LEGARD: Technical Overview and User Guide

Lower-Extremity Guided and Assisted Rehabilitation Device

Ricardo Garcia
ricardo.garcia@cosmiac.org
University of New Mexico

January 8, 2026

Contents

1 Overview

LEGARD (Lower-Extremity Guided and Assisted Rehabilitation Device) was conceptualized to bridge the gap between clinical physical therapy and home exercises. The system digitizes the rehabilitation process, allowing for quantitative tracking of progress through sensor fusion and real-time feedback. The LEGARD is an innovative rehabilitation tool designed to aid in post-surgical hip rehabilitation by improving hip strength and range of motion. Developed by a collaborative team of UNM physical therapy students, engineering students, and recent graduates, the LEGARD project aims to provide an effective in-home alternative to traditional rehabilitative care.

2 Improvements

- **Hardware and Firmware Enhancements:** The integration of the Center of Pressure (COP) platform is a key addition to the system. The user now utilizes a rigid metal board that acts as a controller to measure stability and balance. The microcontroller firmware utilizes four analog-to-digital converters to acquire raw weight data from load cells. This data is filtered to reject abnormal spikes based on historical readings, and the Center of Pressure is calculated locally using normalized differentials, then displayed with a following trail.

- **Threading and Background Processes:** Taking advantage of a threaded architecture, the application ingests this data into a queue, associating it with timestamped angle measurements. The LEGARD system is now a true dual-input system, simultaneously processing balance data from proprietary firmware and motion data from the gyroscope angle sensor. The new approach utilizes dedicated background processes to pass data from hardware threads to the UI/Logic thread, preventing any interface freezes, which the old approach of blocking calls often caused.

- **App Architecture:** The project has successfully transitioned from a rigid, procedural script to a modular, object-oriented, multi-threaded application. The new architecture separates the user interface from hardware operations and business logic, resulting in a stable, scalable, and maintainable codebase.

- **Algorithm and Logic:** We have shifted from using physical limit switches to set exercise difficulty to an adaptive algorithm based on continuous angle sensor data. A state machine now governs the routine, intelligently detecting the start, peak, and return of the users stretches. The noisy raw data is filtered, and graphical displays apply moving averages to ensure smooth lines that glide across the screen.

- **Data Logging and Analysis:** Unstructured .txt logging has been replaced by structured .csv logging with proper headers (Set, Time, Angle, Velocity). A newly improved history section parses these .csv's to render interactive graphs for Center of Pressure and angle over time. This feature includes a timeline scrubbing tool, allowing users to click and drag to review specific moments in a session.

- **Other UI Improvements:** What once required halting the program, editing the Python source code, and restarting for any small change in hardware, user environment, or sensor behavior is now handled seamlessly via the settings menu. Furthermore, the user's PIN is now encrypted, guaranteeing confidential access to overall analytics data.

3 Quick Start User Guide

This section provides a step-by-step guide for new users, covering account setup, login, and the calibration process necessary for accurate exercise tracking.

3.1 Account Management

3.1.1 User Registration

Before starting any session, you must create a user profile if you haven't already.

1. Launch the application and navigate to the registration screen.
2. Enter a unique **Username** and a 4-digit **PIN**.
3. Provide your **First Name**, **Last Name**, and **Gender**.
4. Click **Register**. A successful registration will create a user entry and hash your PIN for security.

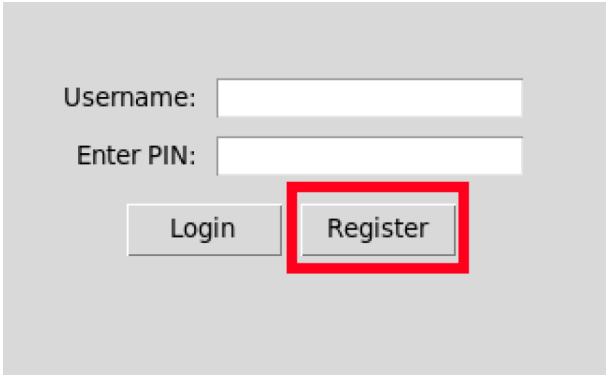

Figure 1: Register new user

Figure 2: User registration

3.1.2 User Login

To access the main Dashboard, log in with your credentials.

