

# CONTROLUL UNUI ANIMAL FANTASTIC FOLOSIND SENZORI IMU

**Autori:** Paul-Mihai Nerges <sup>1</sup>

Nicoleta-Cristina Zaicanu <sup>2</sup>

[p4ulnerges@gmail.com](mailto:p4ulnerges@gmail.com)

[dyanad14@gmail.com](mailto:dyanad14@gmail.com)

**Coordonator:** asist.cercet.drd.ing. Victor TRIOHIN <sup>3</sup>

prof.univ.dr.habil.dr.ing. Monica LEBA <sup>4</sup>

<sup>1</sup> Universitatea din Petrosani, Facultatea IME, specializarea: Automatica si Informatica Aplicata , anul 1

<sup>2</sup> Universitatea din Petrosani, Facultatea IME, specializarea: Calculatoare, anul 1

<sup>3</sup> Universitatea din Petrosani, Facultatea IME, Departamentul ACIEE

## Rezumat

În cadrul acestei cercetări, am folosit o combinație de hardware și senzori avansați pentru a controla animalul fantastic creat în mediul virtual. Unul dintre componente este platforma de dezvoltare Arduino UNO, care a servit ca bază pentru implementarea sistemului de control. Arduino UNO oferă o interfață flexibilă și ușor de programat, facilitând integrarea și comunicarea cu alte dispozitive. Pentru a detecta și măsura mișcările animalului fantastic, am utilizat un senzor de accelerație Serial 6-Axis Accelerometer SEN0386, special conceput pentru platforma Arduino. Acest senzor oferă măsurători precise ale accelerației și a mișcării în șase axe diferite, permițând o monitorizare detaliată a comportamentului animalului. Comunicarea între Arduino UNO și senzorul de accelerație s-a realizat prin intermediul interfeței de comunicație serială. Arduino UNO a preluat datele de la senzor și a procesat informațiile pentru a ajusta mișcările animalului fantastic în mediul virtual. Această comunicare bidirecțională a permis o interacțiune eficientă între hardware și software, contribuind la controlul precis al entității fantastice.

## Cuvinte cheie

*IMU, Blender, control, Arduino*

### 1. Introducere

Obiectivul principal al acestei lucrări este de a dezvolta și implementa un sistem care să permită controlul mișcării unui obiect digital, creat în mediul virtual al programului Blender, utilizând un senzor fizic conectat la o placă de dezvoltare Arduino UNO.

Prin integrarea acestui sistem hardware-software, ne propunem să demonstrăm fezabilitatea și eficacitatea controlului unui obiect digital prin intermediul mișcărilor detectate de senzorul fizic. Astfel, utilizatorul va putea manipula și controla mișcările obiectului digital în mediul virtual al Blender-ului folosind mișcările detectate de senzor.

Implementarea acestui sistem va presupune dezvoltarea unui algoritm de comunicație și de control între Arduino UNO și Blender, permițând transmiterea datelor de la senzor către aplicația Blender pentru a ajusta poziția și orientarea obiectului digital în funcție de mișcările detectate.

Scopul final al acestui proiect este de a demonstra posibilitatea și utilitatea integrării tehnologiilor hardware și software în domeniul animației și graficii computerizate, deschizând noi perspective în interacțiunea dintre lumea reală și mediile virtuale.

### 2. Proiectare in Blender

În acest proiect Blender, ne-am propus să creăm un model 3D al unei creaturi fantastice prin sintetizarea elementelor din diferite animale. Am inițiat modelul nostru prin construirea corpului creaturii, începând cu cinci sfere principale care conturau capul, gâtul, trunchiul și picioarele din față și din spate. Folosind modul Sculpt cu simetria axei X activată, am modelat aceste sfere, utilizând instrumentul Grab pentru dezvoltarea formei generale și instrumentul Smooth pentru rafinarea suprafeței.

