Robot-assisted stroke/ankle rehabilitation

Table of Contents

Introduction 2
Aim 2
Literature Review 2
Methodology 3
    Block Diagram 3
    Electrode placement 4
    Data Acquisition 5
    Data Analysis and Filtering 7
    Feature extraction 8
    Machine learning model 9
Real time Implementation 10
Limitations and Improvement: 11
Novel Ideas of the project (Future) 11
        Bioinspired Robotic Designs for Enhanced Rehabilitation 11
        Virtual Reality Gamification for Enhanced Rehabilitation 12
Conclusion 12
References 12
Appendix 15
ANN Model: 15
Real time implementation (testing with the new signal) 15

Introduction

Stroke-induced injuries often result in severe motor deficits and functional limitations, with the ankle joint being particularly susceptible to impaired mobility and impaired gait. (Cruz et al., 2009) In response to the urgent need for effective neurorehabilitation interventions, integrating robotic technologies is gaining increasing attention, offering a promising avenue for personalized and targeted treatment for individuals aiming to regain mobility and independence. In this context, the development of advanced robotic-assisted stroke/ankle rehabilitation systems is a critical step in providing comprehensive, adaptive solutions to the complex challenges faced by stroke survivors during the rehabilitation process.

Aim

The purpose of this project is to create a framework algorithm to be applied to provide intensive, comprehensive, and novel resistance therapy for patients with movement disorders and muscle atrophy due to stroke and foot drop. The ability to control and rotate a servo motor in real time using EMG data from the subject. (Franklin & Su, 2023) By developing an algorithm that connects human movement to mechanical motors, it is hoped that this can one day be applied to the design and manufacture of biomedical instruments used to assist stroke patients in ankle rehabilitation.

Literature Review

It has been shown that robot-assisted rehabilitation has a significant role in promoting improved motor function and gait parameters in stroke survivors. They were analysed in the literature, highlighting the positive impact of robotic interventions in enhancing neuroplasticity and promoting functional recovery in stroke survivor populations (Lo et al., 2017).

Advanced technologies in the field of robotic rehabilitation have proven versatile and effective in addressing the complex needs of stroke survivors, providing tailored interventions that cover all aspects of motor recovery, including gait training and upper and lower extremity rehabilitation (Cano-de-la-Cuerda & Alguacil-Diego, 2013). Additionally, the study highlights the importance of patient engagement and adherence to robotic rehabilitation programs, highlighting the interactive feedback mechanisms and gamified exercises in increasing patient motivation and active participation throughout the rehabilitation process (Farshid Amirabdollahian et al., 2014). These journals' evidence emphasizes the growing importance of robotic-assisted stroke/ankle rehabilitation systems, highlighting their potential to revolutionize the field of neurorehabilitation and improve the overall quality of life for people recovering from stroke-induced injuries.

Project specification

Hardware requirements:

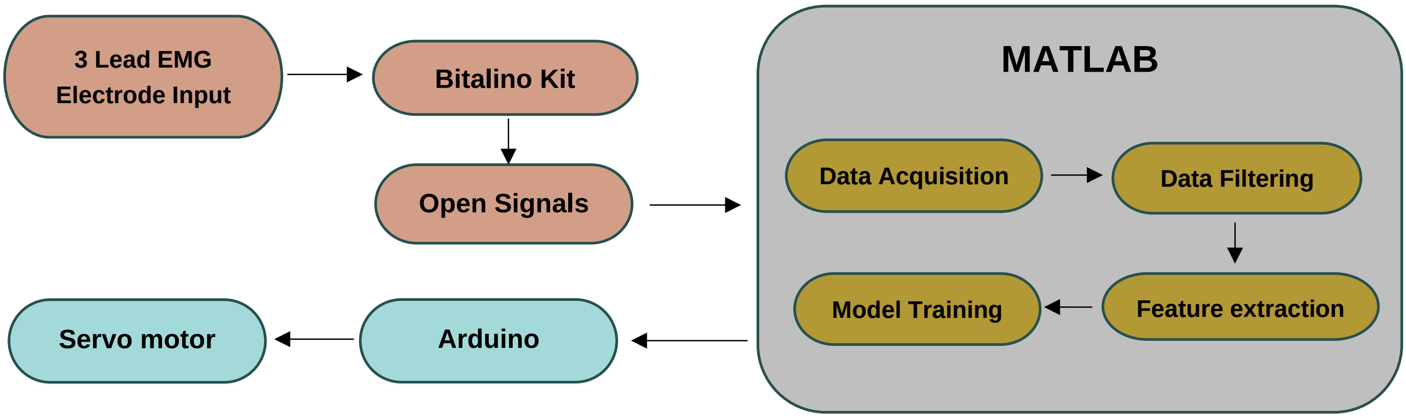
- Computer
- Servo Motor- SG90
- Arduino Uno
- BITalino (r)evolution kit with Bluetooth
- EMG sensor
- 3-Lead Electrode Cable
- EMG gel type electrodes

Software requirements:

- Open signals (for Bitalino) for PC
- Arduino IDE for PC
- MATLAB for PC
- Arduino support package in MATLAB
- EMG Feature extraction Toolbox in MATLAB
- Deep Learning Toolbox in MATLAB

Methodology

Block Diagram



Electrode placement

The data acquistion of the EMG signals is an essential aspect of our project to correctly determine the different signals collected corresponding to the 3 different positions of the ankle. The unique placements of the electrode is in accordance with the research during the literature review (EMG Practicum 1: Electrode location and placement. nd). These placements are on different muslces in the leg and ankle to accurately stimulate the correct signals of the muscles.