1. Enter your registered **Username** and **PIN**.
2. Click **Login**.
3. Upon successful validation, you will be directed to the main Dashboard, displaying your Profile, Analytics, History, and Settings tabs.

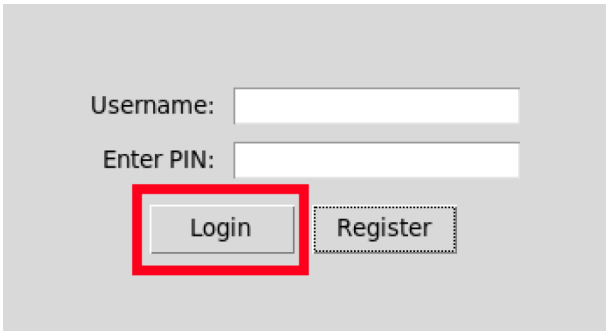

Figure 3: User login

Figure 4: Dashboard

3.2 The Routine

Accurate tracking of your exercise angle and Center of Pressure (CoP) requires establishing a baseline angle and determining your maximum range of motion.

3.2.1 Starting a New Rountine

The calibration process uses the `CalibrationWindow` to zero the sensor and find your maximum angle.

1. On the main Dashboard, navigate to the routine tab
2. Keeping the scale clear and leg lever at the starting position, start the calibration process by clicking **Start New Routine**
3. The application will now zero the leg lever and COP system.

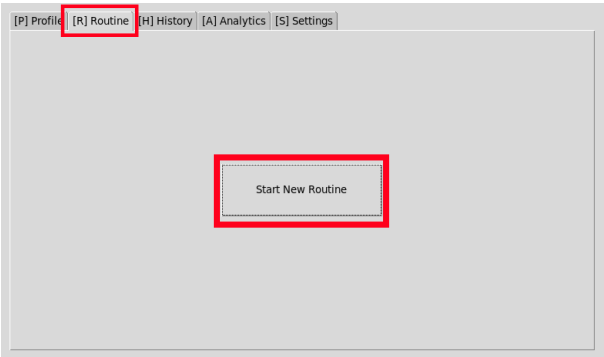

Figure 5: Starting the routine

3.2.2 Establish a Baseline

- **Instruction**: Lift your leg as high as you can.
- The system starts calculating your **relative angle** (current angle minus the `initial_angle`).
- As you move your leg, the highest positive angle reached is recorded as your `max_angle`.
- **Visual Feedback**: Monitor the on-screen progress bar. The green fill shows your current angle, and the red line indicates the maximum angle achieved.

3.2.3 Successful Calibration

1. Once you have reached your highest comfortable range of motion, click **Complete Calibration**.
2. The system saves the `initial_angle` and `max_angle`. The `max_angle` is used to calculate the **Target Angle Threshold** for your exercise routines, typically set as a percentage (e.g., 90%) of this maximum.

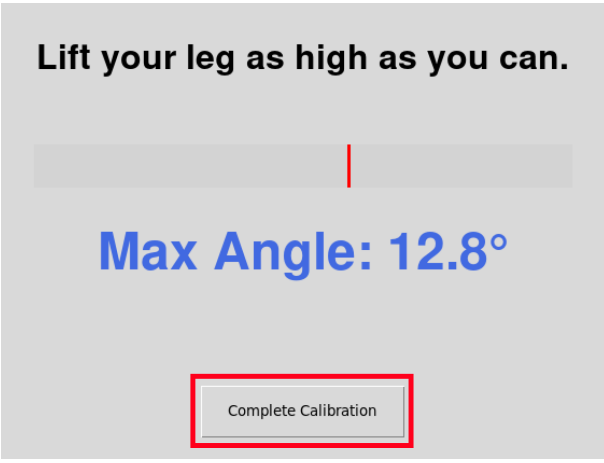

Figure 6: Calibration

3.2.4 Routine Instructions

1. Follow on screen prompts
2. Note that for each new set you will need to press the **Start** button, in the top left

3. Once session is complete, it will be automatically saved

4. To exit simply press the **Exit** in the bottom left at any time

3.3 Reviewing Session History and Analytics

Once a session is completed and saved, you can review your performance data through the **History** and **Analytics** tabs located on the main Dashboard. These tools allow for both review of specific sessions and high-level tracking of progress over time.

