Journal
Bachelor Thesis

Fabiola Marín Rivera
TEC ID: 2014086018
UCID: 30108542

Introduction

The following document will be used as a Journal for the project ”Development of an e-Nose System for Cannabis Detection”. In this you can find all records of the executed work for this project and other important documents.

Date: September 02, 2019

I worked on the composition of the final report, I put in order all archives and documents that will be used, this can be seen at the following link: `https://drive.google.com/open?id=1vSPXzggPYw4jijCzH1Wnrc0dCRjeB_-r`

Date: September 03, 2019

Reading of the first chapter of the thesis ”Develpment of a Highly Selective Single Sensor Microfluidic-based Gas Detector”.

Chapter 1: Background [1]

Introduction

The state of the art technology available in the industry today for real time gas detection is the electronic nose or e-nose, which functions based on sensor arrays coupled with pattern recognition systems.

Odor and sense of smell

Odor is a sensation generated when odorant 8gas) molecules interact with receptors in the olfactory neurons in the (human) nose which can sense more than 10000 different odors. The nose is the only external part of our olfaction system, the odorant molecules enter through the nasal cavity and reach the olfactory receptors which are placed between our eyes. The receptors neurons are connected to the olfactory bulb which is part of our brain and the olfactory bulb is connected to the olfactory cortex which is in charge of signal analysis and patter recognition on different smells. While the air goes through the nasal cavity, the temperature of the air reaches the temperature of body, in the presence of any odorant in the air the molecules of the odorant react with the olfactory receptors and that results into the sense of smell.

In essence the olfactory receptors are the main part of the smelling process since they produce the electrical pulse which is transferred through the olfactory bulb to the neurons and hence the brain for signal analysis and pattern recognition. Olfactory receptors are neuron cells which are placed in epithelium, there are more than 10 million olfactory receptor cells in epithelium which has a surface area of 2-4 cm^2 and is cover with a 10-40 μm thick layer of mucus. The gas molecules are adsorbed to this later, diffuse into, and reach the olfactory receptors, the end of each cell reaches the surface of the epithelium through tiny hairs which are called cilia.

Each olfactory receptor responds only to certain gas molecules, therefore, by exposing the olfactory system to a chemical compound, some of the olfactory receptors get activated and the rest are deactivated. In essence, if there are "n" receptors and each has two modes (active and non-active) 2^n scenarios migth occur. The results in encoding different smells in the olfactory system.

Machine Olfaction

Gas chromatography (GC) and mass spectrometry (MS) are the most commonly used "macro" systems developed for analysis of Volatile Organic Compounds, VOCs. Although GC and MS have high accuracy and reliability, these methods are expensive and time consuming and require bulky devices and highly skilled personnel to run them. Also the sample extraction and experimental process involved using these methods are sophisticated and time consuming. The miniaturization of the system such as gas chromatography and mass spectrometry in a low-cost fashion has not been realized yet. These methods are not called e-nose as their operation principle is not similar to the biological model.

- Gas Chromatography: A GC is an analytical instrument that measures the content of a mixture of different gas components in a sample, the sample injected into the instrument is transported by a stream of gas (carrier gas) into a separation channel known as the column, different components are then separated along the column. The detector placed at the end of the column measures the quantity of the components that exit the column.
- Mass Spectrometry: The sensitivity of this method is high, and it is usually used in series which a GC to identify different compounds of components in a gas mixture.
- Electronic nose (e-nose): Operate based on sensor arrays coupled with patter recognition systems, as for the sensory part, an array of several sensors are used. The goal of the first e-nose was to develop portable device for detection of flammable gases.

The name of the e-nose is chosen because of its similarity to the human nose. Five stages of smell detection sing human olfaction system are:

1) Sniffing the smell during which the odorant molecules are adsorbed to receptor cells and stimulate them;

2) Stimulation of the olfactory receptors and producing an electrical pulse;
3) Transferring the electrical signal to mitral cells through glomerulus, which is part of the olfactory bulb;
4) Sending the neuron signals to the brain through the mitral cells;
5) Performing signal analysis and pattern recognition in the brain, resulting in smell detection.

The five components of smell detection using an e-nose are:

1) Sample extraction and delivering the gas molecules to the chamber where the the sensor array is located;
2) Reaction of the gas molecules with the sensor sensing pallet, resulting in electrical signals;
3) Amplification and recording of the electrical signals using interface circuit;
4) Converting the recorded signals to digital data that is fed to a computer;
5) Analyzing the data, extracting the features from the signals, and recognizing the pattern of each signal (using different pattern recognition techniques), resulting in smell detection.

The analyte is extracted in the chamber where the conditions of the sample (temperature and humidity) are controlled. If the components of the sensor array are correctly chosen the responses of the different sensor to the analyte are different which results in a general finger print for the examined sample.

Gas sensors are devices which perform based on changing one or few of their physical characteristics (such as mass, conductivity or capacitance) when they are exposed to a gas. Converting these changes to electrical signals results in producing the response of the sensor to different gases. Gas sensors are evaluated based in their performance indicators which are as follows:

- **Sensitivity:** Shows how precise the sensor can detect the target gas.
- **Detection limit:** The minimum volume concentration of the target which can be detected by gas sensor, showing how sensitive is the sensor.
- **Selectivity:** The ability of the gas sensor to distinguish between different components of a mixture and detect a single specific gas.
- **Response time:** The time which is required for the sensor to create a signal from reaction to a specific concentration of the target gas.
- **Recovery time:** The time which takes for the sensor response to return to its baseline.

- **Power consumption:** The power dissipated by the sensor heater and sensing pallet.
- **Dynamic range:** The difference between the maximum concentration of the gas that could be detected and minimum detection limit.
- **Life cycle:** The period of time which a sensor can operate without stopping.
- **Drift:** The gradual change in the sensing capability of a gas sensor over time.

The most frequently used type of gas sensors is metal oxide semiconductor (MOS), in the basic configuration of MOS sensors, figure 1, a chemo-resistor is made by deposition of a thick film metal oxide pallet ant thick film thermo-resistor micro-heater on opposite surfaces of a millimeter-scale ceramic substrate.

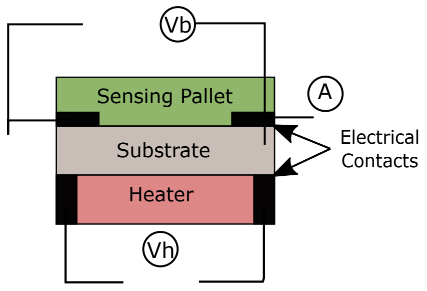


Fig. 1: MOS Gas Sensor Schematic

The behaviour of a MOS sensor in DC bias can be modeled as a variable resistance R_s, figure 2, the value of this resistance depends on the type of gas molecule, the gas concentration and temperature of the sensing pallet. The resistance of the sensor resistance of the sensor in the clean air R_{air} (baseline) The sensitivity (S) of such a sensor is defined by equation 1

$$S = \frac{R_{air}}{R_{gas}} \tag{1}$$

in which R_{air} and R_{gas} are the resistances of the sensing pallet measured in the clean and contaminated air, respectively.

The selectivity between two gases (i, j) is defined by equation 2. Current gas sensors are either made to be evenly sensitive to different gases or fabricated for detecting a specific target. Hence differentiating among different gases or a

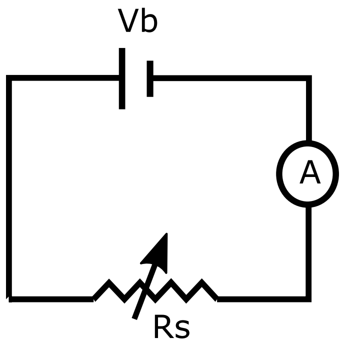


Fig. 2: MOS Gas Sensor Electrical Circuit in DC Bias

mixture of gases using single sensor is very challenging, if not impossible as the transient responses of the sensor to two different gases are almost the same.

$$Sel(i,j) = \frac{S_i}{S_j} \tag{2}$$

Different methods are used for data classification, and hence gas identification, methods such as k-nearest neighbor (KNN), neural network (NN) and support vector machine (SVM).

KNN is one of the most commonly used methods for sample identification, in this one the distances of each feature vector to different clusters are measured and based on the measured distances the appropriate cluster is selected (the minimun distance). This method is simple but computationally expensive, and it needs mathematical analytical tools as the distance of the feature vector from all other points in the feature map is required to be measured and compared.

The past recap of the gas sensor thesis will be very useful for the development of the background in the final report.

Date: September 11, 2019

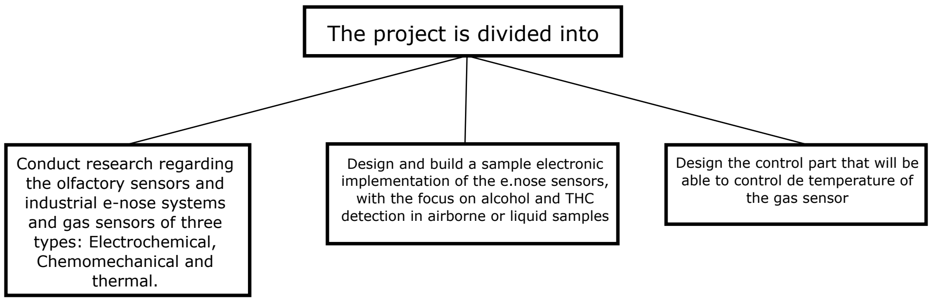

The project is divided into
Conduct research regarding the olfactory sensors and industrial e-nose systems and gas sensors of three types: Electrochemical, Chemomechanical and thermal.
Design and build a sample electronic implementation of the e.nose sensors, with the focus on alcohol and THC detection in airborne or liquid samples
Design the control part that will be able to control de temperature of the gas sensor

Fig. 3: Project division

Date: September 11, 2019

In [2] mention three main types of metal oxide-based gas sensors that can be distinguished as:

- Electrochemical: Potential or resistance changes through charge transfer.
- Chemomechanical: mass change due to the adsorption.
- Thermal: Temperature changes through chemical interaction.

