A Prototype Development of an ESP32-Based Locomotor Device for Gait Rehabilitation

Caparro, Mark Darius S., David, Lexcie Jude T., Echavez, Matthew Broderick D., Monterde, Lawrence, Tañala, Alexa A.

co.renso.m@gmail.com
lexciejudedavid@gmail.com
dariusmarkcaparro@gmail.com
mateoechavez7@gmail.com
tanala.2324A16786.bsap@gmail.com

Abstract
This study describes a low-cost and easy to deploy gait rehabilitation system that uses an ESP32 microcontroller with an integration of a smartphone-accessible graphical user interface (GUI). Thus, utilizing seamless wireless communication via Bluetooth or Wi-Fi protocols inherent to the ESP32 module. A gait rehabilitation device in Arduino platform is presented to provide cost-effective, portable, and user-centric prototype for locomotor-training and related fields testing. This prototype successfully simulated a complete gait cycle with servo-driven joint motion and real-time visualization through an interactive GUI which confirms the feasibility of a low-cost ESP32/Arduino-based gait rehabilitation prototype. This project is intended as a proof-of-concept and feasibility testing device rather than a clinical-grade system.

Keywords: *gait rehabilitation device, arduino platform, proof-of-concept*

Introduction

The Philippines faces persistent challenges in providing effective rehabilitation for Filipinos with mobility impairments. With the improvements in health care in the last few decades, residents are now living longer, some with multiple, often complex, locomotor difficulties. As a developing country, healthcare systems are inaccessible to the marginalized communities, much so to specialized and advanced rehabilitation technologies. Gait assessment and rehabilitation remain largely infamous, whereas the cohort of "baby boomers" in the Philippines is now reaching an age at which they will begin to severely stress and demand the Philippine health care system with new, updated approaches for common impairments—leaving gait concerns and needs unattended. This creates a gap between clinical need and available care for a country of limited access.

The country's geographic location and tropical climate contribute to a high incidence of cerebrovascular disease, among which is stroke. Stroke remains as the leading cause of long-term disability with gait impairment being a common and debilitating consequence.

These realities arise several motivation for this study:

1. How can gait rehabilitation devices be designed to be accessible and affordable to patients across socioeconomic backgrounds?

*Adviser: Arjo Ladia
College of Computing Studies, Eulogio "Amang" Rodriguez Institute of Science and Technology, Nagtahan, Sampaloc Manila
Full list of authors' information is available at the end of this paper.

2. How can wearable systems address the high prevalence of stoke-related gait impairments in the Philippines?
3. How can continuous, home-based monitoring and rehabilitation be implemented in resource-limited environments?

Background

Gait impairment is a common consequence of neurological disorders, aging, and post-stroke conditions that often result in diminished quality of life. Gait rehabilitation aims to restore or improve walking ability through repetitive, task-specific training that reinforces proper joint motion and timing across the gait cycle. However, conventional gait rehabilitation typically relies on manual assistance from therapists, or commercially available robotic exoskeletons, many of which are costly, complex, and inaccessible in low-resource clinical and educational settings.

Recent advancements in lower-limb exoskeletons and assistive robotic devices have demonstrated potential in supporting gait rehabilitation. Despite these advancements, most existing systems are designed for clinical or industrial use that require sophisticated hardware, and expensive mechanical systems. The idea for this study emerged from the need to support this by developing a low-cost, modular, and educational gait rehabilitation prototype that demonstrates fundamental gait mechanics while remaining accessible to students, researchers, and small rehabilitation facilities. During the initial review of related literature and rehabilitation practices, it was observed that many studies successfully model gait cycles using microcontroller-based systems due to their wide community support. These platforms have been shown to effectively simulate joint kinematics of the hip and knee during different phases of the gait cycle, making them suitable for prototyping and instructional use.

Method

This section explains the methodology used in this study. It details the workflow, architecture, and the prototyping that govern the workings of the locomotion device.