3.3.1 Navigating the History Tab

The History tab is designed for detailed inspection of the Center of Pressure (CoP) and Relative Angle for specific exercise sets.

1. **Select a Session**: Click the **Session** dropdown menu to view a list of all recorded sessions, sorted by date and time (most recent first).

2. **Select a Set**: Once a session is loaded, use the **Set** dropdown to choose the specific exercise set you wish to visualize.

3. **Data Visualization**:

   - **CoP Plot**: Displays the movement of your Center of Pressure. Use this to monitor stability and balance during the exercise.
   - **Relative Angle Plot**: Shows the leg lever angle over time. The dashed green line indicates your Target Angle Threshold.

4. **Interactive Scrubbing**: Click and drag your mouse across the **Relative Angle** plot. A red cursor will appear on both graphs simultaneously, allowing you to see exactly where your balance (CoP) was at any specific moment in the range of motion.

5. **Refresh**: Click **Refresh Data** to update the list if you have just finished a new session.

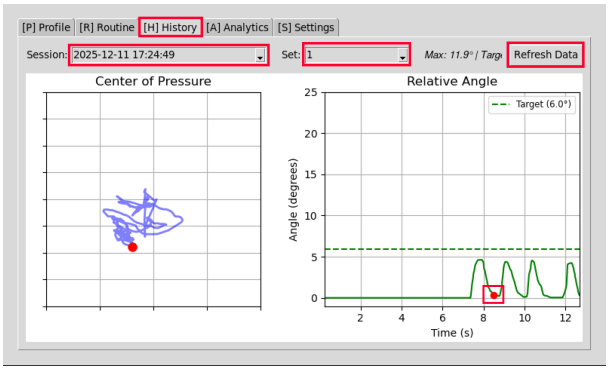

Figure 7: History tab

3.3.2 Navigating the Analytics Tab

The Analytics tab provides a trend of your training consistency and performance trends across multiple sessions.

1. **Select Metric**: Use the **View** dropdown menu to select one of the following tracking graphs:

- **Repetitions per Session**: Track the total volume of work per visit.
- **Sessions per Week**: Monitor your weekly consistency and frequency.
- **Avg Velocity/Max Angle**: Analyze the quality and intensity of your movements over time.

2. **Trend Analysis**: Use these graphs to identify improvements in range of motion or speed, which are key indicators of recovery or strength gains.

3. **Data Update**: If sessions are missing, click **Refresh Data** to re-scan your local user directory for the latest CSV logs.