Figure 1: 3 lead EMG electrode

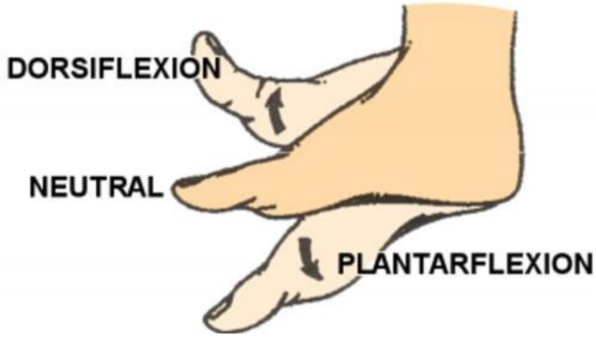
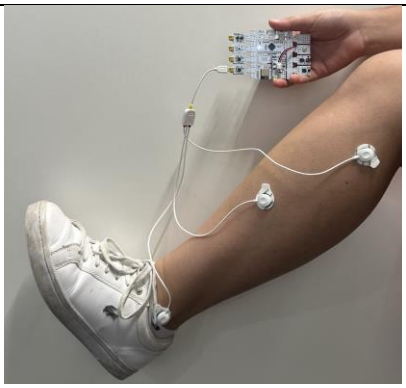
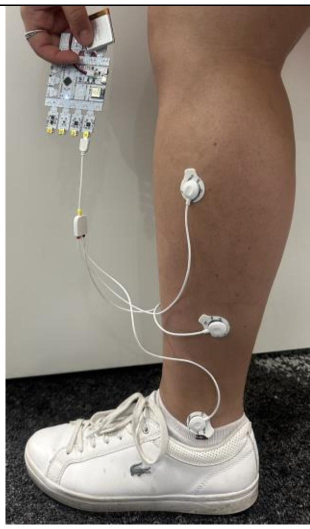


Figure 2: Ankle movement

Position of ankle	Description	Photo of electrode placement
Dorsiflexion of ankle joint	Ankle dorsiflexion is the extension of the foot towards the shin and helps in activities such as walking and running. This maneuver involves muscles such as the tibialis anterior and extensor digitorum longus. It is vital for gait and balance, prevents foot drop, and targeted exercises can improve lower extremity function and mobility.	 *Figure 3: Dorsiflexion*

Plantar flexion of ankle joint	Plantarflexion of the ankle involves plantarflexion of the foot downward, increasing the angle between the foot and the leg. This maneuver works the calf muscles, including the gastrocnemius and flounder muscles, which aid in movements such as walking and propel the body forward during activities such as running and jumping.	 *Figure 4: Plantar Flexion*
Rest position of the ankle	The relaxed posture of the ankle involves neutral alignment with the foot at rest, neither flexed nor extended. The muscles are balanced to ensure stability and balance at rest.	 *Figure 5: Rest position*

Data Acquisition

Data acquisition is a fundamental step in the data preprocessing stage as it ensures data is in a suitable format for further processing and analysis. After finalizing the muscle to take the signal from we held the position for minimum of 5 seconds and took 10 sets of data for each position. The EMG electrode placed on the muscle is connected to the Bitalino kit using a cable. Open signals is a platform which is used to control and acquire the signals from the EMG sensor in the Bitalino kit. The data signals acquired using open signals are saved in the computer. The signals are saved in the H5 (Hierarchical Data Format (HDF)) format. H5 is a binary format that can compress and access data much more efficiently than text formats, which is especially useful when dealing with large datasets. The data collected at the sampling frequency of 1000hz.

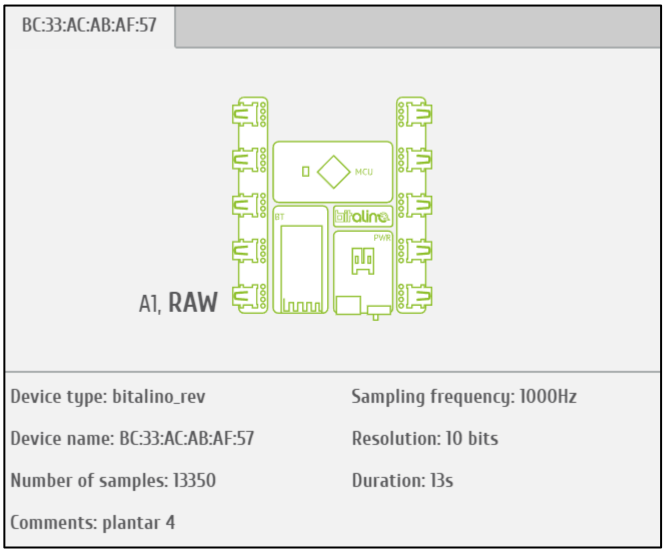


Figure 6: Channel used in the Bitalino and sampling frequency.

