Monitoramento da Quantidade de Carbono no Ar - Carby

Aline Rosa dos Santos Rocha[1], 16/0023076, Sofia Consolmagno Fontes[2], 16/0018234
[1,2]Programa de Engenharia Eletrônica, Faculdade Gama - Universidade de Brasília, Brasil

***Resumo*—O artigo em questão descreve o desenvolvimento de um módulo eletrônico em conjunto com a *Raspberry Pi*, capaz de monitorar a quantidade de gás carbônico (CO_2) no ar, o qual será elaborado para a disciplina de Sistemas Operacionais Embarcados. A proposta surgiu a partir da dificuldade de clínicas médicas e odontológicas verificarem as trocas de ar realizadas na sala de espera durante o período de pandemia do vírus SARS-CoV-2. Por fim, para avaliar o projeto será realizado um protótipo, o qual passará por testes de viabilidade em um consultório odológico de Brasília.**

***Index Terms*—COVID-19, Qualidade do Ar Interior, CO_2, *Raspberry Pi* e Sistema Embarcado**

I. Introdução

Em dezembro de 2019 na China foi diagnosticado o primeiro paciente infectado pelo vírus SARS-CoV-2, causador da doença COVID-19, a qual atualmente constitui uma Emergência de Saúde Pública de Importância Internacional [1]. Por conseguinte, a pandemia se espalhou rapidamente, sendo o primeiro caso confirmado no Brasil no dia 25 de fevereiro de 2020 [2].

A transmissão do vírus se dá pela vias aéreas, por meio de gotículas respiratórias expelidas durante a fala, tosse ou espirros. Dessa forma, é fundamental seguir as principais medidas orientadas pelas autoridades sanitárias, as quais são: isolamento físico ou domiciliar, assepsia, cuidados individuais, utilização de máscaras e respiradores do tipo N95 e em ambientes fechados deve-se realizar a transferência e substituição do ar possivelmente contaminado do interior pelo ar exterior [3].

Consequentemente, a maior parte da transmissão do vírus SARS-CoV-2 ocorre em ambientes fechados, principalmente pela inalação de partículas transportadas pelo ar que contêm o coronavírus. Além do mais, o dióxido de carbono (CO_2) existe naturalmente na atmosfera, é uma molécula produzida pelo corpo humano através da respiração [4]. Logo, para ambientes fechados os níveis de CO_2 podem ser utilizados para aferir se o ambiente está sendo preenchido por exalações potencialmente infecciosas [5].

Normalmente, o nível de CO_2 no ambiente é estável e tem-se uma variação desse nível a partir da exalação humana, assim é possível estimar se há uma quantidade suficiente de ar fresco entrando no espaço [6]. Por conseguinte, em ambientes externos os níveis de CO_2 são, em média, de 400 partes por milhão (ppm), e em um ambiente interno bem ventilado terá aproximadamente 800 ppm, dessa forma, um número maior do que esse indica que o ambiente precisa de mais ventilação [7].

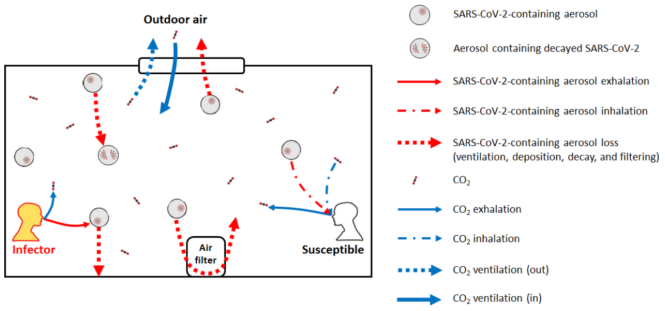

Fig. 1. Esquema da ilustração da expiração, inalação e outros processos de perda de SARS-CoV-2 contendo aerossóis em um ambiente interno. Fonte: Exhaled CO2 as COVID-19 infection risk proxy for different indoor environments and activities [5]

II. Justificativa

Após um ano de pandemia e sem medidas de isolamento social efetivas, é inevitável que haja circulação de pessoas e possibilidades de aglomeração, além de que existem pessoas assintomáticas não diagnosticados mas que continuam dispersando o vírus. Com tais características, os ambientes fechados como clínicas e hospitais que apresentam uma grande fluxo de pessoas diariamente, sofrem com os altos riscos de contaminação tanto para os pacientes quanto para a equipe de saúde [8]. Assim, clínicas odontológicas podem provocar a infecção cruzada devido ao uso de instrumentos que produzem aerossóis, gotículas e secreções de saliva e sangue [9].

