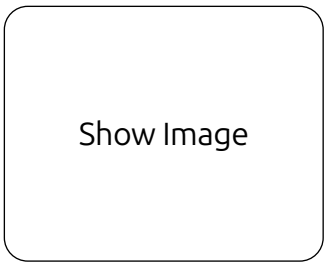
Servo Motor Digital Twin

A real-time digital twin system for servo motors, enabling bidirectional communication between a physical servo connected to an Arduino and a 3D visual representation.



Overview

This project creates a digital twin for a servo motor, allowing:

- Real-time synchronization between the physical servo and 3D visualization
- Bidirectional control - changes to the physical servo are reflected in the visualization and vice versa
- Interactive control via keyboard and mouse
- Standalone operation in simulation mode when hardware is unavailable

Features

- **Bidirectional Control:** Control the servo via the 3D interface or directly through the physical hardware
- **3D Visualization:** Realistic 3D rendering of the servo with real-time angle display
- **Multiple Control Options:**
  - Arrow keys for incremental rotation
  - Mouse wheel for smooth adjustment
  - Direct angle input (terminal version)
- **Auto-detection:** Automatically finds and connects to Arduino hardware
- **Simulation Mode:** Runs without physical hardware for demonstration or development
- **Robust Communication:** Reliable serial protocol with error recovery

System Architecture

The system consists of three main components:

1. **Arduino Firmware (`arduino.ino`):**
   - Controls the physical servo motor
   - Handles serial communication with the Python application
   - Reports current servo position
   - Processes movement commands
2. **Digital Twin GUI (`digital_twin.py`):**
   - Provides 3D visualization of the servo using OpenGL
   - Accepts keyboard and mouse input for servo control
   - Synchronizes with the physical servo
   - Handles bidirectional communication with Arduino
3. **Terminal Control Application (`rotate_servo.py`):**
   - Simple text-based interface for servo control
   - Alternative to the GUI version
   - Useful for headless systems or troubleshooting

Requirements

Hardware

- Arduino board (Uno, Nano, Mega, etc.)
- Standard servo motor
- USB cable
- Optional: Potentiometer for manual control

Software Dependencies

- Python 3.6+
- PySerial
- PyGame
- PyOpenGL
- Arduino IDE

Installation

1. **Install Python Dependencies:**

   ```bash
   pip install pyserial pygame pyopengl
   ```

2. **Upload Arduino Firmware:**
   - Open `arduino.ino` in the Arduino IDE
   - Connect your Arduino board
   - Upload the sketch

3. **Connect Hardware:**
   - Connect servo to pin 9 on the Arduino
   - Ensure proper power supply for the servo
   - Optional: Connect potentiometer to pin A0

Usage

Running the Digital Twin GUI

```bash
python digital_twin.py
```

Options:

- `--port PORT`: Specify Arduino port (e.g., COM3, /dev/ttyACM0)
- `--baud RATE`: Set baud rate (default: 9600)
- `--list-ports`: List available serial ports and exit
- `--sim`: Run in simulation mode without Arduino

Running the Terminal Controller

```bash
python rotate_servo.py
```

Options:

- `--port PORT`: Specify Arduino port (e.g., COM3, /dev/ttyACM0)
- `--baud RATE`: Set baud rate (default: 9600)
- `--list-ports`: List available serial ports and exit

Controls

Digital Twin GUI:

- Arrow keys: Rotate servo in 10° increments
- Mouse wheel: Fine rotation control
- ESC: Exit application

Terminal Interface:

- 1: Rotate 5° right
- 2: Rotate 5° left
- 3: Set specific angle
- 4: Show available ports
- q: Quit

Communication Protocol

The Arduino and Python applications communicate using a simple text-based protocol:

- **Commands to Arduino:**
  - `S:angle\n` - Set servo to the specified angle (0-180)
- **Updates from Arduino:**
  - `A:angle\n` - Report current servo angle
  - `ACK:angle\n` - Acknowledge command

Troubleshooting

No Arduino Connection:

- Check USB cable
- Verify correct port selection
- Run with `--list-ports` to see available ports
- Try unplugging and reconnecting the Arduino

Servo Not Responding:

- Check servo wiring (signal to pin 9, power to 5V, ground to GND)
- Ensure servo receives adequate power
- Test servo with a basic Arduino sketch

Display Issues:

- Update OpenGL drivers
- Try running in a different resolution
- Check for PyGame/OpenGL compatibility issues with your system

Project Structure

```
servo-digital-twin/
├── arduino.ino          # Arduino firmware
├── digital_twin.py      # 3D visualization and control GUI
├── rotate_servo.py      # Terminal-based control interface
└── README.md            # This documentation
```

Future Enhancements

- Multiple servo support
- Data logging and performance analysis
- Advanced motion profiles and trajectories
- Network/cloud connectivity for remote monitoring
- Support for additional sensor inputs

License

This project is available under the MIT License.

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

Acknowledgments

- PyGame and PyOpenGL communities
- Arduino community for libraries and support