Capul creaturii a fost detaliat prin sculptarea botului și formarea gurii cu instrumentul Crease. Urechile au fost adăugate folosind o altă sferă, oglindită pentru simetrie și detaliate din interior folosind instrumentul Draw cu tasta Control. Nasul a fost creat dintr-o sferă separată, modelată pentru a completa natura mistică a creaturii noastre.

Pentru membre, am aplicat tehnici de oglindire pentru consistență și am folosit instrumentele Grab și Smooth pentru sculptare și, respectiv, netezire. Coada, inspirată de anatomia dragonului, a fost realizată dintr-un cerc și instrumental Path, cu alt instrument Bevel Object folosit pentru a proiecta forma cercului de-a lungul căii. După ajustări pentru formă și dimensiune, acesta a fost îmbinat pe corp.

Aripile au fost inspirate de cele ale pasării Phoenix, construite dintr-un cerc și instrumental Path modificate pentru a obține pene cu vârful ascuțite. Procesul a fost repetat pentru a crea mai multe straturi de pene, adăugând profunzime și realism. Coarnele, asemănătoare cu cele ale unei căprioare, au fost, de asemenea, create cu un cerc și instrumental Path, realizate individual și îmbinate pentru a forma structura complicată.

În cele din urmă, labele, desenate din picioare de dragon, au început cu sfere pentru fiecare deget, detaliate în Modul Sculpt, iar ghearele au fost adăugate folosind un cerc și instrumental Path, apoi unite împreună.

Acest proiect Blender exemplifica integrarea unor caracteristici anatomice variate într-un model coeziv și dinamic, demonstrând capacitatea noastră de a manipula forme simple în structuri complexe prin sculptură atentă și ajustări detaliate.



**Fig 1.** Animal fantastic



**Fig 2.** Animal fantastic

### 3. Proiectare hardware

Senzorul Serial 6-Axis Accelerometer SEN0386 este un dispozitiv de măsurare a accelerației și mișcării, proiectat special pentru utilizarea cu platforma de dezvoltare Arduino. Cu o gamă largă de aplicații și funcționalități, SEN0386 oferă o soluție versatilă și precisă pentru detectarea și monitorizarea mișcării în mediul fizic.

Acest senzor este echipat cu un accelerator triaxial și un giroscop triaxial, permițând măsurători precise în șase axe diferite. Accelerometrul detectează accelerația liniară în trei axe (X, Y și Z), în timp ce giroscopul măsoară viteza de rotație în jurul acestor axe. Această combinație de măsurători oferă informații detaliate despre mișcările și orientarea unui obiect în spațiu.

Unul dintre avantajele principale ale SEN0386 este interfatarea ușoară cu platforma Arduino, facilitată de interfața serială integrată. Acest lucru permite o integrare rapidă și simplă în proiectele de dezvoltare hardware, precum și comunicarea eficientă cu alte dispozitive și sisteme.

Fiabilitatea și precizia sunt două caracteristici esențiale ale SEN0386. Senzorul oferă măsurători stabile și consistente, ceea ce îl face ideal pentru aplicații care necesită detectare și control precis al mișcării. De asemenea, este echipat cu funcții avansate de calibrare și compensare a erorilor, asigurând rezultate fiabile în diverse condiții de utilizare.

Datorită caracteristicilor sale tehnice și versatilității, SEN0386 este folosit într-o varietate de domenii și aplicații, cum ar fi roboți mobili, dispozitive de realitate virtuală, echipamente de monitorizare a sănătății și multe altele. Performanța sa remarcabilă și ușurința în utilizare îl fac un instrument indispensabil pentru dezvoltarea și implementarea soluțiilor tehnologice moderne.