System and Operational Flow

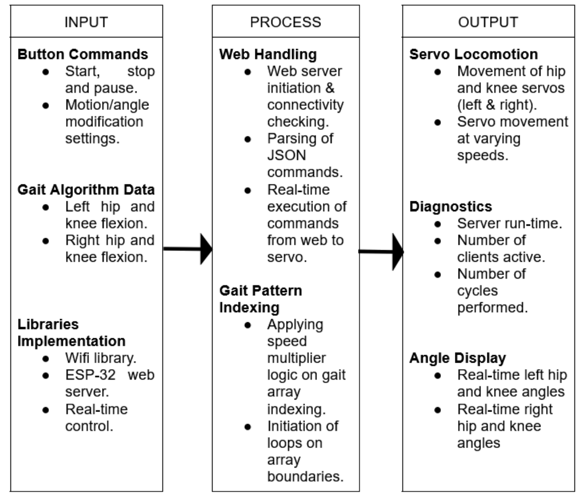


Figure 1.1. System and Operational Workflow in IPO format.

Figure 1.1 shows a model of a comprehensive workflow that was implemented in this study. The input section involves the implementation of various button commands etched on the web server GUI which dictates the overall functionality of the exoskeletal model using servo motors. ESP-32, as a main board, proved to be different from the usual Arduino UNO in terms of efficiency and its use, so this study made use of several libraries (e.g., <WiFi.h>,<ESPAsyncWebServer.h>,<ESP32Servo.h>, etc.) for harnessing its functionality in accordance with our needs. Additionally, this stage also involves feeding the gait algorithm data that we have to the ESP-32 so that it may be used as a main reference to the movement of the servo motors.

The process is about establishing a connection between the web server and the ESP-32 so that the exoskeleton can be controlled wirelessly. This involves handling commands in .JSON format that the web server generates in response to the clicking of command buttons. Through this process, the computer can easily read/write the commands and the transmission of data between the ESP-32 and web server will be smoothly processed thanks to the lightweight nature of the format. Another crucial part of the process in making the servos move is by gait pattern indexing. With the use of the gait data restored in arrays, the system reads a “beat” that it follows and loops whenever needed. This creates the pattern of motion per phase that the four servos follow in moving the hips and joints of both legs.

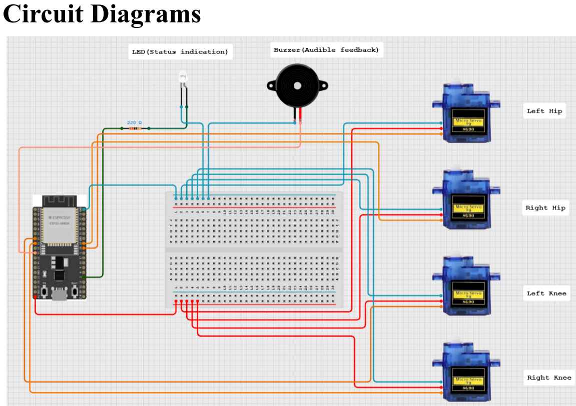
This overall flow leads to the formulation of movement patterns based on gait algorithms that causes the leg model to move as intended. Both the hips' and knees' servos attached to the exoskeleton will be the ones responsible for inducing movement on the leg. Additionally, the complexity of the process also leads to an interactable webpage. The webpage GUI contains virtual buttons intended for different functionalities, these buttons are capable of starting, stopping and pausing the device at work. Also, it contains virtual buttons that are dedicated for user-desired modifications, specifically, these buttons allow the user to modify the speed of the servo movements and even adjust the angle of the servos manually.

Finally, included in the GUI are several diagnostics that show various information vital to the operation of the device. This includes the angle that the servos move in accordance to and the current phases that they are performing in real-time. Server run-time, number of clients active and the number of cycles performed are also shown as additional information should the user be curious about those. These inputs, processes and outputs are what makes up the inner and outer-workings of the locomotor device highlighted in this study.

System Design and Architecture

The hardware architecture employs a star topology with the ESP32 microcontroller as the central hub, interfacing directly with four servo motors for bilateral gait simulation.

Circuit Diagrams



The system implements a dual-power architecture with separate voltage regulation for the ESP32 (3.3V) and servo motors (5V). The circuit was designed using Cirkit Designer and validated through iterative prototyping.