Dessa forma, com a finalidade de diminuir a dispersão da COVID-19 este projeto consiste no desenvolvimento de um módulo eletrônico capaz de detectar quantas trocas de ar deverão ser realizadas por hora em uma clínica odontológica, para obter uma maior segurança do profissional da saúde e dos pacientes. As variáveis que compõem as realizações das trocas, são: quantidade de pessoas, medidas do espaço físico, concentração de CO_2, temperatura e umidade.

III. Objetivo

Proporcionar aos funcionários e clientes de uma clínica odontológica segurança contra a propagação do vírus dentro do ambiente fechado. Para isso, tem-se por objetivo projetar e prototipar um equipamento que atue na sala de espera de uma clínica odontológica, o qual pode ser acionado pelo comando de voz do usuário para monitorar a qualidade do ar por meio de sensores de CO_2, temperatura e umidade. Além do mais, esse processamento deve ser realizado pela Single-Board Computer

(SBC) Raspberry Pi 3B, e será possível ao usuário observar um gráfico da concentração de CO_2 nas últimas 24 horas e deverá emitir alertas quando houver a necessidade das trocas de ar.

IV. Metodologia

Com a finalidade de de acompanhar e analisar o desenvolvimento do projeto dividiu-se os marcos em quatro pontos de controle, conforme descritos abaixo:

- **PC1:** proposta do projeto (justificativa, objetivos, requisitos, benefícios, revisão bibliográfica);
- **PC2:** protótipo funcional do projeto, utilizando as ferramentas mais básicas da placa de desenvolvimento, bibliotecas prontas etc;
- **PC3:** refinamento do protótipo, acrescentando recursos básicos de sistema (múltiplos processos e threads, pipes, sinais, semáforos, MUTEX etc.);
- **PC4:** refinamento do protótipo, acrescentando recursos de Linux em tempo real.

Além do mais, para facilitar o desenvolvimento do protótipo, o projeto será dividido em quatro áreas de trabalho: módulos de aquisição, controle, alimentação e estrutura, e Software.

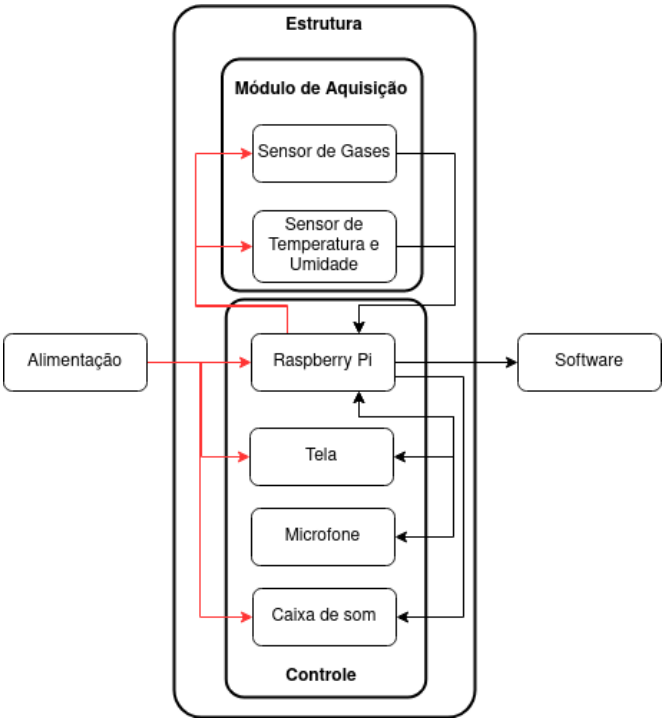

Fig. 2. Divisão das áreas de trabalho do protótipo

A. Módulo de aquisição

A partir do módulo de aquisição que é composto por sensores de CO_2, temperatura e umidade é possível obter os dados do ambiente fechado e retornar para o servidor.