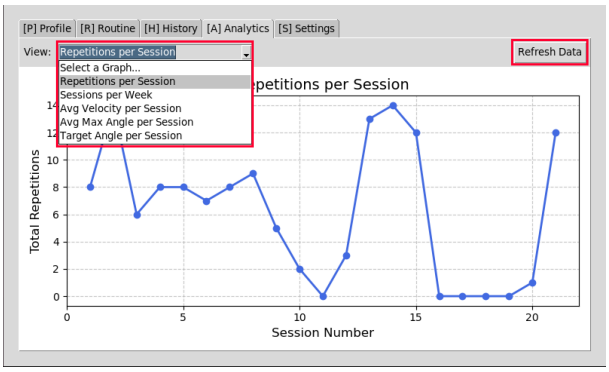

Figure 8: Analytics tab

4 Hardware Setup

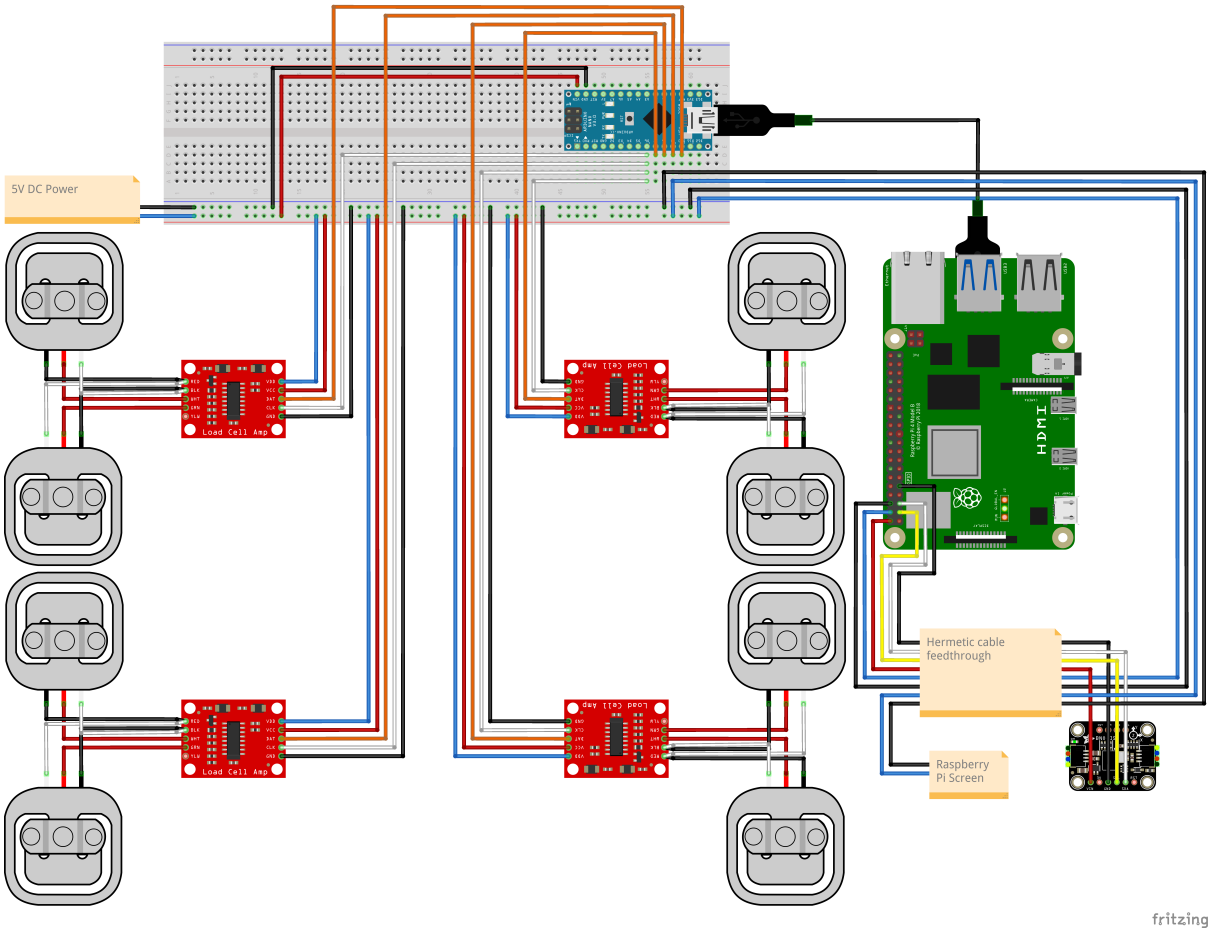

Figure 9: Full wiring diagram

The following specifications define the minimum plate thickness and dimensions utilized during testing. These parameters were selected to optimize structural rigidity and ensure maximum force transfer to the load cells.