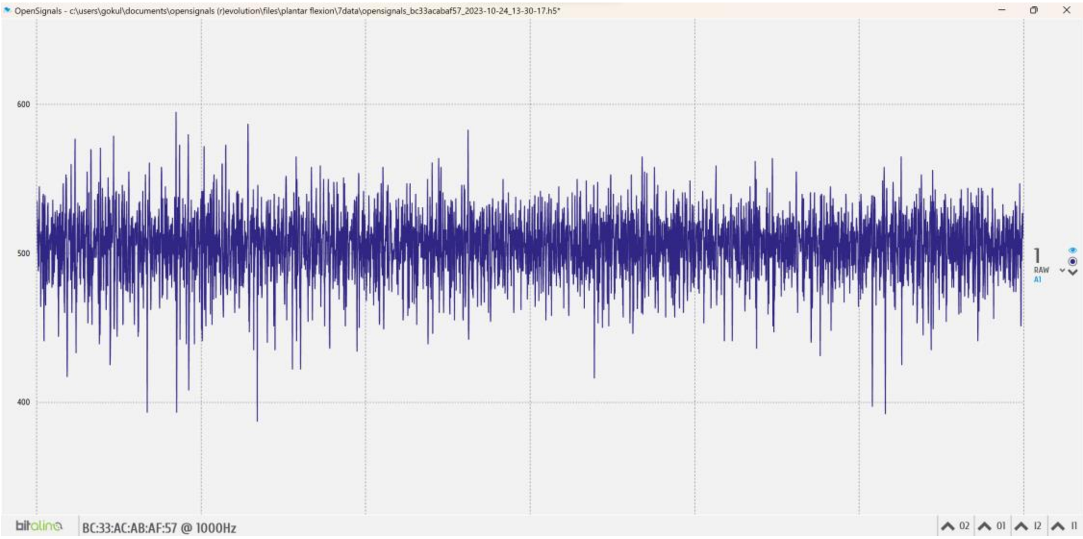


Figure 7: Raw Plantarflexion signal

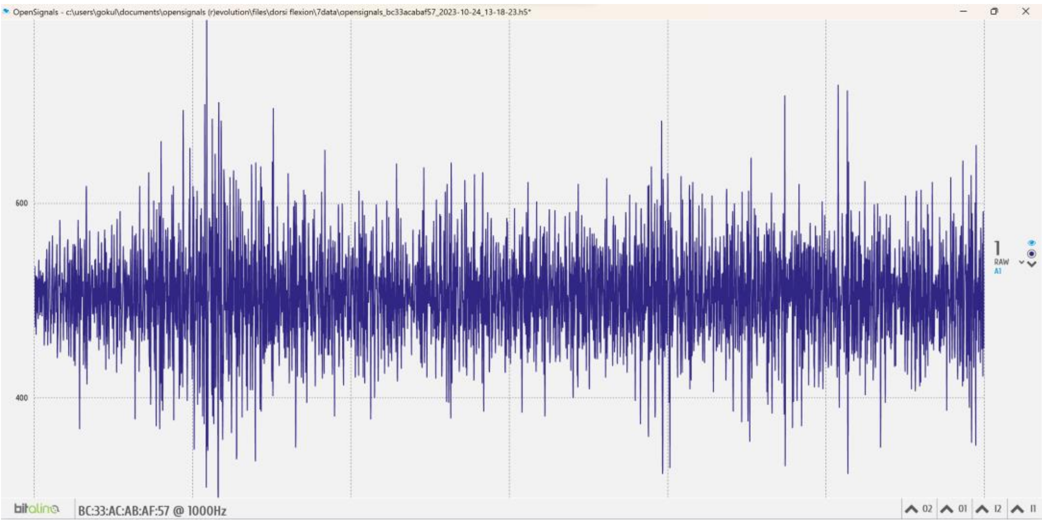


Figure 8: Raw Dorsiflexion signal

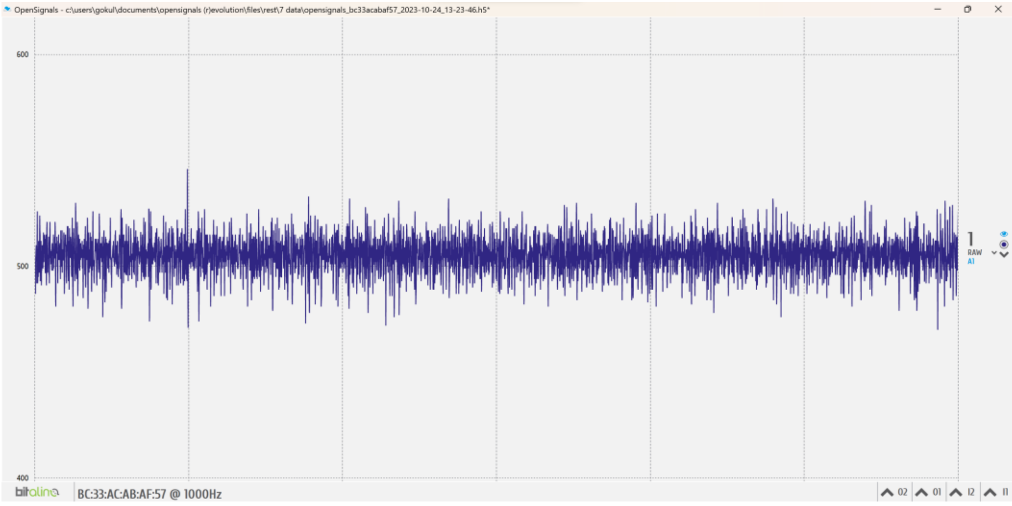


Figure 9: Raw Rest position signal

Data Analysis and Filtering

After data acquisition we will load the stored raw signals onto MATLAB and analyze it. Below are the raw waveforms of different signals seen in time domain before filtering. After visualizing we will filter the signal to remove the unwanted spikes and concentrations in the signals. First, we will do a bandpass filter to remove the frequency below 10hz and after 450hz as we got a big spike at the start and to keep all the signals trimmed to same length. Then we will do apply a notch filter. Usually, the notch filter is used to remove the spike at 50hz due to electronics interference but in our signal, we had spikes at an interval of 50hz throughout the signal. So, we applied notch with the modification that includes a start frequency-50hz and end frequency- 450hz and interval 50hz. This modification in the code will allow us to remove the spike throughout the signal at 50hz intervals. Then we convert the signal from time domain to frequency domain. Doing FFT (Fast Fourier Transform) allows visualization of the frequency content of the filtered data. Before giving the signal to the next step the data length of all the signals are different. So, we make changes to the data to make it to the closest multiple of 100’s. This will be extremely useful in the feature extraction step.

