Adrian Ioan Costea

Facultatea de Electronică, Telecomunicații și Tehnologia Informației

Scaner 3d optic

Link proiect github: https://github.com/adivd15/LaserScanner

Cuprins

[1. Introducere 2](#_Toc122287621)

[1.1. Descriere 2](#_Toc122287622)

[1.2. Principii de funcționare 2](#_Toc122287623)

[1.3. Tehnologia aleasă 2](#_Toc122287624)

[2. Scanner 3D optic 3](#_Toc122287625)

[2.1. HARDARE 3](#_Toc122287626)

[2.1.1. Electronică 3](#_Toc122287627)

[2.1.2. Mecanică 7](#_Toc122287628)

[2.2. Software 8](#_Toc122287629)

[3. Concluzii 17](#_Toc122287630)

[4. Planuri pentru o următoare versiune 17](#_Toc122287631)

[Bibliografie 18](#_Toc122287632)

# Introducere

## Descriere

Scanarea 3D este procesul de analizare a unui obiect pentru a colecta date în legătura cu forma acestuia. Datele colectate pot fi folosite apoi pentru a construi modele 3D[[1]](#footnote-1). (Wikipedia)

Scanarea 3D poate fi făcută prin diferite tehnologii, cum ar fi:

* Industrial Tomography Scanning
* Structured-light 3D scanners
* LiDAR
* ToF (time of flight)

Datele colectate pot fi folosite în aplicații precum cinematografie, realitate augmentată, recunoaștere de gesturi, cartografierea unei încăperi, prostetice, etc.

## Principii de funcționare

Scopul unui scaner 3D este în general creare unui model 3D din datele colectate de acesta. În general scanerul 3D creează dintr-un Point Cloud[[2]](#footnote-2) cu ajutorul cărora se poate face reconstrucția obiectului.

## Tehnologia aleasă

Pentru acest proiect am ales construirea unui scaner 3D, folosind un senzor ToF optic. Am ales această tehnologie din următoarele motive:

* Timp redus de implementare
* Cost redus
* Materialele disponibile în orice magazin de electronică

A picture containing text, floor, indoor

Description automatically generated

Figură 1‑1Scanerul 3D în stagiul actual

# Scanner 3D optic

Trebuie să obținem un Point Cloud pentru a putea reconstrui obiectul, deci va trebui să obținem coordonatele unor puncte de pe obiect intr-un sistem de axe , unde centul obiectului este centrul sistemului de coordonate.

Pentru a rezolva această problemă putem să măsuram distanța de la diferite puncte de pe obiect până într-un punct fix, ales de noi, apoi prin câteva calcule matematice putem obține coordonatele.

Am împărțit această problemă în două părți:

* HARDWARE, care se poate împărți în:
  + Electronică
  + Mecanica
* SOFTWARE

## HARDARE

### Electronică

#### Senzor de distanță Sharp GP2Y0A21YK0F

În primul rând trebuie să rezolvăm problema măsurării distanței. Pentru aceasta am ales să folosesc un senzor optic, ce funcționează pe principiul ToF.

Un senzor de distanță optic Time of Flight se folosește de principiul reflexiei undelor de lumină. Acesta emite o undă, așteaptă ca lumina să se lovească de obiect și să se reflecte înapoi la senzor. Cunoscând viteza luminii și timpul scurs de la emiterea razei de lumină până la recepționarea acesteia, distanța poate fi calculată cu formula, știind că unda de lumină parcurge drumul dus-întors:

Diagram

Description automatically generated

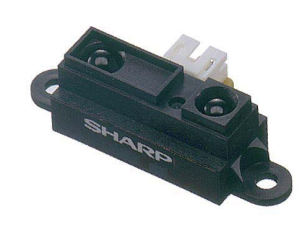
Figură 2‑1Exemplu de funcționare senzor ToF IR

Problema însă cu acești senzori este că pot fi afectați de condițiile de lumină din încăpere, însă au un cost redus, iar această problemă poate fi rezolvată prin acoperirea scanerului cu o carcasă pentru a evita fluctuații de lumină.