Key Circuit Features:

- ESP32 Core: Dual-core 240MHz processor with integrated Wi-Fi/Bluetooth
- PWM Signal Distribution: Direct GPIO-to-servo connections with independent PWM channels
- Power Management: External 5V/3A power bank with 1000µF decoupling capacitor
- Safety Circuitry: Over-current protection and voltage regulation

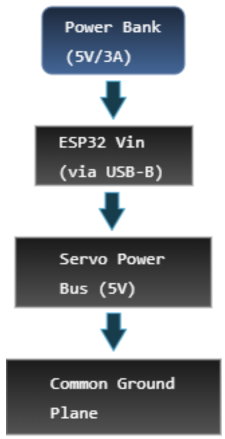
Pin Mapping Table:

Component	GPIO Pin	PWM Channel	Function
Left Hip Servo	GPIO 18	Channel 0	Primary joint actuation
Left Knee Servo	GPIO 19	Channel 1	Primary joint actuation
Right Hip Servo	GPIO 32	Channel 2	Primary joint actuation
Right Knee Servo	GPIO 33	Channel 3	Primary joint actuation
Buzzer	GPIO 26	N/A	Auditory feedback
Status LED	GPIO 2	N/A	System status indication

Wiring

The wiring scheme prioritizes signal integrity and mechanical robustness:

1. Power Distribution:



2. Signal Routing:
    - Servo control wires (orange) routed separately from power lines
    - Cable strain relief at all servo connection points
    - Color coding: Red (5V), Blue (GND), Orange/Yellow (Signal)

Software Architecture

The software employs a multi-layered asynchronous architecture with clear separation between communication, control logic, and hardware abstraction.

Communication Architecture

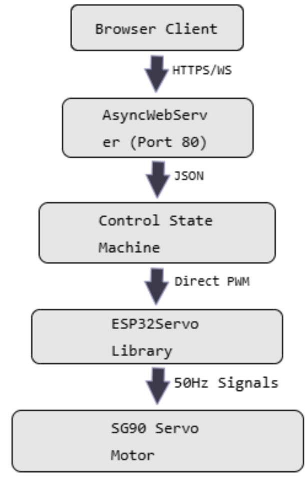
The system implements a hybrid communication model using both HTTP and WebSocket protocols:

1. Wi-Fi Access Point Mode:

```
// ESP32 creates standalone network
WiFi.softAP("ExoskeletonAP", "rehab2025");
```

- SSID: `ExoskeletonAP`
- Password: `rehab2025`
- IP Address: `192.168.4.1`

2. Protocol Stack:



3. WebSocket Implementation:

```
// Real-time bidirectional communication
AsyncWebSocket ws("/ws");
ws.onEvent(onWebSocketEvent);            //
Event-driven callbacks
```

- Bidirectional JSON messaging
- 100ms update interval for real-time control

Design Principles:

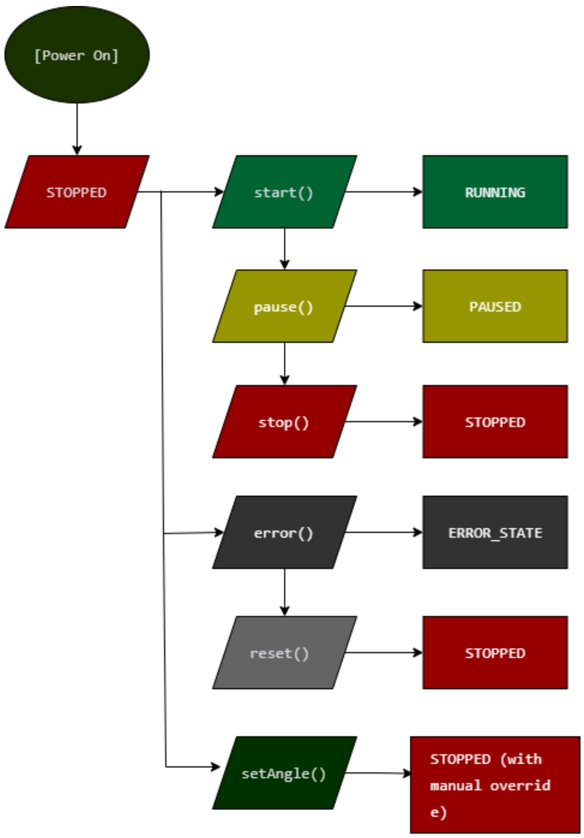
- **Neumorphic Design:** Soft shadows and rounded elements
- **Responsive Layout**: Adapts to mobile and desktop screens
- **Color Coding:** Distinct colors for different system states