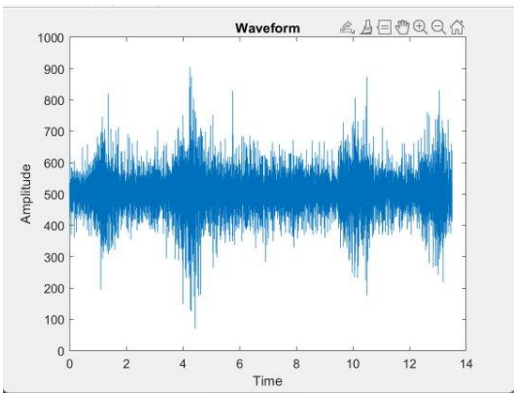


Figure 10: Raw Dorsiflexion signal

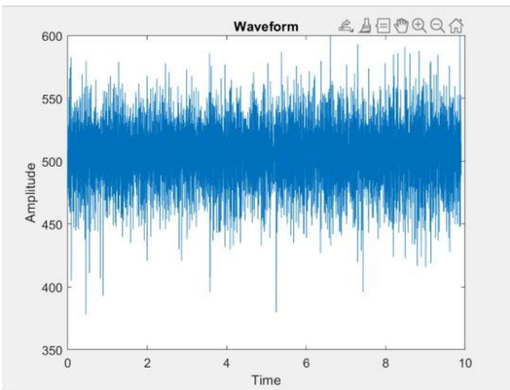


Figure 11: Raw Plantarflexion signal

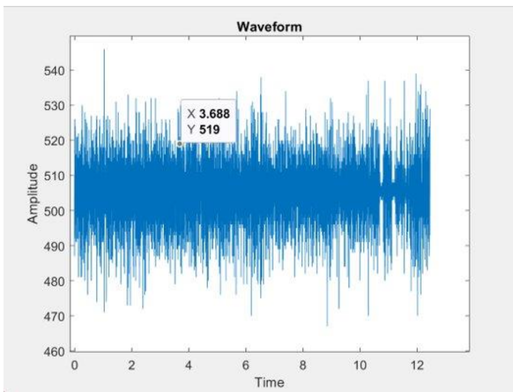


Figure 12: Raw Rest signal

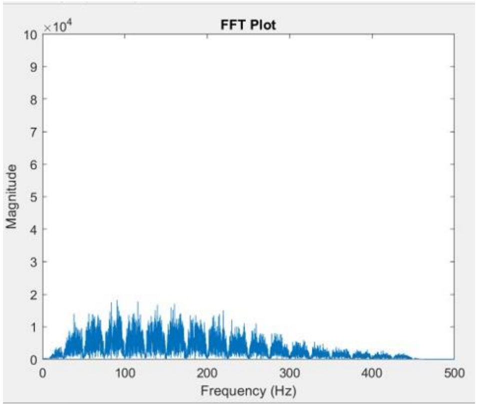


Figure 13: Filtered Dorsiflexion signal

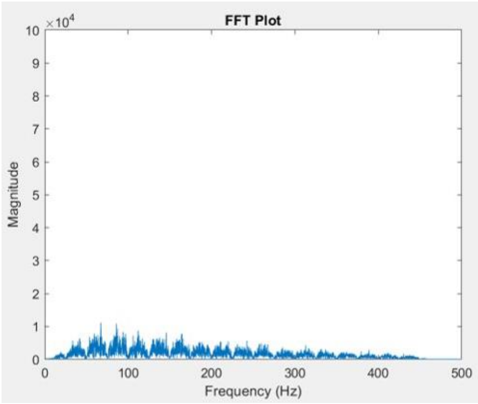


Figure 14: Filtered Plantarflexion signal

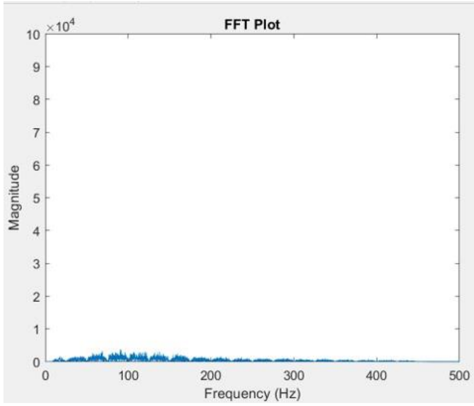


Figure 15: Filtered Rest signal

Feature extraction

To make sense of EMG signals, we extract features. A messy EMG signal can be reduced to useful information by extracting its features. The features of EMG signals help us analyze and understand muscle activity. As a result, raw data can be transformed into more manageable, tangible data that can be analyzed and recognized better. We are extracting 10 features which are commonly extracted for EMG signals. The features extracted are,

- Root Mean Square (RMS)
- Variance (VAR)
- Mean Absolute Value (MAV)
- Standard Deviation (SD)
- Zero Crossing (ZC)
- Integrated EMG (IEMG)
- Simple Square Integral (SSI)
- Waveform Length (WL)
- Willison Amplitude (WA)
- Slope Sign Change (SSC).

We extract the features using the EMG feature extraction toolbox in the MATLAB. The features are extracted for every 100 data points. This process makes it easier to analyze the EMG signal without using too much data, making the information easier to work with. That’s why we cropped the signal to its nearest multiple of 100. We also label 1-Plantarflexion, 2-Rest, and 3-Dorsiflexion the features extracted from each different to identify them.

RMS	VAR	MAV	SD	ZC	IEMG	SSI	WA	WL	SSC	Movement
15.19423	231.3259	11.71364	15.2094	36	1171.364	23086.47	99	1244.736	45	1
15.63603	246.4252	12.3044	15.69793	27	1230.44	24448.55	99	1122.067	45	1
18.2378	335.6146	14.49323	18.31979	28	1449.323	33261.75	99	1422.327	49	1
17.34476	303.8491	13.57269	17.43127	35	1357.269	30084.08	99	1366.144	42	1
18.51117	345.7243	12.96868	18.59366	33	1296.868	34266.33	99	1189.094	42	1

Table 1: feature extracted from plantar signal.

RMS	VAR	MAV	SD	ZC	IEMG	SSI	WA	WL	SSC	Movement