- **Top Plate Material:** Aluminum
- **Plate Thickness:** $d = \frac{1}{4}$ in
- **Surface Dimensions:** 1 ft $\times$ 6 $\frac{5}{8}$ in

The required electronic hardware

- **Load Cell Amplifier - HX711:** `https://www.sparkfun.com/sparkfun-load-cell-amplifier-hx711.html`
- **Microcontroller - Arduino Nano:** `https://store-usa.arduino.cc/products/arduino-nano?srsltid=AfmBOoqHG8TmDiNVRxF2wcbw8GoCBR1ZO089K6dOIg9_UZ02xWuCrqgN`
- **Load Cells - Strain Gauge:** `https://www.digikey.com/en/products/detail/sparkfun-electronics/10245/5843757?gclsrc=aw.ds&gad_source=1&gad_campaignid=20232005509&gbraid=0AAAADrbLliXJS3RC2fv9q4iT9Pibizxm&gclid=Cj0KCQiAyP3KBhD9ARIsAAJLnnamyUgkKCobKhcfy1BFLCSAiQZMHj22ZVEp6isocmjUzSWKEOcI0SgaAsJoEALw_wcB`

- **IMU - BNO055:** https://www.digikey.com/en/products/detail/adafruit-industries-llc/2472/5699182?gclsrc=aw.ds&gad_source=1&gad_campaignid=20243136172&gbraid=0AAAADrbLlgS32nBB_3EcmLIWzB4bOlDX&gclid=Cj0KCQiAyP3KBhD9ARIsAAJLnnavn4jpbrmRiYHJ-2YMiWkE2-1Tp1omlcGwEvi0g0qgJa_Fr4HYoVcaAh2BEALw_wcB
- **Power Supply - 5 V:** https://www.amazon.com/SHNITPWR-Converter-Adapter-Transformer-WS2812B/dp/B07TZ2TRRB?th=1
- **Raspberry Pi 4:** https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
- **Raspberry Pi Touch Display:** https://www.raspberrypi.com/products/raspberry-pi-touch-display/

Note that by default the Sparkfun HX711 chip SJ2 is closed, data rate set to 10SPS, open the jumper to set to 80SPS, increasing the noise per readings. This is not optional is neccsarry for proper function.