Some important links.

- `https://media.digikey.com/pdf/Data%20Sheets/Sensirion%20PDFs/SEK-SGPxx_Eval_Kit_Overview.PDF`
- `https://www.mouser.co.cr/ProductDetail/Sensirion/SEK-SGPC3?qs=YCa%2FAAYMW01bmSevPnV7WA==`
- `https://www.sensirion.com/en/environmental-sensors/evaluation-kit-sek-environmental-se`
- `https://www.sensirion.com/controlcenter/`
- `https://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/15_Environmental_Sensing/Sensirion_Environmental-Sensing_SEK_Overview.pdf`
- `https://www.sensirion.com/?id=1053`

Date: December 3, 2019

The first step of the project will be the detection of ethanol, first a yes no question and when that is ready I will try to detect the concentration, this can be done by using a certain amount of thresholds, in which each threshold will indicate an specific concentration.

Dr. Svetlana gave me the gas sensor that I will be using but the Phd. student that was suppose to help me with the cannabis identification part hasn't arrive yet. This is the main reason of why I will be focusing on ethanol for now.

`https://www.mdpi.com/1424-8220/18/4/1052` REVISAR

Date: December 5, 2019

Basic chemestry review for the chemical background of the project.

Date: December 6, 2019

Table 1 shows the inventory of the sensirion evaluation kit for the SGPCx that is going to be used for the develpment of the project, figures 4 and 5 depicts all the kit components.

Quantity	Name
1	SEK sensirion bridge
1	USB to micro USB cable
1	SGP30 - 2.5K
1	SHT31 - DIS - BIE 1-101199-01
1	STS31 - DIS 3.000.049
1	SGPC3 1-101679-01
1	SHTC3 3.000.048
1	SHTW2 1-101-483-01
2	Sensirion bridge to sensor cables

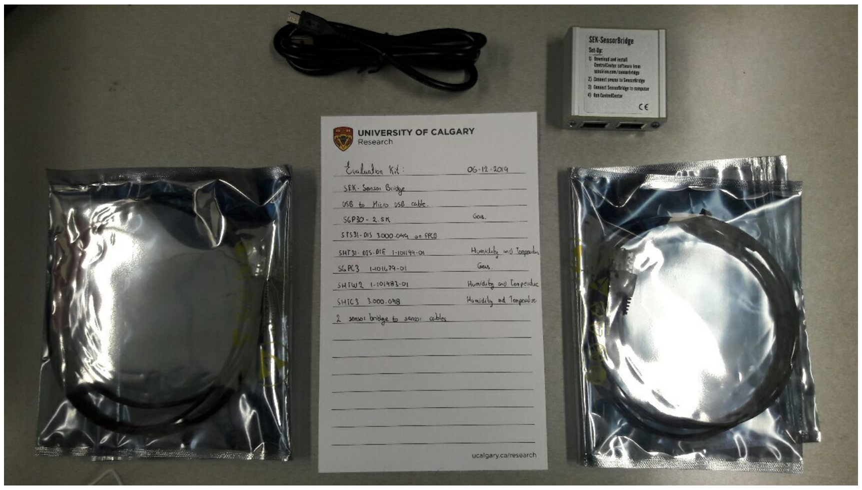
Tab. 1: Evaluation kit for SGPCx Inventory

In order to use the evaluation kit is necessary to download the control center software, the main goal of this is to help with the sensor evaluation in a plug and play way.

Control Center software: `https://www.sensirion.com/en/controlcenter/`

Control Center Documentation `https://www.sensirion.com/de/controlcenter/documentation-control-center/`

Fig. 4: Evaluation kit for SGPCx Inventory 1

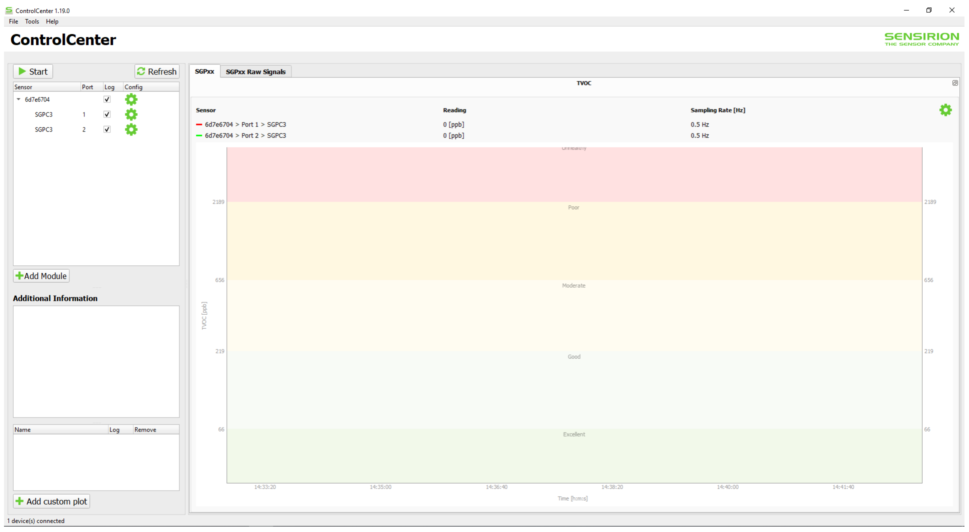


UNIVERSITY OF CALGARY
Research

Evaluation Kit: 06-12-2019
SEK-Sensor Bridge
USB to Micro USB cable.
SGP30 - 2.5K Gas.
STS31-DIS 3.000.049 on FPCB
SHT31-DIS-BIE 1-101199-01 Humidity and Temperature
SGPC3 1-101679-01 Gas.
SHTW2 1-101483-01 Humidity and Temperature
SHTC3 3.000.048 Humidity and Temperature
2 sensor bridge to sensor cables

ucalgary.ca/research

Fig. 5: Evaluation kit for SGPCx Inventory 2



Sensor	Port	Log	Config
6d7e6704		✓	
SGPC3	1	✓	
SGPC3	2	✓	

Sensor	Reading	Sampling Rate [Hz]
🟥 6d7e6704 > Port 1 > SGPC3	0 [ppb]	0.5 Hz
🟩 6d7e6704 > Port 2 > SGPC3	0 [ppb]	0.5 Hz

Name	Log	Remove

Fig. 6: Control center interface

The SGPC3 is a gas sensors from the multi-pixel gas sensor platform SGP of Sensirion, this is not the one that is going to be used but it was tested with the control center, this because it already came with the directed connection to be plug to the sensor bridge, which can be connected directly to the computer. Figure 6 shows the Control Center interface when the two SGPC3 sensors are connected 7. 8 and 9 shows the specifications for the sensor bridge and the gas sensors.

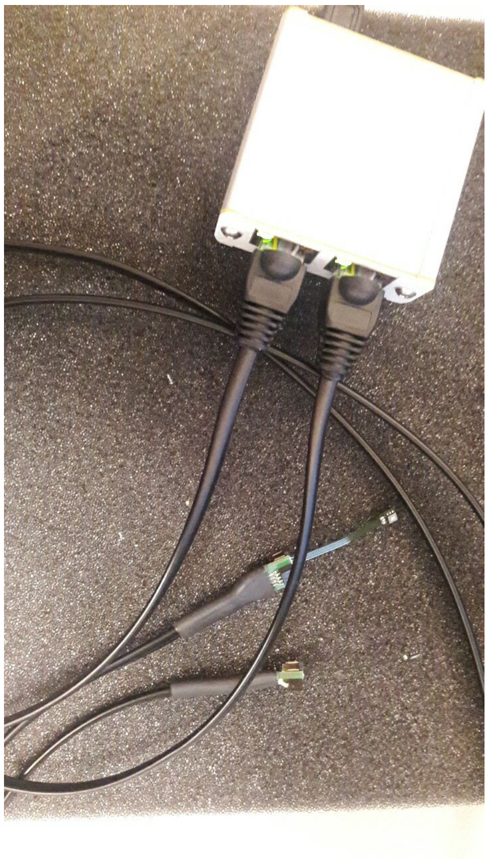
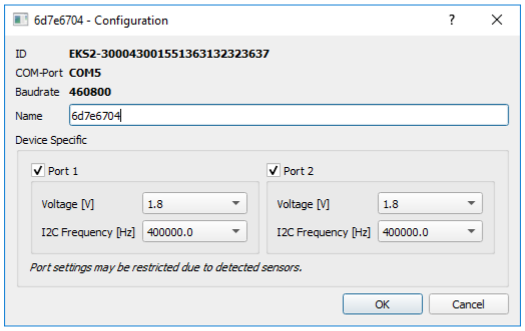


Fig. 7: Connected sensors



6d7e6704 - Configuration

ID EKS2-300043001551363132323637
COM-Port COM5
Baudrate 460800
Name 6d7e6704

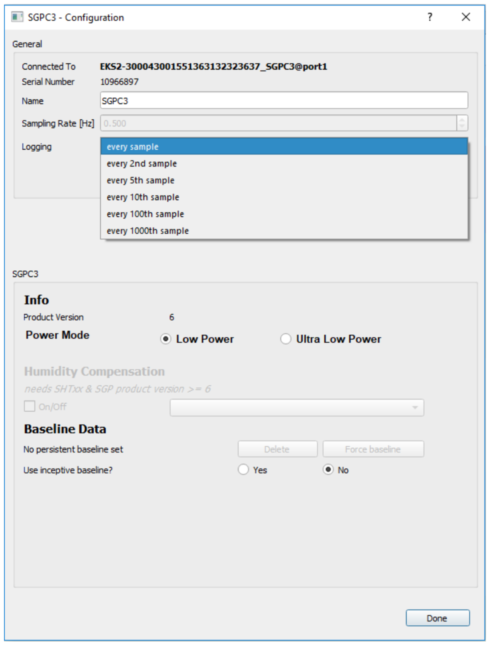
Device Specific

✓ Port 1
Voltage [V] 1.8
I2C Frequency [Hz] 400000.0

✓ Port 2
Voltage [V] 1.8
I2C Frequency [Hz] 400000.0

Port settings may be restricted due to detected sensors.