Control States

The system implements a finite state machine (FSM) with four distinct states:

```
enum SystemState {
STOPPED, // System idle, servos at 90° home position
RUNNING, // Active gait pattern playback
PAUSED, // Motion paused, maintains current position
    ERROR_STATE  // System fault detected
};
```

State Transition Logic:



Gait Control Implementation:

```
// Array-based gait pattern storage
float hipLeft[100] = { /* 100 samples from OpenSim
2354 model */ };
float kneeLeft[100] = { /* 100 samples */ };
float hipRight[100] = { /* 100 samples */ };
float kneeRight[100] = { /* 100 samples */ };

// Update logic with speed control
void updateGaitSimulation() {
  if (currentState != RUNNING) return;
  int updateInterval = 50 / gaitSpeed;// 20-200ms
range
  currentData.gaitProgress = gaitCycleIndex;
   // Direct array indexing for bilateral control
  currentData.leftHip.targetAngle=
hipLeft[gaitCycleIndex];
  currentData.leftKnee.targetAngle =
kneeLeft[gaitCycleIndex];
  currentData.rightHip.targetAngle =
hipRight[gaitCycleIndex];
  currentData.rightKnee.targetAngle =
kneeRight[gaitCycleIndex];
    // Cycle detection and counting
  if (lastProgress > 95 && gaitCycleIndex < 5) {
        completedCycles++;  // Increment cycle
counter
    }
}
```

Prototyping and Implementation

This phase translated the proposed system architecture into a functional physical prototype. The focus here is on the materialization of those designs in consideration of physical integration, assembly decisions, and presentation designs.

The prototyping process followed an iterative engineering approach. Figure 2.1 demonstrates the due process the authors followed. Starting with low-fidelity sketches and progressing toward an assembled working model.

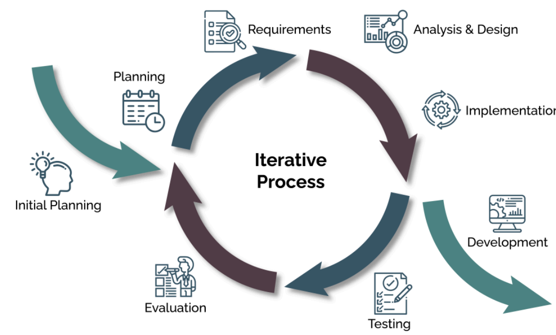
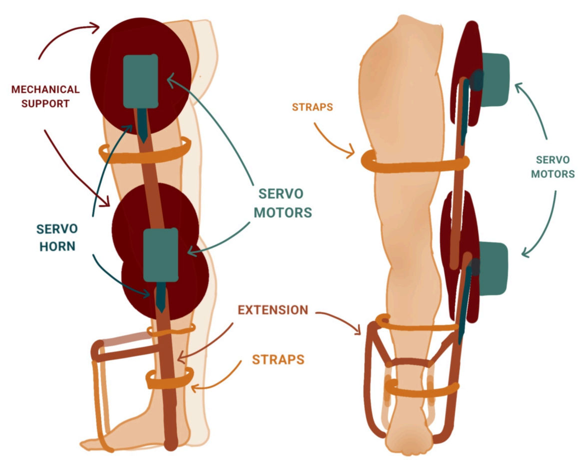


Figure 2.0. *Iterative Process Model (Radiant, n.d.)*

Initial conceptual drawings were used to establish component placement, mechanisms, and spatial relationships to assess the need for supporting structures. Figure 3.0 displays the sketches with guided naming conventions for identification and color for separation of parts. This was later on subjected to refinements during assembly to ensure compatibility between electronic components and mechanical elements.