6.292598	39.97943	5.289738	6.322929	27	528.9738	3959.68	99	439.5328	41	2
6.12247	37.86315	4.371792	6.153304	33	437.1792	3748.464	98	458.8686	44	2
8.115496	66.51357	6.733001	8.155585	27	673.3001	6586.127	99	532.9543	37	2
6.314284	40.26093	4.810204	6.34515	31	481.0204	3987.018	99	474.2781	39	2
7.683684	59.60475	6.026598	7.720411	28	602.6598	5903.901	99	518.0327	36	2

Table 2: feature extracted from rest signal.

RMS	VAR	MAV	SD	ZC	IEMG	SSI	WA	WL	SSC	Movement
69.85308	4927.798	55.00724	70.19828	33	5500.724	487945.2	99	5340.825	46	3
80.51314	6547.817	65.57478	80.91858	30	6557.478	648236.5	99	5952.176	41	3
80.52321	6547.169	64.34833	80.91458	28	6434.833	648398.7	99	6041.353	42	3
85.52087	7385.955	68.69889	85.94158	33	6869.889	731382	99	6584.165	41	3
104.656	11050.45	79.21477	105.1211	33	7921.477	1095289	99	7657.075	40	3

Table 3: feature extracted from Dorsi signal.

Machine learning model

Feature extraction was used to select and prepare a training data set. Machine Learning will learn how to solve the problem from that data as we feed it to it. Labelling our data will allow the model to determine its characteristics and whether it is correct. Choosing an algorithm for the training dataset is the second step. Training data determines the algorithm and the problem to be solved. Lastly, the algorithm must be trained. Iteratively, this process takes place. The algorithm runs variables, and the results are compared to what is expected. By adjusting the "weights" and bias of the model, the accuracy of the results can be increased. In most cases, the algorithm will produce the correct result after rerunning the variables. Machine Learning is a trained algorithm.

It is now time to use and improve the model. Different data sets will allow our model to learn and become more accurate. Specifically, several hidden layers are used in pattern recognition neural networks. We chose an Artificial neural network (ANN) for pattern recognition of EMG signals since we were trying to identify different movements. As a result, the network can identify similarity and redundancy in the data. Here we use 70% of the data to train the model, 10% validate and 10% for testing it.