B. Controle

A área de controle será o foco principal do projeto e contará com a *Raspberry Pi 3B* para realizar toda comunicação entre os módulos e o usuário a partir da tela, do microfone e da caixa de som.

C. Alimentação e Estrutura

Esta área é responsável pela elaboração do circuito de alimentação do protótipo. Além do mais, outro enfoque dessa área é o correto posicionamento dos sensores na estrutura para possibilitar uma melhor medição. Assim, o protótipo será construído visando uma interface amigável e intuitiva para o usuário.

D. Software

O software indica ao usuário por meio de avisos quando é necessário realizar as troca de ar do ambiente e, ainda, será possível observar um gráfico da concentração de CO_2 nas últimas 24 horas.

V. Requisitos

Dado que o projeto será desenvolvido para um fim específico e com um público alvo bem definido, os requisitos devem ser definidos de forma clara e objetiva para que o módulo eletrônico supra e alcance as condições e as capacidades em que ele foi projetado.

A. Requisitos de Materiais e Custos

O projeto deve ser viável economicamente para o escopo da disciplina e restrições da universidade. Assim, uma análise de custos mais detalhada será feita em fases mais avançadas do projeto.

B. Requisitos Técnicos

1) **Hardware**
   O hardware deverá adquirir os dados de temperatura, umidade e concentração de CO_2 corretamente, além de que é necessário conter pelo menos um conversor A/D para que haja uma entrada digital vinda do sensor de gás analógico para a *Raspberry*. O projeto do hardware deverá ser acessível para o usuário, construído visando uma interface amigável e intuitiva, mas que também forneça a ele ferramentas e funcionalidades que supram os objetivos que o sistema se propõe. O protótipo resultante do projeto deve ser robusto, portátil e funcional.
2) **Software**
   Os comandos de voz de entrada, interpretação e os comandos de voz de saída serão processados com o auxílio do *Google Assistent DSK* com a configuração da biblioteca de voz pelo *ActionPackage*. O software disponibilizará via web, por comunicação MQTT, um gráfico com o nível da concentração de $CO2$ nas últimas 24 horas, sendo assim, é fundamental a integração entre a *Raspberry Pi* e o armazenamento dos dados. Com o decorrer do projeto pode-se adicionar algumas funcionalidades no software, tais como a criação de um servidor, que gerencie diversos módulos de aquisição.

VI. Benefícios

O projeto se mostra importante e favorável na travessia deste momento de Emergência de Saúde Pública de Importância Internacional, uma vez que, irá beneficiar diretamente secretárias, dentistas, técnicos, pacientes e demais pessoas que frequentam a clínica odontológica, além de que, por meio do monitoramento é possível garantir uma maior segurança. Outrossim, também beneficia indiretamente a sociedade, já que tem-se a produção e conhecimento científico e experimental do monitoramento de vírus, principalmente o SARS-CoV-2, em ambientes fechados.

O grande diferencial do projeto proposto e pensando no conforto e segurança dos usuários, a ativação por comando de voz possibilita que o dentista, por exemplo, que for ativar o início da verificação da qualidade do ar do ambiente não precise parar sua atividade em desenvolvimento e tocar no dispositivo.

Além disso, há um compromisso custo $versus$ benefício. O protótipo completo é estimado no valor de $R\$490,00$, sendo que o microcomputador é o que adiciona maior custo. Com uma maior liberdade de custos, podem ser adicionadas mais funcionalidades que interessem aos usuários como verificação da quantidade de pessoas no ambiente por meio de sensores de presenças, automação da ventilação do ambiente ou adicionar módulos ao protótipo para que seja verificada a qualidade do ar em diferentes ambientes por meio da recepção dos dados através de comunicação Wi-Fi pelo microcomputador, operando como um servidor.

VII. Revisão Bibliográfica

É de extrema importância conhecer os produtos já existentes no mercado para avaliar os pontos fortes e fracos, bem como as funcionalidade existentes. Consequentemente, essa análise pode ser utilizada como referencia para criar estrategias para o desenvolvimento do próprio produto.