Considerând cele de mai sus am decis să aleg un senzor Sharp GP2Y0A21YK0F, având următoarele caracteristici:

* Range:
* Tip: IR

Este compus dintr-un emițător IR, un PSD și un circuit de procesare a semnalului.

Diagram, schematic

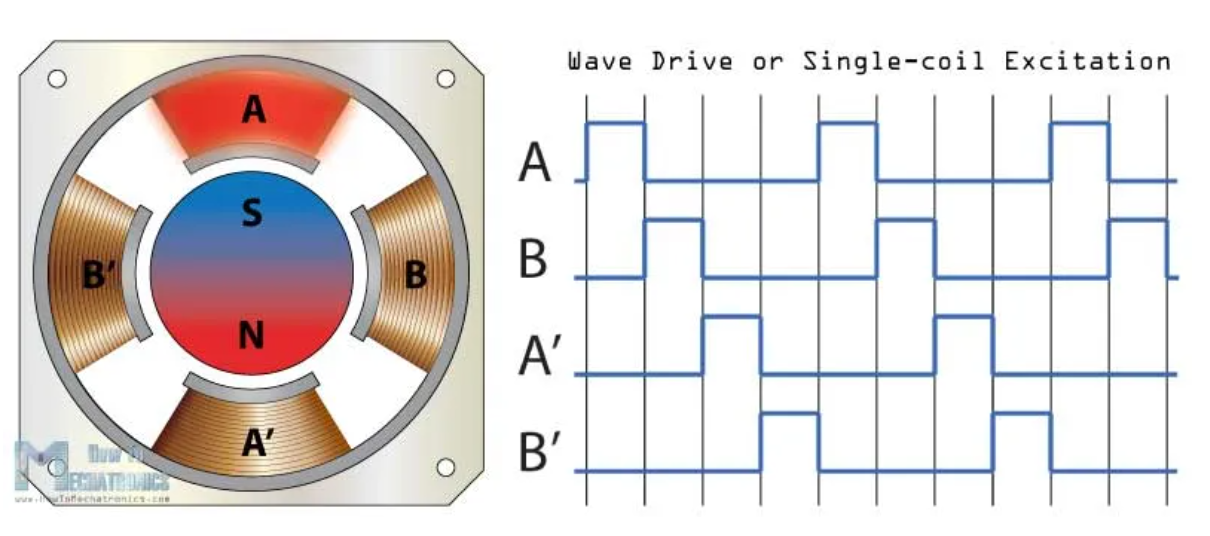
Description automatically generated

Figură 2‑2Sharp sensor Figură 2‑3Schema bloc

#### Motor stepper si drivere de motoare

Acuma că am rezolvat problema măsurării dintanței trebuie rezolvată trebuie să rezolvăm și problema poziționării senzorului pentru a măsura diferite puncte de pe obiect. Pentru aceasta cea mai potrivită soluție vine din zona de imprimante 3D și anume motoarele stepper.

Un motor stepper este un motor DC cu perii ce permite poziționarea precisă a axului fără ajutorul unor componente precum encoderele. Acesta este format dintr-o serie de bobine, care după ce sunt energizate permit rotirea rotorului cu o precizie și cuplu mare.



Figură 2‑4Principiu de functionare motor stepper

Pentru proiect am ales doua steppere Nema.

A picture containing adapter

Description automatically generated

Figură 2‑5Steperele nema alese

Un pas al acestor motoare va învârti rotorul cu 1.8 grade.

Pentru o utilizare mai ușoară a acestora am ales să folosesc două drivere pentru acest tip de motoare ce vor putea fi controlate de microcontroler, deoarece acesta nu poate oferi suficientă putere pentru a controla motoarele de unul singur.