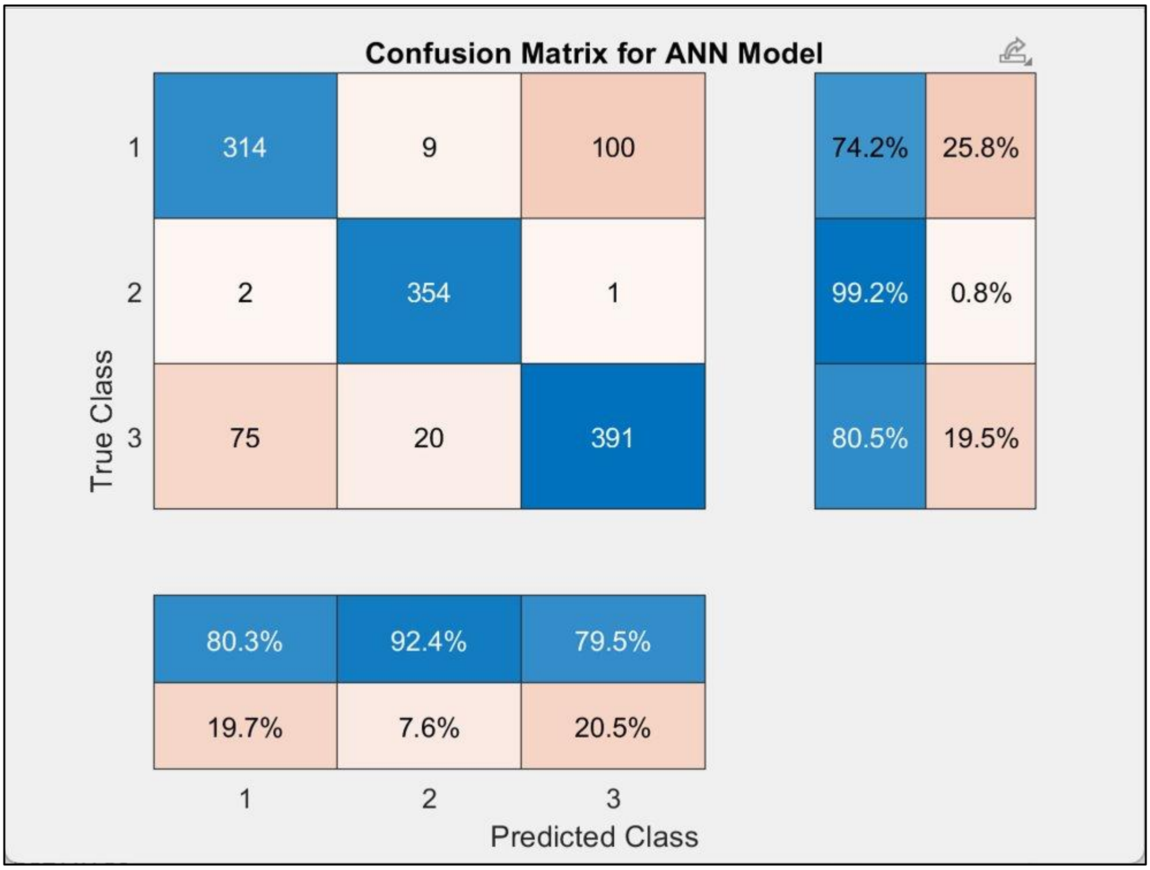
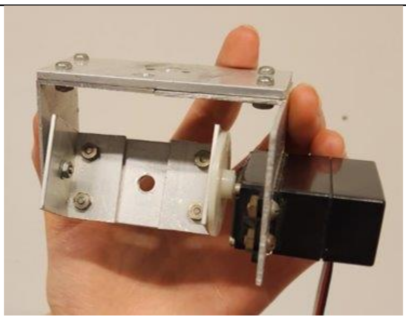
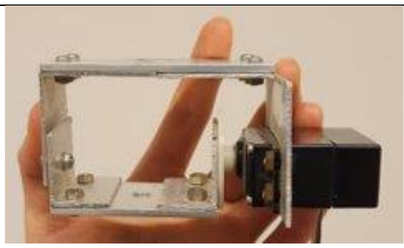
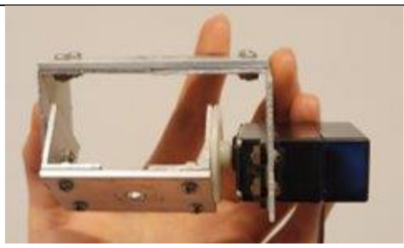


Figure 16: Confusion Matrix

Real time Implementation

After training the model, we connected the servo motor, which was pre-calibrated for 3 different angles. The wire that controls the servo motor is linked to the D3 pin on the Arduino, and this connection is also set up in the programming code. Once we made sure the motor is moving, we gave a new signal which was recorded and stored just for testing our model. The new signal was analyzed, filtered and feature extracted. Now it is fed the model to predict, and the output was seen with the motor moving to predetermined angles as shown in the images below.

Position of ankle	Description	Position of servo motor
Plantar flexion of ankle joint	45º	 *Figure 17: Plantar Flexion*

Rest position of the ankle	0º	 *Figure 18: Rest position*
Dorsiflexion of ankle joint	-45º	 *Figure 19: Dorsiflexion*

```
Command Window
Size of the ANN input layer: 10
Size of the test data: 10   124
The identified classes are: 2
Current motor position is 72 degrees
```

Figure 20: the model predicting the correct movement of the ankle from the given new signal.

