
PROIECT DE DIPLOMĂ

FACULTATEA DE ELECTRONICĂ, TELECOMUNICAȚII
ȘI TEHNOLOGIA INFORMAȚIEI
UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FACULTATEA DE ELECTRONICĂ

TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

Specializarea:

Proiect de diplomă

Absolvent,



**Decan,
Prof.dr.ing. Ovidiu POP**

Președinte comisie,

2014

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA
FACULTATEA DE ELECTRONICĂ
TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

Departamentul

Titlul proiectului de diplomă:

Descrierea temei:

Locul de realizare:

Data emiterii temei:

Data predării temei:

Absolvent,



Director departament,

Conducător,

Declarație pe proprie răspundere privind autenticitatea lucrării de diplomă

Subsemnatul

legitimat cu seria nr. CNP

în calitate de autor al lucrării cu titlul

elaborată în vederea susținerii examenului de finalizare a studiilor de licență, la
Facultatea de Electronică, Telecomunicații și Tehnologia Informației
Specializarea
Universitatea Tehnică din Cluj-Napoca
sesiunea a anului

declar pe proprie răspundere că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării și în bibliografie.

Declar că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor. Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență/diplomă/disertație.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv anularea examenului de diplomă.

Nume și prenume

Data

Semnătura

Absolvent:

Conducător:

SINTEZA PROIECTULUI DE DIPLOMĂ

Avizul conducerii

Conducător,

Absolvent,



Cuprins

Cuprins	1
1 Summary	4
1.1 State of the Art	4
1.1.1 Antenna Positioning Systems	4
1.1.2 Types of Antennas	4
1.1.3 Advanced Materials and Manufacturing techniques	4
1.1.4 Categories of Antenna Positioning Systems	5
1.2 Theoretical Foundation	5
1.2.1 Antenna	5
1.2.2 Azimuth and Elevation angles	6
1.2.3 Design Considerations	6
1.2.4 Antenna Gain and Directivity	6
1.2.5 IoT-Enabled Control	6
1.2.6 Mechanical Design	7
1.2.7 3D Printing	7
1.2.8 Programming Languages	7
1.3 Implementation of the Adopted Solution	7
1.3.1 System Design Overview	7
1.3.2 Hardware Implementation	7
1.3.3 Software Implementation	8
1.3.4 System Integration	8
1.3.5 Motor Steps to Angle Conversion Formula and Vice Versa	8
1.3.6 Serial Ports and Communication	8
1.3.7 Web Interface Development	9
1.3.8 Accessing localhost from external sources	9
1.4 Experimental Results	10
1.5 Conclusions	10
2 Planificarea activităților	11
3 Stadiul actual	12
3.1 Sisteme de poziționare antene	12
3.2 Tipuri de antene	12
3.2.1 Clasificare antene în funcție de frecvență	13

3.2.2	Clasificare antene în funcție de polarizare	13
3.3	Categorii de sisteme de poziționare antene	13
3.3.1	Clasificare după tipul mișcării.....	14
3.3.2	Clasificare după metoda de control	14
3.4	Materiale avansate si tehnici de fabricație	15
3.5	Aplicațiile sistemelor de poziționare a antenelor	15
4	Fundamentarea teoretică.....	16
4.1	Antena.....	16
4.2	Caracteristicile antenei	16
4.2.1	Câștigul	16
4.2.2	Directivitatea	17
4.2.3	Eficiența antenei.....	17
4.2.4	Diagrama de radiație	17
4.3	Unghiul de elevație	18
4.4	Unghiul de azimut	18
4.5	Componente mecanice	19
4.5.1	Roți dințate	19
4.5.2	Raportul de transmisie (Gear Ratio).....	20
4.5.3	Rulmenți	20
4.5.4	Arbori.....	21
4.6	Arduino UNO	22
4.7	Motoare pas cu pas.....	23
4.8	Tehnologia IoT (Internet of Things).....	24
4.9	Imprimare 3D	25
4.10	Limbaje de programare pentru interfețe web	26
5	Implementarea soluției adoptate.....	27
5.1	Descrierea proiectului.....	27
5.2	Implementarea hardware	28
5.2.1	Roțile dințate	28
5.2.2	Realizarea și dimensionarea pieselor printate la imprimanta 3D	30
5.2.3	Motoarele pas cu pas NEMA17	32
5.2.4	Rulmenți și arbori	34
5.3	Implementarea software	35
5.3.1	Comunicarea prin port serial.....	35
5.3.2	Serverul Node.js	36
5.3.3	Interfața Web	38

5.3.4	Utilizarea ngrok pentru acces extern	39
5.3.5	Conversia din unghiuri în pași pentru motoare și viceversa	39
6	Rezultate experimentale	41
6.1	Redimensionarea componentelor printate	41
6.2	Implementarea procedurilor de detectare și tratare a erorilor interfeței web.....	42
6.3	Testarea și verificarea conexiunii seriale	43
6.4	Calibrarea motoarelor pas cu pas	44
6.5	Rezultatele testelor de precizie	44
6.6	Configurația finală	45
6.7	Îmbunătățiri și optimizări	46
7	Concluzii	47
8	Bibliografie	48
9	Anexe	49
9.1	Codul Arduino pentru motoarele pas cu pas	49
9.2	Codul HTML, CSS și JavaScript pentru interfața web	51
9.3	Codul JavaScript pentru serverul Node.js	56

1 Summary

1.1 State of the Art

The project builds on existing technologies and research in the field of antenna positioning systems. Current systems utilize various methods and technologies to gain precise control over antenna orientation. These systems often utilize advanced motorization techniques for accuracy and reliability, and are commonly managed by microcontrollers or other control units.

1.1.1 Antenna Positioning Systems

Antenna positioning systems are necessary for various applications, including telecommunications, satellite transmissions, and both military and civilian uses. These systems have evolved to meet the need for precise control and real-time monitoring of antenna orientation, ensuring efficient and reliable communication. In telecommunications, antenna positioning systems optimize signal quality and coverage in mobile networks and other wireless services.

1.1.2 Types of Antennas

Antennas convert electromagnetic signals between free space and an electrical circuit, granting the transmission or reception of radio, microwave, or light signals. Various types include:

- **Dipole Antennas:** A simple wire antenna divided into two equal parts.
- **Parabolic Antennas:** Use a parabolic reflector for satellite communication and radar.
- **Helical Antennas:** A spiral conductor used in communication and GPS.
- **Microstrip Patch Antennas:** A flat conductor on a dielectric substrate, used in wireless communication.
- **Horn Antennas:** Utilized for microwave applications and satellite communication.
- **Plasma Antennas:** Use plasma to generate and control signals.
- **MIMO Antennas:** Multiple input and output antennas to improve wireless network capacity and efficiency

1.1.3 Advanced Materials and Manufacturing techniques

- **Advanced Structural Materials:** The development of lightweight, high-strength structural materials is significant for the design of agile and robust antenna positioning systems. Advanced composites, nanomaterials, and lightweight alloys are being studied for their potential to reduce weight and increase structural integrity, enabling agile and efficient antenna positioning systems for aerospace, automotive, and maritime applications.
- **Additive Manufacturing for Positioning Mechanisms:** Additive manufacturing technologies, such as 3D printing, offer remarkable design flexibility and customization for the fabrication of complex positioning mechanisms. Researchers are exploring the use of additive manufacturing to produce lightweight, compact, and

customizable components for antenna azimuth and elevation actuators, gears, and linkages.

- **Flexible and Stretchable Materials for Conformal Positioning Systems:** Flexible and stretchable materials are enabling the development of conformal antenna positioning systems capable of adapting to curved surfaces and irregular geometries.
- **Smart Materials and Actuators:** Smart materials, such as shape memory alloys, electroactive polymers, and magnetostrictive materials, offer unique properties that can be used to develop adaptive and responsive positioning actuators.

1.1.4 Categories of Antenna Positioning Systems

Antenna positioning systems can be classified based on movement type, purpose, and control methods.

Classification by Type of Movement:

- Single-axis Systems: Rotate the antenna in a single plane (azimuth or elevation).
- Dual-axis Systems: Enable combined movements in azimuth and elevation.
- Multifunctional Systems: Integrate multiple axes for complex adjustments, used in advanced applications.
- Integrated Control Systems: Use computers and algorithms for automatic position adjustments.

Classification by Control Method:

- Manual Control Systems: Require human intervention for adjustments.
- Semi-automatic Control Systems: Partially automated but require some human input.
- Automatic Control Systems: Fully automated, using programmable controllers for real-time adjustments.
- Closed-loop Control Systems: Use sensor feedback for constant precision adjustments.
- Open-loop Control Systems: Follow a predetermined path without feedback verification.

1.2 Theoretical Foundation

The theoretical foundation explores the basic principles leading the design and operation of an antenna positioning system. It covers antenna theory, mechanical components, 3D printing for prototypes, and various programming languages used in implementing IoT-enabled control systems. Key focuses include precise control of azimuth and elevation angles.

1.2.1 Antenna

An antenna is an electrical device that converts electromagnetic signals between free space and an electrical circuit. It functions as an interface between electromagnetic waves propagating through the air and electronic circuits, thus enabling the transmission and reception of radio, microwave, or light signals. Antennas are used in various applications, including telecommunications, radar, navigation, and other communication systems. They are designed to efficiently emit and receive electromagnetic waves, ensuring reliable and effective communication. Figure 1 shows a parabolic antenna, an example of a directional antenna used in

signal transmission and reception applications. Parabolic antennas are known for their ability to focus signals on a specific point.

1.2.2 Azimuth and Elevation angles

The azimuth angle represents the horizontal orientation of the antenna, measured in degrees from a reference direction (typically true north). Precise control over the azimuth angle is essential for aligning the antenna with the signal source, especially in applications where the source may be located in different directions. The elevation angle represents the vertical orientation of the antenna, measured in degrees from the horizontal plane. Control over the elevation angle is crucial for ensuring the antenna is aligned with the signal source at different heights or altitudes. The design of the positioning system incorporates mechanisms to adjust both the azimuth and elevation angles, allowing for full 3D spatial orientation of the antenna. This dual-angle adjustment capability is vital for optimizing the antenna's performance in diverse environments.

1.2.3 Design Considerations

The design of the antenna positioning system incorporates the following considerations to accommodate high-gain and highly directive antennas:

- **Structural Integrity:** The mechanical components, including gears, mounts, and bearings, are designed to support the physical dimensions and weight of the antennas. This ensures stability and minimizes vibrations that could affect the precision of the positioning.
- **Precision Motors and Control:** Stepper motors are chosen for their ability to provide fine-grained control over the antenna's orientation in both azimuth and elevation angles.

1.2.4 Antenna Gain and Directivity

Antenna gain refers to the ability of the antenna to focus energy in a particular direction, enhancing signal strength in that direction. Directivity is a measure of how focused the antenna beam is. These concepts are crucial for optimizing signal reception and transmission, ensuring that the antenna is accurately aligned to maximize performance.

1.2.5 IoT-Enabled Control

The integration of IoT-based control mechanisms allows for remote adjustments of the antenna orientation. This is achieved through a central control unit that communicates with the antenna system via a web interface. The system is capable of real-time monitoring and adjustments, ensuring that the antenna maintains optimal alignment for the best signal reception and transmission.

1.2.6 Mechanical Design

The mechanical design of the system includes a set of gears, bearings, and shafts that translate the rotational motion of the motors into precise movements of the positioning system. The gears ensure smooth and consistent motion, while the bearings reduce friction and support the rotating elements.

1.2.7 3D Printing

3D printing is a manufacturing process in which material is deposited layer by layer to form a three-dimensional object. It is considered an additive process because the object is built from scratch, unlike subtractive processes where material is cut, drilled or milled. It is used in this project for 3D printing the structural components of the positioning system.

1.2.8 Programming Languages

Programming languages are essential tools for writing instructions that computers understand to perform tasks. Within this project, these languages like HTML for structuring web content, CSS for styling and layout, and JavaScript for adding interactivity are used to create a user-friendly web interface.

1.3 Implementation of the Adopted Solution

The implementation phase of the project involved translating the design concepts into a functional prototype. This included developing both the hardware and software components, integrating them into a cohesive system. The primary goal was to create a reliable, accurate, and user-friendly antenna positioning system controlled via a web interface.

1.3.1 System Design Overview

The system was designed to incorporate an Arduino Uno microcontroller to interface with stepper motors that control the azimuth and elevation angles of the antenna. The web interface allows users to input desired angles, which are then transmitted to the Arduino via a serial connection.

1.3.2 Hardware Implementation

The hardware consisted of stepper motors, gears, bearings, an Arduino Uno, and a 3D-printed structure to house and support the components. The mechanical design included a geared system to translate motor rotation into precise antenna movement. The components were assembled to ensure accurate and smooth operation.

1.3.3 Software Implementation

The software included the Arduino code for motor control, a Node.js server to handle web requests, and a user-friendly web interface. The communication between the web interface and Arduino was facilitated through serial communication.

1.3.4 System Integration

The hardware and software were integrated to function as a cohesive unit. The system was tested and calibrated to ensure accurate positioning of the antenna based on user inputs from the web interface.

1.3.5 Motor Steps to Angle Conversion Formula and Vice Versa

In the implementation of the antenna positioning system, precise control over the azimuth and elevation angles is crucial for accurate alignment of the antenna. To achieve this, the system relies on stepper motors to drive the movement of the antenna in both azimuth and elevation directions. To translate the desired angles into motor movements and vice versa, conversion formulas were utilized. These formulas play a fundamental role in the functioning of the system, allowing for seamless communication between the user input (in angles) and the motor control mechanism (in motor steps). The motor steps to angle conversion formulas enable the system to accurately determine the number of motor steps required to achieve a specific angle of rotation and vice versa. By establishing this relationship, the system can interpret user commands entered through the web interface and translate them into precise motor movements. Conversely, the angle to motor steps conversion formulas facilitate the translation of measured angles into corresponding motor steps, enabling the system to provide feedback on the current position of the antenna. This bidirectional conversion process ensures that the antenna positioning system operates with precision and reliability, meeting the requirements of various applications and scenarios.

1.3.6 Serial Ports and Communication

Serial communication is a fundamental method for data exchange between electronic devices, enabling reliable communication over wires or wireless connections. In the context of our antenna positioning system, serial communication plays a crucial role in facilitating communication between the web interface and the Arduino Uno microcontroller. The hardware setup for serial communication involves connecting the Arduino Uno to the host computer using a USB cable. This establishes a physical connection between the microcontroller and the computer, allowing data to be transmitted and received via the serial port. In our implementation, serial communication is handled in the Node.js environment using the `SerialPort.js` library. This library provides a simple and efficient way to interact with serial ports, allowing us to send commands from the Node.js server to the Arduino Uno and receive responses or sensor data back from the microcontroller. To establish a serial connection between the Node.js server and the Arduino Uno, we configure the serial port settings such as baud rate, data bits, parity, and stop

bits. We then open the serial port and handle any errors that may occur during the connection process. Bi-directional communication enables data to be exchanged between the Node.js server and the Arduino Uno in both directions. This allows us to send commands from the web interface to the Arduino Uno to control the antenna positioning system and receive real-time feedback or sensor data from the Arduino Uno back to the server for monitoring or further processing.

1.3.7 Web Interface Development

- **Frontend Development:** The frontend development process involves creating the user interface components and interactions using HTML(HyperText Markup Language), CSS (Cascading Style Sheets) , and JavaScript.
- **User Interface Design:** User interface design principles guide the development of the web interface, focusing on factors such as layout, color scheme, and navigation.
- **Integration with Backend:** The web interface interacts with the backend Node.js server to send commands to the Arduino Uno and receive real-time feedback or updates. This integration enables the communication between the frontend and backend components of the system, ensuring efficient control and monitoring of the antenna positioning system.
- **User Interaction:** Users interact with the web interface through various input methods such as buttons and input forms. The interface is designed to be intuitive and user-friendly, allowing users to easily control the antenna positioning system and access relevant information or settings.

1.3.8 Accessing localhost from external sources

Accessing a local server (localhost) from external sources is a common necessity for developers, especially during remote testing. The existence of methods that enable this is important to ensure that locally developed applications can be accessed and tested in different environments or by different people remotely. These methods range from network configuration to the use of tunneling services. One such method is port forwarding, which involves configuring the router to direct traffic from an external IP address and a specific port to a specific port on a computer within the local network. This is one of the most common methods for accessing a local server from external sources. Although effective, this method requires access to router settings and can expose the local server to security risks. Another method is represented by tunneling services, which offer a simple and secure way to access local servers via the internet. These services create a tunnel between the local computer and a public address, allowing access from external sources without modifying network settings. Additionally, using a VPN allows access to local resources through a secure remote connection. By configuring a VPN server, users can access the local server as if they were part of the same local network.