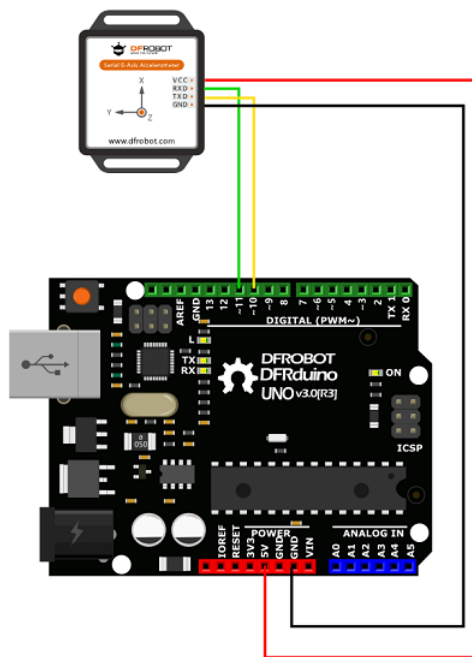
Arduino este o platformă open-source de dezvoltare hardware, proiectată pentru a fi accesibilă și ușor de utilizat de către pasionații de electronică, inginerii amatori și profesioniști deopotrivă. Creată inițial în 2005 de către Massimo Banzi și David Cuartielles, Arduino a devenit rapid un instrument de bază în domeniul prototipării și dezvoltării de proiecte electronice.

Platforma Arduino constă dintr-o serie de plăci de dezvoltare bazate pe microcontrolere AVR (Atmel) sau ARM (chipuri de la producători precum Microchip și Intel). Aceste plăci sunt echipate cu interfețe digitale și analogice, care permit conectarea la o varietate de senzori, module și componente electronice.

Unul dintre avantajele majore ale platformei Arduino este mediul de dezvoltare software ușor de înțeles și de utilizat. Mediul Arduino IDE (Integrated Development Environment) oferă o interfață grafică simplă și intuitivă pentru scrierea și încărcarea codului pe plăcile Arduino. Cu ajutorul unei biblioteci extinse de coduri predefinite și exemple practice, utilizatorii pot crea rapid și eficient proiecte complexe de hardware și software.

Arduino promovează conceptul de "open-source hardware", ceea ce înseamnă că specificațiile tehnice ale plăcilor Arduino sunt publice și accesibile tuturor. Acest lucru încurajează colaborarea și partajarea de cunoștințe între comunitatea Arduino, stimulând inovația și dezvoltarea continuă a platformei.

Datorită versatilității și accesibilității sale, Arduino este folosit într-o varietate de domenii și aplicații, inclusiv roboți, dispozitive IoT, sisteme de control automat și multe altele. De la hobbyiști la profesioniști, platforma Arduino continuă să inspire și să faciliteze creația de soluții inovatoare în lumea electronică.



**Fig 3.** Conectare Arduino cu SEN0386

Pentru a conecta senzorul la placa Arduino, am urmat următoarele conexiuni:

VCC (Tensiune de alimentare): Am conectat firul VCC de la senzor la pinul de alimentare 5V de pe placa Arduino. Această conexiune furnizează tensiunea necesară pentru alimentarea senzorului.

GND (Pământ): Firul GND de la senzor a fost conectat la pinul de pământ (GND) de pe placa Arduino. Această conexiune asigură o legătură comună pentru circuit, necesară pentru a completa circuitul electric.

RXD (Recepție de date): Firul RXD de pe senzor a fost conectat la pinul 10 (recepție) de pe placa Arduino. Acesta este pinul pe care Arduino primește date de la senzor.

TXD (Transmisie de date): Firul TXD de la senzor a fost conectat la pinul 11 (transmisie) de pe placa Arduino. Acesta este pinul pe care Arduino transmite date către senzor.

```
#include <DFRobot_WT61PC.h>
#include <SoftwareSerial.h>

// SoftwareSerial RX and TX pins
#define RX_PIN 11
#define TX_PIN 10

// Create a SoftwareSerial object
SoftwareSerial softSerial(RX_PIN, TX_PIN); // RX, TX

DFRobot_WT61PC sensor1(&softSerial);

void setup()
{
    Serial.begin(9600); // Use Serial for debugging
    softSerial.begin(9600);

    sensor1.modifyFrequency(FREQUENCY_10HZ); // Set data output frequency for sensor 1
}

void loop()
{
    if (sensor1.available()) {
        Serial.print(sensor1.Angle.X); Serial.print(",");
        Serial.print(sensor1.Angle.Y); Serial.print(",");
        Serial.println(sensor1.Angle.Z);
    } else {
        Serial.println("Sensor 1 not available");
    }
}
```