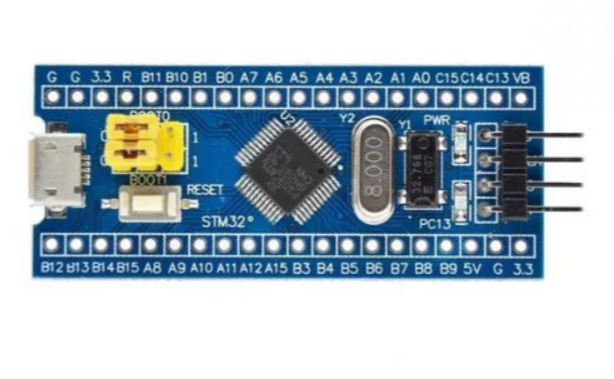
A picture containing text, electronics, circuit

Description automatically generated

Figură 2‑6Driver pentru motoare pas cu pas A4988

#### Microcontroler STM32F103C8T6 (bluepill)

Ultima problemă ce trebuie rezolvată constă într-un circuit de control. Circuitul ales a fost o placă de dezvoltare bazată pe microcontrolerul STM32F103C8T6. Am ales acest microcontroler pentru prețul său relativ redus, viteză mare de procesare și pentru faptul că este un microcontroler pe 32 de biți. De asemenea ARM oferă un pachet software ce conține un tool pentru generare de cod și un tool pentru a programa microcontrolerul (STM32CubeMx; STM32CubeProgrammer).