Fig. 8: Sensor bridge software specifications



SGPC3 - Configuration

General

Connected To	**EKS2-30004300155136313232637_SGPC3@port1**
Serial Number	10966897
Name	SGPC3
Sampling Rate [Hz]	0.500
Logging	every sample

SGPC3

Info

Product Version 6

Power Mode [✓ Low Power] Ultra Low Power

Humidity Compensation

needs SHTxx & SGP product version >= 6

On/Off

Baseline Data

No persistent baseline set

Use inceptive baseline? Yes [✓ No]

Fig. 9: SGP30 software specifications

Date: December 11, 2019

The SGP platform is a good option because the long-term stability in the presence of siloxanes, a bad stability will deteriorate their accuracy over time. The evaluation kit counts with an SGP30 gas sensor that can be used for the detection of a desire compound. Nevertheless, this sensor does not come ready to be used, which gives me two options.

1) Design the PCB.

2) Buy the sensor that is ready to be connected.

This has to be discuss with the lab director on Monday 16 because she is on a business trip.

Date: December 12, 2019

I studied the SGP30 datasheet, [3].

The measurement range for Ethanol signal goes from 0ppm - 1000ppm an the specified range from 0.3ppm - 30ppm, this specified measurement range covers the gas concentrations expected in indoor air quality applications. The accuracy of the concentration c can be determined by 3. For Ethanol has a typical accuracy tolerance of 15% of the measure value.

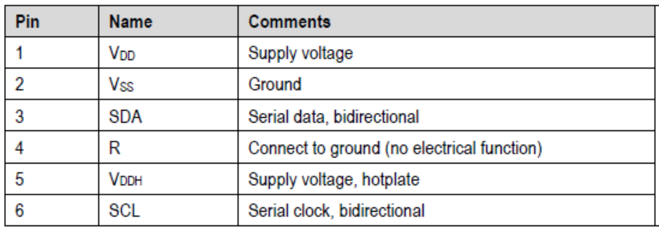
$$ln = (\frac{c}{c_{ref}}) = \frac{s_{ref} - s_{out}}{a} \tag{3}$$

- $a = 512$.
- s_{out}: Ethanol signal output.
- s_{ref}: Ethanol output at 0.5ppm H_2.
- $c_{ref} = 0.4$ppm

The long term drift, change of accuracy per year of operation, has a typical value of 1.3% of measure value, the resolution, a 2% and the sampling frequency has a max value of 40Hz.

The sensor shows best performance when operated within recommended normal temperature and humidity range of $5 - 55C^{\circ}$ **and** $4 - 20g/m^3$

The SGP30 comes in a 6 pin DFN package, figure 10.



Pin	Name	Comments
1	V_{DD}	Supply voltage
2	V_{SS}	Ground
3	SDA	Serial data, bidirectional
4	R	Connect to ground (no electrical function)
5	V_{DDH}	Supply voltage, hotplate
6	SCL	Serial clock, bidirectional

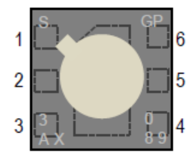


Fig. 10: Interface specifications SGP30

The typical application circuit is shown in 11, The power supply pins must be decoupled with a 100nF capacitor that shall be placed as close as possible to pin VDD, VDD and VDDH pins should be shorted. SCL is used to synchronize the communication between the microcontroller and the sensor. SDA is used to transfer the data to and from the sensor. The timing specifications defined in the I^2C manual must be met for safe communication. To avoid signal contention, the microcontroller must only drive SDA and SCL low and the external pull-up resistors are required to pull the signal high.

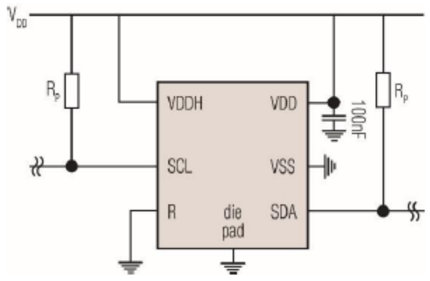


Fig. 11: Typical application circuit for an SGP30 sensor

Date: December 13, 2019

El SGP30 supports I^2C fast mode, all the commands and data are mapped to a 16-bit address space and all the data and commands are protected with a CRC checksum. The 16-bit commands that are sent to the sensor already include a 3-bit CRC checksum. Data sent from and received by the sensor is always succeeded by and 8-bit CRC.

- **Checksum** Method of ensuring that data is correct, checksum also provide a form of authentication because an invalid checksum suggests that the data has been compromised in some fashion. A checksum is determined in one of two ways. Let's say the checksum of a packet is 1 byte long. A byte is made up of 8 bits, and each bit can be in one of two states, leading to a total of 256 (28) possible combinations. Since the first combination equals zero, a byte can have a maximum value of 255. [4]

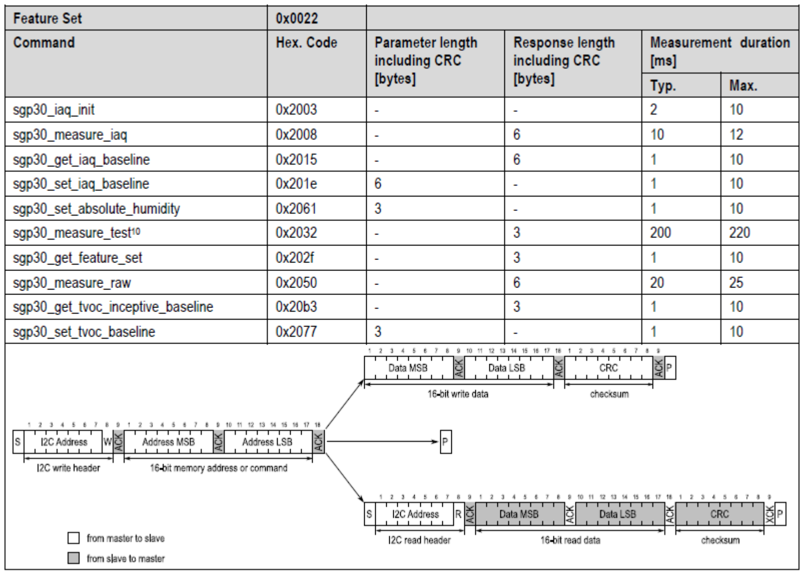
  - If the sum of the other bytes in the packet is 255 or less, then the checksum contains that exact value. [4]
  - If the sum of the other bytes is more than 255, then the checksum is the remainder of the total value after it has been divided by 256. [4]

- **Cyclic Redundancy Check (CRC):** CRCs are similar in concept to checksums, but they use polynomial division to determine the value of the CRC, which is usually 16 or 32 bits in length. The good thing about CRC is that it is very accurate. If a single bit is incorrect, the CRC value will not match up. Both checksum and CRC are good for preventing random errors in transmission but provide little protection from an intentional attack on your data. Symmetric- and public-key encryption techniques are much more secure. [4].

The sensor starts powering up after reaching the power up voltage (V_{POR}), after reaching this threshold voltage the sensor needs the time up (t_{PU}) to enter the idle state. Once the idle state is entered it is ready to receive commands from the master. Each transmission sequence begins with the START condition and ends with a STOP one.

Date: December 16, 2019

A measurement communication sequence consists of a START condition, the I^2C write header (7-bit I^2C device address plus 0 as the write bit) and a 16-bit measurement command. The sensor pulls the SDA pin low (ACK bit) after falling edge of the 8th SCL clock to indicate the reception, with this ACK of the measurement command the sensor starts measuring. Measurement commands can be seen on figure 12.



Feature Set	0x0022				
Command	Hex. Code	Parameter length including CRC [bytes]	Response length including CRC [bytes]	Measurement duration [ms]	
				Typ.	Max.
sgp30_iaq_init	0x2003	-	-	2	10
sgp30_measure_iaq	0x2008	-	6	10	12
sgp30_get_iaq_baseline	0x2015	-	6	1	10
sgp30_set_iaq_baseline	0x201e	6	-	1	10
sgp30_set_absolute_humidity	0x2061	3	-	1	10
sgp30_measure_test[10]	0x2032	-	3	200	220
sgp30_get_feature_set	0x202f	-	3	1	10
sgp30_measure_raw	0x2050	-	6	20	25
sgp30_get_tvoc_inceptive_baseline	0x20b3	-	3	1	10
sgp30_set_tvoc_baseline	0x2077	3	-	1	10

Fig. 12: Measurement commands

The SGP30 uses a dynamic baseline compensation algorithm and on-chip parameters to provide two complementary air quality signals. Based on the sensor signals a TVOC and a CO_2eq are calculated. The command *sgp30_iaq_init* starts the air quality measurement, after this one is necessary to sent a *sgp30_measure_iaq* command in regular intervals of 1s to ensure proper operation of the dynamic baseline compensation algorithm.

The command *sgp30_get_iaq_baseline* returns the baseline values for the two air quality signals. The sensor responds with 2 data bytes (SMB first) and 1 CRC byte for each of the two values in order CO_2eq and TVOC. *sgp30_measure_raw* is intended for part verification and testing purposes. It returns the sensor raw signals which are used as inputs for the on-chip calibration and baseline compensation algorithms. Figure 13 depicts the command sequence for starting and repeating measurements.