Acest cod este destinat să comunice cu senzorul SEN0386 pentru a obține date despre orientarea acestuia în spațiu. Funcționarea acestui cod constă în:

Include bibliotecile necesare:

#include <DFRobot\_WT61PC.h>: Această linie include biblioteca pentru senzorul SEN0386, care conține funcțiile necesare pentru a comunica cu senzorul și a obține datele sale.

#include <SoftwareSerial.h>: Această linie include biblioteca pentru comunicarea serială software, necesară pentru a comunica cu senzorul folosind pini digitali.

Defineste pini pentru comunicarea serială software:

#define RX\_PIN 11: Această linie definește pinul RX (recepție) folosit pentru comunicarea serială software.

#define TX\_PIN 10: Această linie definește pinul TX (transmisie) folosit pentru comunicarea serială software.

Inițializează obiectul SoftwareSerial:

SoftwareSerial softSerial(RX\_PIN, TX\_PIN);: Această linie creează un obiect SoftwareSerial cu pini RX și TX specificați.

Inițializează obiectul DFRobot\_WT61PC:

DFRobot\_WT61PC sensor1(&softSerial);: Această linie creează un obiect DFRobot\_WT61PC și îl asociază cu obiectul SoftwareSerial creat anterior.

Configurează și inițializează serialul:

Serial.begin(9600);: Inițializează comunicarea serială pentru portul serial hardware (conectat la USB pentru debugging).

softSerial.begin(9600);: Inițializează comunicarea serială software cu senzorul la o viteză de transmisie de 9600 de biți pe secundă.

Configurează frecvența de ieșire a datelor pentru senzor:

sensor1.modifyFrequency(FREQUENCY\_10HZ);: Această linie configurează frecvența de ieșire a datelor de la senzor la 10 Hertz, ceea ce înseamnă că datele sunt trimise de senzor la fiecare 0,1 secunde.

Bucleaza pentru a citi și afișa datele de la senzor:

În bucla loop(), codul verifică dacă datele de la senzor sunt disponibile folosind sensor1.available().

Dacă datele sunt disponibile, acesta le citește și le afișează utilizând sensor1.Angle.X, sensor1.Angle.Y și sensor1.Angle.Z pentru a obține unghiurile de rotație pe cele trei axe (X, Y și Z).

Dacă datele nu sunt disponibile, se afișează un mesaj de eroare.

Prin urmare, acest cod inițializează comunicarea cu senzorul SEN0386 și afișează datele despre orientarea acestuia pe diferite axe în terminalul serial al Arduino IDE.