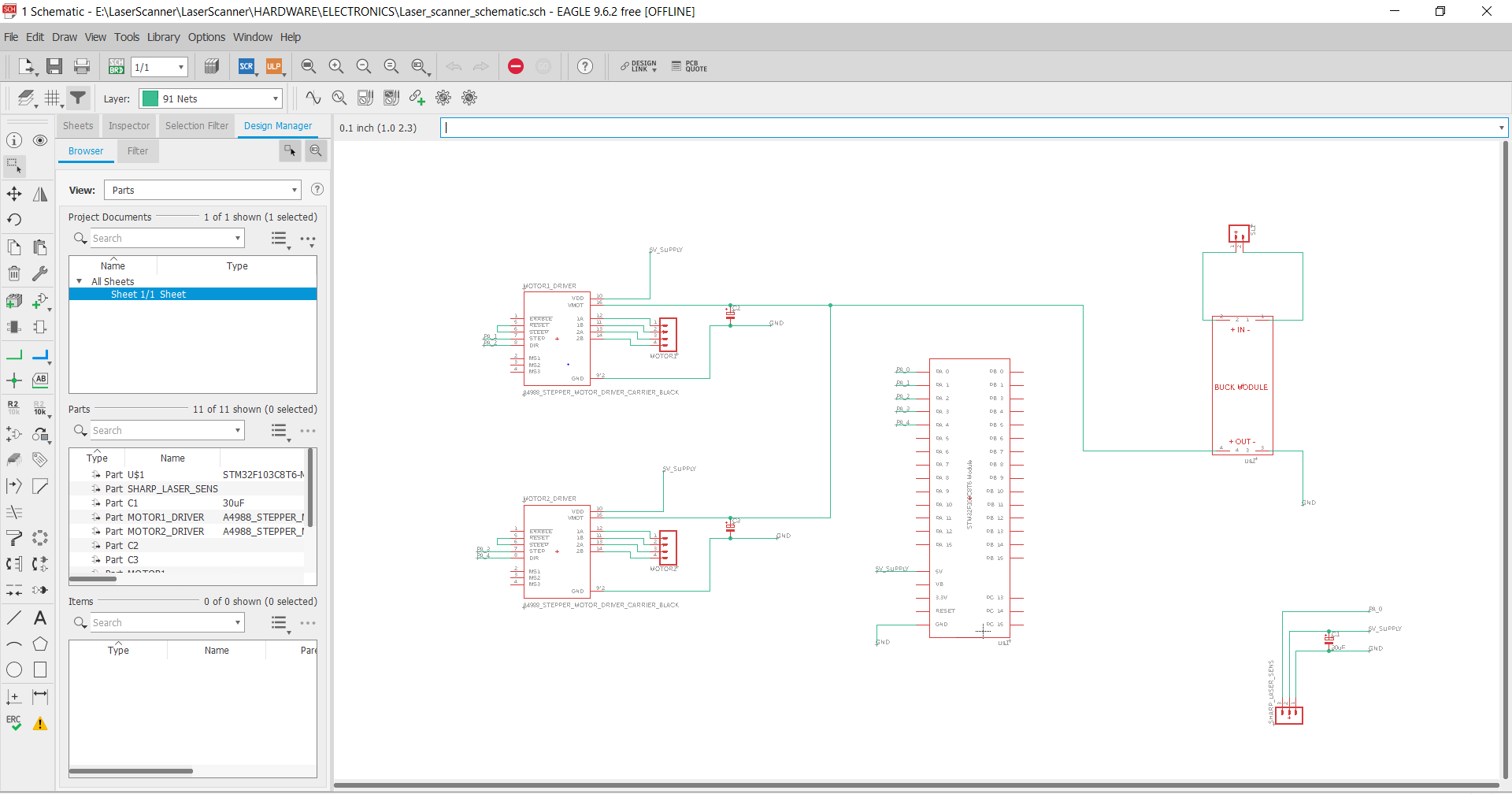


Figură 2‑7STM32F103C8T6

#### Schema electronică si realizarea cablajului

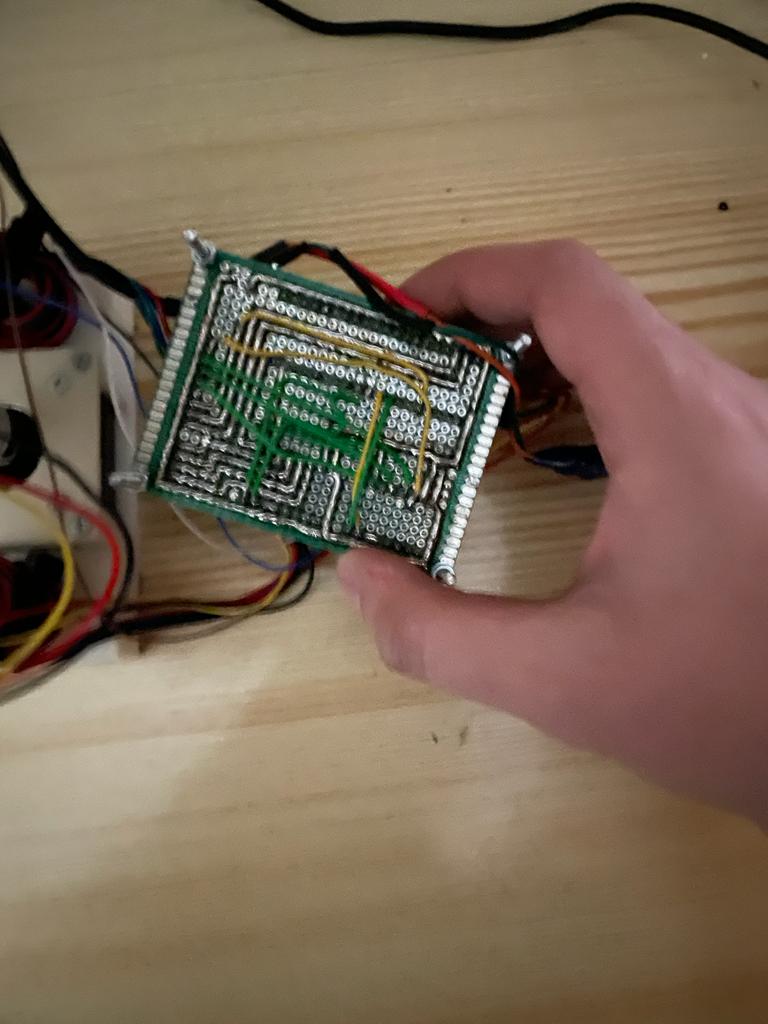
Am realizat schema electrică cu ajutorul aplicației Eagle.

Datasheetul producătorilor recomandă montarea unui condensator de decuplare pe partea de alimentare a senzorului și a driverelor de steppere, pentru a stabiliza alimentare.



Figură 2‑8Schema electronică

Am realizat cablajul pe o placă de prototipare.

