

Facultatea de Automatică și Calculatoare

Calculatoare și Tehnologia Informației

**BRAT ROBOTIC**

**- DOCUMENTAȚIE -**

**Vezendan Alexandra Maria**

**Grupa 30226**

**Anul 2**

**2019-2020**

**1.Cerinta si obiectiv**

Sa se realizeze un proiect care sa simuleze un brat robotic controlat de o placut Arduino UNO prin intermediul a 3 potentiometre si ilustrat intr-o aplicatie OpenGL. Obiectivul acestui proiect consta in realizarea comunicatiei seriale intre programul in Arduino si aplicatie OpenGL din Visual Studio, pentru a prelua datele transmise de potentiometre sub forma unor grade si de a roti bratul.

**2.Analiza problemei/Proiectare**

In ceea ce priveste codul scris in arduino, acesta implica citirea unor date de pe 3 pini analogici A0,A2 si A4 si afisarea lor prin interfata seriala. Complexitatea proiectului consta insa in proiectarea bratului robotic care a fost lasata la latitudinea programatorului, asa ca am ales sa simulez ideea bratului prin intermediul a 3 paralelipipede, reprezentand bartul, antebratul si mana bartul, antebratul si mana, fiecare reprezentand cate un obiect separat creat in programul Blender si importat in scena din aplicatia OpenGL.