1.4 Experimental Results

In this section, I will present the measures I have taken and the tests I have conducted to ensure the correct and optimal functioning of the system. These measures include resizing the printed components to fit with other 3D components and more, verifying and testing the serial connection, handling web interface errors, checking data consistency, and verifying the accuracy of the angles movements through repeated tests. Additionally, I monitored the overall performance of the system, including response time and reliability. Furthermore, I conducted stress tests to evaluate the durability of the components under long-term operating conditions.

1.5 Conclusions

The development of this antenna positioning system highlights the crucial role of precise antenna alignment in improving signal reception and transmission. This system has showcased exceptional precision and reliability, achieved through the use of stepper motors, an Arduino Uno, and a user-friendly web interface. Future enhancements could aim at boosting the system's responsiveness and broadening its capabilities, such as incorporating real-time feedback mechanisms and advanced control algorithms. The primary challenges involved optimizing synchronization between hardware and software components and maintaining smooth communication via the serial port. These issues were addressed through iterative testing and refinement, resulting in effective solutions that ensured the system's stability and accuracy.

2 Planificarea activităților

Activitate	Data începerii	Data finalizării	Durata(zile)
Alegerea temei	02.10.2023	06.10.2023	6
Documentarea temei	09.10.2023	30.10.2023	22
Documentarea componentelor necesare	02.11.2023	04.12.2023	33
Achiziția componentelor	10.01.2024	12.03.2024	63
Implementarea practică	15.03.2024	02.05.2024	49
Implementarea software	15.03.2024	03.04.2024	20
Redactarea documentației	20.05.2024	30.06.2024	42

3 Stadiul actual

3.1 Sisteme de poziționare antene

Sistemele de poziționare a antenelor sunt esențiale pentru diverse aplicații, de la telecomunicații și transmisii satelitare, până la aplicații militare și civile. Evoluția acestor sisteme a fost determinată de necesitatea unui control mai precis și a unei monitorizări în timp real a orientării antenelor, pentru a asigura o comunicare eficientă și fiabilă. În industria telecomunicațiilor, sistemele de poziționare a antenelor sunt utilizate pentru a maximiza acoperirea și calitatea semnalului în rețelele de telefonie mobilă și în alte servicii de comunicații wireless. Aceste sisteme permit ajustarea continuă a orientării antenelor pentru a compensa interferențele și pentru a optimiza transferul de date în timp real. În domeniul militar, sistemele de poziționare a antenelor sunt esențiale pentru comunicațiile tactice și pentru supravegherea radar. Capacitatea de a controla și monitoriza poziția antenelor în timp real poate face diferența între succesul și eșecul unei operațiuni militare. În aplicațiile civile, cum ar fi monitorizarea mediului și tehnologia de navigație, sistemele de poziționare a antenelor sunt utilizate pentru colectarea datelor și pentru furnizarea informațiilor vitale în timp real.

3.2 Tipuri de antene

Antena este un dispozitiv electric care convertește semnalele electomagnetice între spațiul liber și un circuit electric sau o linie de transmisie și recepție. Aceasta acționează ca o interfață între unde electromagnetice propagându-se în spațiu și circuitele electronice, permitând transmiterea sau recepția semnalelor radio, microunde sau luminoase. Antenele pot fi clasificate în funcție de structură și modul de funcționare [1]. Câteva tipuri de antene sunt: antene dipol – un conductor drept împărțit în două părți egale, antene parabolice – un reflector parabolic, utilizat pentru comunicații prin satelit și radar, antene helicoidale – un conductor spiralat, utilizat în comunicații și navigație GPS, antene patch (microstrip) – un conductor plat pe substrat dielectric, utilizat în comunicații wireless și mobile, antene horn – utilizate pentru aplicații de microunde și satelit, antene plasma – care utilizează plasma pentru a genera și controla semnalele, și antene MIMO – multiple intrări și ieșiri, utilizate pentru a crește capacitatea și eficiența rețelelor wireless [1].

3.2.1 Clasificare antene în funcție de frecvență

- Antene VLF (Very Low Frequency) - Acestea sunt antene proiectate pentru frecvențe foarte joase, situate în intervalul 3 kHz - 30 kHz. Sunt utilizate în special în aplicații de comunicare subacvatică și în cercetările geofizice [1].
- Antene LF (Low Frequency) - Acestea sunt antene adaptate pentru frecvențe joase, în intervalul 30 kHz - 300 kHz. Sunt utilizate în principal pentru comunicații maritime și în alte aplicații specifice care necesită penetrarea eficientă a apei și a suprafețelor solide [1].
- Antene HF (High Frequency) - Acestea sunt antene optimizate pentru frecvențe înalte, între 3 MHz și 30 MHz. Sunt utilizate în comunicațiile pe distanțe lungi, inclusiv în radiodifuziunea internațională și în comunicațiile maritime și aeronaute [1].
- Antene VHF (Very High Frequency) - Acestea sunt antene proiectate pentru frecvențe foarte înalte, în intervalul 30 MHz - 300 MHz. Sunt utilizate în principal pentru comunicații terestre, radiodifuziune locală, radar și alte aplicații [1].
- Antene de microunde - Acestea sunt antene adaptate pentru frecvențe de microunde, situate în spectrul de la 1 GHz până la 300 GHz. Sunt utilizate într-o gamă largă de aplicații, inclusiv comunicații de bandă largă, radar, sisteme de televiziune prin satelit și alte tehnologii avansate [1].

3.2.2 Clasificare antene în funcție de polarizare

Polarizarea undelor electromagnetice este definită ca orientarea vectorului câmpului electric în spațiu în raport cu timpul [2]. Există trei tipuri de polarizare a undelor electromagnetice:

- **Polarizare verticală:** Când vectorul câmpului electric (E) al unei unde electromagnetice este perpendicular pe Pământ, unda electromagnetică este considerată polarizată vertical. Acest lucru înseamnă că câmpul electric oscilează într-un plan vertical, perpendicular pe suprafața Pământului [2].
- **Polarizare orizontală:** Când vectorul câmpului electric (E) al unei unde electromagnetice este paralel cu Pământul, unda electromagnetică este considerată polarizată orizontal. Acest lucru înseamnă că câmpul electric oscilează într-un plan orizontal, paralel cu suprafața Pământului [2].
- **Polarizare circulară:** Când câmpurile electric (E) și magnetic (H) ale unei unde electromagnetice au aceeași amplitudine și au o diferență de fază de 90 de grade, unda este considerată polarizată circular. Acest tip de polarizare înseamnă că vectorii câmpurilor electric și magnetic descriu un cerc în timp ce se propagă [2]

3.3 Categorii de sisteme de poziționare antene

Sistemele de poziționare a antenelor pot fi clasificate în funcție de mai mulți factori, inclusiv mecanismele de mișcare, scopul utilizării și coordonatele utilizate pentru orientare.

3.3.1 Clasificare după tipul mișcării

- Sisteme cu un singur ax: Permite rotirea antenei într-un singur plan (azimut sau elevație).
- Sisteme cu două axe: Permite mișcări combineate în azimut și elevație, oferind o acoperire sferică mai completă.
- Sisteme multifuncționale: Integrează mai multe axe pentru ajustări complexe și precise, adesea utilizate în aplicații avansate cum ar fi telescoapele radio sau comunicațiile satelitare.
- Sisteme de control integrat: Utilizează computere și algoritmi avansați pentru a monitoriza șiajusta automat poziția antenei pentru a menține o aliniere optimă [3].

3.3.2 Clasificare după metoda de control

- **Sisteme de control manual:** Aceste sisteme necesită intervenția umană pentru a ajusta poziția antenei. Operatorul poate folosi comenzi mecanice, cum ar fi roți sau manivele, pentru a schimba orientarea antenei. Aceste sisteme sunt adesea utilizate în aplicații unde schimbările frecvente de orientare nu sunt necesare și unde costul și simplitatea sunt priorități.
- **Sisteme de control semi-automat:** În aceste sisteme, anumite funcții sunt automatizate, dar este încă necesară o intervenție umană pentru anumite ajustări sau initializări. De exemplu, un sistem poate avea motoare controlate electric pentru mișcări de finețe, dar necesită ca un operator să stabilească poziția de bază sau să inițieze comenziile de mișcare.
- **Sisteme de control automat:** Aceste sisteme folosesc controlere programabile pentru a gestiona complet poziționarea antenei fără intervenție umană. Sistemele de control automat utilizează microcontrolere, PLC-uri (Controlere Logice Programabile), sau FPGA-uri (Matrici de Porturi Programabile în Câmp) pentru a procesa semnalele de la senzori și pentru a trimite comenzi către actuatoare. Aceste sisteme sunt capabile să efectueze ajustări precise în timp real, bazându-se pe algoritmi de control avansați, cum ar fi controlul PID (Proporțional-Integral-Derivat) sau metode de control adaptiv.
- **Sisteme de control prin buclă închisă:** Sistemele de control prin buclă închisă utilizează feedback de la senzori pentru a ajusta continuu poziția antenei, asigurând astfel o precizie ridicată. Aceste sisteme monitorizează constant poziția actuală a antenei și o compară cu poziția dorită, efectuând ajustări necesare pentru a corecta orice deviații. Sunt ideale pentru aplicații care necesită o acuratețe foarte mare și stabilitate în timp.
- **Sisteme de control prin buclă deschisă:** Sistemele de control prin buclă deschisă nu folosesc feedback pentru a ajusta poziția antenei. Odată ce o comandă de mișcare este dată, sistemul își urmează traseul prestabilit fără a verifica dacă poziția dorită a fost atinsă. Aceste sisteme sunt mai simple și mai puțin costisitoare, dar pot fi mai puțin precise decât cele cu buclă închisă.

3.4 Materiale avansate și tehnici de fabricație

- **Materiale Structurale Avansate:** Cercetarea asupra materialelor structurale ușoare și de înaltă rezistență, cum ar fi compozitele avansate, nanomaterialele și aliajele ușoare, este esențială pentru dezvoltarea sistemelor robuste de poziționare a antenelor. Aceste materiale sunt deosebit de utile în aplicații aerospatiale, auto și maritime datorită capacitații lor de a reduce greutatea și de a îmbunătăți integritatea structurală.
- **Fabricarea Aditivă pentru Mecanisme de Poziționare:** Tehnologiile de fabricație aditivă, cum ar fi imprimarea 3D, oferă o flexibilitate semnificativă de proiectare și personalizare pentru crearea mecanismelor complexe de poziționare. Acest lucru include fabricarea componentelor ușoare, compacte și personalizabile pentru actuatoare de azimut și elevație ale antenelor, angrenaje și articulații.
- **Materiale Flexibile și Extensibile pentru Sisteme de Poziționare:** Dezvoltarea materialelor flexibile și extensibile permite crearea sistemelor de poziționare conformale ale antenelor. Aceste sisteme se pot adapta la suprafețe curbe și geometrii neregulate, fiind benefice pentru aplicații care necesită integrare conformală și flexibilă.
- **Materiale și Actuatoare Inteligente:** Materialele inteligente, cum ar fi aliajele cu memorie de formă (SMA), polimerii electroactivi și materialele magnetostrictive, posedă proprietăți unice care sunt utile pentru dezvoltarea actuatoarelor adaptive. SMA-urile sunt utilizate în aplicații care necesită control precis și adaptabilitate, cum ar fi dispozitivele medicale și componentele aerospatiale. Aceste materiale pot fi activate de stimuli externi precum temperatura sau câmpurile magnetice, oferind funcționalitate și capacitați de integrare ridicate [4].

3.5 Aplicațiile sistemelor de poziționare a antenelor

- **Comunicații prin Satelit:** Urmărirea sateliților în orbite geostaționare, de Pământ mediu sau de Pământ scăzut.
- **Radiotelescoape:** Observarea obiectelor cerești.
- **Sisteme Radar:** Urmărirea aeronavelor, navelor sau altor obiecte.
- **Stații Terestre:** Comunicarea cu nave spațiale sau platforme aeriene.

4 Fundamentarea teoretică

4.1 Antena

O antenă este un dispozitiv electric care convertește semnalele electromagnetice între spațiul liber și un circuit electric. Aceasta funcționează ca o interfață între undele electromagnetice care se propagă prin aer și circuitele electronice, permitând astfel transmiterea și receptia semnalelor radio, microunde sau luminoase. Antenele sunt utilizate în diverse aplicații, inclusiv telecomunicații, radar, navigație, precum și alte sisteme de comunicare. Ele sunt proiectate pentru a emite și receptiona unde electromagnetice într-un mod eficient, asigurând o comunicare fiabilă și eficientă. În Figura 1, este prezentată o antenă parabolică, un exemplu de antenă direcțională utilizată în aplicații de transmisie și recepție de semnale. Antenele parabolice sunt cunoscute pentru capacitatea lor de a focaliza semnalele într-un punct specific.



Figura 1. Antenă parabolică (preluată din [1])

4.2 Caracteristicile antenei

Caracteristicile antenei sunt parametri care determină performanța și eficiența acesteia în transmiterea și receptia semnalelor electromagnetice. Aceste caracteristici sunt fundamentale în proiectarea și utilizarea antenelor pentru diverse aplicații de comunicații, de la telecomunicații, la sisteme radar și sateliți.

4.2.1 Câștigul

Câștigul unei antene, fără a lua în considerare eficiența, este definit ca raportul dintre intensitatea maximă de radiație într-o direcție dată și intensitatea maximă de radiație de la o antenă de referință produsă în aceeași direcție cu aceeași putere de intrare. Câștigul este definit ca fiind creșterea puterii semnalului pe măsură ce acesta este procesat de antenă pentru un unghi de incidentă dat. De obicei exprimat în decibeli, poate fi negativ. Ecuația generală pentru câștig este dată de ecuația (1) [1].

$$G = \eta \left(\frac{4\pi}{\lambda^2} \right) A_p \quad (1)$$

4.2.2 Directivitatea

Directivitatea unei antene este definită ca raportul dintre intensitatea maximă de radiație și intensitatea medie de radiație. Relația dintre directivitatea (D) și câștigul (G) antenei este dată de ecuația (2) [1]:

$$G = \eta D \quad (2)$$

4.2.3 Eficiența antenei

Eficiența unei antene este definită ca raportul dintre puterea radiată și puterea totală la intrare furnizată antenei. Ea este dată de ecuația (3) [1]:

$$\eta = \frac{\text{puterea radiată}}{\text{puterea totală la intrare}} \quad (3)$$

4.2.4 Diagrama de radiație

Diagrama de radiație a unei antene determină aria de acoperire a acesteia în spațiul liber. Diagrama de radiație a unei antene este ilustrată în Figura 2. Diagrama de radiație este necesară pentru înțelegerea performanței unei antene, deoarece oferă informații despre direcționalitatea și forma fasciculului emis [1]. Caracteristicile diagramă de radiație sunt utilizate pentru a optimiza plasarea și orientarea antenelor în aplicații practice.

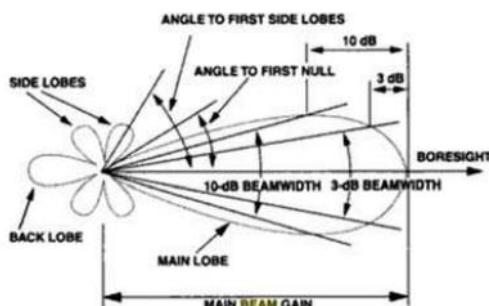


Figura 2. Parametrii antenei sunt definiți pe baza geometriei modelului de câștig al antenei (preluată din [1])

4.3 Unghiul de elevație

Unghiul de elevație este un parametru esențial în determinarea direcției unei antene în plan vertical. Este definit ca fiind unghiul dintre linia orizontală și linia de vizibilitate către un obiect observat, după cum reiese din Figura 3. Practic, acest unghi măsoară cât de sus sau jos față de orizont se află obiectul observat. Acest unghi este crucial pentru optimizarea performanței antenei, în special în aplicațiile de comunicații prin satelit și radar, unde antenele trebuie orientate cu precizie pentru a asigura o recepție și transmitere eficiente a semnalelor. Unghiul de elevație poate varia de la 0 grade, când antena este orientată direct pe orizontală, până la 90 de grade, când antena este orientată direct în sus. În practică, setarea corectă a unghiului de elevație permite antenei să maximizeze acoperirea și să minimizeze interferențele de la alte surse de semnal.

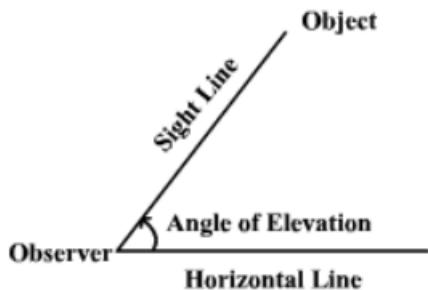


Figura 3. Unghiul de elevație (preluată din [10])

4.4 Unghiul de azimut

Unghiul de azimut este definit ca unghiul dintre direcția nordului geografic și linia de vizibilitate către un obiect observat, măsurat în plan orizontal, după cum este ilustrat în Figura 4. Acesta este un alt parametru esențial pentru orientarea antenelor, permitând determinarea direcției orizontale în care trebuie îndreptată antena. Azimutul este măsurat în grade, în sensul acelor de ceasornic, de la 0 la 360 de grade, acoperind întregul orizont. În combinație cu unghiul de elevație, azimutul permite o orientare tridimensională precisă a antenei.