For more information refer to the SGP30 datasheet, version 0.92, April 2019, [3].

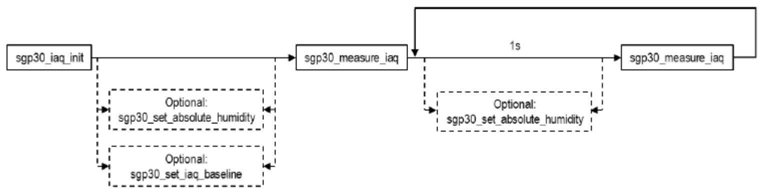


Fig. 13: Command sequence for starting and repeating measurements

`https://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/0_Datasheets/Gas/Sensirion_Gas_Sensors_SGP30_Datasheet.pdf`

First revision

December 17, 2020.

Date: December 19, 2019

It was decided to use the Adafruit SGP30 gas sensor. According to the datasheet provided by mouser electronics, the online distributor from which the sensor was bought, the SGP30 is easy to use with python.

Some important links:

- `https://www.instructables.com/id/Raspberry-Pi-I2C-Python/`
- `https://circuitpython.readthedocs.io/projects/sgp30/en/latest/`

Date: December 30, 2019

`https://learn.adafruit.com/circuitpython-on-raspberrypi-linux/installing-circuitpython-on-r`

`https://learn.adafruit.com/adafruit-sgp30-gas-tvoc-eco2-mox-sensor/circuitpython-wiring-test`

`https://pi4j.com/1.2/pins/model-3b-rev1.html`

`https://learn.adafruit.com/welcome-to-circuitpython/installing-circuitpython`

Date: January 02, 2020

There were some problems with the raspberry pi so I had to format and install everything again.

Date: January 03, 2020

Everything was installed in the Raspberry pi 3, including all the libraries that are going to be used and it looks like is working, nevertheless, I won't be able to do tests until Monday or Tuesday of next week because I do not have a way to solder the sensor pins.

Date: January 06, 2020

The sensor pins were solder and the sensor was tested. According to the datasheet, it needs to work approximately 12 hours to establish the real baseline values.

Date: January 07, 2020

There was an error in the code used for the calibration, instead of using the line $sgp30.get_iaq_baseline()$, the example that comes in the datasheet used $sgp30.set_iaq_baseline(0x8973, 0x8aae)$, therefore, the baseline wasn't set correctly.

The code used for the second try can be seen in fig 1, all the baseline, eCO2 and TVOC values are written in an output text because the first attempt to calibrate the sensor was failed, it will try again the night of January 07. Nevertheless, data was acquired for approximately two hours and it will be compared to the one acquired after 12 hours of continuous operation. At the moment I haven't still decided which type of model will I use for the data analysis, however, I'm investigating the posibility of using support vector machine, this because in many of the paper that I have already read, this is one of the most effective analysis techniques.

Listing 1: Data Acquiring Script 01-07-2020.

```python
import time
import board
import busio
import adafruit_sgp30


i2c = busio.I2C(board.SCL, board.SDA, frequency=100000)

# Create library object on our I2C port

sgp30 = adafruit_sgp30.Adafruit_SGP30(i2c)

Datos = open("Datos_010720.txt", "w")
Base = open("Baseline_010720.txt", "w")


print("SGP30 serial #", [hex(i) for i in sgp30.serial])

sgp30.iaq_init()
sgp30.get_iaq_baseline()

elapsed_sec = 0

while True:

        print("eCO2 = %d ppm \t TVOC = %d ppb" %
            (sgp30.eCO2, sgp30.TVOC))
        Datos.write("eCO2 = %d ppm \t TVOC = %d
                ppb" % (sgp30.eCO2, sgp30.TVOC))
        time.sleep(1)
        elapsed_sec += 1
        if elapsed_sec > 10:
            elapsed_sec = 0
            print("Baseline values: eCO2 = 0x%x,
     TVOC = 0x%x" % (sgp30.baseline_eCO2,
     sgp30.baseline_TVOC))
            Base.write("Baseline values: eCO2 = 0x%x,
                TVOC = 0x%x" % (sgp30.baseline_eCO2,
                sgp30.baseline_TVOC))
```

Date: January 08, 2020

I worked on the final report introduction and let the sensor working from 10:00am to 3:30 pm, the baseline values were:

- $eCO_2 = 0x90a3$
- $TVOC = 0x95d4$

Date: January 09, 2020

The sensor worked from 8:00pm of 08/01/20 to 9:30am of 09/01/20, the baseline values were:

- $eCO_2 = 0x90e6$
- $TVOC = 0x9637$

I acquire two set of data from the sensor for clean air, using the baseline values above. Reorganization of the final report and reading for the background and related work.

Date: January 10, 2020

With the baseline values that where obtain on December 09 I collected Carbon dioxide equivalent (eCO_2) and Total Volatile Organic Compound (TVOC) data for clean air, figure 14. For eCO_2 we can notice that the value keeps constant between 400 and 500, as for the TVOC, which is between 100 and 160.

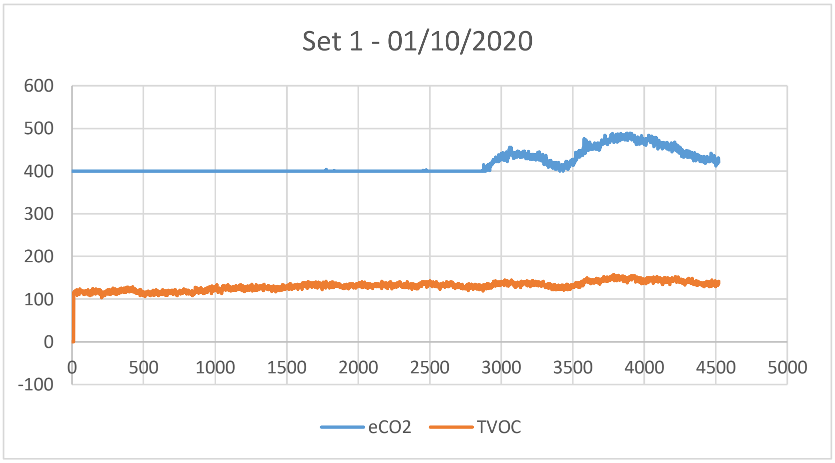


Fig. 14: eCO_2 and TVOC vs Number of samples, January 10, 2020 V1

For the second group of data, a swab with Ethanol (unknown concentration) was put close to the sensor for a few seconds, this was between samples 1500 and 2000, both eCO_2 and TVOC values got up to 60000, 15.

The idea is to collect as much data as possible and then use something like support vector machine or other processing algorithm for the Ethanol data, and eventually do the same for THC, this because there is no that much background on this subject so we need to do some research and test before we can considered the design of the controller part.

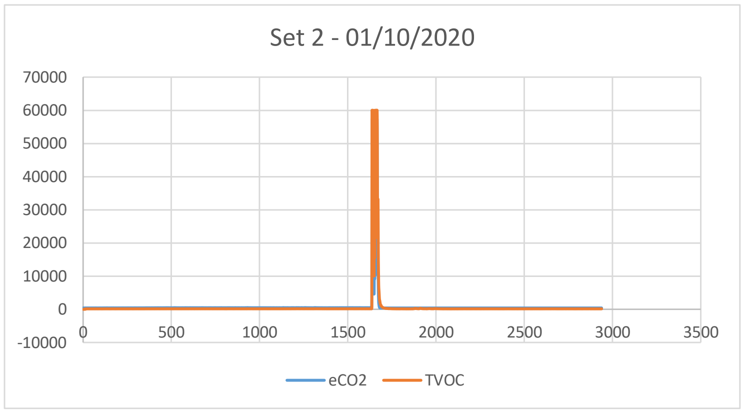


Fig. 15: eCO_2 and TVOC vs Number of samples, January 10, 2020 V2

Date: January 13, 2020

The first set of data was acquire from 11:00am to 11:30 am, eCO_2 was constant, in 400 ppb and the TVOC went from 0 ppm to 200 and then decreased again.

For the second set of data the TVOC values went from 0 to 800 in the first measurements, and then started to decreased an stabilized between 0 and 50ppb. eCO_2 stay stable in 400ppb

There was a mistake with all the data acquire after the second set, nevertheless, I notice that the samples change when air ventilation system starts working, they can go from 14 ppb to 300 ppb. Up until now, this can be considered as noise because when I test ethanol the values went to saturation level, which is 60000 ppb, I'm considering the option of building a little chamber, to limit the air circulation changes.

Date: January 14, 2020

I worked on a new proposal for the project, also in an overview document, this in other to have the most useful ideas out of the best papers related to THC detection using gas sensors.