A picture containing electronics

Description automatically generated

Figură 2‑9Cablajul final

### Mecanică

Am decis că o soluție bună pentru a scana obiectul ar fi ca obiectul să fie poziționat pe o bază rotativă, iar senzorul să fie pozitionat pe o baza ce se ridică și coboară în pe axa Z.

Pentru realizarea părții de mecanică am folosit SolidWorks. Toate componentele au fost printate 3D.

A picture containing engineering drawing

Description automatically generated

Figură 2‑10Imagine cu modelul 3D in stagiul actual

Pentru mișcarea senzorului pe axa Z am folosit un ax filetat de 8mm cuplat la unul dintre steppere.

## Software

Pentru software-ul microcontrolerului am decis să folosesc o arhitectură layered.

A picture containing table

Description automatically generated

Figură 2‑11Arhitectura aleasă

In partea de HAL (Hardware Abstraction Layer) se află driverele generate prin intermediul lui STM32CubeMx. În stratul SRV (Services) se află codul ce preia datele de la senzor și controlează stepperele. În partea de HMI (Human Machine Interface) se află partea din cod ce se ocupă cu trimiterea datelor de la microcontroler la laptop.

Pașii pe care trebuie să îi facă sistemul sunt următorii:

1. Inițializează componentele
2. Ia o măsurătoare de distanță
3. Calculeaza coordonatele si trimite-le pe portul USB al laptopului
4. Daca baza a făcut o rotație completă, ridică senzorul
5. Daca baza nu a realizat o rotație completă, învârte baza cu un anumit număr de grade

Primul pas aste generarea codului:

Graphical user interface

Description automatically generated with medium confidence

Figură 2‑12Configuratia selectată

Trebuie să selectăm pinii PA3, PA4, PA5, PA6 ca fiind ieșiri (GPIO\_OUTPUT), pinii aceștia fiind folosiți pentru a controla driverele de motoare. De asemenea vom folosi unul dintre cele doua ADC-uri ale microcontrolerului pentru a citi datele de la senzor. În cazul de față am folosit ADC1, care este legat la pinul PA0.

De asemenea trebuie să selectăm clock-ul microcontrolerului ca fiind rezonatorul de cristal de pe placă, să selectăm obțiunea de usb device și să ne asigurăm că la portul USB ajunge o frecvență de 48MHz.

Graphical user interface

Description automatically generated

Figură 2‑13Configuratia ceasului

După care putem genera codul (selectăm în opțiunile de generare să genereze un proiect cu makefile) și putem deschide proiectul cu VS-code iar apoi îl putem compila folosind arm gnu gcc compiler.

A screenshot of a computer

Description automatically generated with medium confidence

Figură 2‑14Imagini cu proiectul final

Pentru a putea controla motoarele am scris o funcția

void *motorControl\_spinMotor*(SRV\_MotorControl\_def\_motor selectMotor, uint8\_t numOfSteps, uint8\_t stepDelayuS, uint8\_t direction, uint8\_t delay)

{

*if* (direction == 1)

    {

*HAL\_GPIO\_WritePin*(selectMotor.motor\_dirPort, selectMotor.motor\_dirPin, GPIO\_PIN\_SET);

    }

*else*

    {

*HAL\_GPIO\_WritePin*(selectMotor.motor\_dirPort, selectMotor.motor\_dirPin, GPIO\_PIN\_RESET);

    }

*for* (int i = 0; i < numOfSteps; i++)

    {

*HAL\_GPIO\_WritePin*(selectMotor.motor\_stepPort, selectMotor.motor\_stepPin, GPIO\_PIN\_SET);

*HAL\_Delay*(stepDelayuS);

*HAL\_GPIO\_WritePin*(selectMotor.motor\_stepPort, selectMotor.motor\_stepPin, GPIO\_PIN\_RESET);

*HAL\_Delay*(delay);

    }

}

Funția primește ca intrare motorul pe care dorim să îl învârtim, numărul de pași cu care vrem să învârtim motorul și direcția.