Assim, após uma pesquisa de mercado encontrou-se alguns detectores de gás carbônico para ambientes internos que custam na faixa de R$ 800,00, os quais apresentavam na tela os dados de temperatura e umidade, concentração de CO_2, data e hora em tempo real, gráfico de tendência, configuração de alarme, registro de dados de medição de intervalo de tempo, possui bateria de lítio recarregável por meio do USB externo.

O periódico *Sustainability* publicou um artigo em 2020 de um sistema de monitoramento multi-paramétrico e simultâneo da qualidade do ar. Sendo assim, esse sistema monitora a maioria dos gases poluentes, o que é crítico e essencial, além de que os desafios enfrentados por equipamentos convencionais existentes na medição de múltiplas concentrações de poluentes em tempo real incluem alto custo, capacidade de implantação limitada e detectabilidade de apenas alguns poluentes [10]. Ele apresenta, então, um sistema módulo sensor abrangente de monitoramento da qualidade do ar interno usando um *Raspberry Pi* de baixo custo. O sistema personalizado mede 10 ambientes internos, condições de temperatura, umidade relativa, material particulado , incluindo poluentes: NO_2, SO_2, CO_2, O_3 e compostos orgânicos voláteis.

Fig. 3. Detector de Gás Carbônico Portátil

Ainda, a *startup* Omni-eletrônica anunciou em 2020 um sistema de monitoramento da presença do coronavírus em ambientes internos, desenvolvido a partir da parceria com o Hospital das Clínicas (HC) da Faculdade de Medicina da Universidade de São Paulo e apoiado pelo Programa FAPESP. A empresa oferece o SPIRI, serviço de monitoramento da qualidade do ar, por meio de uma assinatura. Um aparelho instalado no local com vários sensores integrados, os quais enviam as informações para a central, que gera laudos on-line em tempo real. Além do mais, os técnicos podem instruir o cliente sobre a melhor forma de aumentar a circulação do ar quando ela não está adequada [11].

VIII. Desenvolvimento

A. Descrição de Hadware

Este projeto fará uso dos seguintes componentes para a solução de Hardware:

- Raspberry Pi 3 Model B+ com cartão micro SD. **Preço:** $R\$300,00$;
- Sensor de gases MQ-135. **Preço:** $R\$19,90$;
- Sensor de temperatura e umidade DHT11. **Preço:** $R\$12,90$;
- Conversor Analógico para Digital MCP3008. **Preço:** $R\$34,90$;
- Display Oled.**Preço:** $R\$37,90$;
- Caixa acrílica ou caixa de impressão 3D.
- Adaptador de áudio USB plugável com 3,5 mm. **Preço:** $R\$20,00$;
- Dispositivo de entrada: Microfone. **Preço:** $R\$14,50$;
- Dispositivo de saída: Caixa de som com conector USB ou P2 3.5 mm. **Preço:** $R\$25,90$;

*1) **Conexão Módulo de Aquisição - Raspberry**:* O sensor de temperatura e umidade, DHT11, é um sensor digital, e consequentemente os seus dados foram adquiridos a partir da conexão do sensor na *Raspberry* conforme representado na figura 4.

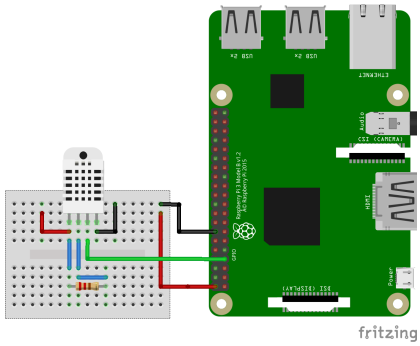

Fig. 4. Conexão sensor DHT11 com Rasberry Pi 3b+

Já o sensor de gás MQ-135, é um sensor analógico e precisa de calibração. Como a *Raspberry* não tem pinos de entrada analógica, faz-se necessário o uso de um conversor AD (MCP3008), conforme a figura 5.