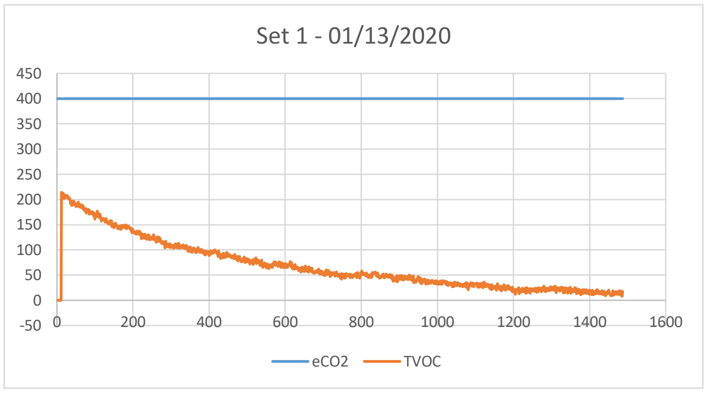


Fig. 16: eCO_2 and TVOC vs Number of samples, January 13, 2020 V1

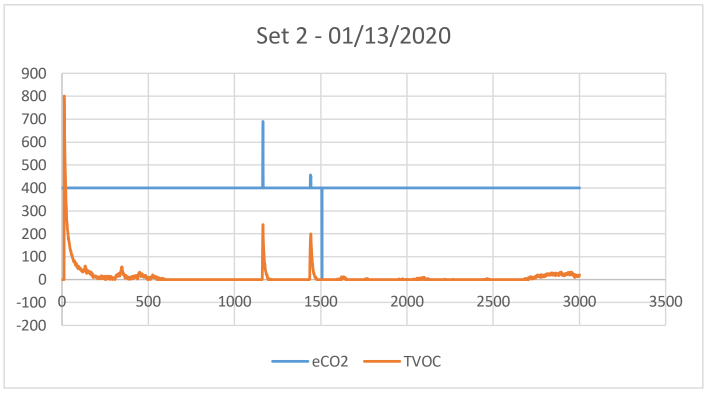


Fig. 17: eCO_2 and TVOC vs Number of samples, January 13, 2020 V2

Date: January 17, 2020

Listing 2: Data Acquiring Script 01-17-2020.

```
import time
import board
import busio
import adafruit_sgp30
import csv
import sys

i2c = busio.I2C(board.SCL, board.SDA, frequency=100000)

# Create library object on our I2C port

sgp30 = adafruit_sgp30.Adafruit_SGP30(i2c)

print("SGP30 serial #", [hex(i) for i in sgp30.serial])

sgp30.iaq_init()
sgp30.set_iaq_baseline(0x90e6, 0x9637)

elapsed_time = 0

with open(r'/home/pi/01-20-20/Datos_012020_V2.csv', 'a')\
    as b:
    Datos = csv.writer(b)
    Datos.writerow(['Number of sample','eCO2','TVOC'])

while True:

    print("Number of sample = %d \t eCO2 = %d ppm \
           \t TVOC = %d ppb"% (elapsed_time, sgp30.eCO2,\
                                sgp30.TVOC))
    with open(r'/home/pi/01-20-20/Datos_012020_V2.csv',\
              'a') as b:
        Datos = csv.writer(b)
        Datos.writerow([elapsed_time,sgp30.eCO2, \
                        sgp30.TVOC])

    elapsed_time +=1
    time.sleep(1)

    if elapsed_time == 1000:
        sys.exit()
```

Date: January 20, 2020

Right now we have isopropyl rubbing alcohol at 70% v/v concentration, in order to obtain differents concentrations we will use the equation 4.

$$Vd = \frac{Va(Ca - Cd)}{Cd} \tag{4}$$

Were

Vd = Volume of water that we need to add.

Va = Actual volume.

Ca = Actual concentration.

Cd = Desired concentration

Hence, the amount of distilled water in order to get different ethanol concentration are.

- **For 5%:** $V_{d5\%} = \frac{10ml(70\%-5\%)}{5\%} = 130ml$
- **For 10%:** $V_{d10\%} = \frac{10ml(70\%-10\%)}{10\%} = 60ml$
- **For 35%:** $V_{d35\%} = \frac{10ml(70\%-35\%)}{35\%} = 10ml$
- **For 40%:** $V_{d40\%} = \frac{10ml(70\%-40\%)}{40\%} = 7.5ml$

Unfortunately, I don't have the necessary equipment to make the solutions, therefore I have to ask for accesses to one of the labs and in order to get it I need to complete some safety courses for unsupervised personnel, which I plan to do today. I'm still collecting data for clean air.

Second revision

January 20,2020.

Date: January 23, 2020

Python libraries are not the best option to use the sensor, therefore I had to choose between changing to either C++ or Arduino. Because of simplicity I decided to use Arduino. There is a laboratory that has a chamber that we might use, we have to schedule an appointment in order to check it.

Date: January 27, 2020

The teacher brought me an Arduino Uno however, it didn't work because it doesn't have enough memory to even make the basic measurements. We still haven't had an answer from the person in charge of the laboratory that has the chamber that we might use.

Date: January 31, 2020

We check the chamber and it is perfect for what we need, we can control the temperature, relative humidity and CO_2, there is also a port so the Arduino cable can go out. I had to take several courses to get access to the lab.

- Incident Reporting and Investigation Training
- WHMIS 2015
- Laboratory Safety Training (Classroom Feb 06)
- Spill Response Training
- Bio-safety (Program) Training
- Bio-safety (Biohazards handling) Training (Classroom March 12)
- Bio-safety (Booldborne Pathogens) training

I also work in the background for the final report and got Isopropyl and Methanol samples at different concentrations.

Date: February 04, 2020

Listing 3: TVOC and eCO2eq Script for the acquisition Arduino 02-04-2020.

```
#include <Wire.h>
#include "Adafruit_SGP30.h"

Adafruit_SGP30 sgp;


void setup() {
  Serial.begin(9600);
  Serial.println("SGP30 test");


  if (! sgp.begin()){
    Serial.println("Sensor not found :(");
    while (1);
  }
  Serial.print("Found SGP30 serial: ");
  Serial.print(sgp.serialnumber[0], HEX);
  Serial.print(sgp.serialnumber[1], HEX);
  Serial.println(sgp.serialnumber[2], HEX);
  Serial.print("TVOC ppb ");
  Serial.println("eCO2 ppm ");

}

int counter = 0;

void loop() {

  if (! sgp.IAQmeasure()) {
    Serial.println("Measurement failed");
    return;
  }
  Serial.print(sgp.TVOC); Serial.print(" ppb\t");
  Serial.print(sgp.eCO2); Serial.println(" ppm");

delay(1000);

  counter++;
  if (counter == 30) {
    counter = 0;

    uint16_t TVOC_base, eCO2_base;
    if (! sgp.getIAQBaseline(&eCO2_base, &TVOC_base)) {
      Serial.println("Failed to get baseline readings");
      return;
    }
    Serial.print("****Baseline values: eCO2: 0x");
    Serial.print(eCO2_base, HEX);
    Serial.print(" & TVOC: 0x");
    Serial.println(TVOC_base, HEX);
  }
}
```

Listing 4: Baseline Script Arduino 02-04-2020.

```
#include <Wire.h>
#include "Adafruit_SGP30.h"

Adafruit_SGP30 sgp;


void setup() {
  Serial.begin(9600);
  Serial.println("SGP30 test");


  if (! sgp.begin()){
    Serial.println("Sensor not found :(");
    while (1);
  }
  Serial.print("Found SGP30 serial: ");
  Serial.print(sgp.serialnumber[0], HEX);
  Serial.print(sgp.serialnumber[1], HEX);
  Serial.println(sgp.serialnumber[2], HEX);
//Serial.print("TVOC ppb ");
//Serial.println("eCO2 ppm ");

}
int counter = 0;

void loop() {

  if (! sgp.IAQmeasure()) {
    Serial.println("Measurement failed");
    return;
  }
//  Serial.print(sgp.TVOC); Serial.print(" ppb\t");
//  Serial.print(sgp.eCO2); Serial.println(" ppm");

delay(1000);

  counter++;
  if (counter == 2) {
    counter = 0;

uint16_t TVOC_base, eCO2_base;
if (! sgp.getIAQBaseline(&eCO2_base, &TVOC_base)) {
  Serial.println("Failed to get baseline readings");
  return;
  }
  Serial.print("****Baseline values: eCO2: 0x");
  Serial.print(eCO2_base, HEX);
  Serial.print(" & TVOC: 0x");
  Serial.println(TVOC_base, HEX);


}
}
```

Date: February 05, 2020

Python program to access and save data acquired from the port COM6, the Arduino.

Date: February 11, 2020

Listing 5: Data Acquiring (TVOC) Script 02-11-2020 Python.

```
# -*- coding: utf-8 -*-
"""
@author: Fabi
"""
import time
import csv
import serial


###Abrir el puerto COM6


ser = serial.Serial('/dev/ttyACM0', 9600)
count = 0
elapsed_time = 0
with open(r'Data_file_01.csv', 'a') as b:
    Datos = csv.writer(b,delimiter=',')
    Datos.writerow(['Sample','TVOC','eCO2'])

while True:
    line = ser.readline()
    line_str = str(line)
    largo = len(line_str)
    line_fix=line_str[2:largo-5]
    data = str(elapsed_time)+','+line_fix
    print(line_fix)
    with open(r'Data_file_01.csv', 'a') as b:
        Datos = csv.writer(b,delimiter=' ')
        Datos.writerow([data])


    elapsed_time +=1
    time.sleep(1)

    if elapsed_time == 100:
        break
```

Listing 6: Data Acquiring (Baseline) Script 02-11-2020 Python.

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Feb  7 10:16:09 2020

@author: Fabi
"""
import time
import csv
import serial

#Abrir el puerto COM6

ser = serial.Serial('/dev/ttyACM0', 9600)

count = 0
elapsed_time = 0
with open(r'Baseline_file_01.csv', 'a') as b:
    Datos = csv.writer(b,delimiter=',')
    Datos.writerow(['Baseline value TVOC[HEX]',
                    'Baseline value eCO2[HEX]'])

while True:
    line = ser.readline()
    print(line)
    line_str = str(line)
    largo = len(line_str)
    line_fix=line_str[2:largo-5]
    print(line_fix)

    with open(r'Baseline_file_01.csv', 'a') as b:
        Datos = csv.writer(b,delimiter=' ')
        Datos.writerow([line_fix])

    time.sleep(1)
```

Listing 7: Data Acquiring (TVOC) Script 02-11-2020 Arduino.