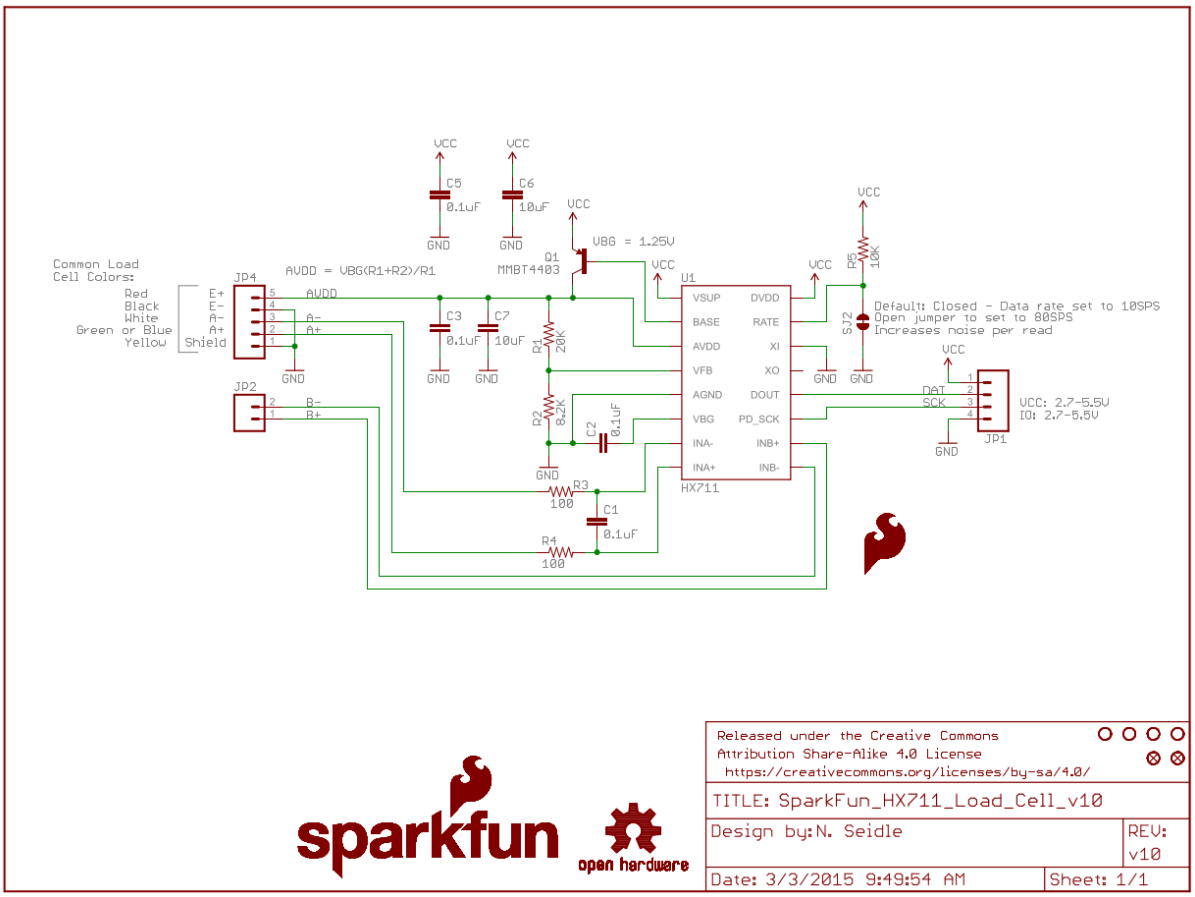

Figure 10: HX711 Schematic

5 Codebase

5.1 Full code

Codebase for this project: `https://github.com/rickwgarcia/LEGARD`.

5.2 COP Firmware

For the center of pressure firmware: `https://github.com/rickwgarcia/cop_controller`

6 Multithreading and Hardware I/O (`data_inputs.py`)

To ensure a responsive user interface, all slow hardware interactions are offloaded to dedicated background threads.

6.1 SerialThread

- **Purpose**: Manages the connection to the serial device, the center of pressure system.
- **Operation**: Continuously reads line-delimited data via `serial.readline()`.
- **Data Flow**: Received data lines are placed into a shared, thread-safe `queue.Queue` for subsequent processing.
- **Commands**: Includes a `send` method to transmit commands (e.g., 'z' for tare, 'c' for start) to the serial device.

6.2 SensorThread

- **Purpose**: Manages high-speed polling of an I2C-based Inertial Measurement Unit (IMU), such as the BNO055.
- **Operation**: Runs a loop to read the sensor's quaternion data via the `.quaternion` property.

7 Live Session Management

7.1 Calibration (`calibration_window.py`)

The calibration process is a two-step procedure that establishes the operational range for the user.

7.1.1 Step 1: Sensor Zeroing (`zero_sensor`)

- **Initial Angle**: The absolute angle is calculated based on an average of 10 quaternion W-component (q_w) readings while the user is stationary.
- **Calculation**: The absolute angle θ in degrees is derived from q_w using:

$$\theta = 2 \arccos(q_w) \cdot \frac{180}{\pi}$$

  This establishes the `initial_angle` (the $\theta = 0$ baseline).

7.1.2 Step 2: Max Angle Tracking (`track_max_angle`)

- **Relative Angle**: The relative angle is calculated as the difference between the current absolute angle and the stored `initial_angle`.
- **Maximum Range**: The peak positive value achieved during this step is recorded as the `max_angle`.
- **Visual Feedback**: A custom Tkinter canvas provides a dynamic progress bar visualization of the current and maximum angle achieved.

7.2 Data Processing and Rep Counting (`routine_window.py: DataProcessor`)

This is a critical, high-frequency thread that processes and analyzes the raw data streams.