#### 4. Sistemul integrat

Patrea de cod Python pentru realizarea conexiunii între obiectul proiectat în Blender și senzor prin intermediul plăcii Arduino Uno este:

```
import bpy
import serial
import time
from math import radians

# Set up serial connection (Replace 'COM4' with your Arduino's port)
try:
    ser = serial.Serial('COM5', 9600, timeout=1)
except serial.SerialException as e:
    print("Serial exception:", e)
    raise e

# Define the object you want to manipulate
cube = bpy.data.objects['Cube']
initial_angles = None

def update_initial_angles():
    global initial_angles
    print("Waiting for initial sensor data...")
    while initial_angles is None:
        initial_angles = read_sensor_data()
        time.sleep(0.1) # Wait a bit before trying again to prevent spamming the output
    print("Initial angles captured:", initial_angles)

def read_sensor_data():
    """Read data from the serial port and return it as a list of floats."""
    while ser.in_waiting > 0:
        line = ser.readline().decode('utf-8').strip()
        print("Debug - Line Read:", line) # Debug print
        if "not available" not in line:
            try:
                angles = list(map(float, line.split(',')))
                return angles
            except ValueError:
                print("Received malformed data:", line)
                return None
    else:
        print(line)
        return None
    return None

def apply_transformation(angles):
    """Apply transformations relative to initial angles."""
    global initial_angles
    if angles and len(angles) == 3 and initial_angles is not None:
        # Calculate the difference from the initial angles
        angle_differences = [(angles[i] - initial_angles[i]) % 360 for i in range(3)]
        angle_differences = [angle - 360 if angle > 180 else angle for angle in angle_differences]
```

```

        # Apply radians transformation
        cube.rotation_euler = (radians(angle_differences[0]), radians(angle_differences[1]),
radians(angle_differences[2]))

    # Debugging outputs
    print("Cube Rotation (radians):", cube.rotation_euler)

def update_scene():
    """Update the scene with new sensor data."""
    angles = read_sensor_data()
    print("Debug - Angles:", angles) # Check what angles are received
    if angles:
        apply_transformation(angles)
        bpy.context.view_layer.update()

# Ensure continuous update using a modal operator
class ModalOperator(bpy.types.Operator):
    bl_idname = "object.modal_operator"
    bl_label = "Sensor Data Modal Operator"

    def modal(self, context, event):
        if event.type == 'TIMER':
            update_scene()
        return {'PASS_THROUGH'}

    def invoke(self, context, event):
        self.timer = context.window_manager.event_timer_add(0.1, window=context.window)
        context.window_manager.modal_handler_add(self)
        return {'RUNNING_MODAL'}

def register():
    bpy.utils.register_class(ModalOperator)

def unregister():
    bpy.utils.unregister_class(ModalOperator)

if __name__ == "__main__":
    register()
    update_initial_angles() # Make sure this captures data before proceeding
    if initial_angles is None:
        print("Initial angles not set. Check sensor connection and data.")
    else:
        bpy.ops.object.modal_operator('INVOKE_DEFAULT')

```

Acest cod Python este conceput pentru a interacționa între aplicația Blender și un senzor conectat la portul serial al Arduino-ului. Iată o explicație a funcționării acestui cod:

**Conectarea Serială:** Codul începe prin a încerca să stabilească o conexiune serială cu Arduino-ul conectat la portul COM5 la o viteză de transfer de 9600 de biți pe secundă.

**Inițializarea Obiectului 3D:** Definiște un obiect 3D în Blender pe care doriți să îl manipulați. În acest caz, este vorba despre un cub numit 'Cube'.

**Actualizarea unghiurilor inițiale:** Așteaptă și citește datele inițiale de la senzor până când sunt disponibile și le salvează ca unghiuri inițiale.

**Citirea Datelor de la Senzor:** Citirea și decodificarea datelor de la senzor prin portul serial. Datele sunt transmise sub formă de unghiuri pe cele trei axe (X, Y și Z).

**Aplicarea Transformărilor:** Calcularea diferenței între unghiurile inițiale și cele citite de la senzor și aplicarea acestor diferențe ca transformări ale obiectului 3D în Blender.

**Actualizarea Scenei:** Actualizarea continuă a scenei Blender cu noile date citite de la senzor.

**Operator Modal:** Crearea unui operator modal pentru a asigura actualizarea continuă a scenei în timp real, utilizând un temporizator pentru a citi și aplica datele de la senzor în mod repetat.

Prin urmare, acest cod facilitează interacțiunea între datele citite de la senzorul SEN0386 și transformările aplicate asupra unui obiect 3D în aplicația Blender, permițând astfel manipularea obiectului 3D în funcție de mișcările detectate de senzor.