Figura 4. Unghiul de azimut (preluată din [9])

4.5 Componente mecanice

Componentele mecanice constituie baza fizică a sistemelor de poziționare a antenelor, asigurând operarea precisă și corectă a acestora în diverse condiții. În această secțiune, se va analiza rolul crucial al fiecărei componente mecanice, de la rulmenți și arbori până la angrenaje și montaje, subliniind importanța lor în asigurarea performanței și fiabilității sistemelor de poziționare a antenelor.

4.5.1 Roți dințate

O roată dințată este o componentă circulară rotativă, dotată cu dinți în structura sa, folosită pentru a transfera cuplu și viteză de la un arbore la altul. Aceste dispozitive mecanice operează pe principiul avantajului mecanic, dinții asigurând transferul eficient al cuplului și prevenind alunecarea. Atunci când mai multe roți dințate sunt conectate și funcționează într-o succesiune, ele formează un sistem numit transmisie sau angrenaj, așa cum se poate observa în Figura 5. Roțile dințate sunt clasificate în funcție de poziția axelor și a perechilor de roți dințate. Există trei clasificări principale:

1. **Configurația Axelor paralele**: puterea este transmisă între axe paralele cu o eficiență de 98% până la 99,5%. Roțile dințate sunt montate pe arbori în același plan, iar roata condusă se mișcă în sens opus celei conduceătoare. Tipurile de roți dințate utilizate în această configurație includ roți dințate drepte, elicoidale, herringbone și dublu elicoidale. Această configurație oferă fiabilitate optimă, întreținere ușoară și necesită un număr minim de componente. Totuși, roțile dințate drepte pot genera zgomot și au limitări de cuplu, iar roțile dințate elicoidale generează o forță axială care necesită măsuri suplimentare pentru gestionare.
2. **Configurația Axelor intersectate**: axele se intersectează la un punct într-un plan comun, fiind utilizate pentru schimbarea cuplului și vitezei. Roțile dințate conice bevel sunt cele mai frecvent utilizate în acest tip de configurație. Acestea permit economisirea spațiului, reducerea uzurii și reducerea stresului asupra dinților. Cu toate acestea, aceste sisteme pot avea costuri mai mari și o capacitate de transmitere a cuplului mai mică decât configurațiile paralele, necesitând o aliniere precisă pentru o funcționare cât mai eficientă.
3. **Configurația axelor neparalele și neintersectante**: implică axe care nu sunt paralele și nu se intersectează. În acest caz, puterea este transmisă prin alunecare relativă între suprafețele dinților. Tipurile de angrenaje utilizate în această configurație includ angrenajele melcate. Principalul avantaj al acestei configurații constă în capacitatea de a permite transmiterea unui unghi de mișcare diferit, ceea ce este util în aplicații specifice. Însă, dezavantajele includ o eficiență redusă în transmiterea puterii și o complexitate mai mare în procesul de proiectare.

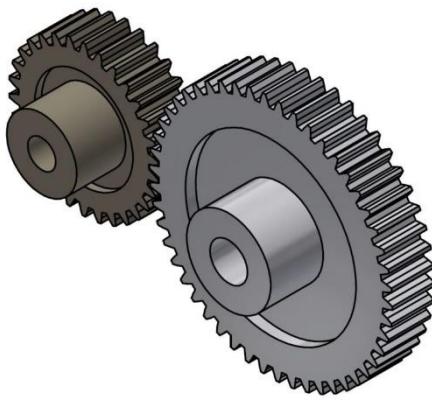


Figura 5. Angrenaj roți dințate
(preluată din [11])

4.5.2 Raportul de transmisie (Gear Ratio)

Formula raportului de transmisie este un instrument foarte important în ingineria mecanică, utilizat pe scară largă pentru proiectarea și analiza sistemelor de angrenare. Această expresie matematică furnizează o măsură obiectivă a eficienței sistemului, permitând inginerilor să calculeze și să optimizeze performanța transferului de putere într-un angrenaj. După cum se poate observa în ecuația (4), acest raport este exprimat pentru un angrenaj format din două roți dințate, dar acesta se poate modifica în funcție de numărul de componente.

$$\text{Raportul de transmisie} = \frac{\text{Numarul de dinti de pe roata condusa}}{\text{Numarul de dinti de pe roata conducatoare}} \quad (4)$$

4.5.3 Rulmenți

Rulmenții sunt componente esențiale care facilitează rotația obiectelor. Aceștia sprijină axul care se rotește în interiorul diferitelor mecanisme. Rulmenții sunt utilizati într-o gamă largă de aparate, inclusiv automobile, avioane, generatoare electrice și multe altele. Ei sunt folosiți chiar și în aparatelor electrocasnice pe care le folosim în viața de zi cu zi, cum ar fi figiderele, aspiratoarele sau aparatelor de aer condiționat. Rulmenții susțin arborii ai roților, turbinelor, rotoarelor și altor componente din aceste mașinării, permitându-le să se rotească mai ușor. Rulmenții îndeplinesc două funcții majore esențiale pentru funcționarea eficientă a mașinilor și echipamentelor:

- **Reducerea frecării și facilitarea rotației mai linie:** Frecarea este inevitabilă între arbore și suportul său. Pentru a diminua frecarea și a permite o rotație mai lină, se utilizează rulmenți între aceste două componente. Acest lucru reduce consumul de energie, constituind funcția principală a rulmenților.

- **Protejarea componentei de susținere și menținerea poziției corecte a arborelui:** Între arbore și componenta de susținere este necesară o forță considerabilă. Rulmenții joacă un rol crucial în prevenirea deteriorării acestei componente de susținere și asigură menținerea precisă a poziției arborelui. Aceasta permite mașinilor să funcționeze în mod repetat și durabil pe perioade îndelungate.

Rulmenții sunt alcătuși din mai multe componente principale, precum inele, corpuri de rostogolire, colivii, care se pot vedea formând rulmenții în Figura 6. Fiecare având un rol esențial în funcționarea eficientă a rulmenților:

- **Inelele interioare și exterioare:** De obicei, inelul interior este montat pe arbore, iar inelul exterior pe carcăsă. Aceste inele se pot găsi și sub denumirea de șaibe. Suprafața inelelor, pe care se deplasează elementele de rostogolire, este cunoscută sub numele de cale de rulare. Această suprafață poate avea forme variate, fiind fie sferică, fie plată, pentru a facilita mișcarea lină și eficientă a corpuri de rostogolire.
- **Elementele de rostogolire:** Aceste corpuri de rostogolire vin sub formă de bile sau role și sunt în direct contact cu inelele rulmentului prin intermediul căilor de rulare. Ele au rolul principal de a permite mișcarea fluidă între inelele rulmentului. Ele reduc în mod semnificativ frecarea și uzura dintre suprafețele de contact ale rulmentului, contribuind astfel la o funcționare mai lină, mai eficientă și mai durabilă a sistemului mecanic în care sunt integrate.
- **Coliviile:** Structuri care cuprind corpurile de rostogolire și mențin o distanță egală între ele, dar le și ghidează mișcarea pentru a evita contactul direct între acestea.



Figura 5. Rulmenți formați din inele, bile, colivii (preluată din [12])

4.5.4 Arbori

Un arbore mecanic este un element de transmisie a puterii și a mișcării rotative, de obicei cu secțiune transversală circulară, fie solid, fie gol pe interior. Arborii mecanici sunt componente esențiale în diverse mașini și echipamente, utilizate pentru a transmite cuplul și mișcarea rotativă de la un dispozitiv la altul. Pe arbori se montează diverse elemente mecanice precum roți dințate,

scripete, volante, ambreiaje și lanțuri pentru a transmite puterea de la dispozitivul de acționare, cum ar fi un motor sau un motor termic [5]. Un exemplu reprezentativ al unui arbore mecanic este arborele liniar, ilustrat în Figura 6. Arborii mecanici sunt clasificați în patru tipuri principale: **Arbore de transmisie** - este un component esențial al mașinilor, care asigură axa de rotație și oscilație și regleză geometria mișcării. **Axul (axul de roată)** - este o versiune non-rotativă a unui arbore, care susține scripetele și roțile rotative, dar nu transmite cuplu. Axul este o grindă statică și poate fi luat drept o grindă susținută. **Arborele de fus** - este un arbore rotativ cu un dispozitiv de fixare pentru reținerea unui instrument (sau a unei piese de prelucrat în cazul fusurilor de frezare, rectificare sau găurire). Fusul servește drept suport pentru instrument sau piesă de prelucrat, poziționare și acționare rotativă. **Arborele mașinii** - acești arbori sunt un element inherent al mașinii și se află în interiorul ansamblului. Un exemplu este arborele cotit al unui motor de automobil [5].



Figura 6. Arbori liniari (preluată din [13])

4.6 Arduino UNO

Placa Arduino UNO este una dintre plăcile standard ale familiei Arduino. A fost prima placă USB lansată de Arduino. Se bazează pe microcontrolerul ATmega328P. Placa include 20 de pini de intrare/ ieșire digitali și analogici (I/O) dintre care 14 sunt digitali iar 6 sunt analogici, conține un conector USB, un conector de alimentare și un header ICSP (In-Circuit Serial Programming). Această placă este programată utilizând un IDE (Integrated Development Environment), care poate fi utilizat atât online, cât și offline. Toate componentele acestei plăci electronice se pot vedea în Figura 7.

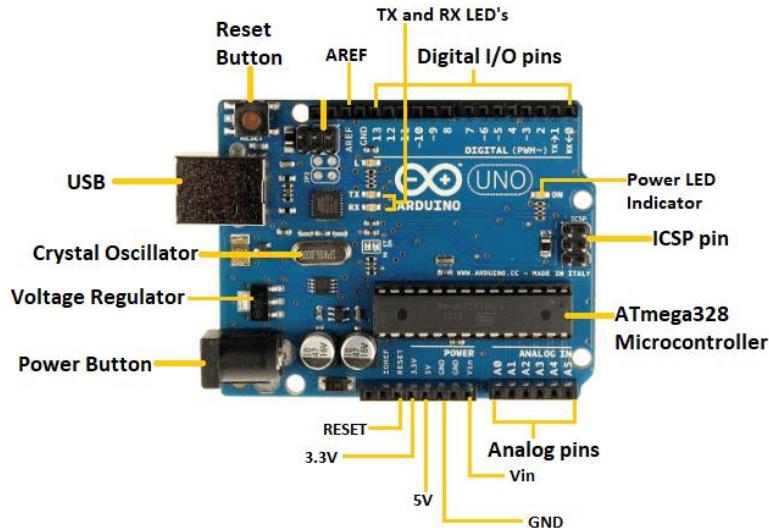


Figura 7. Placa Arduino Uno și componentele sale (preluată din [14])

4.7 Motoare pas cu pas

Un motor pas cu pas este un motor electric sincron fără perii care transformă impulsurile digitale în rotațiile unui arbore mecanic. Mișcarea obișnuită a arborelui constă în mișcări angulare discrete și de magnitudine uniformă, atunci când este alimentat de o sursă de curent continuu comutată secvențial. Motorul pas cu pas funcționează ca un dispozitiv digital de intrare-ieșire, fiind foarte eficient în aplicații unde semnalele de control sunt impulsuri digitale, nu tensiuni analogice. Fiecare impuls digital trimis către un driver sau un traductor al motorului produce o mișcare angulară precisă. Pe măsură ce frecvența acestor impulsuri crește, mișcarea pas cu pas devine o rotație continuă. Fiecare rotație completă a motorului pas cu pas este divizată într-un număr specific de pași discreți, de obicei 200, iar motorul necesită un impuls separat pentru a realiza fiecare pas. Motorul efectuează un singur pas la un moment dat, fiecare având aceeași dimensiune. Deoarece fiecare impuls provoacă o rotație exactă a motorului, de obicei de $1,8^\circ$, poziția acestuia poate fi controlată cu precizie fără a necesita un mecanism de feedback. Când frecvența impulsurilor digitale crește, mișcarea pas cu pas devine o rotație continuă, iar viteza de rotație este direct proporțională cu frecvența acestor impulsuri. Motoarele pas cu pas sunt utilizate frecvent în aplicații industriale și comerciale datorită costului redus, fiabilității ridicate, cuplului mare la viteze scăzute și construcției simple și robuste. Un exemplu de acest gen de motor poate fi găsit în Figura 8. Limitările acestui tip de motor includ:

- **Eficiență redusă:** motoarele pas cu pas consumă o cantitate constantă de curent, indiferent de sarcină. Ele trag cel mai mult curent atunci când sunt inactive, ceea ce duce la o generare mai mare de căldură în comparație cu motoarele DC, făcându-le astfel mai puțin eficiente în general [6].
- **Cuplu limitat la viteza mare:** motoarele pas cu pas au un cuplu mai scăzut la viteze mari în comparație cu vitezele mici. Deși anumite motoare pas cu pas sunt concepute pentru a funcționa mai eficient la viteze mari, acestea necesită un driver adecvat pentru a obține performanțe optime [6].
- **Lipsa feedback-ului:** Spre deosebire de motoarele servo, majoritatea motoarelor pas cu pas nu au mecanisme integrate de feedback pentru monitorizarea poziției. Deși controlul precis poate fi realizat într-o configurație în buclă deschisă, componente suplimentare,

cum ar fi întrerupătoare de limită sau detectoare de poziție inițială, sunt adesea necesare pentru a asigura siguranța și pentru a stabili o poziție de referință [6].

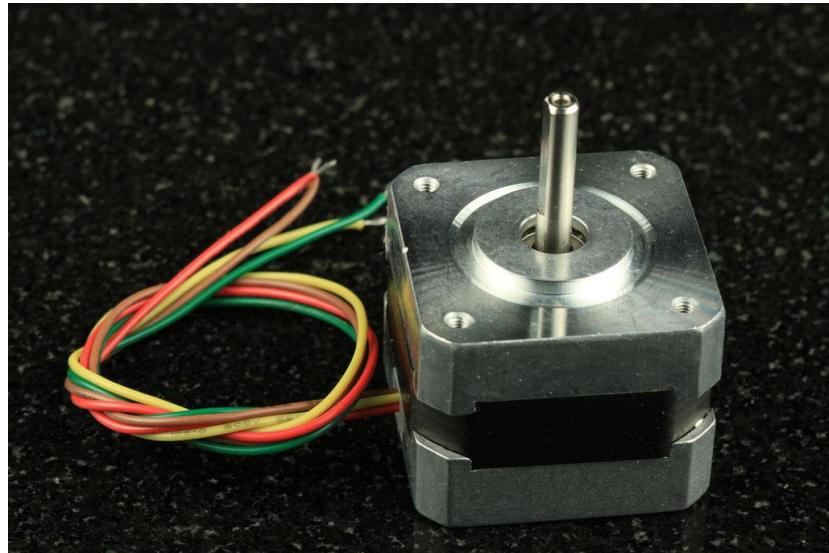


Figura 8. Motor pas cu pas (preluată din [6])

4.8 Tehnologia IoT (Internet of Things)

Internetul Obiectelor, sau IoT, se referă la rețeaua globală de dispozitive fizice conectate la internet care colectează și distribuie date. Datorită dezvoltării rapide a microcipurilor ieftine și a rețelelor wireless omniprezente, aproape orice obiect, de la cele mai mici dimensiuni până la cele mai mari, de exemplu, un avion, poate fi integrat în IoT. Prin dotarea acestor obiecte cu senzori și capacitați de conectivitate, acestea pot transmite și primi date în timp real fără intervenție umană. IoT transformă obiectele obișnuite în dispozitive inteligente, capabile să comunice și să reacționeze la mediu, făcând astfel lumea mai interconectată și mai eficientă. Acest lucru îmbină universurile digital și fizic, permitând o gamă largă de aplicații inovatoare și sporind considerabil eficiența operațională în diverse domenii [7]. IoT este format din mai multe elemente esențiale, printre care:

- **Dispozitive și senzori:** Acestea sunt elementele fizice care adună date din mediul înconjurător. Senzorii pot monitoriza temperatură, umiditate, lumină, mișcare și alți parametri.
- **Conecțitate:** Datele culese de senzori sunt transmise prin rețele wireless sau prin cabluri către alte dispozitive sau către o platformă centralizată.
- **Platformă de date:** Aici, datele colectate sunt stocate, procesate și analizate. Platformele pot fi locale sau se pot afla în cloud.
- **Interfață utilizator:** Utilizatorii interacționează cu sistemele IoT prin aplicații mobile, interfețe de bord web sau alte interfețe digitale.