Pentru a selecta direcția în care vrem să învârtim motorul setam pinul dir de pe driverul de motoare (conectat la PA3 și PA5 de pe microcontroler) cu HIGH pentru a învârti în sensul acelor de ceasornic și LOW pentru a învârti invers acelor de ceasornic.

Iar pentru a învârti motorul cu n număr de pași trebuie să trimitem n impulsuri pe pinul STEP de pe driver (conectal la PA4 și PA6 de pe microcontroler).

Pentru a citi datele de pe senzor am scris funcția

void *srv\_sens\_main*(void)

{

    uint16\_t raw;

    float sensor\_data\_sum;

*for*(int i = 0; i < 20; i++)

    {

        //*read sensor data*

*HAL\_ADC\_Start*(&hadc);

*HAL\_ADC\_PollForConversion*(&hadc, 100);

        raw = *HAL\_ADC\_GetValue*(&hadc);

*HAL\_ADC\_Stop*(&hadc);

        sensor\_readings[i].raw\_sensor\_voltage =(float)raw\*(3.3/4096);

        //*convert raw\_sensor\_voltage to sensor\_data*

/\*

*using curve expert we know that: raw\_sensor\_voltage = 7.747/(1+sensor\_data/4.271)*

*so we can calculate sensor\_data using the following formula: sensor\_data = 33.087/raw\_sensor\_voltage-7.747*

\*/

        sensor\_readings[i].sensor\_data = 33.087/sensor\_readings[i].raw\_sensor\_voltage-7.747;

        //*get medium value from sensor\_data*

        sensor\_data\_sum += sensor\_readings[i].sensor\_data;

        //*wait 5ms*

*HAL\_Delay*(5);

    }

    //*calculate medium value*

    data\_meadian.sensor\_data = sensor\_data\_sum/20;

    data\_meadian.sensor\_decimal = (int)(data\_meadian.sensor\_data);

    data\_meadian.sensor\_unit = (int)((data\_meadian.sensor\_data - data\_meadian.sensor\_decimal)\*1000);

    sensor\_data\_sum = 0;

}

Pentru a crește acuratețea măsurătorilor, vom lua un număr de 20 de măsurători și vom face media acestora. Pentru a obține o măsurătoare citim ADC-ul la care este legat senzorul, convertim valoarea citită pentru a afla tensiunea pe care o primim de la senzor cu formula:

Iar pentru a aproxima valoarea citită cu distanța folosim formula:

Pentru a obține această formula a fost nevoie de doua tool-uril:

* Plot digitizer
* Curve expert

În datasheet-ul senzorului putem observa următorul grafic tensiune-distanță:

Chart, line chart

Description automatically generated

Figură 2‑15Caracteristică tensiune distantă a senzorului

Chart, line chart

Description automatically generated

Figură 2‑16Extragerea punctelor de pe caracteristica cu plot digitizer

Folosind plot digitizer putem extrage punctele de pe caracteristică pe care le încărcăm apoi în curve expert pentru a ne genera formula caracteristicii.

Graphical user interface, application

Description automatically generated

Figură 2‑17Curve expert

Putem apoi să folosim algoritmul următor pentru a controla sistemul:

float angle\_rad = 0;

*HAL\_Delay*(4000);

*CDC\_Transmit\_FS*("Waiting a litlle bit to start\r\n",32);

    point.z = 0;

*for*(int i = 0; i < 260; i++){

*for*(int i = 0; i < 40; i++){

*srv\_sens\_main*();

        sens\_readings = *get\_sensor\_data*();

        point.x = (19.0-sens\_readings.sensor\_data)\**sin*(alpha);

        point.y = (19.0-sens\_readings.sensor\_data)\**cos*(alpha);

        point.x\_int = (int)point.x;

        point.y\_int = (int)point.y;

        point.z\_int = (int)point.z;

*if*(point.x<0){

          point.x\_dec = -1\*(int)((point.x - point.x\_int)\*1000);

        }

*else*{

          point.x\_dec = (int)((point.x - point.x\_int)\*1000);

        }

*if*(point.y<0){

          point.y\_dec = -1\*(int)((point.y - point.y\_int)\*1000);