7.2.1 Data Processing Pipeline

1. **Acquisition**: Consumes Center of Pressure (CoP) data from `data_queue` and angle data from `SensorThread`.
2. **Smoothing**: A moving average filter with a window size of `SMOOTHING_WINDOW` (configured via `config.ini`) is applied to the relative angle.
3. **Velocity**: Angular velocity is calculated from the change in smoothed angle over time. A second moving average filter (`VELOCITY_SMOOTHING_WINDOW`) is applied to this velocity.

7.2.2 Rep Counting State Machine

The exercise detection logic is a 4-state state machine:

- **State 0 (Ready)**: Waiting to start. Transitions to State 1 if velocity exceeds `VELOCITY_POS_THRESHOLD`.
- **State 1 (Positive Movement)**: Records the peak angle achieved during the upward motion. Transitions to State 3 upon reversal (velocity falls below `VELOCITY_NEG_THRESHOLD`).
- **State 2 (Negative Movement)**: Records the peak angle achieved during the downward motion. Transitions to State 3 upon reversal (velocity rises above `VELOCITY_POS_THRESHOLD`).
- **State 3 (Reversal/Transition)**: A repetition is counted if the smoothed velocity returns to a near-zero state (not moving) and the recorded peak angle (`max_angle_for_current_rep`) is greater than the `target_angle_threshold`.