IoT are aplicații diverse, inclusiv:

- **Case inteligente:** Dispozitivele conectate din locuințe permit controlul la distanță al luminilor, termostatelor, sistemelor de securitate și electrocasnicelor.

- **Sănătate:** Monitorizarea pacienților în timp real prin dispozitive care colectează date despre semnele vitale și le transmit doctorilor.
- **Industria:** Automatizarea proceselor de producție și mențenanță a echipamentelor prin monitorizarea condițiilor de operare.
- **Orașe inteligente:** Gestionarea traficului, iluminării stradale, colectării deșeurilor și alte servicii urbane prin utilizarea datelor colectate de senzori.

Deși IoT oferă numeroase beneficii, există și provocări semnificative, în ceea ce privește securitatea și confidențialitatea datelor:

- **Securitatea datelor:** Dispozitivele IoT sunt predispuse la atacuri cibernetice. Fără măsuri de securitate adecvate, acestea pot fi compromise, ducând la pierderi de date sau control neautorizat asupra dispozitivelor.
- **Confidențialitate:** Colectarea masivă de date impune probleme legate de confidențialitate. Este crucial ca utilizatorii să fie informați despre modul în care datele lor sunt colectate, stocate și utilizate.
- **Interoperabilitate:** Diversele dispozitive și platforme IoT trebuie să fie capabile să comunice între ele pentru a crea sisteme funcționale și integrate. Lipsa standardelor comune poate cauza probleme de compatibilitate.

4.9 Imprimare 3D

La cel mai elementar nivel, imprimarea 3D este un proces de fabricație în care materialul este depus strat cu strat pentru a forma un obiect tridimensional. Acesta este considerat un proces aditiv, deoarece obiectul este construit de la zero, spre deosebire de procesele subtractive în care materialul este tăiat, forat, frezat sau prelucrat. O astfel de imprimantă 3D este evidențiată în Figura 9. Deși imprimantele 3D utilizează o varietate de materiale (precum plastic sau metal) și tehnici, ele au în comun capacitatea de a transforma fișiere digitale care conțin date tridimensionale — fie create într-un program de proiectare asistată de calculator (CAD), fie dintr-un scanner 3D — în obiecte fizice [8]. Imprimantele 3D folosesc o varietate de tehnologii. Cea mai cunoscută este modelarea prin depunere fuzionată (FDM), cunoscută și sub numele de fabricație prin filament fuzionat (FFF). În acest proces, un filament din ABS (acrilonitril butadien stiren), PLA (acid polilactic) sau alt termoplastice este topit și depus printr-o duză de extrudare încălzită, strat cu strat. Primele imprimante 3D, lansate pe piață la mijlocul anilor 1990 de către Stratasys cu ajutorul IBM, au folosit FDM (termen marca Stratasys), la fel ca majoritatea imprimantelor 3D destinate consumatorilor. O altă tehnologie utilizată în imprimarea 3D este stereolitografia. În acest proces, un laser UV este direcționat într-un rezervor de fotopolimer sensibil la ultraviolete, trasând obiectul care urmează să fie creat pe suprafața sa. Polimerul se solidifică oriunde este atins de fasciculul laser, iar fasciculul „printează” obiectul strat cu strat conform instrucțiunilor din fișierul CAD. O variație a acestei tehnologii este imprimarea 3D cu proiecție de lumină digitală (DLP). Această metodă expune un polimer lichid la lumina de la un proiectoare de procesare digitală a luminii. Polimerul se întărește strat cu strat până când obiectul este construit, iar polimerul lichid rămas este drenat. Modelarea cu jet multiplu este un sistem de imprimare 3D asemănător cu cel al imprimantelor cu jet de cerneală, care pulverizează un liant colorat, asemănător unui adeziv, pe straturi succesive de pulbere unde urmează să fie format obiectul. Aceasta este una dintre cele mai rapide metode și una dintre puținele care suportă imprimarea multicoloră [8].

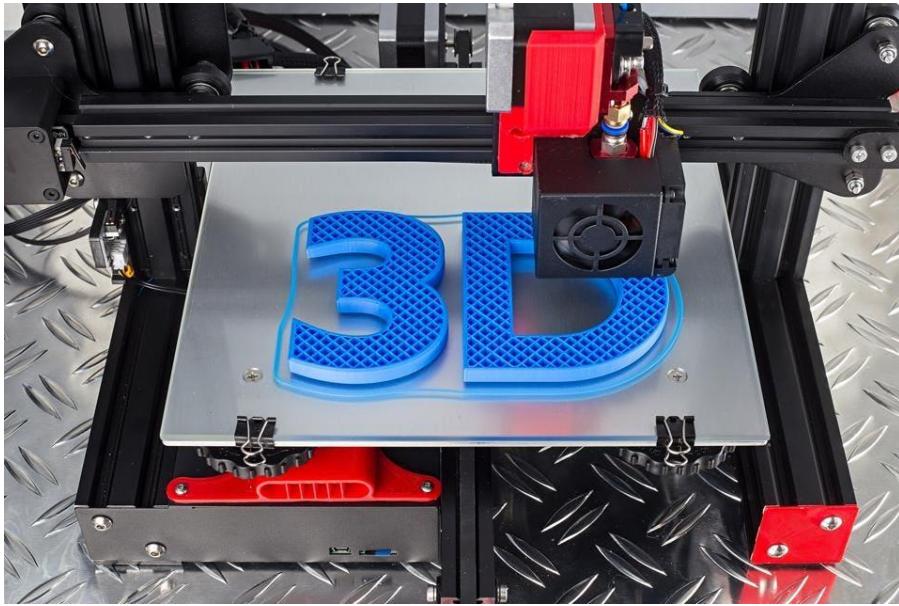


Figura 9. Imprimantă 3D cu rezultatul printat (preluată din [15])

4.10 Limbaje de programare pentru interfețe web

În dezvoltarea interfețelor web, sunt utilizate mai multe limbaje de programare fundamentale pentru structurarea, stilizarea și adăugarea interacțiunilor paginilor web. HTML este limbajul de programare folosit pentru a structura conținutul unei pagini web. Prin intermediul său, dezvoltatorii definesc elemente precum titluri, paragrafe, butoane, câmpuri de input, câmpuri de text sau imagini în cadrul paginii. HTML este fundamentalul pe care se construiește orice pagină web. CSS este folosit pentru a stiliza aspectul și prezentarea elementelor HTML. Prin intermediul său, se definesc proprietăți precum culori, fonturi, dimensiuni, margini și altele. CSS permite o personalizare flexibilă a aspectului vizual al unei pagini web, facilitând crearea unui design atractiv și coerent. JavaScript este limbajul de programare folosit pentru a adăuga interactivitate paginilor web. Aceasta permite dezvoltatorilor să manipuleze dinamic elementele HTML, să răspundă la acțiunile utilizatorilor (cum ar fi click-urile de mouse sau completarea formularelor) și să actualizeze conținutul paginii fără a necesita o încărcare completă a acesteia. JavaScript este esențial pentru crearea aplicațiilor web interactive și dinamice. Aceste limbaje de programare sunt fundamentale în dezvoltarea oricărui sistem de interfață web, oferind instrumentele necesare pentru a construi pagini web bine structurate, atrăgătoare și funcționale. Fiecare limbaj își aduce aportul diferit în procesul de dezvoltare, contribuind la crearea unei experiențe de utilizare plăcute și eficiente pentru utilizatori.

5 Implementarea soluției adoptate

5.1 Descrierea proiectului

Proiectul presupune implementarea unui sistem de poziționare pentru o antenă bazat pe unghiurile de elevație și de azimut. Soluția adoptată optează pentru utilizarea unui sistem care se bazează pe îmbinarea componentelor hardware cu partea de control software. Implementarea hardware presupune utilizarea unui motor pas cu pas pentru poziționarea pe fiecare unghi în parte. De asemenea, aceste motoare pas cu pas vor fi conectate la un angrenaj format din roți dințate care vor genera mișcarea propriu-zisă a sistemului pe cele două unghiuri. Pentru a facilita rotirea se utilizează rulmenți și arbori rotativi. Structura acestui sistem este realizată folosind componente imprimate 3D, inclusiv roțile dințate, blocuri de susținere, blocuri de prindere și alte componente necesare stabilității sistemului. Partea de control software se realizează cu ajutorul unei plăcuțe Arduino UNO care controlează motoarele pas cu pas, iar ea la rândul ei este conectată prin port serial cu o interfață web implementată cu ajutorul mai multor tehnologii software. La finalul implementării, scopul este ca sistemul să fie complet funcțional, să îndeplinească cerințele și funcțiile sale, precum și să fie ușor de utilizat și de întreținut. Toată această conexiune între tehnologii și acest sistem este prezentat vizual în diagrama din Figura 10.

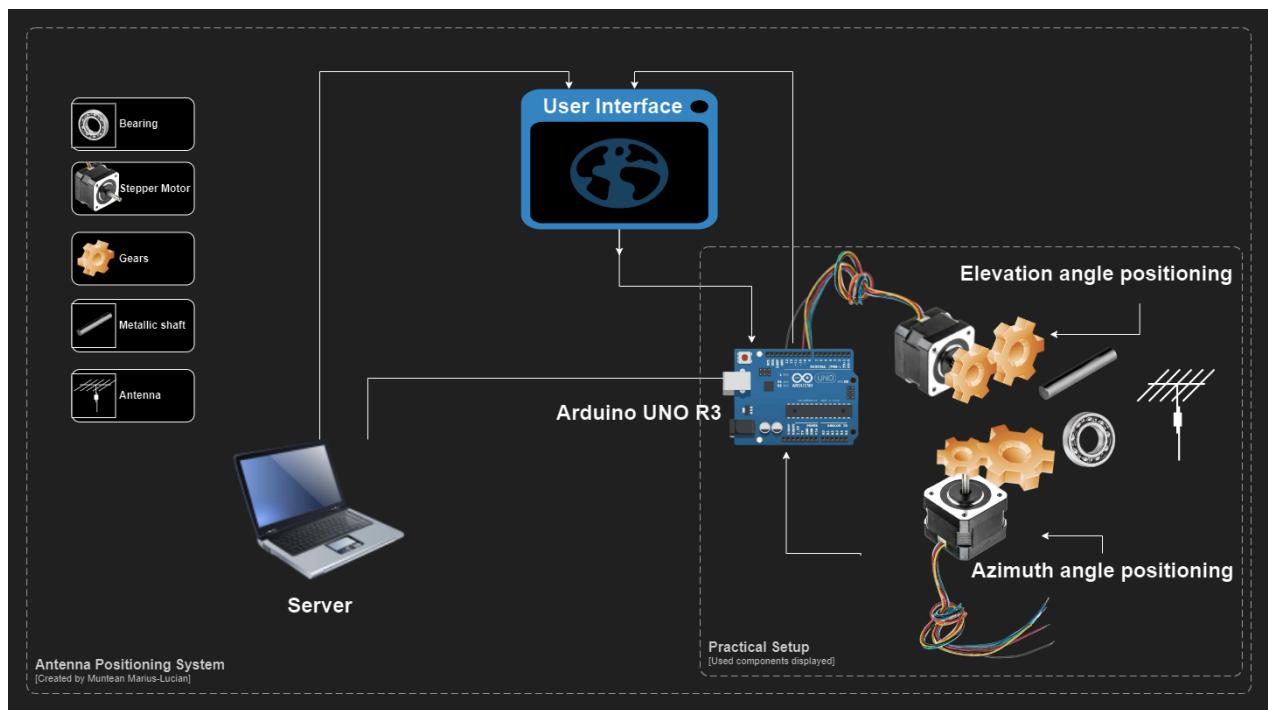


Figura 10. Diagrama de conexiuni și componente ale proiectului

5.2 Implementarea hardware

Partea hardware constă în realizarea structurii propriu-zise a sistemului, începând de la gândirea și proiectarea acesteia. În faza de concepție, a fost realizată o schiță care să ilustreze viziunea asupra aspectului sistemului, având ca obiectiv principal funcționalitatea, utilizând ca parametri esențiali unghurile de azimut și de elevație. Schița poate fi vizualizată în Figura 11, oferind o bază pentru dezvoltarea ulterioară proiectului. Următorul pas a fost identificarea și evaluarea componentelor esențiale necesare pentru a construi sistemul de la zero. Această etapă a implicat selectarea fiecărei piese în funcție de specificațiile tehnice și de compatibilitatea cu proiectul.

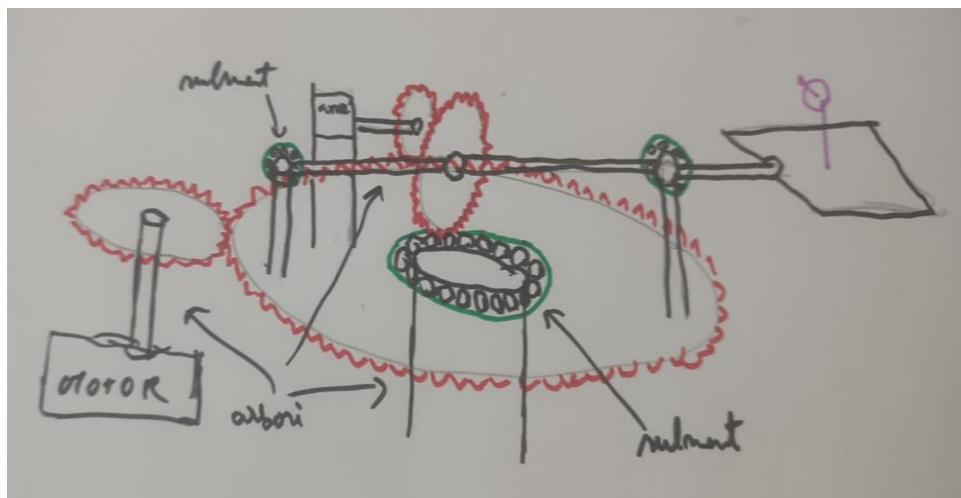


Figura 11. Schiță prototip sistem de poziționare

5.2.1 Roțile dințate

Roțile dințate reprezintă cea mai importantă și complexă componentă a sistemului, esențială pentru funcționarea corectă și eficientă a mecanismului. Alegerea roților dințate a fost făcută datorită capacitații lor de a transfera cuplu și mișcare între componentele sistemului, asigurând astfel sincronizarea necesară pentru poziționarea antenei pe unghurile de elevație și azimut. Precizia în proiectarea și realizarea roților dințate este foarte importantă pentru a reduce pierderile de energie și pentru a maximiza eficiența operațională. Am utilizat OpenSCAD pentru a crea roțile dințate necesare proiectului. În cadrul acestui software, am scris un cod care definește parametrii pentru dinții roților, cum ar fi pasul, înălțimea dinților și unghiul pantei dinților. Acestea sunt necesare pentru a asigura un angrenaj precis și eficient. Codul creează un model 3D al roților dințate, pe care l-am imprimat ulterior folosind o imprimantă 3D. Codul definește mai întâi parametrii inițiali, inclusiv toleranțele și diametrele pentru arborii utilizați. Apoi, se stabilesc parametrii dinților, cum ar fi pasul (distanța dintre doi dinți consecutivi), înălțimea dinților și unghiul pantei acestora. Un factor de ajustare este utilizat pentru a asigura îmbinarea corespunzătoare a dinților. Punctele de contur ale unui dintă sunt definite prin coordonate specifice, care conturează forma fiecărui dintă. Funcțiile DiametruInterior și OrientareDinți sunt utilizate pentru a calcula diametrul interior al roții și unghiul de rotație pentru fiecare dintă, pe baza numărului total de dinți. Modulul Roata2D generează profilul 2D al unei roți dințate, calculând diametrul interior și plasând fiecare dintă în poziția corectă prin rotire și translație. De asemenea, desenează un cerc pentru marginea interioară a roții. Modulul RoataDințiDePrintat

extrudează acest profil 2D pentru a crea un model 3D al roții și decupează un cilindru în centru pentru a forma gaura pentru arbore. În final, modulul ModelImbinare creează un angrenaj compus din două roți dințate, poziționându-le și rotindu-le corespunzător pentru a se îmbina corect. Rezultatul se poate vedea în Figura 12.

```

Suprapunere = 0.01;
Ax3mm = 3.2;
Ax4mm = 10;
Ax5mm = 5.2;

// Parametri dinților
Pas = 2.0;
ÎnălțimeDinți = 1.5;
UnghiPantaDinți = 75;
FactorImbinare = Pas / 20;

LățimePanta = ÎnălțimeDinți / tan(UnghiPantaDinți);
LățimeDinți = Pas / 2 - LățimePanta - FactorImbinare;

P0 = [0, 0];
P1 = [LățimePanta, ÎnălțimeDinți];
P2 = [Pas / 2, ÎnălțimeDinți];
P3 = [Pas / 2 + LățimePanta, 0];

/* Parametri: (Numărul de dinți, Grosimea roții, Diametrul axului interior) */
RoataDinți3DDePrintat(40, 5, Ax4mm);

function DiametruInterior(NumarDinti) = (NumarDinti * Pas) / PI;
function OrientareDinți(NumarDinti) = 360 / NumarDinti;

module Roata2D(NumarDinti) {
    DiametruInteriorRoata = DiametruInterior(NumarDinti);

    for (a = [1:NumarDinti]) {
        rotate([0, 0, OrientareDinți(NumarDinti) * a]) {
            translate([-Pas / 4 - FactorImbinare / 2, DiametruInteriorRoata / 2]) polygon([P0, P1, P2, P3]);
        }
    }
    circle(DiametruInteriorRoata / 2 + FactorImbinare, $fn = NumarDinti * 4);
}

module RoataDințiDePrintat(NumarDinti, GrosimeRoata, DiametruAx) {
    difference() {
        linear_extrude(GrosimeRoata) Roata2D(NumarDinti);
        translate([0, 0, -Suprapunere]) cylinder(d = DiametruAx, h = GrosimeRoata + 2 * Suprapunere, $fn = 36);
    }
}

module ModelImbinare() {
    RoataDințiDePrintat(80, 5, Ax5mm);
    translate([(DiametruInterior(80) + DiametruInterior(40) + 2.5 * ÎnălțimeDinți) / 2, 0, 0]) {
        rotate([0, 0, (OrientareDinți(20)) / 1.5]) RoataDințiDePrintat(40, 5, Ax5mm);
    }
}
ModelImbinare();