        }

*else*{

          point.y\_dec = (int)((point.y - point.y\_int)\*1000);

        }

        point.z\_dec = (int)((point.z - point.z\_int)\*1000);

*sprintf*(msg,"%d.%d, %d.%d, %d.%d\r\n",point.x\_int,point.x\_dec,point.y\_int,point.y\_dec,point.z\_int,point.z\_dec);

*CDC\_Transmit\_FS*(msg,*strlen*(msg));

*motorControl\_spin4Steps*();

        alpha = alpha + 0.157;

*HAL\_Delay*(10);

      }

*motorControlSens\_spin4Steps*();

      point.z = point.z + 0.2;

    }

    alpha = 0;

Pentru a obține coordonatele x și y putem folosi următoarele formule

        point.x = (19.0-sens\_readings.sensor\_data)\**sin*(alpha);

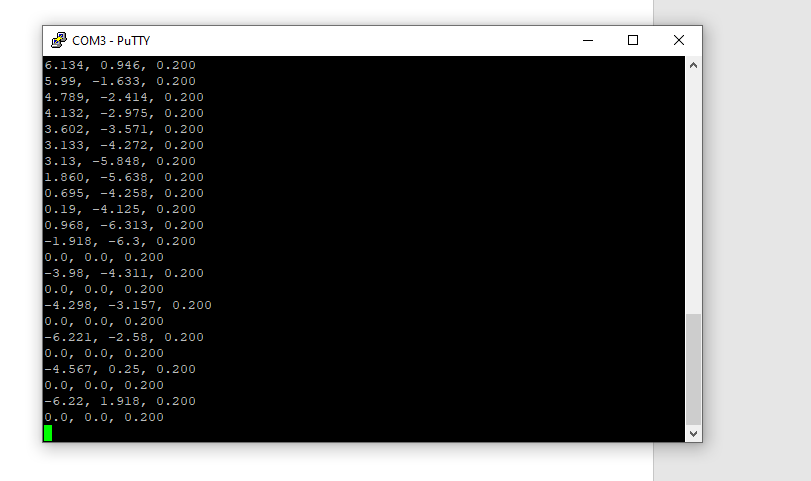
        point.y = (19.0-sens\_readings.sensor\_data)\**cos*(alpha);

unde 19.0 este dinstanța de la senzor până la centrul obiectului.

Pentru a obține coordonata z, după fiecare rotație completă

Deoarece motorul va ridica senzorul cu 0.2 cm după fiecare rotație completă a obiectului

Putem apoi salva datele intr-un fișier .log, folosind un monitor serial cum ar fi Putty



Figură 2‑18Datele primite prin serial monitor

Pentru a vizualiza point cloud-ul putem folosi un script de python

Text

Description automatically generated

Figură 2‑19Script de python pentru a vizualiza point cloud-ul

# Concluzii

În concluzie putem folosi acest scanner 3D pentru a crea point cloud-uri ce pot ulterior fi folosite pentru a reconstrui obiecte 3D.

# Planuri pentru o următoare versiune

1. Crearea unei arhitecturi software mai scalabile
2. Incastrarea sistemului intr-o carcasă
3. Găsirea unei soluții pentru a mișca senzorul în jurul obiectului pentru a putea păstra obiectul cât mai stabil
4. Realizarea unui cablaj imprimant pentru circuit electronic

# Bibliografie

*How to mechatronics*. Preluat de pe https://howtomechatronics.com/

*Sharp*.Preluat de pe https://global.sharp/products/device/lineup/selection/opto/haca/diagram.html

*STM32-base*.Preluat de pe https://stm32-base.org/boards/STM32F103C8T6-Blue-Pill.html

*Wikipedia*. Preluat de pe https://en.wikipedia.org/wiki/3D\_modeling

1. Modelarea 3D este procesul de creare a unui model matematic bazat pe coordonate pentru a reprezenta un obiect; [↑](#footnote-ref-1)
2. Un set de puncte discrete în spațiu ce pot reprezenta o formă sau un obiect; [↑](#footnote-ref-2)