Limitations and Improvement:

We couldn't connect the Bitalino to MATLAB to make the servo motor move in real time as there were compatible issues with Bitalino and MATLAB. We tried analyzing alternate ways to make a connection across them and found LSL (Lab Streaming Layer) to be effective as we can stream the EMG signal taken from the Bitalino straight to the MATLAB for processing.

Novel Ideas of the project (Future)

Bioinspired Robotic Designs for Enhanced Rehabilitation

The incorporation of bioinspired robotic designs in our robot-assisted stroke/ankle rehabilitation system aims to replicate natural human biomechanics and enhance the overall rehabilitation experience. Biomimetic actuators enable lifelike movements, while compliant mechanisms prioritize patient safety and comfort during interactions. Optimization of biomechanics and kinematics ensures efficient and intuitive rehabilitation exercises, fostering a user-friendly environment. Advanced sensorimotor integration facilitates personalized therapy, adapting to individual progress and needs.

By fostering familiarity and comfort, our system promotes patient confidence and active engagement, ultimately contributing to a more effective and engaging rehabilitation process for stroke survivors.

Virtual Reality Gamification for Enhanced Rehabilitation

The integration of Virtual Reality (VR) gamification in our robot-assisted stroke/ankle rehabilitation system introduces interactive and immersive gaming elements, fostering engaging and effective therapeutic experiences. This approach promotes patient motivation and adherence to the rehabilitation program through enjoyable and purposeful gameplay activities. Real-time performance monitoring and customizable rehabilitation modules within the VR platform enable personalized and adaptive therapy sessions tailored to individual patient needs. By leveraging multisensory stimulation and interactive gameplay, VR gamification contributes to neuroplasticity, facilitating motor learning and cognitive recovery. Through the implementation of VR gamification, our project aims to establish a dynamic and patient-centric rehabilitation platform, enhancing both physical recovery and overall well-being for stroke survivors.

Conclusion

In conclusion, using EMG data, this project successfully developed a framework to control a servo motor. This indicates potential applications in ankle rehabilitation for stroke patients. The systems were developed with various hardware and software elements, including an Arduino Uno, an SG90 servo motor, a BITalino (revolution kit), and MATLAB for feature extraction and data analysis. We trained a machine-learning model capable of controlling motor position in real-time with electrode placement, filtering, and processing of EMG signals. Although we encountered challenges with real-time data streaming between BITalino and MATLAB, we overcame this with alternative solutions like LSL. Future work includes exploring bioinspired robotic designs and virtual reality gamification to enhance rehabilitation outcomes.

References

1. *(PDF) EMG Practicum 1: Electrode location and placement · EMG Practicum 1: Electrode location and placement . Objectives . Learning to - Find the correct muscles - Find the correct electrode*. (n.d.). Dokumen.tips. Retrieved November 8, 2023, from https://dokumen.tips/documents/emg-practicum-1-electrode-location-and-placement-emg-practicum-1-electrode-location.html?page=1

2. Cano-de-la-Cuerda, R., & Alguacil-Diego, I. M. (2013). Emerging Therapies in Stroke Technologies and Neurorehabilitation: State of the Art. *Biosystems & Biorobotics*, *1*, 1091–1095. https://doi.org/10.1007/978-3-642-34546-3_179
3. Cruz, T. H., Lewek, M. D., & Dhaher, Y. Y. (2009). Biomechanical impairments and gait adaptations post-stroke: Multi-factorial associations. *Journal of Biomechanics*, *42*(11), 1673–1677. https://doi.org/10.1016/j.jbiomech.2009.04.015
4. Farshid Amirabdollahian, Ates, S., Basteris, A., Cesario, A., Buurke, J. H., Hermens, H. J., Hofs, D., Johansson, E., Mountain, G., Nasrin Nasr, Nijenhuis, S. M., Prange, G. B., Rahman, N., Sale, P., Schatzlein, F., B.W. van Schooten, & Arno. (2014). Design, development and deployment of a hand/wrist exoskeleton for home-based rehabilitation after stroke - SCRIPT project. *Robotica*, *32*(8), 1331–1346. https://doi.org/10.1017/s0263574714002288
5. Franklin, D. R., & Su, S. (2023, August 30). *49261 Biomedical Instrumentation Major Project*. Login.uts.edu.au. https://canvas.uts.edu.au/courses/28552/pages/project-descriptions?module_item_id=1413591
6. Lo, K., Stephenson, M., & Lockwood, C. (2017). Effectiveness of robotic assisted rehabilitation for mobility and functional ability in adult stroke patients. *JBI Database of Systematic Reviews and Implementation Reports*, *15*(12), 3049–3091. https://doi.org/10.11124/jbisrir-2017-003456
7. Spiewak, C.*, Islam, M.R., Zaman, M.A.-U., & Rahman, M.H. (2018). A Comprehensive Study on EMG Feature Extraction and Classifiers. Volume(1).
8. Azhiri, R. B., Esmaeili, M., & Nourani, M. (Year). EMG-Based Feature Extraction and Classification for Prosthetic Hand Control
9. Phinyomark A, N Khushaba R, Scheme E. Feature Extraction and Selection for Myoelectric Control Based on Wearable EMG Sensors. Sensors (Basel). 2018 May 18;18(5):1615. doi: 10.3390/s18051615 . PMID: 29783659; PMCID: PMC5982518.