```

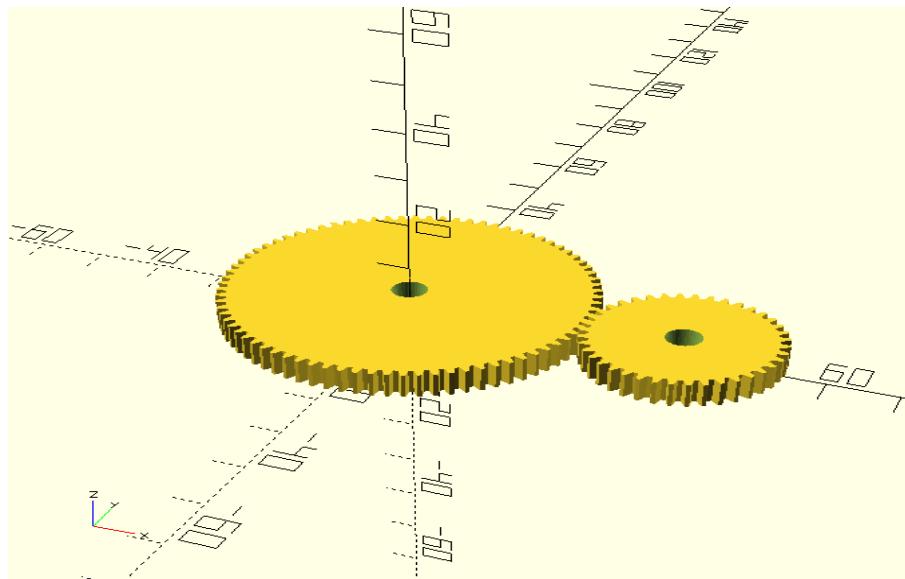


Figura 12. Rezultatul randării 3D al codului reprezentând două roți dințate îmbinante

A fost ales acest tip de configurație a angrenajului (configurație cu axe paralele), deoarece implică un număr redus de componente, este caracterizat prin fiabilitate optimă și este ușor de întreținut.

5.2.2 Realizarea și dimensionarea pieselor printate la imprimanta 3D

Piese de susținere, piesele de prindere și roțile dințate, precum și platforma antenei, au fost proiectate și dimensionate în OpenSCAD și pregătite pentru imprimare în UltiMaker Cura. Dimensiunile acestor piese au fost ajustate pentru a se potrivi cu specificațiile roților dințate, cu accent pe asigurarea funcționării corecte, îmbinarea eficientă a componentelor și garantarea rezistenței și durabilității sistemului. În procesul de proiectare în OpenSCAD, am luat în considerare îmbinarea pieselor care trebuie printate, cu cele care asigură rotația, fluiditatea și mișcarea, cum ar fi motoarele pas cu pas, rulmenții și arborii. Astfel, dimensionarea fiecărei piese s-a realizat prin aplicarea unor măsurători constante în raport cu celelalte piese, astfel garantând interacțiunea corectă între toate elementele sistemului. Interfața de lucru în OpenSCAD este foarte practică, permitând definirea fiecărei componente prin intermediul codului. Aceasta oferă un control precis asupra geometriei și dimensiunilor fiecărei piese, permitând ajustări rapide și repetabile. Procesul de modelare se bazează pe definirea formelor și a structurilor utilizând operații de baza precum extrudarea, rotația și tăierea. În Figura 13 se poate observa interfața acestui software foarte util, care conține funcții de previzualizare, randare, imprimare, precum și funcție de export STL (stereolitografie).

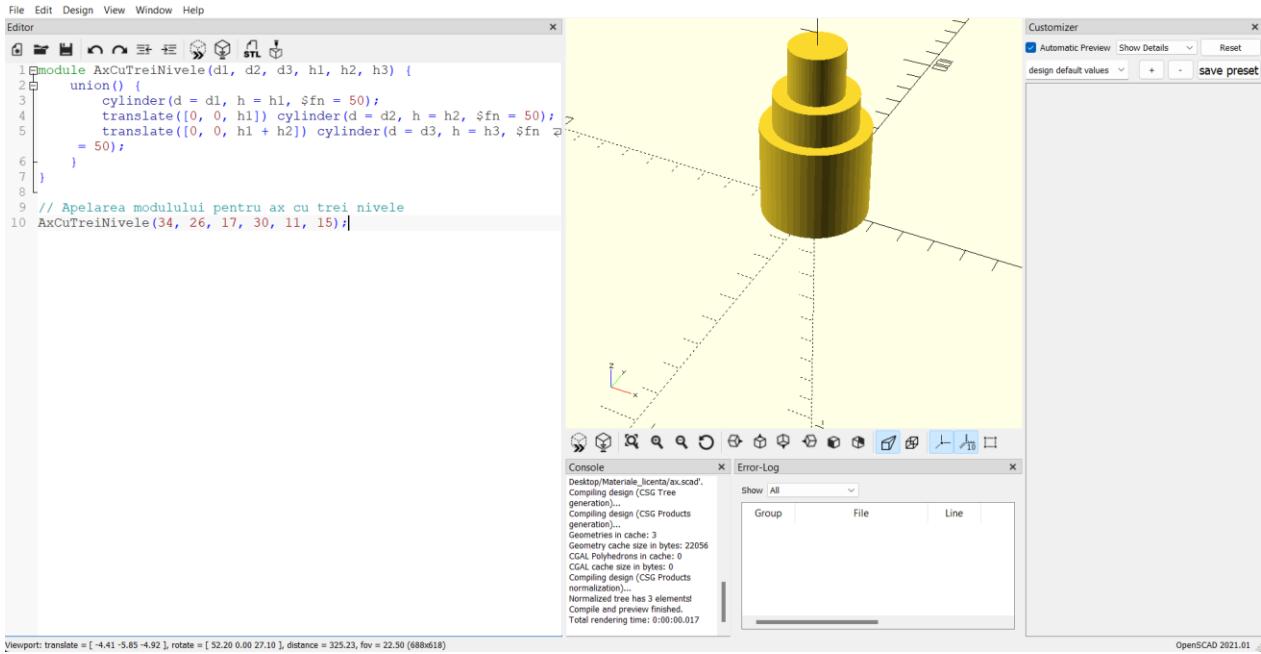


Figura 13. Interfața software-ului OpenSCAD

Pe lângă proiectarea digitală, imprimarea 3D a permis crearea acestor componente cu o precizie foarte mare. Fiecare piesă a fost testată pentru a asigura potrivirea și funcționarea corectă în ansamblu. Materialul utilizat pentru imprimare este PLA (acid polilactic), un material durabil și ușor, ideal pentru acest sistem. În Figura 14 sunt prezentate imagini ale pieselor printate, incluzând blocuri de prindere, platforma pentru antenă, blocuri de sușinere, arborele care suține întreaga structură, precum și lagările pentru rulmenți, care asigură fixarea acestora și permit rotația ușoară a arborelui.

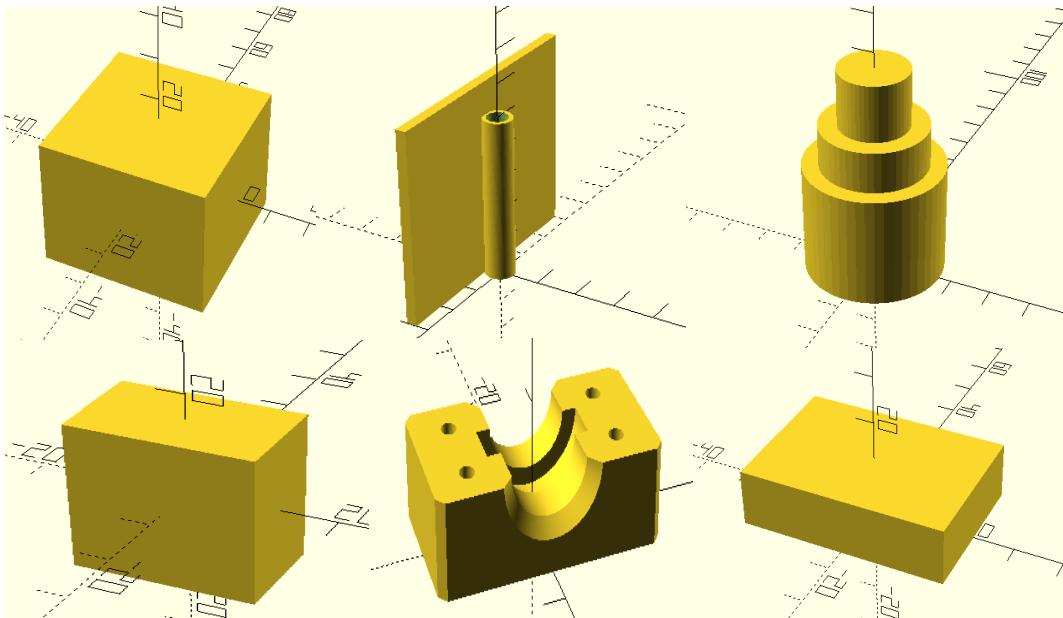


Figura 14. Componentele structurii gata de printat, afișate în OpenSCAD

Înainte de a printa componente, s-a profitat de faptul că fișierele în format 3D se pot manipula digital. Au fost combinate toate componente și a fost realizată mai întâi structura în format digital, pentru a verifica dacă dimensiunile sunt corecte și cum se îmbină toate piesele. Acest proces a permis realizarea unor ajustări și modificări necesare înainte de imprimare, asigurându-se astfel că toate componente se vor potrivi perfect și vor funcționa corespunzător în ansamblul final. De asemenea, acest proces a oferit o idee clară despre cum ar putea arăta sistemul la final, ajutând la identificarea eventualelor probleme de design și optimizării fiecărei piese dacă este nevoie. Această schemă 3D care conține toate componente se poate vedea în Figura 15.

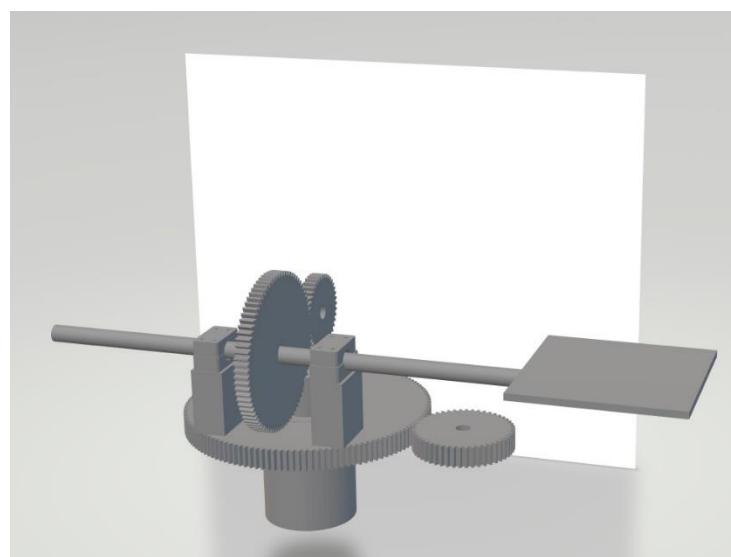


Figura 15. Schema 3D care conține toate elementele printate

5.2.3 Motoarele pas cu pas NEMA17

În acest proiect, s-au utilizat două motoare pas cu pas (stepper) NEMA 17 pentru controlul poziției azimutului și elevației antenei. Ele sunt fixate în suporturi speciale de prindere realizate din metal în structura sistemului. Motoarele pas cu pas sunt esențiale pentru mișările precise și repetabile necesare în acest tip de sistem. Un astfel motor NEMA 17 este format dintr-un rotor cu magneți permanenți și un stator cu mai multe perechi de bobine. Aceste motoare au fost alese, deoarece oferă un control precis al poziției, util pentru orientarea antenei. Motoarele pas cu pas sunt cunoscute pentru fiabilitatea și durabilitatea lor în diverse aplicații industriale și nu numai. Motoarele NEMA 17 oferă 200 de pași per revoluție, ceea ce înseamnă aproximativ 1.8 grade per pas. În Figura 16 se regăsește unul dintre aceste motoare. Ele sunt folosite frecvent în imprimante 3D, CNC-uri (Computer Numerical Control), robotică și automatizări industriale. Aceste motoare au adesea patru, șase sau opt fire, în funcție de configurația bobinelor interne. Cele alese pentru acest sistem au patru fire și se conectează pe porturile 4, 5, 6, 7, respectiv 8, 9, 10, 11 ale placii Arduino UNO. Controlul se realizează printr-un cod scris în Arduino IDE, asigurându-se astfel o precizie mare în mișările necesare pentru ajustarea antenei.

```

const int stepsPerRevolution = 200; // nr. pași per revoluție a motoarelor NEMA 17
// Inițializează bibliotecile stepper pentru motoarele de azimut și elevație pe pinii 4 până la 11:
Stepper azimuthStepper(stepsPerRevolution, 8, 9, 10, 11);
Stepper elevationStepper(stepsPerRevolution, 4, 5, 6, 7);
void setup() {
// Setează viteza la 15 rpm pentru motorul de pe azimut și la 30 rpm pentru motorul de pe elevație:
    azimuthStepper.setSpeed(15);
    elevationStepper.setSpeed(30);
// Inițializează portul serial:
    Serial.begin(9600);
    Serial.println("Arduino este pregătit!");
}
void loop() {
if (Serial.available() > 0) {
    // Citește inputul de la monitorul serial
    String inputString = Serial.readStringUntil('\n');
    inputString.trim(); // Elimină spațiile albe de la început și sfârșit
    Serial.print("Input primit: ");
    Serial.println(inputString); // Mesaj de debugging
    if (inputString.length() > 0) {
        // Convertește inputul în pași de azimut și elevație
        int commaIndex = inputString.indexOf(',');
        if (commaIndex != -1 && commaIndex < inputString.length() - 1) {
            String azimuthStr = inputString.substring(0, commaIndex);
            String elevationStr = inputString.substring(commaIndex + 1);
            int azimuthSteps = azimuthStr.toInt();
            int elevationSteps = elevationStr.toInt();
            Serial.print("Input convertit - Pasi Azimut: ");
            Serial.print(azimuthSteps);
            Serial.print(", Pasi elevatie: ");
            Serial.println(elevationSteps);
            Serial.println("Pornire motoare...\"");
            azimuthStepper.step(azimuthSteps);
            elevationStepper.step(elevationSteps);

            delay(1000); // Delay pentru a permite motoarelor să se miște
        } else {
            Serial.println("Format de input invalid. Inputul ar trebui să fie în formatul
'azimuthSteps,elevationSteps\"");
        }
    }
    delay(500); // Delay pentru a preveni citirea prea rapidă a serialului
}

```

Codul folosește biblioteca Stepper pentru a controla două motoare pas cu pas NEMA 17, fiecare având 200 de pași per revoluție. În setup(), motoarele sunt inițializate cu o viteză de 20 rpm, iar conexiunile seriale sunt deschise pentru comunicare. În loop(), se citește inputul serial, se convertește în pași pentru azimut și elevație. Motorul de azimut și cel de elevație sunt mișcate conform numărului de pași primiți. O mică întârziere este introdusă pentru a permite motoarelor să se miște înapoi de a citi din nou inputul serial. Codul utilizează o conexiune serială pe portul COM3 al plăcii Arduino atât pentru a încărca codul în plăcuță, cât și pentru a comunica cu interfața web.