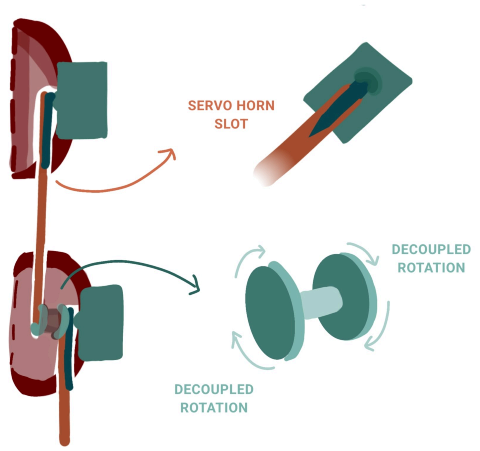
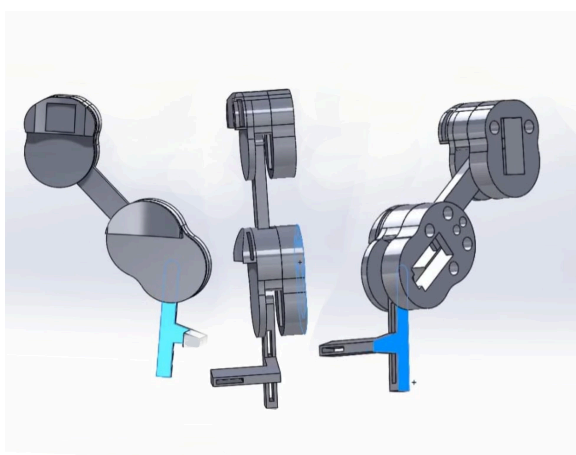


Figure 3.0. *Conceptual Designs*

Following the low-fidelity stage, select components were either obtained from (1) recycled materials, and/or (2) fabricated using 3D Printing. These artefacts were carefully chosen to serve its intended purpose such as mechanism and structural support, mechanical work initiation, and placements for actuators. 3D printed materials were modeled and printed from the authors' chosen partner, The LayerLab. The 3D Model design is presented in Figure 4.0.



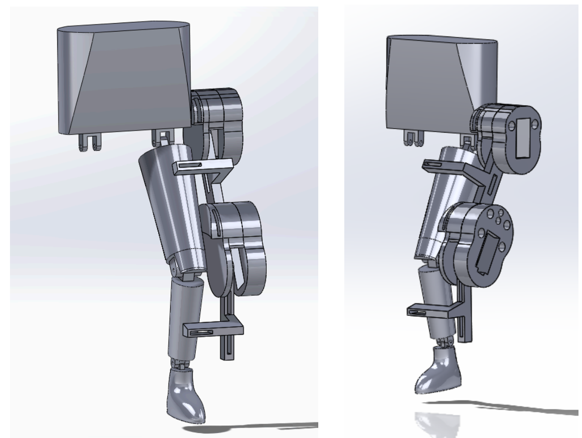


Figure 4.0. *3D Model of the Concept Design*

At the stage presented in Figure 5.0, 3D models were manifested into tangible components that shifted the authors' focus to assembly. These printed components were printed using PLA filaments for its matte finish. It was first inspected for surface quality and appropriate spaces for wiring, straps, and support for mechanical work. The primary plan presented in concept designs was adapted in establishing the structural frame. Mechanical extensions were then secured using fasteners using knots, straps, and available strings. Once securely attached to the dummy leg, electric modules and components were mounted onto pre-designed allocations or brackets. The wiring engineering was carefully managed to ensure it does not affect the simulation of gait patterns. Intermediate testing of the components, especially servo motors were tested for functionality prior to assembly. Once the prototype has been assembled, servo motors are powered and their resposes have been observed and compared against expected simulation and/or design specifications. The authpurs observed major discrepancies in movement and alignment of the leg in its first trial, and were subjected to localized adjustments and underwent subsequent trials with each having a modification. There were 15 trials performed. These repeated trials allowed the prototype to perform its operational purpose.

Figure 5.0. *3D Printed Materials by The LayerLab.*

The final assembly was performed as a fully integrated protpyupe, in consideration of visual appeal andpresentation. In order to highlight the key purpose of the project, components and additional artefacts were constructed to build an appealing prototype. These artefacts were accumulated from available materials such as boxes, styrofoams, obsolete, and recycled materials retrieved from the authors'. This consideration is intended to illustrate the authors' desire to converge design to material realization, operation to presentation, and attention to integration in translating conceptual designs into demonstrable physical systems.

Gait Data Acquisition

Our gait data was obtained from the article's mathematical modeling of normal human walking, where hip and knee joint motions over one complete gait cycle were represented using sinusoidal curve-fitting. The authors analyzed typical gait patterns and derived continuous equations for hip and knee angular position, which describe how each joint angle changes with time. These equations serve as the reference model for realistic lower-limb motion and define the expected joint behavior during different phases of the gait cycle.