```
#include <Wire.h>
#include "Adafruit_SGP30.h"

Adafruit_SGP30 sgp;

void setup() {
  Serial.begin(9600);
//  Serial.println("SGP30 test");


  if (! sgp.begin()){
    Serial.println("Sensor not found :(");
    while (1);
  }
}

int counter = 0;

void loop() {

  if (! sgp.IAQmeasure()) {
    Serial.println("Measurement failed");
    return;
  }
  Serial.print(sgp.TVOC); Serial.print(",");
  Serial.println(sgp.eCO2);

delay(1000);
}
```

Listing 8: Data Acquiring (Baseline) Script 02-11-2020 Arduino.

```
#include <Wire.h>
#include "Adafruit_SGP30.h"

Adafruit_SGP30 sgp;

void setup() {
  Serial.begin(9600);
//  Serial.println("SGP30 test");


  if (! sgp.begin()){
    Serial.println("Sensor not found :(");
    while (1);
  }
}

int counter = 0;

void loop() {

  if (! sgp.IAQmeasure()) {
    Serial.println("Measurement failed");
    return;
  }

  uint16_t TVOC_base, eCO2_base;
  if (! sgp.getIAQBaseline(&eCO2_base, &TVOC_base)) {
    Serial.println("Failed to get baseline readings");
    return;
  }
  Serial.print(eCO2_base, HEX); Serial.print(",");
  Serial.println(TVOC_base, HEX);
  delay(1000);
  }
```

Date: February 13, 2020

The approach that I'm going to take will consist in data analysis, as I said before. I talk to the Sensirion Company and they assure me that the sensor will provide different selectivities for each VOC. Therefore, I will do two different types of tests.

- Controlled Chamber: As said before, A chamber with controlled temperature, relative humidity and clean air.
- Not Controlled Chamber: A 3D printing chamber where measurements will be taken without abrupt changes in air currents and temperature. For this one I will use a sensor to measure the actual temperature and relative humidity, this way, even if I can't control the values, I can still set the values in my code, in order to get more accurate measurements.

The Sensirion gas kit that I mentioned before have some humidity and temperature sensors, I have to chose between them.

Date: February 14, 2020

- SHTW2:. Wafer level chip sale package. humidity sensor, offering a complete digital humidity and temperature sensor system in a package so tiny that it fits into virtually any app. Comes in a flip chip package that represents one of the simplest and smallest possible ways of packing for a semiconductor chip.

  Complete sensor system on a simple chip, consisting of a capacitive humidity sensor, a band-gap temperature analog and digital communication interface supporting I^2C fast mode.

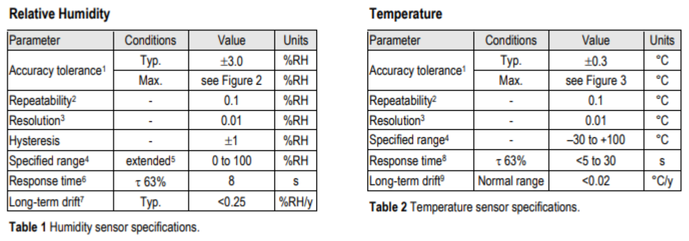
  The humidity sensor covers a humidity measurement of $-30°$ to 100°C and 0 to 100%RH with ±0.3°C and $\pm 3^{\%}$RH. The operating voltage is 1.8V and has a low power consumption that makes the sensor suitable for wearable technologies that run on the tightest power budgets.
- SHT31: ± 2 @ $0 - 100\%$ RH, ± 0.2 @ $0 - 90°$C, VDD $2.15 - 5.5V$, I^2C, $2.5 \times 2.5\text{mm}^2$.
- SHTC3: Complete sensor system on a single chip, consisting of a capacitive humidity sensor, a bandgap temperature sensor, analog and digital signal processing A/D converter, calibration data memory and digital communication interface supporting I^2C fast mode $2 \times 2 \times 0.75\text{mm}\ ^3$. Cover a humidity measurement range of $-40°$ to 125°C and 0 to 100%RH with ±0.2°C and ±2% RH. The supply voltage of 1.62 to 3.6V and energy budget below $1\mu J$ per measurements.

- STS31: Just temperature.

The difference between the SHTW2 and the SHTC3 is the power and temperature range, they are, basically, the same sensor. The SHTW2 is low power and the SHTC3 has multiple modes of use. Nevertheless, I think this is unnecessary, is better to use the smallest one.

SHTW2:

According to the datasheet the humidity and temperature specifications are, figure 18.



Relative Humidity

Parameter	Conditions	Value	Units
Accuracy tolerance[1]	Typ.	±3.0	%RH
	Max.	see Figure 2	%RH
Repeatability[2]	-	0.1	%RH
Resolution[3]	-	0.01	%RH
Hysteresis	-	±1	%RH
Specified range[4]	extended[5]	0 to 100	%RH
Response time[6]	τ 63%	8	s
Long-term drift[7]	Typ.	<0.25	%RH/y

Table 1 Humidity sensor specifications.

Temperature

Parameter	Conditions	Value	Units
Accuracy tolerance[1]	Typ.	±0.3	°C
	Max.	see Figure 3	°C
Repeatability[2]	-	0.1	°C
Resolution[3]	-	0.01	°C
Specified range[4]	-	–30 to +100	°C
Response time[8]	τ 63%	<5 to 30	s
Long-term drift[9]	Normal range	<0.02	°C/y

Table 2 Temperature sensor specifications.

Fig. 18: SHTW2 humidity and temperature specifications

There is a table with typical accuracy of RH measurements given the RH% for temperatures from 0° to 80°C. The sensor shows best performance when operated within recommended normal temperature and humidity range of 5° to 60°C and 20 to 80% RH respectively.

The sensor shall not get in close contact with volatile organic chemicals such s solvents or other organic compounds like ethanol, methanol, isopropanol, acetone, etc.