**3.Implementare**

**3.1. Resurse**

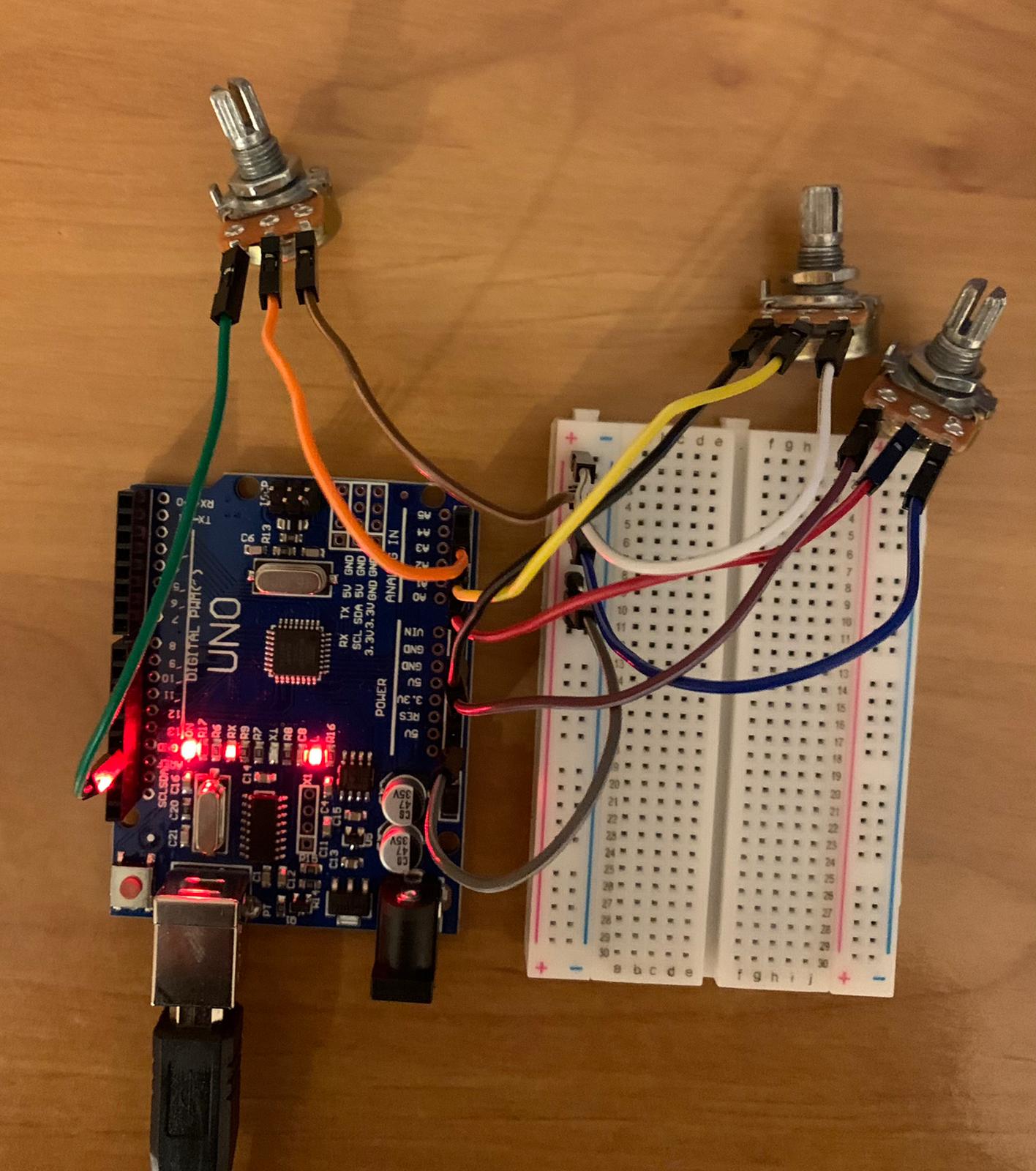
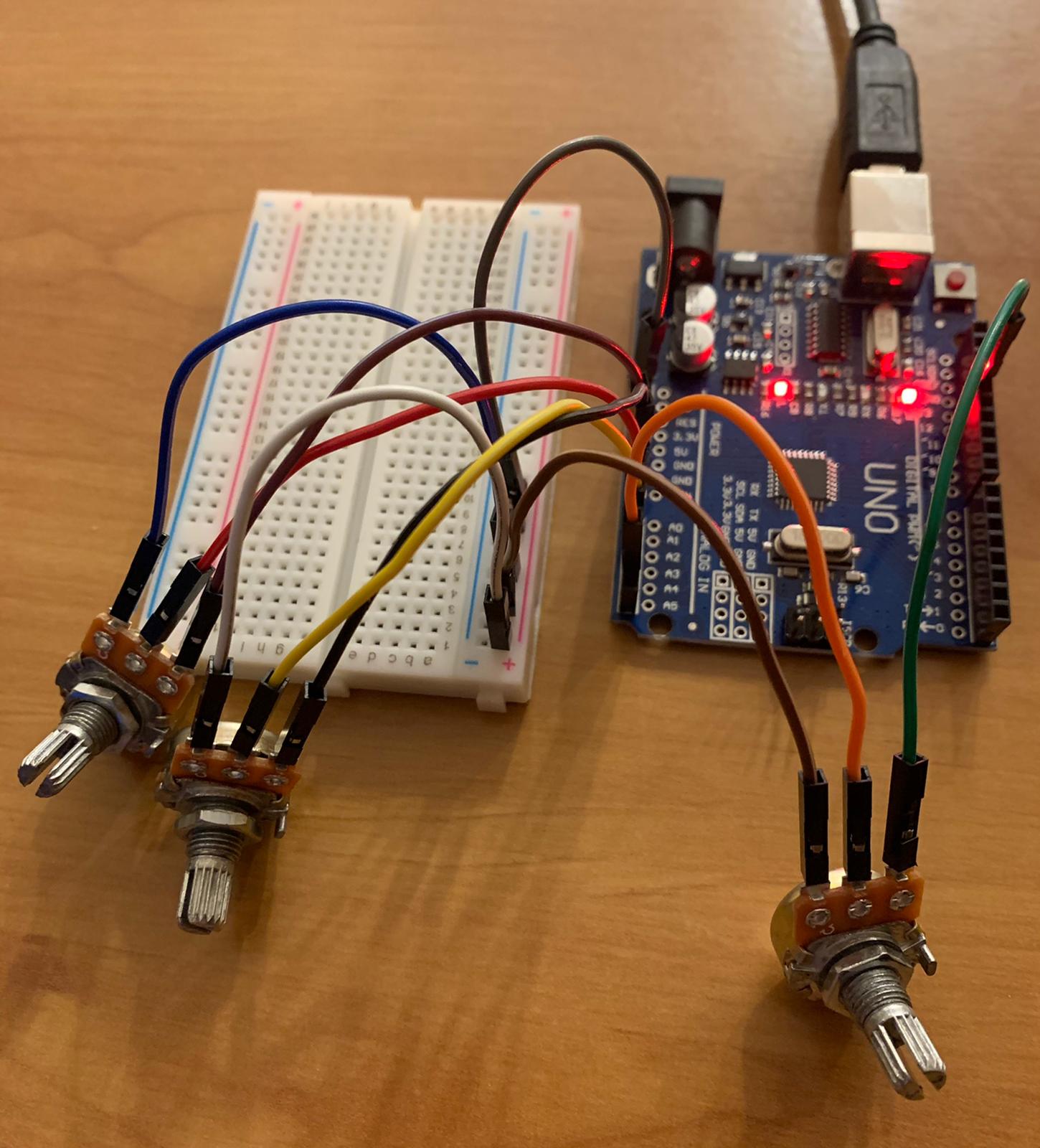
Resursele necesare implementarii fizice proiectului sunt reprezentate de placa de dezvoltare UNO R3 compatibil Arduino, 3 potentiometre liniare ,10 fire mama-tata si un breadbord.