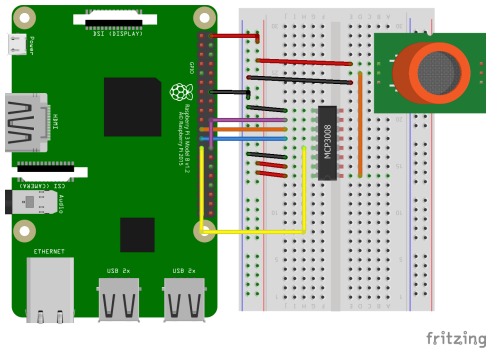

Fig. 5. Conexão sensor MQ135 com Rasberry Pi 3b+

Para a calibração do MQ-135, a figura 6 mostra a característica típica de sensibilidade do sensor para os gases de amônia, dióxido de carbono, benzeno, óxido nítrico, fumaça e álcool. Sendo assim, usa-se como parâmetro: temperatura de 20°, umidade de 65% e uma resistência de $20K\Omega$ [12].

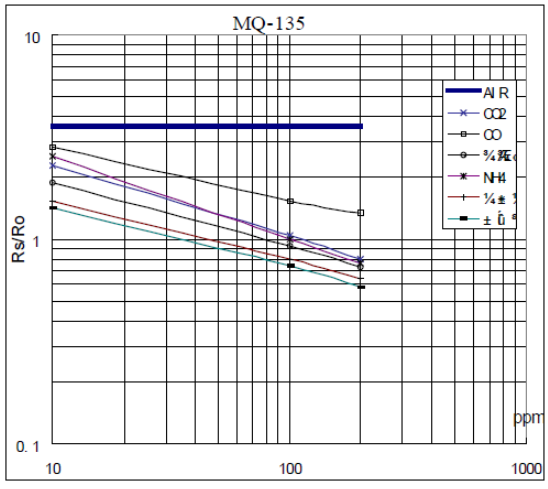

Fig. 6. Características de sensibilidade do sensor de gases MQ-135

Fundamentando no gráfico da figura 6, para realizar a calibração e da leitura do sensor é necessário marcar dois pontos da linha de CO_2, aplicar o log e posteriormente realizar o cálculo do coeficiente angular da reta [13], conforme a equação 1, com os pontos $x_1 = 100$ e $x_2 = 200$:

$$\frac{Log_{10}(0.8) - log_{10}(1.01)}{log_{10}(200) - log_{10}(100)} = -0.336 \quad (1)$$

Além do mais, como no gráfico de referência usa-se um resistor de $20K\Omega$ e no *datsheet* é também é sugerido, esse valor de resistência foi utilizado no potenciômetro do módulo do sensor.

2) Conexão Tela OLED - Raspberry Pi: Para realizar a conexão da *Raspberry* com o *Display* OLED, é necessário conectar os pinos de VCC em 3.3V (pino 3), o GND (pino 14), SDA (pino 3) e SCL (pino 5), conforme ilustrado na figura 7.

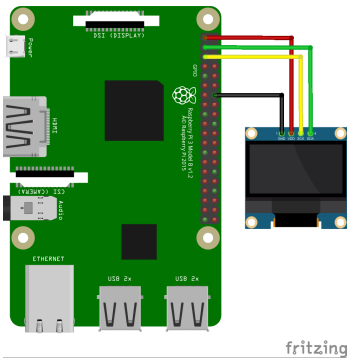

Fig. 7. Conexão *display* OLED com Rasberry Pi 3b+

B. Descrição de Software

*1) **Comunicação sensor DHT11 com a Raspberry**:* Para a leitura dos dados do sensor DHT11, foi desenvolvido um script em Python. A leitura e interpretação do sensor foram feitas facilmente através da biblioteca $Adafruit\_DHT11$, a qual é utilizada para obter leituras dos sensores de umidade e temperatura DHT11, DHT22 e AM2302.

Dessa forma, foi setado o pino GPIO e a umidade e a temperatura foram obtidas pela biblioteca. Esse sensor é comum apresentar falhas de leitura, então, em um $while$, enquanto as variáveis umidade e temperatura forem diferentes de $None$, é impresso na tela os valores a cada 3 segundos.

