// ----- getting all readings in period end ------

// ----------- calculating average ------------------

for (int thermistor = 0; thermistor < NUMBER\_OF\_THERMISTORS; ++thermistor) {

average\_voltage\_readings[thermistor] = total\_voltage\_readings[thermistor] / NUMBER\_OF\_SAMPLES;

}

// --------- calculating average end --------------

// ------------ printing output ------------------

Serial.print(average\_voltage\_readings[0]);

for (int thermistor = 1; thermistor < NUMBER\_OF\_THERMISTORS; ++thermistor) {

Serial.print(", ");

Serial.print(average\_voltage\_readings[thermistor]);

}

Serial.println("");

// ---------- printing output end ----------------

}

* The four temperature sensors are placed at different locations in a room to better represent the temperature distribution
* Green box is pc 2 Arduino stick out1 arduino rep indoor master 1 Arduino rep outdoor master
* For indoor master connected to one sensor and one motor and connect to another Arduino(indoor slave)
* Indoor slave connect to 1 sensor and 1 motor
* For the outdoor master, connected to 1 sensor and 1 fan and another Arduino(/outdoor slavery)
* Outdoor slave connect to 1 sensor and 1 fan ya wtf I follow this to draw a diagram?gay