**ARduino UNO R3**: este o placă de dezvoltare bazată pe microcontrollerul ATmega328. Are 14 pini de intrare/ieșire (dintre care 6 pot fi folosiți ca ieșiri PWM), 6 intrări analog, un oscilator de 16MHz, o conexiune USB, mufă de alimentare, și un buton de reset.

Poate fi alimentat direct de la calculator, de la portul USB, prin intermediul unei baterii de 9V sau a unui alimentator de 9V. Ceea ce ne intereseaza pe noi sunt intrarile analog ale placii, dintre care noi vom utiliza doar 3 pentru a citi datele venite de la potentiometrul liniar.

**3.2. Montaj**

In ceea ce priveste montajul, acesta este unul foarte simplu. Ne vom folosi de breadboard pentru a multiplica semnalul de Vcc pe care il conectam de la placa, printr-un fir, pe breadboard., astfel, de pe linia + pe verticala ne vom duce 3 semnale de Vcc la fiecare dintre cel 3 potentiometre pe care le folosim, pentru a le alimenta. Deoarece placa dispune de 3 pini de GND (exact cati avem nevoie), om conecta direct fiecare pin GND al placii cu pinul din stanga al potentiometrelor (acesta este pinul GND al lor, cel din dreapta este de Vcc). Dupa ce am facut alimentarea si conectarea la ground, ramane sa legam potentiometrele la pinii analogi ai placii, am ales pinii A0, A2 si A4, urmand ca de pe acesti pini sa citim serial datele.

**3.3.Aplicatia OpenGl**

Folosind bibliotecile glfw si glew simula un brat robotic prin intermediul platformei VisualStudio, brat alcatuit din 3 componente: partea de sus, bratul, partea de mijloc, antebratul si partea de jos, adica mana libera. Fiecare dintre aceste 3 componente este reprezentat de cate un paralelipiped scalat la o marime cat se poate de apropiata de cea naturala, obiecte pe care le-am creat in Blender, un program de modelare, sculptare, randare a obiectelor.

Pentru a importa obiectele in scena si a le pozitiona la locul dorit am efectuat o serie de transformari pe matricea model care va fi transmisa shaderelor prin intermediul unei variabile uniform. Astfel ca pozitia finala a unei componente a bratului robotic trebuie calculata relativ la segmentele precedente: mana libera se va roti cu unghiul dat de potentiometrul corespunzator si mai apoi isi va lua pozitia finala relativa la unghiurile antebratului si a bratului; antebratul se va roti cu unghiul corespunzator lui si dupa cu unghiul bratului, iar bratul este implementat cel mai usor deoarece se roteste doar cu unghiul lui.