**Fig 4.** Schema de interacțiune

#### Schema de Interacțiune:

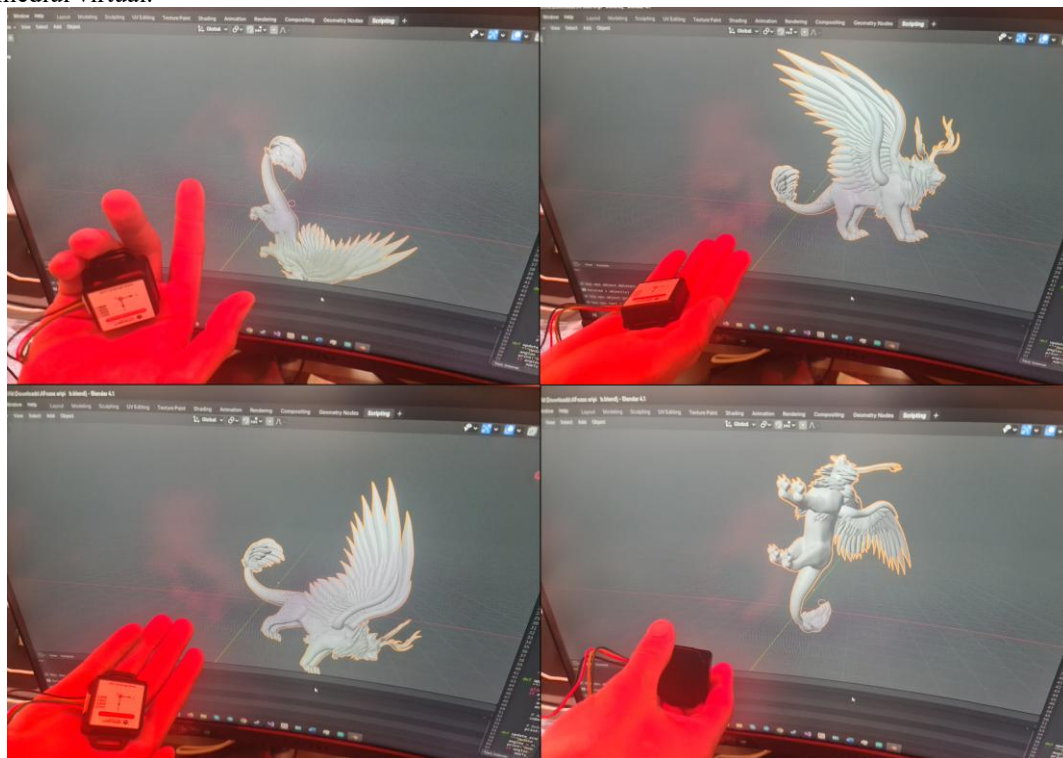
**IMU (Unitate de Măsurare a Inerției) + Arduino:** Unitatea de măsurare a inerției (IMU) este conectată la un Arduino, care servește ca platformă de procesare și intermediere a datelor. IMU este un senzor care măsoară accelerația, viteza unghiulară și orientarea obiectului pe care este montat.

**Arduino:** Arduino-ul preia datele de la IMU și le procesează, utilizând un script specificat pentru a comunica și a interpreta datele de la senzor.

**Script Python:** Un script Python este utilizat pentru a prelua și a interpreta datele de la Arduino. Acest script poate include logica pentru a interpreta mișcările detectate de IMU și poate genera comenzi corespunzătoare pentru manipularea unui personaj virtual.

**Personaj Virtual (Blender):** Personajul virtual, creat în mediul Blender, este manipulat de scriptul Python pe baza datelor primite de la Arduino. Scriptul poate aplica transformări și animații asupra personajului virtual în funcție de mișcările detectate de IMU.

Această schemă simplă ilustrează modul în care Arduino și IMU acționează ca intrări pentru scriptul Python, care la rândul său controlează comportamentul și mișcarea personajului virtual în mediul Blender. Astfel, mișcările detectate de senzorul fizic pot fi traduse în mișcări și acțiuni ale personajului virtual, oferind o interacțiune între lumea reală și mediul virtual.



**Fig 5.** Prezentare proiect

#### Bibliografie:

1. <https://www.rs-online.com/designspark/what-is-arduino-uno-a-getting-started-guide>
2. <https://robu.in/what-is-arduino-uno/>
3. <https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/>
4. <https://core-electronics.com.au/serial-6-axis-accelerometer-for-arduino.html>