*2) **Comunicação sensor MQ-135 com a Raspberry**:* Para a leitura do sensor MQ-135, em um script Python foi usada a biblioteca $SpiDev$ para comunicação serial com protocolo SPI. É definido uma $clase$ para o conversor AD que usa SPI, e uma $clase$ para as variáveis de calibração do MQ. Algumas funções são criadas para leitura, calibração, cálculo da resistência e porcentagem. Essas classes e funções são chamadas na *main*.

*3) **Comunicação da tela OLED com a Raspberry**:* Foi realizado um script em Python com a finalidade da *Raspberry Pi* conectar com a tela para moestar os dados obtidos pelo

módulo de aquisição, e para alcançar essa finalidade foi utilizado o protocolo I_2C. Assim, foi necessário utilizar a biblioteca *Adafruit_GPIO* e *Adafruit_SSD1306*.

Dessa forma, a primeira biblioteca fornece uma interface GPIO de plataforma cruzada no *Raspberry Pi*, enquanto que a segunda é biblioteca para a utilização de telas OLED de 128x64 ou 128x32 pixels baseadas no drive SSD1306 com uma *Raspberry*.

IX. Resultados

1) ***Comunicação sensor DHT11 com a Raspberry****:* A figura 8 mostra o resultado do teste da comunicação com o sensor DHT11. O sensor apresenta erros de leitura em alguns momentos como já era previsto mas no geral faz a leitura correta dos dados de temperatura e umidade.

```
pi@raspberrypi:~ $ python3 sensor_dht11.py
Temp=27.0C Humidity=52.0%
Temp=25.0C Humidity=55.0%
Temp=25.0C Humidity=55.0%
Temp=25.0C Humidity=55.0%
Temp=25.0C Humidity=55.0%
Temp=25.0C Humidity=55.0%
Sensor failure. Check wiring.
Temp=25.0C Humidity=55.0%
```

Fig. 8. Leitura do sensor DHT11

2) ***Comunicação sensor MQ-135 com a Raspberry****:* A figura 9 e 10 compara o resultado do teste de comunicação do sensor MQ-135 para diferentes concentrações de CO2. Assim, os testes foram realizados em um ambiente fechado, e tem-se a comparação dos valores obtidos no caso 1, no qual o sensor fica distante da respiração de uma pessoa, resultando em uma baixa concentração de CO_2. Enquanto que no caso 2, assim que o sensor é aproximado de uma pessoa respirando, é possível visualizar o aumento na concentração de CO_2.

```
pi@raspberrypi:~ $ python sensor_mq135.py
Calibrating...
Calibration is done...

Ro=0.993015 kohm
CO2: 0.00721458 ppm
```

Fig. 9. Leitura do sensor MQ-135 com baixa concentração de CO2

```
pi@raspberrypi:~ $ python sensor_mq135.py
Calibrating...
Calibration is done...

Ro=0.993015 kohm
CO2: 3.1644 ppm
```

Fig. 10. Leitura do sensor MQ-135 com maior concentração de CO2

Ainda é preciso ajustar sua calibração para que os resultados de concentração de CO2 sejam mais verossímeis.

3) ***Comunicação Tela OLED com a Raspberry****:* A figura 11 mostra o resultado do teste da comunicação da *Raspberry Pi* com o *display* OLED, o qual apresentou no *display* a *string* "Teste SOE" e o IP da *Raspberry Pi*.

Fig. 11. Teste com *display* OLED com Rasberry Pi 3b+

X. Considerações Finais

Nota-se que o embasamento teórico do projeto está consolidado e mostra-se como um ponto de partida para o desenvolvimento do projeto e os demais pontos de controle. Portanto, para os seguintes marcos da disciplina os códigos serão adaptados e alterados para a linguagem em C, será realizada a calibração do módulo MQ-135 e acrescentará o comando por voz.