Third revision February 24,2020.

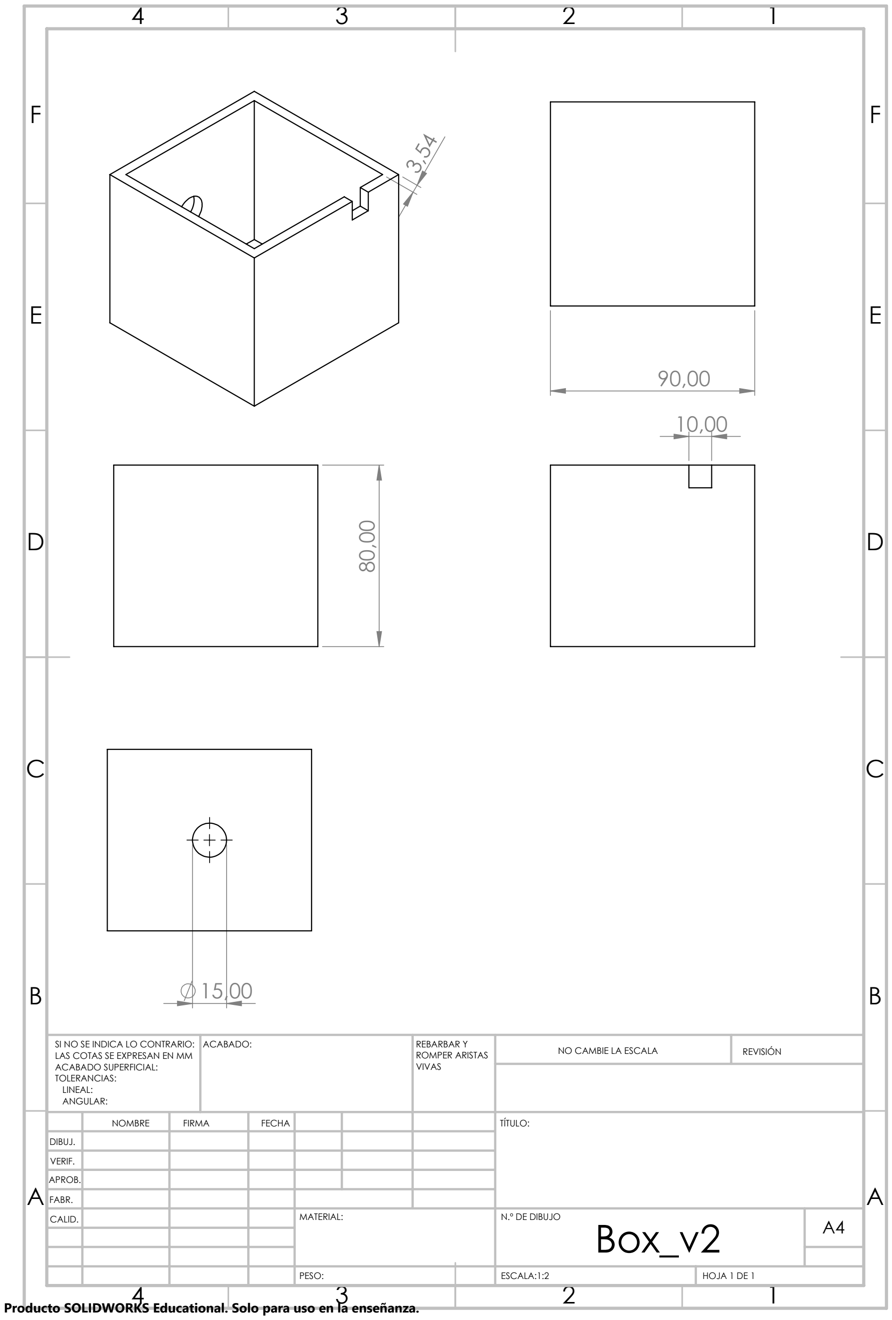
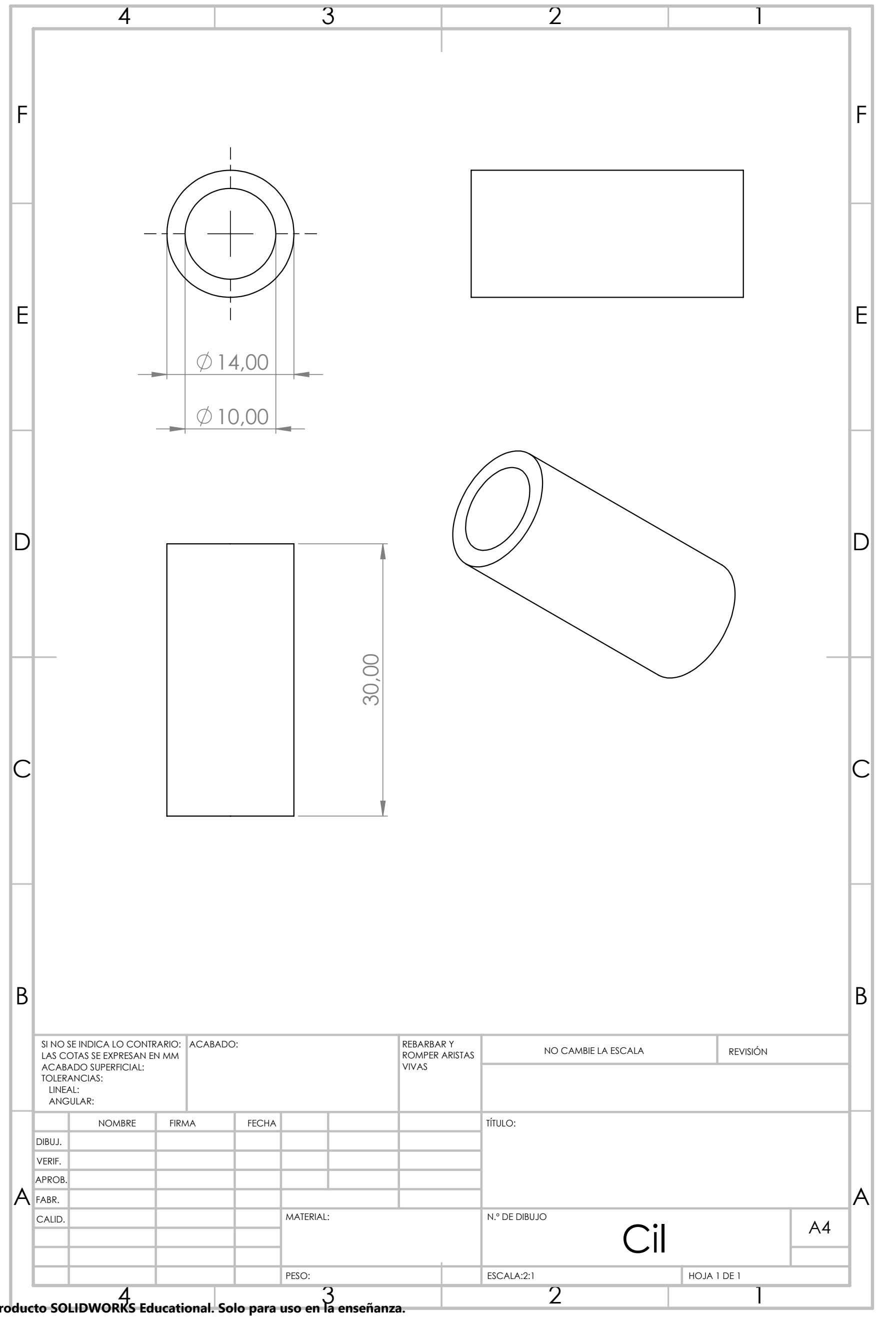

3,54
90,00
10,00
80,00
∅15,00
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:
ACABADO:
REBARBAR Y ROMPER ARISTAS VIVAS
NO CAMBIE LA ESCALA
REVISIÓN
NOMBRE
FIRMA
FECHA
DIBUJ.
VERIF.
APROB.
FABR.
CALID.
TÍTULO:
MATERIAL:
N.º DE DIBUJO
Box_v2
A4
PESO:
ESCALA:1:2
HOJA 1 DE 1
Producto SOLIDWORKS Educational. Solo para uso en la enseñanza.

Fig. 19: Chamber Box Sketch



4
3
2
1
F
E
D
C
B
A
∅14,00
∅10,00
30,00
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:
ACABADO:
REBARBAR Y ROMPER ARISTAS VIVAS
NO CAMBIE LA ESCALA
REVISIÓN
NOMBRE
FIRMA
FECHA
TÍTULO:
DIBUJ.
VERIF.
APROB.
FABR.
CALID.
MATERIAL:
N.º DE DIBUJO
Cil
A4
PESO:
ESCALA:2:1
HOJA 1 DE 1
Producto SOLIDWORKS Educational. Solo para uso en la enseñanza.

Fig. 20: Chamber Cylinder Sketch

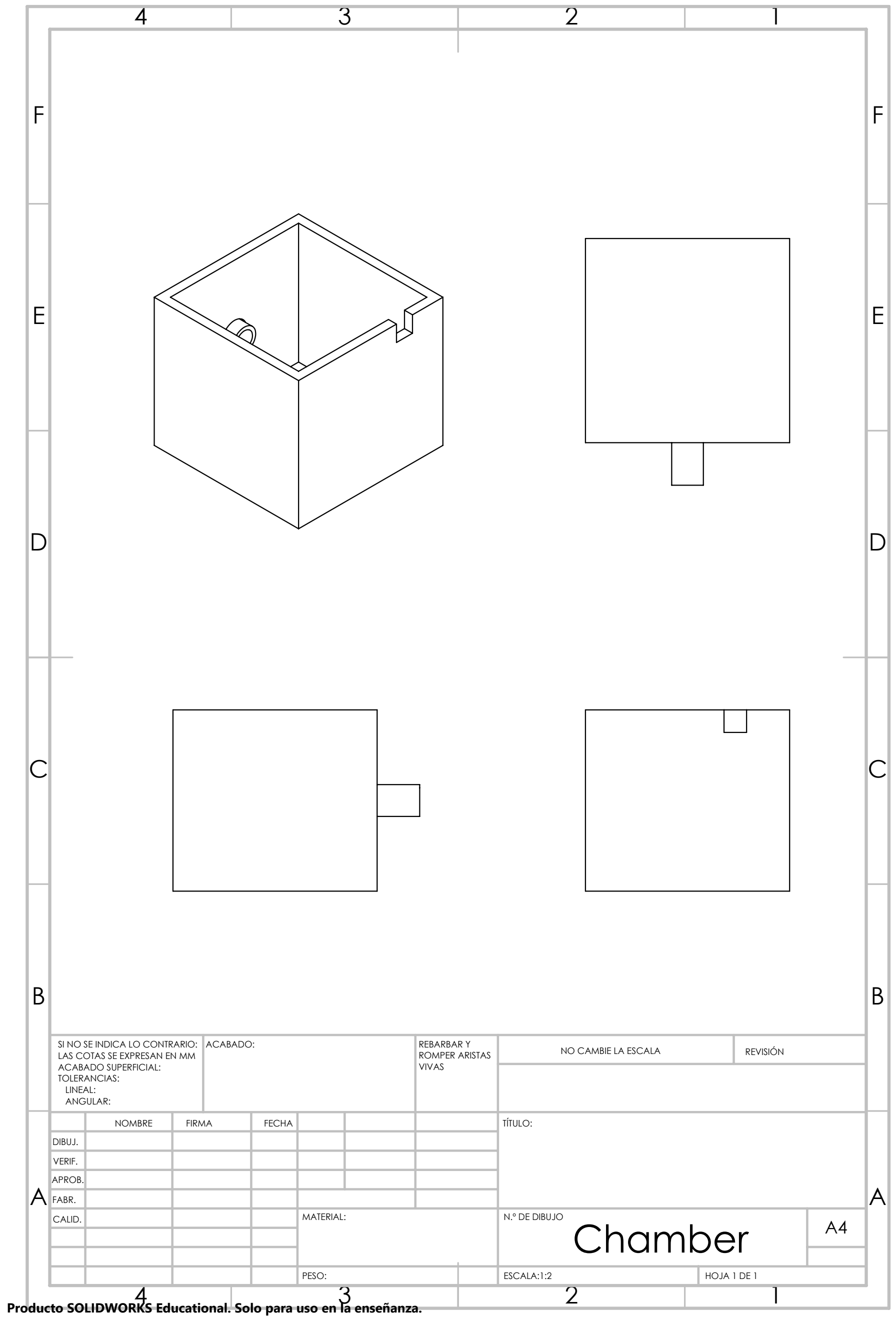


Fig. 21: Chamber Sketch

Date: February 24, 2020

I tried to do 3D printing but I couldn't because due the lack of personnel I still have no access to the lab and it was closed. I wrote to the Sensirion gas company about some questions regarding the baseline values.

Date: February 25, 2020

I tried to do 3D printing but I couldn't because the printers where either not working, all occupied or just there was not enough filament and there was nobody I could ask about it. Calendar for march measurements.

March
2020

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	*Measurement* *2* Set Baseline Values 5pm Run for +12 hours	*Data Analysis* *3*	*Measurement* *4* Set 1 Lab 10am-11:30am Set 1 BTLab 1:00pm-2:30pm	*Data Analysis* *5*	*Measurement* *6* Set 2 10am-11:30am Set 2 BTLab 1:00pm-2:30pm	*7*
8	*Measurement* *9* Set 3 10am-11:30am Set 3 BTLab 1:00pm-2:30pm	*Data Analysis* *10*	*Measurement* *11* Set 4 10am-11:30am Set 4 BTLab 1:00pm-2:30pm	*EHS032 Course* *12*	*Measurement* *13* Set 5 10am-11:30am Set 5 BTLab 1:00pm-2:30pm	*14*
15	*Measurement* *16* Set 6 10am-11:30am Set 6 BTLab 1:00pm-2:30pm	*Data Analysis* *17*	*Measurement* *18* Set 7 10am-11:30am Set 7 BTLab 1:00pm-2:30pm	*Data Analysis* *19*	*Measurement* *20* Set 8 10am-11:30am Set 8 BTLab 1:00pm-2:30pm	*21*
22	*Measurement* *23* Set 9 10am-11:30am Set 9 BTLab 1:00pm-2:30pm	*Data Analysis* *24*	*Measurement* *25* Set 10 10am-11:30am Set 10 BTLab 1:00pm-2:30pm	*Data Analysis* *26*	*Measurement* *27* Set 11 10am-11:30am Set 11 BTLab 1:00pm-2:30pm	*28*
29	*Data Analysis* *30*	*Data Analysis* *31*				

Fig. 22: Chamber Sketch

Date: February 26, 2020

I printed the cylinder but couldn't print the box because something wasn't working, I asked for advice and got a reply so I'll try again tomorrow. I also soldier the second sensor, so I can use two sensor for the two places I want to acquire data from. I also got reply for the Sensirion company about the doubts I had. The THC dissolved in methanol samples arrive.

Date: February 27, 2020

According to the Central limit theorem I need at least 30 samples. For this reason I decided to take 350 samples until I can decide how many I should take, 350 samples are taken in approximately 5 minutes. I finished the printing of the chamber, figure 23.

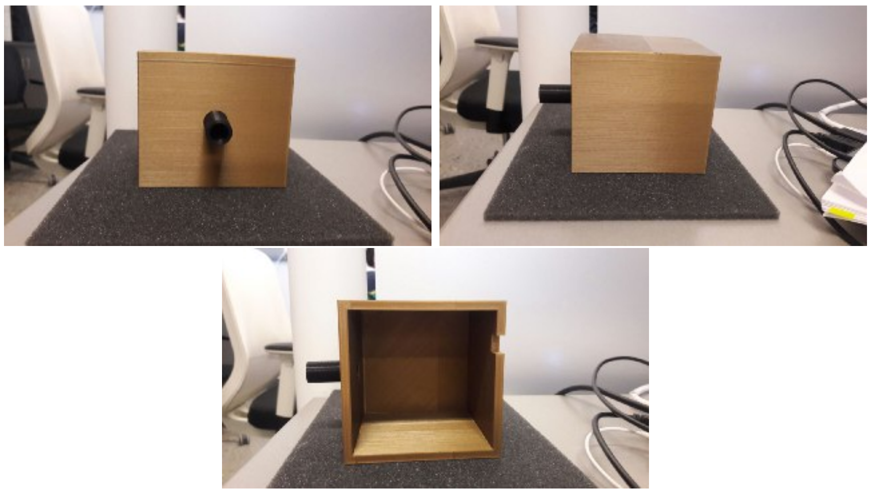


Fig. 23: 3D printed chamber

Date: February 28, 2020

I worked on the overview, read about different types o drug identification matrices. I left the sensor working in order to get the baseline values, for this I set the temperature and RH values.

Temperature ($^\circ C$)	**Relative Humidity RH (%)**
23.178	26.392

Tab. 2: Temperature and Relative Humidity Values for Baseline Calibration

Date: February 29, 2020

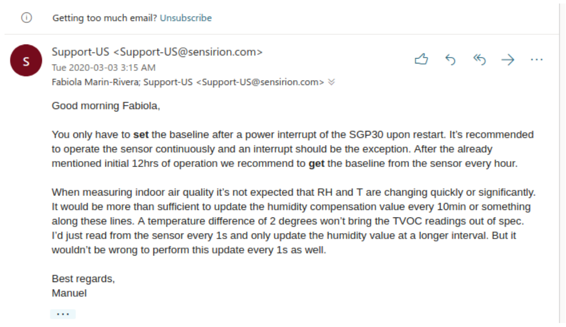
I obtained the baseline values, 3. I took some measurements.

$\mathbf{eCO_2}$	**TVOC**
0x9769	0x92CC

Tab. 3: Baseline Values, february 29

Date: March 02, 2020

I ran the sensor with the values of baseline obtained two days before, under similar conditions the values changed, I had some questions about how should I worked with this base values so I wrote to the company, the answer is in figure 24.



Getting too much email? Unsubscribe

Support-US <Support-US@sensirion.com>
Tue 2020-03-03 3:15 AM
Fabiola Marin-Rivera; Support-US <Support-US@sensirion.com>

Good morning Fabiola,

You only have to **set** the baseline after a power interrupt of the SGP30 upon restart. It's recommended to operate the sensor continuously and an interrupt should be the exception. After the already mentioned initial 12hrs of operation we recommend to **get** the baseline from the sensor every hour.

When measuring indoor air quality it's not expected that RH and T are changing quickly or significantly. It would be more than sufficient to update the humidity compensation value every 10min or something along these lines. A temperature difference of 2 degrees won't bring the TVOC readings out of spec. I'd just read from the sensor every 1s and only update the humidity value at a longer interval. But it wouldn't be wrong to perform this update every 1s as well.