Figura 16. Motor pas cu pas NEMA17

5.2.4 Rulmenți și arbori

Utilizarea rulmenților a fost aleasă pentru acest sistem deoarece aceștia sunt utili în reducerea frecării și asigură mișcarea rotativă a componentelor. De asemenea, aceștia contribuie la creșterea durabilității și eficienței sistemului prin reducerea uzurii componentelor în mișcare. Rulmenții de pe rotirea pe elevație sunt fixați în lagăre proiectate 3D, pentru a permite rotirea facilă a arborelui. Un astfel de rulment este ilustrat în Figura 17. Pe lângă arborii deja existenți în motoarele pas cu pas, care sunt conectați fiecare la o roată dințată, s-au folosit doi arbori, câte unul pentru fiecare angrenaj. Pe elevație a fost folosit un arbore de oțel inoxidabil pe care este plasată o roată dințată condusă, precum și platforma pentru antenă. Este rezistent și se rotește cu ușurință între lagărele cu rulmenți. Celălalt arbore folosit este un arbore special proiectat pentru a permite rotația pe azimut. A fost proiectat să fie în trei nivele. Primul nivel este baza arborelui, care trebuie să aibă o grosime mai mare pentru a putea suporta întregul sistem. Al doilea nivel este realizat pentru a reduce frecarea între roată și arbore, iar cel de-al treilea este făcut în aşa fel încât să corespundă cu diametrul interior al rulmentului, care este plasat în interiorul găurii roții dințate conduse de pe azimut. Acest arbore se poate vedea în Figura 18.



Figura 17. Rulment cu bile simplu
(preluată din [16])



Figura 18. Arbore pe 3 nivele proiectat 3D

5.3 Implementarea software

Implementarea software a sistemului de poziționare al antenei implică dezvoltarea și integrarea mai multor componente importante. Codul Arduino prezentat anterior controlează mișcarea motoarelor pas cu pas pentru a ajusta poziția azimutului și elevației antenei. Acest cod primește comenzi prin intermediul unei conexiuni seriale, care permite comunicarea cu interfața web.

5.3.1 Comunicarea prin port serial

Pe lângă codul Arduino prezentat în capitolul anterior, care conține partea Arduino de conexiune serială, este necesară și o implementare pe partea software a serverului pentru a asigura o comunicare eficientă între interfața web și placa Arduino UNO. Conexiunea serială permite transferul de date între placă și computer în ambele sensuri, facilitând controlul direct al hardware-ului prin comenzi trimise de utilizator dintr-o interfață web. Mai departe, voi explica succint procesul de conectare prin port serial, cu exemple de cod relevante atât din Arduino, cât și din Javascript. Procesul este împărțit în mai multe etape:

- Inițializarea portului serial:** Codul Arduino inițializează portul serial la o viteză de 9600 baud, ceea ce permite recepționarea și trimiterea datelor.

```
Serial.begin(9600);  
Serial.println("Arduino is ready");
```

- Primirea datelor:** Serverul folosește biblioteca "serialport" pentru a deschide o conexiune serială cu Arduino.

```
const port = new SerialPort({ path: 'COM3', baudRate: 9600 });
```

3. **Trimiterea datelor:** Serverul primește datele de la interfața web și le trimit către Arduino sub forma unei comenzi seriale.

```
port.write(` ${data.azimuthSteps}, ${data.elevationSteps}\n`, (err) => {
  if (err) {
    console.error('Eroare scriere pe port serial!', err);
  }
})
```

4. **Prelucrarea Datelor la Arduino:** Arduino primește datele, le procesează pentru a determina pașii necesari pentru motoarele pas cu pas și trimit comenzi motoarelor pentru a ajusta poziția antenei.

```
if (Serial.available() > 0) {
  // Read the input from the serial monitor
  String inputString = Serial.readStringUntil('\n');
```

5.3.2 Serverul Node.js

Serverul reprezintă o componentă importantă a sistemului. El joacă un rol crucial în gestionarea conexiunii seriale și a interfeței web. Un server este un software sau un dispozitiv hardware care primește și procesează cereri prin intermediul unei rețele. Serverul este realizat cu ajutorul bibliotecii Socket.io care constă într-un server Node.js și o bibliotecă client JavaScript pentru browser. În cadrul acestui proiect, serverul are mai multe roluri. Aceste roluri includ:

1. **Gestionarea conexiunii seriale pe partea de JavaScript:** Serverul stabilește o conexiune serială cu placa Arduino UNO prin portul serial COM3. El ascultă și primește date de la Arduino, afișându-le în consola serverului pentru monitorizare.
2. **Interfața Web:** Serverul servește fișierul "project.html" care reprezintă interfața web prin care utilizatorii interacționează cu sistemul.
3. **Gestionarea conexiunii WebSocket:** Folosind biblioteca Socket.io, serverul gestionează conexiunile WebSocket între client și server. WebSocket este un protocol de comunicație care oferă un canal full-duplex cu latență scăzută între server și browser, permitând o comunicare bidirectională în timp real [9].
4. **Transmiterea comenziilor către Arduino:** Serverul primește date de la interfața web, de exemplu, unghiurile de azimut și de elevație și le transmite mai departe către Arduino prin portul serial COM3. Acest lucru permite controlul motoarelor pas cu pas direct din interfața web, fără a fi nevoie de intervenție manuală.

În codul care implementează serverul, se utilizează mai multe biblioteci, inclusiv express pentru a crea serverul web, http pentru a crea serverul HTTP, path pentru a gestiona căile fișierelor, socket.io pentru a gestiona conexiunile WebSocket și serialport pentru a comunica cu portul serial. Când portul serial se deschide, se afișează un mesaj în consolă. Când se primesc date de la Arduino, acestea sunt afișate în consolă. Serverul gestionează conexiunile WebSocket pentru a

permite comunicarea în timp real cu interfața web. La conectarea unui utilizator, se afișează un mesaj în consolă, iar la deconectarea acestuia, un alt mesaj este afișat. Când serverul primește date de la client prin evenimentul sendSteps, acestea sunt trimise către Arduino prin portul serial. Serverul rulează pe portul 3000.

```
const express = require('express');
const { createServer } = require('node:http');
const { join } = require('node:path');
const { Server } = require('socket.io');
const { SerialPort } = require('serialport');
const app = express();
const server = createServer(app);
const io = new Server(server);
const port = new SerialPort({ path: 'COM3', baudRate: 9600 });

port.on('open', () => {
  console.log('Port serial deschis!');
});
port.on('data', (data) => {
  console.log('Date primite de la Arduino:', data.toString());
});
app.get('/', (req, res) => {
  res.sendFile(join(__dirname, 'proiect.html'));
});
io.on('connection', (socket) => {
  console.log('Utilizator conectat!');
  socket.on('disconnect', () => {
    console.log('Utilizator deconectat!');
  });
  socket.on('sendSteps', (data) => {
    console.log('Date primite de la client:', data);
    port.write(` ${data.azimuthSteps}, ${data.elevationSteps}\n`, (err) => {
      if (err) {
        console.error('Eroare scriere pe port serial!', err);
      } else {
        console.log('Date trimise la Arduino:', ` ${data.azimuthSteps}, ${data.elevationSteps}`);
      }
    });
  });
});

server.listen(3000, () => {
  console.log('Serverul rulează la http://localhost:3000');
});
```

5.3.3 Interfață Web

Interfața web a fost realizată utilizând HTML, CSS și JavaScript. HTML-ul este folosit pentru structura paginii web, CSS-ul pentru stilizarea și aranjarea vizuală a elementelor, iar JavaScript-ul pentru a adăuga funcționalitate și interactivitate paginii. Fișierul HTML conține elementele de bază ale interfeței, cum ar fi câmpuri de input pentru a introduce valorile pentru azimut și elevație, precum și un buton pentru a trimite aceste valori către server. CSS-ul este utilizat pentru a oferi o prezentare atractivă și intuitivă, asigurând că interfața este ușor de utilizat. Partea de JavaScript gestionează funcționalitatea aplicației web și permite interacțiunea utilizatorilor cu pagina. Prin utilizarea bibliotecii socket.io, interfața web stabilește o conexiune WebSocket cu serverul. Aceasta permite o comunicare bidirectională în timp real, ceea ce înseamnă că datele pot fi trimise și primite fără a fi necesară reîncărcarea paginii. JavaScript-ul colectează valorile introduse de utilizator pentru azimut și elevație, le validează și apoi le trimite serverului printr-un eveniment WebSocket. Această interfață web se poate vedea în Figura 19. Logica acestei interfațe web constă în introducerea valorilor pe azimut și elevație în câmpurile de input specifice, și trimiterea acestora la placa Arduino prin apăsarea butonului "Trimite unghiurile", buton care pune în mișcare toată funcționalitatea explicitată anterior. De asemenea, în această interfață regăsim și un buton de revenire la poziție inițială, precum și un loc în care sunt afișate pozițiile curente pe cele două unghiuri. Pe lângă partea estetică și funcțională, interfața web gestionează și tratarea erorilor și furnizarea de feedback utilizatorilor. De exemplu, dacă sunt introduse valori invalide, utilizatorii primesc mesaje de eroare clare. De asemenea, după trimiterea cu succes a comenzi, utilizatorul primește confirmare vizuală, asigurând o experiență de utilizare fără probleme.

Sistem de poziționare antenă

Unghiul de azimut (grade):
0

Unghiul de elevație (grade):
0

Trimite unghiurile

Pozitie initială

Elevație: 0 grade, Azimut: 0 grade

Figura 19. Interfață web pentru controlul sistemului de poziționare

5.3.4 Utilizarea ngrok pentru acces extern

Ngrok este un instrument util pentru expunerea unui server local la internet prin crearea unui tunel securizat. Acest instrument facilitează testarea și dezvoltarea aplicațiilor web, permitând accesul la serverele locale de oriunde, fără a necesita configurarea routerului. Pentru a utiliza ngrok, acesta trebuie descărcat de pe site-ul oficial și apoi instalat. După descărcare, executabilul trebuie pus într-un director accesibil din linia de comandă. În plus, pentru a fi utilizat global în linia de comandă, trebuie adăugată calea către acest director în variabilele de mediu ale sistemului Windows. Autentificarea în ngrok este un pas necesar pentru a utiliza funcțiile sale avansate. După ce se creează un cont pe site-ul ngrok, token-ul de autentificare primit trebuie copiat și utilizat în terminal. Pentru a expune serverul local, se rulează o comandă simplă în terminal. Aceasta va crea un tunel securizat către serverul local care rulează pe un anumit port, în cazul acestui proiect, portul 3000. Ngrok va genera o adresă URL publică ce poate fi folosită pentru a accesa interfața web de oriunde. În proiectul acesta, ngrok a fost folosit pentru a permite accesul la interfața web de la distanță, ceea ce a fost foarte util pentru testare și demonstrație. Integrarea ngrok în proiect a început cu pornirea serverului local folosind Node.js. După ce serverul era activ, ngrok a fost lansat pentru a crea tunelul necesar. URL-ul generat de ngrok a oferit acces imediat și facil la interfața web a sistemului, permitând accesul din afara rețelei locale. Utilizarea ngrok a adus numeroase beneficii, printre care simplificarea accesului la aplicații web locale și facilitarea testării și demonstrațiilor. Cu toate acestea, trebuie luate în considerare aspectele de securitate, deoarece expunerea unui server local pe internet poate introduce riscuri. Configurarea corectă și securizarea serverului împotriva accesului neautorizat sunt critice pentru a evita problemele de securitate.

5.3.5 Conversia din unghiuri în pași pentru motoare și viceversa

Conversia din unghiuri în pași pentru motoarele pas cu pas este esențială pentru funcționarea sistemului de poziționare a antenei. Pentru a asigura o mișcare precisă și repetabilă a antenei, unghurile de azimut și elevație sunt transformate în numărul corespunzător de pași pe care motoarele trebuie să îi efectueze. În acest scop, a fost implementată o metodă de conversie care ia în considerare caracteristicile motoarelor utilizate, și anume NEMA 17, care au 200 de pași per revoluție, ceea ce înseamnă 1.8 grade per pas. Această conversie se bazează pe raportul dintre unghiul total al unei rotații complete (360°) și numărul total de pași necesari pentru a realiza această rotație, ajustat pentru fiecare angrenaj. În primul rând, se calculează pasul unghiular pentru azimut, având în vedere că angrenajul pentru azimut are un raport de transmisie de 2.5 și numărul de pași per revoluție este 200. Ecuația (5) oferă valoarea unghiului corespunzător fiecărui pas al motorului pentru mișcarea de azimut.

$$\text{Pas unghiular pentru azimut} = \frac{360^\circ}{200 * 2.5} \quad (5)$$

În mod similar, pentru calculul pasului unghiular pentru elevație, folosind un raport de 2.04 pentru angrenajul de elevație:

$$\text{Pas unghiular pentru elevatie} = \frac{360^\circ}{200*2.04} \quad (6)$$

Această ecuație (6) oferă valoarea unghiului corespunzător fiecărui pas al motorului pentru mișcarea de elevație. Odată cunoscut pasul unghiular, se poate determina numărul de pași necesari pentru un anumit unghi de azimut. Ecuația (7) convertește unghiul de azimut introdus de utilizator în pași de motor.

$$\text{Numar de pasi pentru azimut} = \frac{\text{Unghiul de azimut introdus}}{\text{Pas unghiular pentru azimut}} \quad (7)$$

Se procedează în același fel și pentru elevație, după cum reiese din ecuația (8).

$$\text{Numar de pasi pentru elevatie} = \frac{\text{Unghiul de elevatie introdus}}{\text{Pas unghiular pentru elevatie}} \quad (8)$$

6 Rezultate experimentale

În această secțiune, sunt prezentate măsurile care au fost luate și testele efectuate pentru a asigura funcționarea corectă și în parametri normali ai sistemului. Aceste măsuri includ: redimensionarea componentelor printate pentru a se potrivi cu celelalte componente 3D și cu celelalte componente, verificarea și testarea conexiunii seriale, tratarea erorilor interfeței web, verificarea consistenței datelor, conversia din unghiurile de azimut și de elevație în pașii necesari mișcării motoarelor pas cu pas și verificarea acesteia prin teste repetate. De asemenea, a fost monitorizată performanța generală a sistemului, inclusiv timpul de răspuns și fiabilitatea. În plus, s-au efectuat teste de rezistență pentru evaluarea durabilității componentelor în condiții de operare pe termen lung.

6.1 Redimensionarea componentelor printate

Software-ul UltiMaker Cura a fost utilizat pentru a ajusta dimensiunile modelelor 3D ale componentelor sistemului. S-au proiectat inițial piesele, s-a testat potrivirea și s-au efectuat ajustări repetitive până când toate componentele s-au îmbinat corect. Un mare ajutor a fost reprezentat de faptul că modelele proiectate în prealabil în OpenSCAD au putut fi introduse în UltiMaker Cura pentru o ajustare mai exactă a dimensiunilor. De exemplu, pentru a asigura că roțile dințate se potrivesc corect pe arbori, s-au modificat dimensiunile interioare ale găurilor din modele până când a fost obținută o potrivire exactă. S-a procedat la fel pentru fiecare componentă, pentru a satisface criteriile de dimensiune și potrivire. În Figura 20, se poate observa posibilitatea de redimensionare pe axele X, Y și Z în interfața UltiMaker Cura, cu o precizie și acuratețe foarte mare. În acest software se pot adăuga mai multe modele 3D deodată, astfel s-a putut verifica fiecare redimensionare de oricâte ori a fost nevoie, pentru a asigura o potrivire perfectă.