Referências

[1] Organização Pan-Americana da saúde, "Folha informativa - covid-19 (doença causada pelo novo coronavírus)," 2020, [Online; accessed 22-julho-2020]. [Online]. Available: https://www.paho.org/bra

[2] V. Aquino and N. Monteiro, "Brasil confirma o primeiro caso da doença," 2020, [Online; accessed 23-julho-2020]. [Online]. Available: saude.gov.br/noticias/agencia-saude/46435-brasil-confirma-primeiro-caso-de-novo-coronavirus

[3] Ministério da Saúde, "Sobre a doença," 2020, [Online; accessed 23-fevereiro-2021]. [Online]. Available: https://coronavirus.saude.gov.br/

[4] ANSES, "Carbon dioxide (co2) in indoor air," 2016, [Online; accessed 25-fevereiro-2021]. [Online]. Available: https://www.anses.fr/en/content/carbon-dioxide-co2-indoor-air

[5] Z. Peng and J. L. Jimenez, "Exhaled co2 as covid-19 infection risk proxy for different indoor environments and activities," *medRxiv*, 2020.

[6] R. K. Bhagat, M. D. Wykes, S. B. Dalziel, and P. Linden, "Effects of ventilation on the indoor spread of covid-19," *Journal of Fluid Mechanics*, vol. 903, 2020.

[7] S. N. R. De Araújo, S. A. R. Farias, D. S. Cruz, and R. Farias, "Concentração de dióxido de carbono em salas de aula da ufcg, climatizadas artificialmente," 2018.

[8] C. Wang, L. Miao, Z. Wang, Y. Xiong, Y. Jiao, and H. Liu, "Emergency management in dental clinic during the coronavirus disease 2019 (covid-19) epidemic in beijing," *International dental journal*, 2021.

[9] S. N. Isha, A. Ahmad, R. Kabir, and E. H. Apu, "Dental clinic architecture prevents covid-19-like infectious diseases," *HERD: Health Environments Research & Design Journal*, vol. 13, no. 4, pp. 240–241, 2020.

[10] V. G. He Zhang, Ravi Srinivasan, "Low cost, multi-pollutant sensing system using raspberry pi for indoor air quality monitoring," *Sustainability*, 2020.

[11] FAPESP, "Sistema monitora presença do novo coronavírus no ar," 2020, [Online; accessed 25-fevereiro-2021]. [Online]. Available: https://pesquisaparainovacao.fapesp.br/1526/boletim

[12] *Technical Data MQ-135 Gas Sensor*.

[13] R. P. Tutorials, "Configure and read out the raspberry pi gas sensor (mq-x)," 2017, [Online; accessed 23-março-2021]. [Online]. Available: https://tutorials-raspberrypi.com/configure-and-read-out-the-raspberry-pi-gas-sensor-mq-x/

APÊNDICE

a) Comunicação sensor DHT11:

```
import Adafruit_DHT
import time

DHT_SENSOR = Adafruit_DHT.DHT11
DHT_PIN = 4

while True:
    humidity, temperature =
    ↪ Adafruit_DHT.read(DHT_SENSOR,
    ↪ DHT_PIN)
    if humidity is not None and
    ↪ temperature is not None:
        print("Temp={0:0.1f}C
        ↪ Humidity={1:0.1f}
    else:
        print("Sensor failure. Check
        ↪ wiring.")
    time.sleep(3)
```