Best regards,
Manuel

Fig. 24: Sensirion Gas Company - answer regarding baseline values.

Date: March 03, 2020

I've had several problems regarding the project.

- The 5$\%v/v$ concentration, which is the lowest that I wanted to try, have a value in ppb much greater than the sensor output range.
- All the samples I have are in $\%v/v$ and in order to get the values of samples that allowed me to use the sensor I need them to be in $\%v/v$, I can make all he conversions but they won't be as accurate as I will like them to be.
- The stabilization time of the sensor is pretty high.
- The laboratory I'm currently in doesn't have the appropriate equipment (fume hood) to work with methanol.

Date: March 04, 2020

I'm going to take the chamber measurements under the datasheet ideal case conditions, which are $t = 25°C$ and $RH = 50\%$. The idea is to leave it working for 12 hours (one night) and then use the next day to take all the measurements. We pay for certain amount of use hours and because the datasheet recommends not to turn the sensor off, I will tr to take all the data the same day.

I'm still having some issues with the values of concentrations I need to use.

The Adafruit sensor and the kit sensor work the same but not under the same values. In other words, if I expose each to the same sample of ethanol, both of them will rise and stabilize at the same time but under different ppb values. For the kit sensor I'm using a temperature and humidity sensor that cannot be exposed to VOCs, for the Adafruit sensor I set the t and RH values. I have to change the code so the baseline values get set every other hour.

Date: March 05, 2020

I tested the Adafruit sensor against the Kit sensor to see how different did they behaved, FOTOS, the sensor stabilize faster if I remove the 3D printed chamber. But still, it takes a lot of time.

Date: March 06, 2020

I've been reading some papers regarding the correlation of breath and THC and breath and alcohol and in most of them the concentration values use for the testing are around the $\mu g/mL$, for ethanol and THC, the detection process was made with GC-MS methods.

I believe I can do 3 different analysis, one to determine if the sensor can differentiate between the compounds, one to see if it can differentiate between different concentrations of the same compound and finally, to see if there is any correlation between the three of them.

Date: March 06, 2020

I've talked to a chemistry PhD student and she recommended me to lower the concentrations to the order of $\times 10^{-6}$w/v.

$$1\frac{mg}{ml} = 1000000ppb \tag{5}$$

The gas sensor output limit is 60000ppb which means that for Ethanol at a concentration:

$$\begin{aligned}\frac{1\%w}{v} &= \frac{1g}{100ml}\\ &= \frac{10mg}{ml}\\ &= 10 \cdot 1 \times 10^6 ppb\\ &= 10000000\end{aligned}$$

Which is way over the saturation value. With a lower concentration:

$$\begin{aligned}\frac{0.001\%w}{v} &= \frac{0.001g}{100ml}\\ &= \frac{0.01mg}{ml}\\ &= 0.01 \cdot 1 \times 10^6 ppb\\ &= 10000\end{aligned}$$

Under these the values of concentration selected are:

- 0.001%w/v = 10000
- 0.002%w/v = 20000
- 0.005%w/v = 50000
- 0.01%w/v = 100000

Date: March 09, 2020

The values selected for the data in order to be representative are really hard to obtain by doing dissolution's, because they are really small, I don't know which is the best way to do the testing. One option is to use micropipette but there are some concepts I'm still having trouble understanding.

This because I'm not sure how do the concentration changes, or if it change at all, if I use a sample as small as 0.05ml (1 drop).

Date: March 11, 2020

I've been doing some research and trying to understand how the concentrations works.

$$c_1 v_1 = c_2 v_2 \tag{6}$$

The student from chemistry recommended me to use the equation 6, where c_1 is the concentration of stock solution, v_1 is how much of my stock I need to take to make v_2 at c_2 concentration.

Date: March 12, 2020

I went to foothills campus to take the Biosafety (Biohazard Handling) Training: 9:00 AM - 12:00 PM Health Science Centre G500.

I had a meeting with Steve, the person in charge of the lab I need to use to take the measurements, I'm ready to start once I figure out the sample concentration problems.

References

[1] M. Paknahad, "Development of highly selective single sensor microfluidic-based gas detector," Ph.D. dissertation, University of British Columbia, 2017.

[2] J. L. G. Fierro, *Metal oxides: chemistry and applications.* CRC press, 2005.

[3] *Datasheet SGP30 Sensirion Gas Platform*, Sensirion: The sensor company, 4 2019, version 0.92.

[4] J. Tyson, "How encryption works," Apr 2001. [Online]. Available: https://computer.howstuffworks.com/encryption7.htm