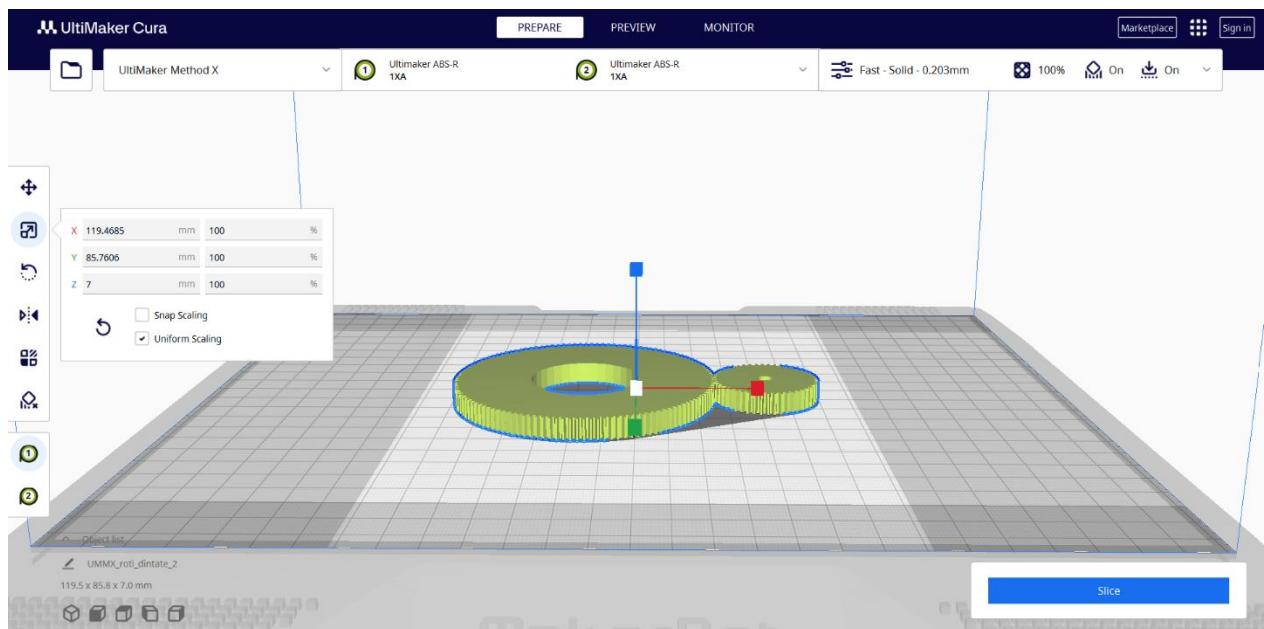


Figura 20. Interfața UltiMaker Cura

6.2 Implementarea procedurilor de detectare și tratare a erorilor interfeței web

Au fost implementate coduri de eroare în interfața web pentru a detecta și gestiona eventualele erori de comunicare și funcționare. S-au realizat trei funcții principale care se ocupă de aceste erori. Funcția validateInput() se concentrează pe verificarea corectitudinii unghiurilor introduse în câmpurile de input. Aceasta testează diferite scenarii, cum ar fi: câmpuri de input lăsate necompletate, unghiuri mai mici de 0 sau mai mari de 360. În cazurile în care se detectează o eroare, va fi afișat un mesaj de eroare corespunzător prin apelarea funcției handleError(errorMsg). Dacă unghiurile sunt introduse corect, utilizatorul va primi un feedback pozitiv printr-un mesaj sugestiv furnizat de funcția displayFeedback(). De asemenea, s-au tratat erorile și în codul serverului, pentru a asigura o funcționare cât mai eficientă. În momentul în care datele sunt trimise la Arduino, a fost implementată o verificare a eventualelor erori prin funcția port.write(). Dacă apare o eroare în timpul scrierii pe portul serial, aceasta este afișată în consolă printr-un mesaj de eroare specific. În caz contrar, dacă datele sunt trimise cu succes, este afișat un mesaj de confirmare în consolă.

Partea de cod interfață web tratare erori:

```
function validateInput() {
    var azimuthAngle = parseFloat(document.getElementById('azimuthAngle').value);
    var elevationAngle = parseFloat(document.getElementById('elevationAngle').value);
    if (isNaN(azimuthAngle) || isNaN(elevationAngle)) {
        return false;
    }
    if (azimuthAngle < 0 || azimuthAngle > 360 || elevationAngle < 0 || elevationAngle > 360) {
        return false;
    }
    return true;
}
function displayFeedback() {
    alert("Unghiurile au fost trimise cu succes!");
}
function handleError(errorMsg) {
    alert("Am întâmpinat o eroare!: " + errorMsg);
}
```

Partea de cod server tratare erori:

```
port.write(` ${data.azimuthSteps}, ${data.elevationSteps}\n`, (err) => {
    if (err) {
        console.error('Eroare scriere pe port serial!', err);
    } else {
        console.log('Date trimise la Arduino:', ` ${data.azimuthSteps}, ${data.elevationSteps}`);
    }
});
});
});
```

6.3 Testarea și verificarea conexiunii seriale

Pentru a asigura o comunicare eficientă și o conexiune fiabilă, a fost efectuată o serie de teste pentru verificarea conexiunii serială. Am verificat conexiunea prin introducerea feedback-ului vizual în consola serverului. Feedback-ul primit confirmă că datele sunt transmise și recepționate între interfața web și Arduino. Acest sistem de feedback a fost implementat pentru a asigura că serverul și conexiunea serială funcționează corect. Feedback-ul este esențial pentru a monitoriza comunicarea între interfața web și placa Arduino. Când se pornește serverul folosind comanda „node server.js” în consolă, apare un mesaj de confirmare care indică faptul că serverul rulează și că portul serial este deschis. În acest moment, Arduino trimite un mesaj de feedback care confirmă că placa este pregătită să primească comenzi. Când este accesată interfața web prin link-ul generat pe portul 3000, se primește un mesaj de confirmare care indică conectarea utilizatorului. De asemenea, dacă se părăsește site-ul, se primește un mesaj care confirmă deconectarea utilizatorului. Pentru a verifica dacă input-ul din interfață a fost primit corect de la utilizator, în consolă se afișează un mesaj sugestiv care confirmă că datele au fost primite și sunt în format corect. După aceasta, datele sunt trimise la Arduino. Arduino trimite un mesaj de feedback pentru a confirma primirea datelor, urmat de un alt mesaj care arată că input-ul a fost împărțit pentru fiecare motor în parte. Acest proces asigură convertirea corectă a unghiurilor în pași pentru motoare. În final, Arduino trimite un mesaj care indică faptul că motoarele sunt pornite și funcționează conform așteptărilor. Toate aceste mesaje de feedback și comenzi se pot observa în Figura 21. Astfel, se poate monitoriza în timp real starea sistemului, asigurându-se că toate componentele acestui sistem funcționează în parametri optimi.

```
PS C:\Users\mariu\Desktop\Materiale_licenta\Interfata_licenta_new> node server.js
Serverul rulează la http://localhost:3000
Port serial deschis!
Date primite de la Arduino: Ard
Date primite de la Arduino: uino
Date primite de la Arduino: est
Date primite de la Arduino: e pr
Date primite de la Arduino: egă
Date primite de la Arduino: tit!
Date primite de la Arduino:

Utilizator conectat!
Date primite de la client: { azimuthSteps: 125, elevationSteps: 102 }
Date trimise la Arduino: 125,102
Date primite de la Arduino: Inp
Date primite de la Arduino: ut p
Date primite de la Arduino: rimi
Date primite de la Arduino: t: 1
Date primite de la Arduino: 25,1
Date primite de la Arduino: 02

Date primite de la Arduino: Inpu
Date primite de la Arduino: t co
Date primite de la Arduino: nver
Date primite de la Arduino: tit
Date primite de la Arduino: - P
Date primite de la Arduino: asi
Date primite de la Arduino: Azim
Date primite de la Arduino: ut:
Date primite de la Arduino: 125,
Date primite de la Arduino: Pas
Date primite de la Arduino: i el
Date primite de la Arduino: evat
Date primite de la Arduino: ie:
Date primite de la Arduino: 102
Date primite de la Arduino:
Por
Date primite de la Arduino: nire
Date primite de la Arduino: mot
Date primite de la Arduino: oare
Date primite de la Arduino: ...
Date primite de la Arduino:
```

Figura 21. Comenzi și mesaje de feedback din consolă

6.4 Calibrarea motoarelor pas cu pas

În urma testării repetitive, s-a ajuns la concluzia că viteza optimă pentru motorul ce ghidează angrenajul pe elevație este de 30 rpm (rotiri pe minut), iar viteza optimă pentru motorul ce ghidează angrenajul pe azimut este de 15 rpm. Calibrarea motoarelor pas cu pas a fost realizată printr-o serie de teste sistematice pentru a determina viteza optimă în funcționarea acestora. S-a început cu viteze diferite pentru ambele motoare și a fost observat comportamentul fiecărui în timpul mișcării. S-au efectuat multiple teste pentru a verifica dacă setările optimizate oferă rezultate mulțumitoare. S-a încercat găsirea unui echilibru între precizie și viteză astfel asigurând funcționarea corectă și eficientă a sistemului de poziționare.

6.5 Rezultatele testelor de precizie

S-au efectuat niște teste de precizie a sistemului pentru a vedea cât de aproape de adevăr este sistemul referitor la poziționare și precizie. Rezultatele acestor teste se pot observa în Tabelul 1, respectiv în Tabelul 2.

Tabelul 1. Deviația pe unghiul de azimut

Unghiul de azimut introdus(°)	Unghiul de azimut măsurat(°)	Deviația (°)
0°	0°	0°
30°	29.8°	0.2°
90°	89.8°	0.2°
180°	178.6°	1.4°
270°	268.6°	1.4°
360°	358.2°	1.8°

Tabelul 2. Deviația pe unghiul de elevație

Unghiul de elevație introdus(°)	Unghiul de elevație măsurat(°)	Deviația (°)
0°	0°	0°
30°	29.7°	0.1°
90°	89.9°	0.1°
180°	178.2°	1.8°
270°	269°	1°
360°	358.2°	1.8°

Aceste teste arată faptul că, chiar dacă poziționarea nu este perfectă, rezultatele sunt foarte aproape de valorile teoretice și de calculele inițiale. În practică, întotdeauna există mici deviații față de valorile teoretice, datorate factorilor mecanici, toleranțelor de fabricație, jocul și frictiunea angrenajelor și altor variabile necontrolabile. Aceste abateri, deși prezente, se situează într-un interval acceptabil de 0.3 până la 2 grade, ceea ce confirmă că sistemul funcționează în parametri normali și oferă o precizie suficientă pentru aplicațiile propuse.

6.6 Configurația finală

Configurația finală constă în combinarea tuturor componentelor și tehnologiilor într-un sistem capabil să funcționeze eficient și precis. Partea practică include nu doar componentele printate 3D, rulmenții, arborii, motoarele pas cu pas și placa Arduino UNO, dar și placa pe care este asamblat tot sistemul, șuruburi de mărimi diferite care fixează suporturile de susținere ale motoarelor pas cu pas de blocurile de prindere printate 3D. Acest sistem care este format din server, interfață web și echipament fizic poate fi vizualizat în Figura 22.

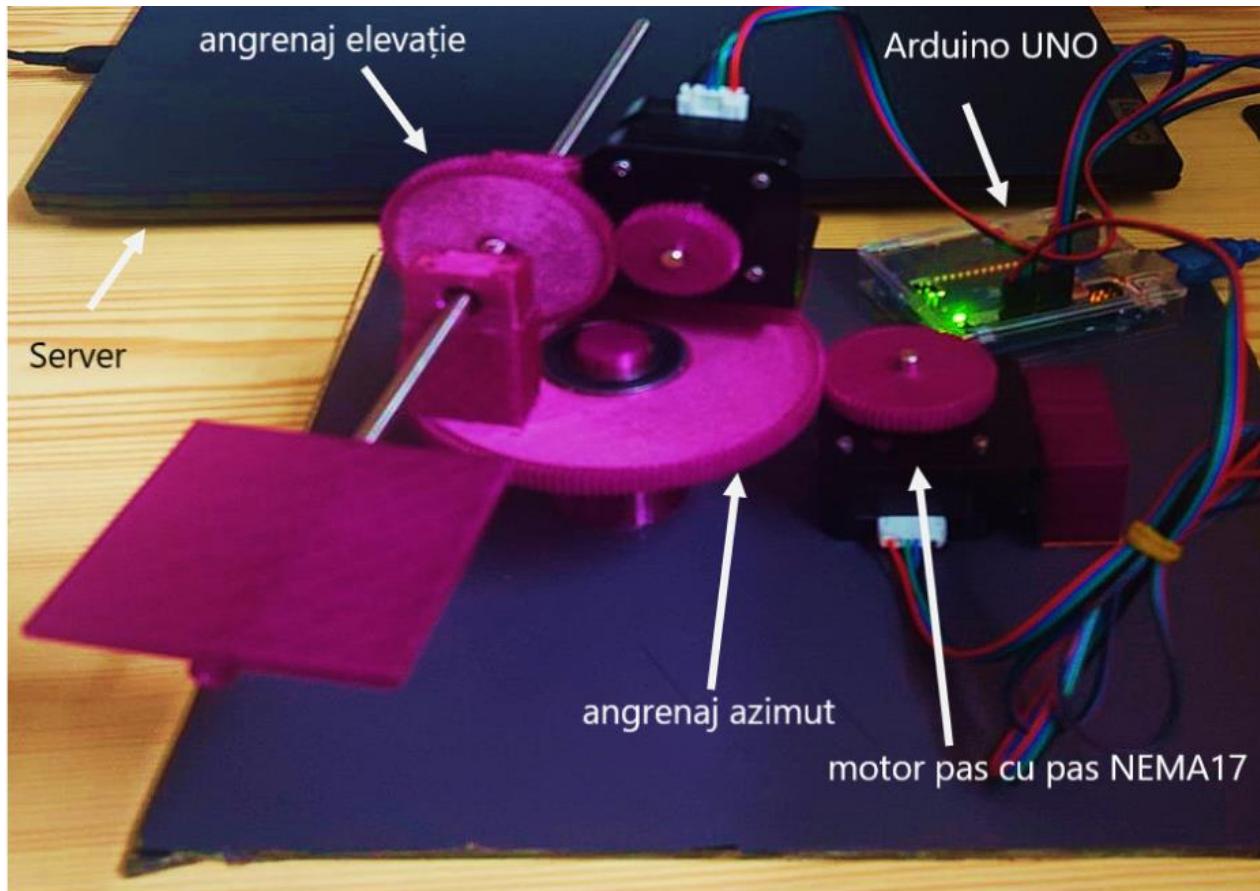


Figura 22. Configurația finală a sistemului de poziționare

6.7 Îmbunătățiri și optimizări

În urma testării și utilizării sistemului, s-au identificat o serie de îmbunătățiri și optimizări care ar putea fi implementate în viitor pentru a crește performanța și fiabilitatea acestuia. Observațiile făcute în timpul testelor practice au arătat mai multe aspecte care ar putea fi ajustate. Una dintre aceste observații a fost legată de influența vibrațiilor asupra preciziei mișcărilor motoarelor. Adăugarea unor amortizoare de vibrații ar putea reduce aceste efecte, asigurând astfel o mișcare mai lină și mai precisă. Aceste amortizoare ar putea fi integrate la nivelul suportului motoarelor și în punctele de contact cu structura principală. Deși algoritmul de control implementat a demonstrat o bună performanță, există posibilitatea de a-l optimiza pentru a compensa mai eficient micile deviații care apar în timpul funcționării. Această optimizare ar putea include ajustarea constantelor de calibrare și implementarea unui algoritm de corecție bazat pe feedback în timp real. Pentru a îmbunătăți și mai mult precizia sistemului, ar putea fi integrat un sistem de feedback bazat pe senzori de poziție. Acești senzori ar permite calibrarea automată și continuă a sistemului, asigurând astfel că orice deviere de la poziția dorită este corectată. Această abordare ar spori fiabilitatea și acuratețea sistemului în aplicații critice. În timpul testelor, anumite componente au arătat semne de uzură prematură sau slăbiciune structurală. Utilizarea unor materiale mai durabile și a unor tehnici de asamblare mai robuste ar putea prelungi durata de viață a sistemului și ar reduce necesitatea de întreținere periodică. De exemplu, roțile dințate din PLA ar putea fi înlocuite cu unele din oțel, astfel să reducă considerabil din fricțiune și ar spori rezistența întregului sistem. Un alt aspect care ar putea fi îmbunătățit este utilizarea unui domeniu propriu securizat și hostat 24/7, în loc de utilizarea unui tunel securizat cu ngrok. Aceasta ar asigura o disponibilitate constantă și securitate sporită pentru interfața web, eliminând dependența de un serviciu terț și oferind o soluție mai profesională. Implementarea unui astfel de domeniu ar permite accesul mai facil și rapid la interfața web, contribuind astfel la o experiență de utilizare mai plăcută și mai eficientă.