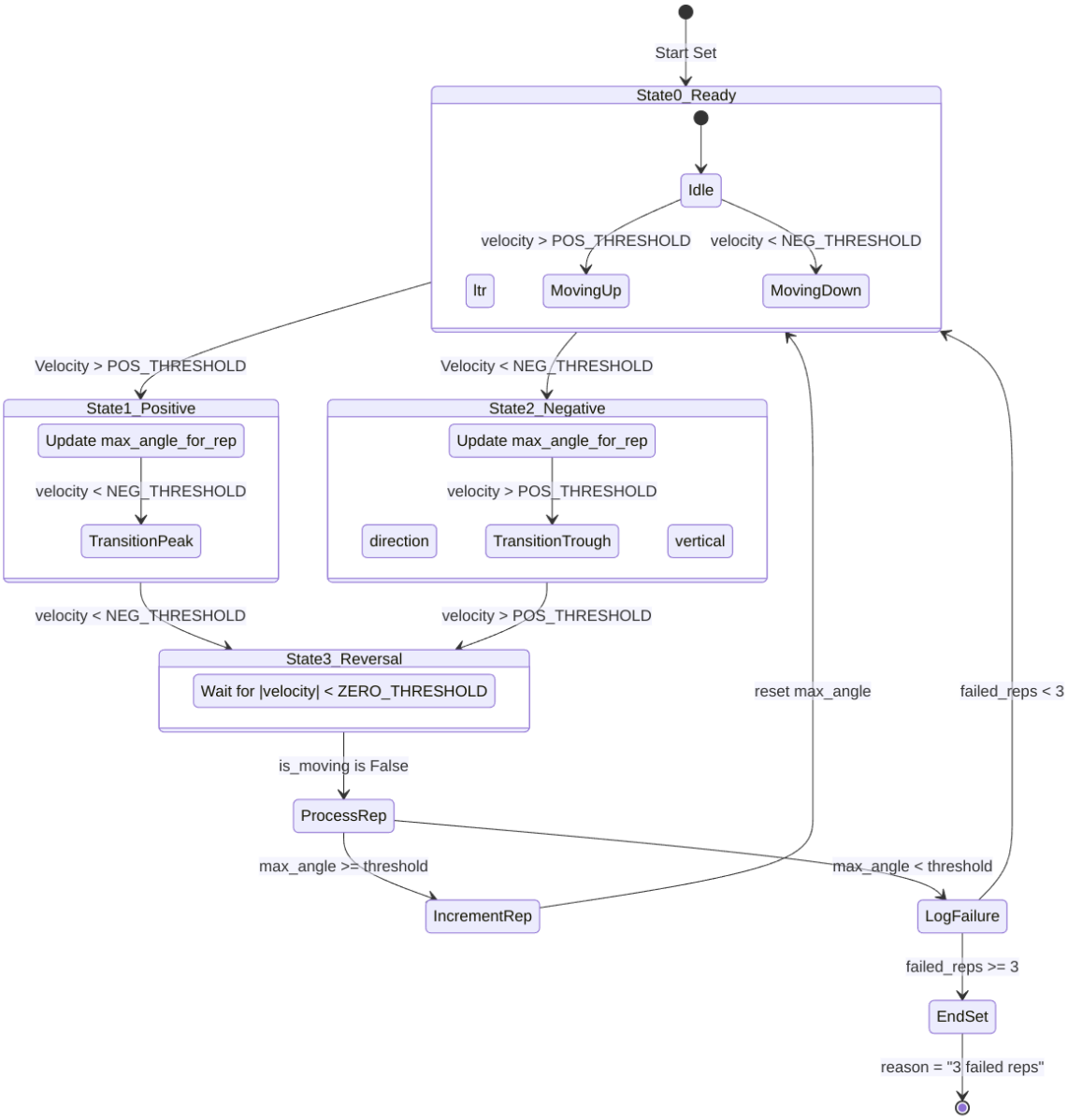

Figure 11: State Machine Diagram

7.2.3 Target Threshold

The `target_angle_threshold` is dynamically calculated from the user's calibration `max_angle`:

$$\text{Target Angle} = \text{max\_angle} \cdot \left(\frac{\text{MAX\_ANGLE\_TOLERANCE\_PERCENT}}{100} \right)$$

7.2.4 Data Logging

Processed data (Time, Angle, Velocity, CoP coordinates, Rep Count, Set Number) is written to a time-stamped CSV file, including metadata for the maximum and target angles.

7.3 Real-time Visualization (`routine_window.py: RoutineWindow`)

- **Animation**: Uses Matplotlib's `FuncAnimation` to update plots at a high frequency.

- **Plots**: Displays two plots side-by-side: Center of Pressure (CoP) X-Y coordinates and Relative Angle vs. Time.
- **Scrolling**: The Angle plot displays a fixed history length defined by `TIME_WINDOW_SECONDS`.
- **Set Flow**: Manages inter-set breaks using the modal `RestTimerWindow`.

8 Historical Analysis and Management

8.1 History Tab (`history_tab.py`)

- **Data Selection**: Scans the user's session directory for `datalog_*.csv` files and allows selection by session and individual set.
- **Interactive Scrubbing**: The primary feature is a linked cursor that updates markers on both the CoP and Angle plots based on the mouse's time-position on the Angle plot.
- **Data Synchronization**: The function `update_cursors` finds the nearest data point in time and uses its corresponding angle and CoP coordinates for the dot positions.

8.2 Analytics Tab (`analytics_tab.py`)

- **Data Aggregation**: The `parse_history` function extracts and calculates session-level statistics, including total repetitions, average positive velocity, average maximum angle per rep, and session frequency (weekly and monthly).
- **Visualization**: Provides line plots over chronological **Session Number** for metrics such as "Repetitions per Session" and "Avg Velocity per Session" using the `draw_line_plot_over_time` function.

8.3 Settings Tab (`settings_tab.py`)

- **Configuration UI**: Provides a user interface for modifying `config.ini` parameters across the Serial, RepCounter, and Plotting sections.
- **File Access**: Allows the user to refresh the list of available serial ports and open the user's session log folder in the native file explorer.
- **Persistence**: The `save_settings` function retrieves values from Tkinter variables, updates the in-memory `config` object, and writes changes back to `config.ini`.

9 Authentication and User Data

9.1 Authentication Manager (`auth_manager.py`)

- **Storage**: User profiles are stored in a local CSV file specified by `USER_DATA_FILE`.
- **Hashing**: User PINs are stored securely as a hash of the user pin.
- **Fields**: The stored data includes `username`, `hashed_pin`, `first_name`, `last_name`, and `gender`.