b) Comunicação sensor MQ-135:

```
from spidev import SpiDev
import time
import math
import sys

class MCP3008:
    def __init__(self, bus=0,
    ↪ device=0):
        self.bus, self.device = bus,
        ↪ device
        self.spi = SpiDev()
        self.open()
        self.spi.max_speed_hz =
        ↪ 1000000  # 1MHz

    def open(self):
        self.spi.open(self.bus,
        ↪ self.device)
        self.spi.max_speed_hz =
        ↪ 1000000  # 1MHz

    def read(self, channel=0):
        cmd1 = 4 | 2 | ((channel & 4)
        ↪ >> 2)
        cmd2 = (channel & 3) << 6
        adc = self.spi.xfer2([cmd1,
        ↪ cmd2, 0])
        data = ((adc[1] & 15) << 8) +
        ↪ adc[2]
        return data

    def close(self):
        self.spi.close()

class MQ():
    MQ_PIN                          = 0
    RL_VALUE                        = 20
    RO_CLEAN_AIR_FACTOR             = 3.8
    CALIBARAION_SAMPLE_TIMES        = 50
    CALIBRATION_SAMPLE_INTERVAL     = 500
    READ_SAMPLE_INTERVAL            = 50
    READ_SAMPLE_TIMES               = 5
    GAS_CO2                         = 0

    def __init__(self, Ro=50,
    ↪ analogPin=0):
        self.Ro = Ro
        self.MQ_PIN = analogPin
        self.adc = MCP3008()

        self.CO2Curve =
        ↪ [2,0.004,-0.34]

        print("Calibrating...")
        self.Ro =
        ↪ self.MQCalibrationn(self.
        ↪ MQ_PIN)
        print("Calibration is
        ↪ done...\n")
        print("Ro=\%f kohm" \%
        ↪ self.Ro)


    def MQPercentage(self):
        val = {}
        read =
        ↪ self.MQRead(self.MQ_PIN)
        val["GAS_CO2"]  =
        ↪ self.MQGetGasPercentage(read
        ↪ /self.Ro, self.GAS_CO2)
        return val

    def MQResistanceCalculation(self,
    ↪ raw_adc):
        return float
        ↪ (self.RL_VALUE*(1023.0-
        ↪ raw_adc)/float(raw_adc))


    def MQCalibration(self, mq_pin):
        val = 0.0
        for i in range(self.
        ↪ CALIBARAION_SAMPLE_TIMES):
            val += self.
            ↪ MQResistanceCalculation
            ↪ (self.adc.read(mq_pin))
```

```
            time.sleep(self.
             ↪  CALIBRATION_SAMPLE_
             ↪  INTERVAL/1000.0)

        val = val/self.
         ↪  CALIBARAION_SAMPLE_TIMES
        val =
         ↪  val/self.RO_CLEAN_AIR_FACTOR
        return val

    def MQRead(self, mq_pin):
        rs = 0.0
        for i in range
         ↪  (self.READ_SAMPLE_TIMES):
            rs += self.
             ↪  MQResistanceCalculation
             ↪  (self.adc.read(mq_pin))
            time.sleep(self.
             ↪  READ_SAMPLE_INTERVAL
             ↪  /1000.0)

        rs = rs/self.READ_SAMPLE_TIMES
        return rs

    def MQGetGasPercentage(self,
     ↪  rs_ro_ratio, gas_id):
        if ( gas_id == self.GAS_CO2 ):
            return
             ↪  self.MQGetPercentage
             ↪  (rs_ro_ratio,
             ↪  self.CO2Curve)
        return 0

    def MQGetPercentage(self,
     ↪  rs_ro_ratio, pcurve):
        return (math.pow(10,(
         ↪  ((math.log(rs_ro_ratio)-
         ↪  pcurve[1])/ pcurve[2]) +
         ↪  pcurve[0])))

def main():
    mq = MQ()
    while True:
        perc = mq.MQPercentage()
        sys.stdout.write("\r")
        sys.stdout.write("\033[K")
        sys.stdout.write("CO2: \%g
         ↪  ppm" \% (perc["GAS_CO2"]))
        sys.stdout.flush()
        time.sleep(0.1)


if __name__ == '__main__':
    main()
```

c) Comunicação Tela OLED:

```
import time
import Adafruit_GPIO.SPI as SPI
import Adafruit_SSD1306
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import subprocess

RST = None
DC = 23
SPI_PORT = 0
SPI_DEVICE = 0

disp = Adafruit_SSD1306.SSD1306_128_32
 ↪  (rst=RST)
disp.begin()
disp.clear()
disp.display()
width = disp.width
height = disp.height
image = Image.new('1', (width,
 ↪  height))
draw = ImageDraw.Draw(image)
draw.rectangle((0,0,width,height),
 ↪  outline=0, fill=0)
padding = -2
top = padding
bottom = height-padding
x = 0
font = ImageFont.load_default()

while True:
    draw.rectangle((0,0,width,height),
     ↪  outline=0, fill=0)
    cmd = "hostname -I | cut -d\' \'
     ↪  -f1"
    IP = subprocess.check_output(cmd,
     ↪  shell = True )
    draw.text((x, top),       "Teste
     ↪  SOE \n IP: " + str(IP),
     ↪  font=font, fill=255)

    # Display image.
    disp.image(image)
    disp.display()
    time.sleep(.1)
```