To apply this model in our system, we discretized the continuous joint angle equations by sampling them at fixed time intervals across one gait cycle. The resulting angle values were organized into arrays for the left and right hip and knee joints and stored directly in the Arduino code. This approach allowed us to avoid real-time mathematical computation while still preserving the shape and timing of the gait motion described in the article. By iterating through these arrays at a constant interval, the servo motors reproduce the article's modeled gait pattern in a practical and stable manner.

User Manual

Written in this section is a beginner-friendly manual that guides the user on how to use the highlighted device on this paper.

Set-up

The ESP-32 is a module capable of generating a local network on its own even without Wi-Fi connectivity. This sub-section is dedicated to teaching a user how to set up a connection between the user's device and the ESP-32 server- the first step in making the device work.

1. Connect a power supply to the ESP-32 to power it up. The port requires a USB-B cable as its connector to function normally; avoid using other types of connector. Connect the cable to a stable power supply. It's highly recommended to use a power bank with 5V/3A output to power up the ESP-32 smoothly (with built-in connectors, preferrably).

   Warning: Do not ignore any of these signs that might damage the ESP-32 or cause unwanted sparks.
   - If the ESP-32 becomes too hot, unplug it immediately and change your power source. You might be using the wrong type of usb or it is receiving too much power than it can handle.
   - If the servo motors produce cranky sounds, change your power source. This indicates that it isn't receiving ample power.

2. Open the network/Wi-Fi section of your device. Search for a network named "`ExoskeletonAP`", once found, click the network and type

“`rehab2025`” as its password. Wait until your device is connected successfully.

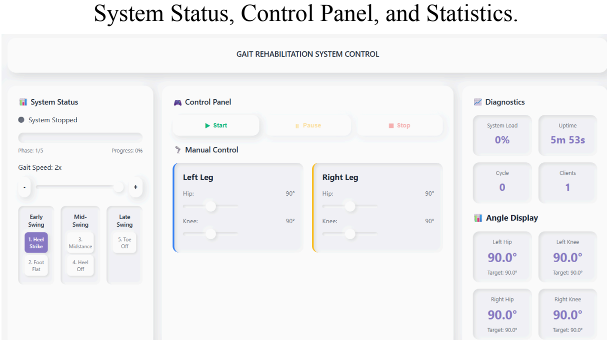
Troubleshooting:
- *No network of the same name found*- Make sure you are close to the device, otherwise, the signal will be blocked by obstacles or distance limitations.
- *Can't connect to the network*- Restarting the ESP-32 by unplugging and plugging the power source might help.
- *“No internet” prompt appeared*- This is normal, click connect anyway or ignore it if connected already.
- *Password is incorrect*- Make sure that you typed the password in lower caps.

3. Open any browser of your choice (except internet explorer) and type/copy this IP address: “xxxxxxxx”. It will redirect you to the web server/control panel of the device.

   Troubleshooting: If the webpage isn't appearing, make sure you typed the IP address/link correctly verbatim. Refresh your browser and try again if the problem persists.

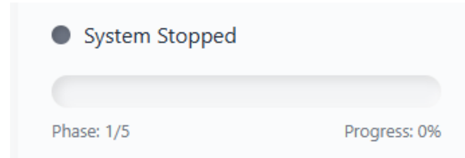
GUI Guide

This section is provided to orient the user of the functions each panel provides for operating and monitoring the device thru a Web-based GUI. This interface is divided into three main panels: System Status, Control Panel, and Statistics.

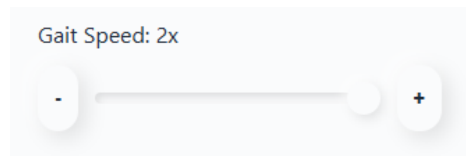


System Status

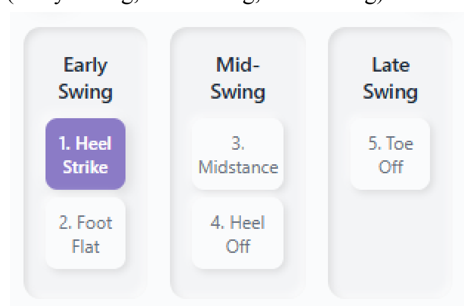
The progress bar shows completion from 0% to 100% of an ideal gait cycle,



Adjust the speed of the gait cycle from 0.2x to 2x using the slider. Use the “–” or “+” buttons for fine adjustments. This will either speed-up or slow down the device in simulating an ideal gait cycle.