Deoarece rotatiile se realizeaza in jurul originii, fiecare dintre cele 3 obiecte au fost pe rand translatate astfel incat in final sa fie pozitionate unul dupa altul, sub forma unui lant.

Pentru a putea sa ne deplasam in interiorul scenei, am implementat cateva functii de miscare ale camerei MOVE\_FORWARD, MOVE\_BACKWARD, MOVE\_RIGHT,MOVE\_LEFT pe carea le-am atasat unor butoane, si anume: left:A, right: D, forward:W,backward: S. Totodata, mouse-ul miscat in interiorul ferestrei de vizualizare va duce la miscarea acesteia, prin functia de mouseCallback.

Toata implementarea are la baza mai multe fisiere .cpp si .hpp/.h de care aplicatia se foloseste pentru a realiza miscarea camerei, incarcarea obiectelor si lucrul cu atributele varfurilor fiecarui obiect in parte.

Citirea datelor de la placa Arduino UNO se realizeaza prin intermediul unei comunicatii seriale intre cele doua dispozitive, placa si PC. Pentru acest lucru avem nevoie de un port caruia sa ii specificam portul PC-ului de la care va primi date si viteza de comunicatie in biti/secunda, in cazul nostru: COM13 este portul si viteza de comunicatie de 9600 biti/s. Apelam port.Open() pentru a deschide conexiunea si inainte de randarea scenei, avem nevoie sa ne obtinem valorile unghiurilor in format double.

La apelul functiei port.ReadLine(), ni se va stoca intr-o structura de tipul String^ datele trimise pe portul serial de catre placuta prin Serial.println(). Pentru a reusi sa obtin fiecare valoare in parte, dupa transmiterea valorii analogice citite am trimis si un ‘;’ pentru a putea face un split dupa acesta.

Deoarece tipul de date String^ nu mi-a fost cunoscut, am intampinat mici dificultati in convertirea datelor primite in double, insa functia System::Convert::ToDouble() m-a scos din incurcatura. A apelat aceasta functie de 3 ori, pe fiecare valoare rezultata in urma splitului facut. In acest moment, in variabilele a1,a2 si a3 avem valoarea unghiurilor corespunzatoare fiecarui potentiometru in parte.

**3.3.Fisierul .ino**

Functia de setup al acestui fisier contine doar apelul Serial.begin() care setează viteza de comunicație, in biți/secunda (baud rate), in cazul nostru, de 9600. Se mai poate adaugă un parametru opțional pentru configurare: câți biți sunt, paritatea si numărul biții de stop. Implicit sunt setate următoarele valori: 8 biti de date, no parity, one stop bit.

In interiorul functie loop() vom citi in variabile de tipul int, valorile de pe pinii analogici A0,A2 si A4. Folosim acesti pini deoarece noi avem nevoie sa detectam valori intermediare valorilor extreme. Cu ajutorul functiei map() vom re-mapa valoarea citita dintr-un interval, in altul. Valorile citite sunt in intervalul 0-1024 iar noi avem nevoie de valori intre 0-180 sa reprezinte gradele cu care se vor face rotatiile. Serial.println(value) va transmite valoarea finala prin interfata seriala catre aplicatia noastra din OpenGl pentru a putea fi procesata de aceasta.

**4.Concluzii**

Consider ca am reusit sa indeplindesc cerinta proiectului, dar desigur ca este mereu loc de mai bine. As fi putut imbunatati miscarile bratlui, astfel incat sa de poata roti pe toate cele 3 axe, dar pentru aceasta as fi avut nevoie de un obiect ce avea la dispozitie animatia si ideea de “incheietura”, astfel incat sa simuleze cat mai real miscarile omenesti.