7 Concluzii

Proiectul de dezvoltare a unui sistem de poziționare a antenei, folosind o interfață web, o placă Arduino UNO, motoare pas cu pas și alte componente structurale a demonstrat eficiența și utilitatea acestei abordări pentru controlul mișcărilor mecanice precise. Etapele parcurse, de la planificare și documentare până la implementare și testare, au scos în evidență atât punctele forte, cât și provocările întâmpinate. O realizare importantă a acestui proiect a fost integrarea eficientă a componentelor hardware și software, care a permis controlul antenei printr-o interfață web ușor de utilizat. Conversia unghiurilor în pași pentru motoare s-a dovedit a fi precisă, cu abateri minime, iar testele repetate au confirmat stabilitatea și funcționalitatea sistemului. Implementarea unui sistem de feedback vizual în consola serverului a fost foarte utilă pentru monitorizarea și diagnosticarea corectă a comunicării dintre interfață web și placa Arduino. Acest feedback a permis identificarea eventualelor probleme și a contribuit la optimizarea funcționării sistemului. Cu toate acestea, testele au evidențiat și aspecte ce pot fi îmbunătățite. Integrarea amortizoarelor de vibrații și utilizarea unui algoritm de control mai avansat ar putea spori precizia și fiabilitatea sistemului. De asemenea, înlocuirea tunelului securizat furnizat de ngrok cu un domeniu propriu, securizat și hostat 24/7, ar asigura o disponibilitate continuă și o securitate sporită a conexiunii web. Calibrarea motoarelor pas cu pas s-a dovedit a fi crucială pentru obținerea performanțelor optime. Aceste ajustări sunt esențiale pentru asigurarea unei funcționări precise a sistemului. Deși poziționarea antenei nu este perfectă, cu deviații între 0.3 și 2 grade, rezultatele obținute sunt foarte aproape de valorile calculate teoretic. Diferențele minore pot fi atribuite toleranțelor mecanice și variațiilor de fabricație ale componentelor, indicând că în practică, sistemul funcționează foarte bine. Proiectul a reușit să atingă obiectivele inițiale și a demonstrat viabilitatea soluției propuse, dar există întotdeauna loc pentru îmbunătățiri și optimizări. Implementarea unor sisteme de feedback mai avansate și utilizarea unor materiale și tehnologii de construcție mai durabile ar putea prelungi durata de viață a sistemului. În concluzie, proiectul a reușit să dezvolte un sistem de poziționare a antenei eficient și precis, demonstrând utilitatea sa într-o varietate de scenarii practice. Acest sistem, deși funcțional și performant, are potențialul de a fi și mai robust și fiabil prin implementarea optimizărilor și îmbunătățirilor propuse. Rezultatele obținute nu doar că validează designul și execuția inițială, dar sugerează și multiple direcții pentru dezvoltări ulterioare. Astfel, proiectul se constituie ca o bază solidă pentru inovații viitoare și adaptări care să răspundă mai bine nevoilor diverse ale utilizatorilor.

8 Bibliografie

- [1] T. I. A. T. Chaitali Ingale, "Study of Different Types of Microwave Antenna and Its Applications," *International Journal of Computer Technology and Electronics Engineering*, vol. 3, no. Special, 2013.
- [2] P. Dhande, "Antennas and its Applications," *DRDO Science Spectrum*, pp. 66-78, 2009.
- [3] QuickSet, "Antenna Positioner System," QuickSet Defense Technologies, 2 November 2023. [Online]. Available: <https://www.quickset.com/antenna-positioner-system/>. [Accessed 25 May 2024].
- [4] G. R. Paul Motzki, "Smart Shape Memory Alloy Actuator Systems and Applications," in *Shape Memory Alloys - New Advances*, Saarland, IntechOpen, 2023.
- [5] "Shaft Design," Engineering PRODUCT Design, 20 December 2021. [Online]. Available: <https://engineeringproductdesign.com/knowledge-base/shaft-design-strength/>.
- [6] B. Earl, "All About Stepper Motors," adafruit, [Online]. Available: <https://learn.adafruit.com/all-about-stepper-motors>.
- [7] S. Ranger, "What is the IoT? Everything you need to know about the Internet of Things right now," ZDNET, 3 February 2020. [Online]. Available: <https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/>.
- [8] T. Hoffman, "3D Printing: What You Need To Know," PC Mag, 1 July 2020. [Online]. Available: <https://www.pcmag.com/news/3d-printing-what-you-need-to-know>.
- [9] "Altitude & Azimuth: The Horizontal Coordinate System," timeanddate, [Online]. Available: <https://www.timeanddate.com/astronomy/horizontal-coordinate-system.html>.
- [10] "Angles of Elevation and Depression," VarsityTutors, [Online]. Available: https://www.varsitytutors.com/hotmath/hotmath_help/topics/angles-of-elevation-and-depression.
- [11] shagemann, "The difference between helical gears and straight gears," igusblogs, 3 November 2022. [Online]. Available: <https://blog.igus.eu/the-difference-between-helical-gears-and-straight-gears/>.
- [12] "Bearing Trivia," JTEKT, 13 June 2019. [Online]. Available: <https://koyo.jtektd.co.jp/en/2019/06/column01-01.html>.
- [13] "Linear Motion Shafts," WON Linear Motion System, [Online]. Available: <https://www.wonstore.com/Linear%20Motion%20Shaft>.
- [14] "Arduino Uno," JavaTPoint, [Online]. Available: <https://www.javatpoint.com/arduino-uno>.

9 Anexe

9.1 Codul Arduino pentru motoarele pas cu pas

```
#include <Stepper.h>

const int stepsPerRevolution = 200;

Stepper azimuthStepper(stepsPerRevolution, 8, 9, 10, 11);

Stepper elevationStepper(stepsPerRevolution, 4, 5, 6, 7);

void setup() {

    azimuthStepper.setSpeed(15);

    elevationStepper.setSpeed(30);

    Serial.begin(9600);

    Serial.println("Arduino este pregătit!");

}

void loop() {

    if (Serial.available() > 0) {

        String inputString = Serial.readStringUntil('\n');

        inputString.trim();
    }
}
```

```
Serial.print("Input primit: ");

Serial.println(inputString);

if (inputString.length() > 0) {

    int commaIndex = inputString.indexOf(',');

    if (commaIndex != -1 && commaIndex < inputString.length() - 1) {

        String azimuthStr = inputString.substring(0, commaIndex);

        String elevationStr = inputString.substring(commaIndex + 1);

        int azimuthSteps = azimuthStr.toInt();

        int elevationSteps = elevationStr.toInt();

        Serial.print("Input convertit - Pasi Azimut: ");

        Serial.print(azimuthSteps);

        Serial.print(", Pasi elevatie: ");

        Serial.println(elevationSteps);

        Serial.println("Pornire motoare...");

        azimuthStepper.step(azimuthSteps);

        elevationStepper.step(elevationSteps);

        delay(1000);
    }
}
```

```

    } else {

        Serial.println("Format de input invalid. Inputul ar trebui să fie în formatul
'azimuthSteps,elevationSteps'");

    }

}

delay(500);

}

```

9.2 Codul HTML, CSS și JavaScript pentru interfața web

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Poziționare Antenă</title>
    <style>
        form {
            background-color: #d0e9c6;
            border-radius: 8px;
            padding: 40px;
            box-shadow: 2px 8px 12px rgba(0, 0, 0, 0.2);
            width: 50%;
            margin: auto;
        }
        .input-container {
            text-align: center;
        }
        label {
            display: block;

```

```
margin-bottom: 10px;
color: #333;
font-weight: bold;
font-family: "Segoe UI", Arial, sans-serif;
transition: color 0.3s ease;
}
label:hover {
  color: #4caf50;
}
input[type="number"] {
  width: 50%;
  padding: 12px;
  border: 2px solid #4caf50;
  border-radius: 8px;
  box-sizing: border-box;
  margin-bottom: 20px;
  transition: border-color 0.3s ease;
}
input[type="number"]:focus {
  border-color: #45a049;
}
button {
  background-color: #4caf50;
  color: white;
  padding: 14px 24px;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  font-size: 18px;
  display: block;
  margin: 0 auto;
  transition: background-color 0.3s ease;
  box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2);
}
button:hover {
  background-color: darkgreen;
  transform: scale(1.05);
}
h1 {
  text-align: center;
  color: #4caf50;
  margin-bottom: 30px;
  font-size: 24px;
  text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.1);
}
#currentAngles {
  background-color: white;
  border-radius: 8px;
  padding: 10px;
  box-shadow: 2px 4px 8px rgba(0, 0, 0, 0.1);
  text-align: center;
  width: 50%;
```

```

        margin: 15px auto;
    }
#currentAngles p {
    font-family: "Segoe UI", Arial, sans-serif;
    font-size: 16px;
    color: #333;
    margin: 0;
    padding: 5px;
}
#currentAngles span {
    font-weight: bold;
    color: #4caf50;
}
#goToZeroButton:hover {
    background-color: black;
    transform: scale(1.05);
}
#goToZeroButton {
    background-color: gray;
    color: white;
    padding: 14px 24px;
    border: none;
    border-radius: 6px;
    cursor: pointer;
    font-size: 18px;
    display: block;
    margin: 15px auto;
    transition: background-color 0.3s ease;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.2);
}
</style>
</head>
<body>
<h1>Sistem de poziționare antenă</h1>
<form>
    <div class="input-container">
        <label for="azimuthAngle">Unghiul de azimut (grade):</label>
        <input type="number" id="azimuthAngle" min="0" max="360" step="1" value="0">
    </div>
    <div class="input-container">
        <label for="elevationAngle">Unghiul de elevație (grade):</label>
        <input type="number" id="elevationAngle" min="0" max="360" step="1" value="0">
    </div>
    <button id="sendStepsButton">Trimite unghiurile</button>
    <button id="goToZeroButton">Poziție inițială</button>
    <div id="currentAngles">
        <p><strong>Elevație:</strong> <span id="currentElevation">0</span> grade,<br/>
        <strong>Azimut:</strong> <span id="currentAzimuth">0</span> grade</p>
    </div>
</form>

<script src="/socket.io/socket.io.js"></script>

```

```

<script>
const socket = io();

function calculateAndSendSteps() {
    var azimuthAngle = parseFloat(document.getElementById('azimuthAngle').value);
    var elevationAngle = parseFloat(document.getElementById('elevationAngle').value);

    var azimuthStepAngle = 360.0 / (200 * 2.5);
    var elevationStepAngle = 360.0 / (200 * 2.04);

    var azimuthSteps = Math.round(azimuthAngle / azimuthStepAngle);
    var elevationSteps = Math.round(elevationAngle / elevationStepAngle);

    console.log("Pasi pe azimut: " + azimuthSteps);
    console.log("Pasi pe elevatie: " + elevationSteps);

    var currentAzimuthDisplay = document.getElementById('currentAzimuth');
    var currentElevationDisplay = document.getElementById('currentElevation');

    var currentAzimuthValue = parseFloat(currentAzimuthDisplay.textContent);
    var currentElevationValue = parseFloat(currentElevationDisplay.textContent);

    currentAzimuthValue = (currentAzimuthValue + azimuthAngle) % 360;
    currentElevationValue = (currentElevationValue + elevationAngle) % 360;

    currentAzimuthDisplay.textContent = currentAzimuthValue.toFixed(2);
    currentElevationDisplay.textContent = currentElevationValue.toFixed(2);

    socket.emit('sendSteps', { azimuthSteps, elevationSteps });
}

function validateInput() {
    var azimuthAngle = parseFloat(document.getElementById('azimuthAngle').value);
    var elevationAngle = parseFloat(document.getElementById('elevationAngle').value);

    if (isNaN(azimuthAngle) || isNaN(elevationAngle)) {
        return false;
    }

    if (azimuthAngle < 0 || azimuthAngle > 360 || elevationAngle < 0 || elevationAngle > 360) {
        return false;
    }

    return true;
}

```

```

function displayFeedback() {
    alert("Unghiurile au fost trimise cu succes!");
}

function handleError(errorMsg) {
    alert("Am întâmpinat o eroare!: " + errorMsg);
}
document.getElementById('sendStepsButton').addEventListener('click', function(event) {
    event.preventDefault();
    if (validateInput()) {
        calculateAndSendSteps();
        displayFeedback();
    } else {
        handleError("Input invalid!");
    }
});
document.getElementById('goToZeroButton').addEventListener('click', function(event) {
    event.preventDefault();

    var currentAzimuthDisplay = document.getElementById('currentAzimuth');
    var currentElevationDisplay = document.getElementById('currentElevation');

    var currentAzimuthValue = parseFloat(currentAzimuthDisplay.textContent);
    var currentElevationValue = parseFloat(currentElevationDisplay.textContent);
    var azimuthStepAngle = 360.0 / (200 * 2.5);
    var elevationStepAngle = 360.0 / (200 * 2.04);

    var azimuthStepsToZero = Math.round(-currentAzimuthValue / azimuthStepAngle);
    var elevationStepsToZero = Math.round(-currentElevationValue / elevationStepAngle);

    console.log("Pasi azimut spre 0: " + azimuthStepsToZero);
    console.log("Pasi elevatie spre 0: " + elevationStepsToZero);
    currentAzimuthDisplay.textContent = "0";
    currentElevationDisplay.textContent = "0";
    document.getElementById('azimuthAngle').value = 0;
    document.getElementById('elevationAngle').value = 0;

    socket.emit('sendSteps', { azimuthSteps: azimuthStepsToZero, elevationSteps: elevationStepsToZero });

    displayFeedback();
});
</script>
</body>
</html>

```

9.3 Codul JavaScript pentru serverul Node.js

```
// Importarea modulelor necesare
const express = require('express');
const { createServer } = require('node:http');
const { join } = require('node:path');
const { Server } = require('socket.io');
const { SerialPort } = require('serialport');

const app = express();
const server = createServer(app);
const io = new Server(server);

const port = new SerialPort({ path: 'COM3', baudRate: 9600 });

port.on('open', () => {
    console.log('Port serial deschis!');
});

port.on('data', (data) => {
    console.log('Date primite de la Arduino:', data.toString());
});

app.get('/', (req, res) => {
    res.sendFile(join(__dirname, 'proiect.html'));
});

io.on('connection', (socket) => {
    console.log('Utilizator conectat!');

    socket.on('disconnect', () => {
        console.log('Utilizator deconectat!');
    });
    socket.on('sendSteps', (data) => {
        console.log('Date primite de la client:', data);

        port.write(` ${data.azimuthSteps}, ${data.elevationSteps}\n`, (err) => {
            if (err) {
                console.error('Eroare scriere pe port serial!', err);
            } else {
                console.log('Date trimise la Arduino:', ` ${data.azimuthSteps}, ${data.elevationSteps}`);
            }
        });
    });
});

server.listen(3000, () => {
    console.log('Serverul rulează la http://localhost:3000');
});
```

10 Curriculum Vitae

**Marius-Lucian Muntean**
Nationality: Romanian Date of birth: 14/10/2001
Phone number: (+40) 771665206
Email address: marius.muntean2001@gmail.com
Home: Str. Toamnei, nr. 11, Ilva Mica (Romania)

ABOUT ME

I am a graduate at the Faculty of Electronics and Telecommunications. I am looking forward to learning and gaining experience in my field of study and related fields. I consider myself to be an outgoing, open-minded person who can work as part of a team as well as individually. I am looking for an opportunity to start my career, improve and prove myself.

WORK EXPERIENCE

Intern
ENDAVA
City: Cluj-Napoca | Country: Romania
Completed a 3-month practice program focused on software development. Assisted in developing and testing software applications. Gained hands-on experience with various programming languages and technologies. Learned and applied project management methodologies.

EDUCATION AND TRAINING

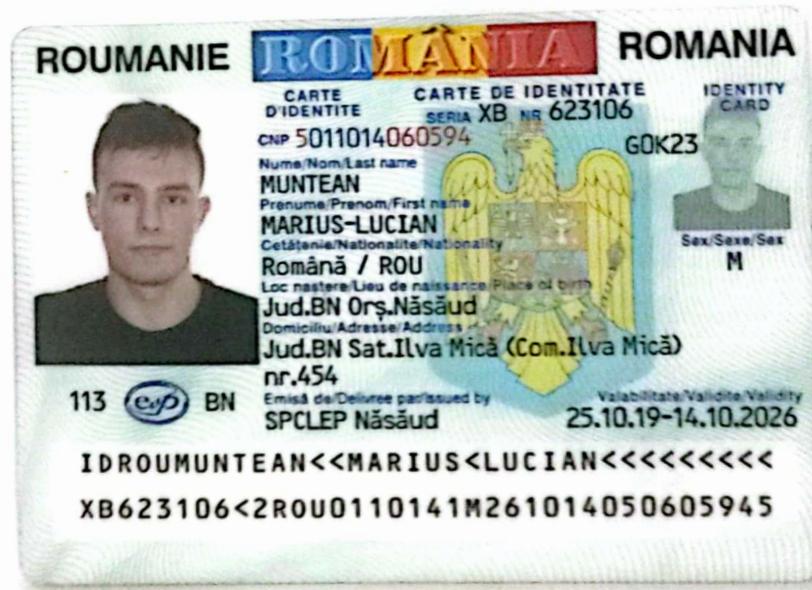
Graduate
National College George Cosbuc, profile: Mathematics-Informatics
City: Nasaud | Country: Romania | Website: <https://www.gcosburnasaud.ro>
Graduate
Faculty of Electronics and Telecommunications
City: Cluj-Napoca | Country: Romania | Website: <https://etti.utcluj.ro/acasa.html>

LANGUAGE SKILLS

Mother tongue(s): Romanian
Other language(s):
English
LISTENING C1 READING C1 WRITING C1
SPOKEN PRODUCTION B2 SPOKEN INTERACTION B2
Levels: A1 and A2: Basic user; B1 and B2: Independent user; C1 and C2: Proficient user

DIGITAL SKILLS

Microsoft Office / Digital Content Creation / Windows and Linux
Soft Skills
Positive attitude / Communication / Problem-solving / Time management / Adaptability
Technical Skills
HTML, Javascript, CSS / Java Programming / C/C++ Programming / SQL
Basic Knowledge
MATLAB / VHDL / OrCAD



ÎN CONFORMITATE CU ORIGINALUL