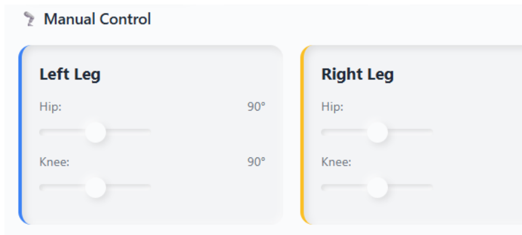
This indicates the current phase of the gait cycle (Early Swing, Mid-Swing, Late Swing).



Control Panel

Start / Pause / Stop: Use these buttons to begin, pause, or stop the gait rehabilitation session.

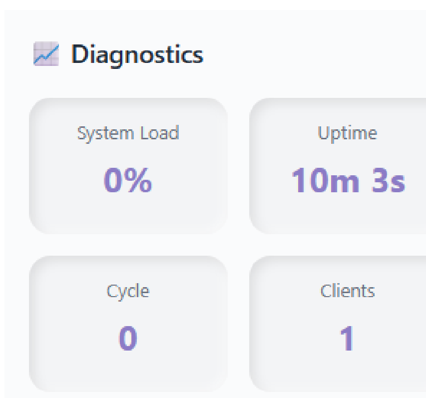
The manual control tab allows direct manipulation of leg joints. Adjust Hip and Knee angles for both Left and Right Legs using sliders. This is intended for physicians or therapists to position the legs precisely.



Diagnostics and Angle Display

Diagnostics

This tab displays the percentage of system resources currently being used presented by the system load tab. Uptime shows the total time the system has been active during the current session. Cycle Count tracks the number of complete gait cycles performed. While clients count indicates the number of connected users or devices.



Angle Display

Shows the current joint angles (Hip and Knee) for both legs, along with the target angles.

Authors' Details

Eulogio "Amang" Rodriguez Institute of Science and Technology-Manila, College of Arts and Science, Department of Bachelor of Science in Applied Physics with Computer Science Emphasis, Batch 2028

[L.M.] Lawrence Monterde
co.renso.m@gmail.com
+639283299854
https://github.com/nzo-m

[L.D.] Lexcie Jude David
lexciejudedavid@gmail.com
+639614951407
https://github.com/lxcjddvd

[D.C.] Darius Mark Roy Caparro
dariusmarkcaparro@gmail.com
+639932731726
https://github.com/darthecodemaster

[M.E.] Matthew Broderick Echavez
mateoechavez7@gmail.com
+639994497680

[A.T.] Alexa Tanala
tanala.2324A16786.bsap@gmail.com
+639857912103

Authors' Contributions

Conceptualization: L.M.; Related Study Research: L.M., L.D.; Methodology: all authors; Software: L.M., L.D., D.C.; Hardware: D.C.; Data Curation: L.M.; Engineering Design: L.M.; External Surveys: A.T., L.D., L.M., D.C.; Writing — Design Draft: L.M.; Writing — Preliminaries: M.E., A.T.; Writing — Original Draft: all authors; Writing — Review & Editing: all authors; Sponsor: all authors; Supervision: A.L.

References

1. Rose, L., Bazzocchi, M. C. F., de Souza, C., Vaughan-Graham, J., Patterson, K., & Nejat, G. (2020). *A framework for mapping and controlling exoskeleton gait patterns in both simulation and real-world.* In **Proceedings of the ASME 2020 Design of Medical Devices Conference (DMD2020)**. American Society of Mechanical Engineers.

   https://doi.org/10.1115/DMD2020-9009

2. Rathod, P., Patil, R., & Chougule, R. (2018). *Design and simulation of lower limb exoskeleton for gait rehabilitation using Arduino*. **International Journal of Health Sciences and Research**

3. Louie, D. R., & Eng, J. J. (2016). Powered robotic exoskeletons in post-stroke rehabilitation of gait: a scoping review. Journal of NeuroEngineering and Rehabilitation.

https://doi.org/10.1186/s12984-016-0162-5

4. Lee, S., Kim, J., Lee, H., & Kim, Y. (2025). A wearable ankle-assisted robot for improving gait function and pattern in stroke patients

https://doi.org/10.1186/s12984-025-01624-w