10. Fundarò, C., Giardini, A., Maestri, R., Traversoni, S., Bartolo, M., et al. (2018). Motor and psychosocial impact of robot-assisted gait training in a real-world rehabilitation setting: A pilot study. PLoS One, 13(2), e0191894. https://doi.org/10.1371/journal.pone.0191894
11. Brambilla, C., Pirovano, I., Mira, R. M., Rizzo, G., Scano, A., et al. (2021). Combined Use of EMG and EEG Techniques for Neuromotor Assessment in Rehabilitative Applications: A Systematic Review. Sensors, 21(21), 7014. https://doi.org/10.3390/s21217014
12. Shi, B., Chen, X., Yue, Z., Yin, S., Weng, Q., et al. (2019). Wearable Ankle Robots in Post-stroke Rehabilitation of Gait: A Systematic Review. Frontiers in Neurorobotics. https://doi.org/10.3389/fnbot.2019.00063
13. Raez MB, Hussain MS, Mohd-Yasin F. Techniques of EMG signal analysis: detection, processing, classification and applications. Biol Proced Online. 2006;8:11-35. doi: 10.1251/bpo115. Epub 2006 Mar 23. Erratum in: Biol Proced Online. 2006;8:163. PMID: 16799694; PMCID: PMC1455479.
14. https://au.mathworks.com/help/signal/ref/butter.html
15. https://au.mathworks.com/matlabcentral/fileexchange/71514-emg-feature-extraction-toolbox
16. https://github.com/JingweiToo/EMG-Feature-Extraction-Toolbox/blob/master/README.md
17. https://au.mathworks.com/help/matlab/matlab_prog/access-data-in-a-cell-array.html
18. https://au.mathworks.com/matlabcentral/fileexchange/53983-bitalino-toolbox
19. https://au.mathworks.com/help/signal/ref/digitalfilter.html
20. https://au.mathworks.com/help/matlab/ref/fft.html
21. https://au.mathworks.com/help/signal/ug/classify-arm-motions-using-emg-signals-and-deep-learning.html
22. https://au.mathworks.com/help/stats/neural-networks-for-classification.html

Appendix

ANN Model:

```
load('Processed_Plantar_Rest_Dorsi_Train_Data.mat');
load('Processed_Plantar_Rest_Dorsi_test_Data.mat');

xTrain = final_data(:,[1:10])';
xTest = final_data_test(:,[1:10])';

yTrain = full(ind2vec(final_data(:,11)'));
yTest = full(ind2vec(final_data_test(:,11)'));

hiddenLayerSize = 10;
net = patternnet(hiddenLayerSize);

% Train the neural network
[net, tr] = train(net, xTrain, yTrain);

% Predict using the ANN model
yPred = net(xTest);

% Convert predictions to the most probable class
yPredClasses = vec2ind(yPred);
yTestClasses = vec2ind(yTest);

% Evaluate the model
confMat = confusionchart(yTestClasses, yPredClasses);
confMat.Title = 'Confusion Matrix for ANN Model';
confMat.ColumnSummary = 'column-normalized';
confMat.RowSummary = 'row-normalized';
```

Real time implementation (testing with the new signal)

```
clear all;
clc;


folder_path = 'C:\Users\gokul\Documents\OpenSignals (r)evolution\files\testsignals';
file_list = dir(fullfile(folder_path, '*.h5'));
test_signal_struct.all_data = {};

for file = 1:length(file_list)
    file_name = fullfile(folder_path, file_list(file).name);

    % Reading the data
    data = h5read(file_name, '/BC:33:AC:AB:AF:57/raw/channel_1');
    data = double(data);

  % Plotting it in time domain
    time = (0:(length(data) - 1)) / fs;
    figure;
    plot(time, data);
    title('Waveform');
    xlabel('Time');
```

```
    ylabel('Amplitude');

    fh = 450;
    fl = 10 ;
    fs = 1000;

    % Bandpass
    [b,a] = butter(6, [fl,fh]/(fs/2));
    data(1, :) = filtfilt(b, a, data(1, :));


    % Notch Filter
    start_frequency = 50;
    end_frequency = 900;
    notch_interval = 50;
    filtered_data = data(1, :);
    for notch_frequency = start_frequency:notch_interval:end_frequency
        notch_frequency_normalized = notch_frequency / fs;
        bandwidth = 0.01;
        [b, a] = iirnotch(notch_frequency_normalized, bandwidth);
        filtered_data = filter(b, a, filtered_data);
    end



    % FFT
    fs = 1000;
    n = length(filtered_data);
    freq = (0:(n/2-1)) * (fs / n);
    p1 = fft(filtered_data(1, :));
    fft_magnitude = abs(p1);
    y_axis_limit = 100000;
    figure;
    plot(freq, fft_magnitude(1:n/2));
    ylim([0, y_axis_limit]);
    title('FFT Plot');
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');

% Truncate the data length to be a multiple of 100
   test_signal_struct.all_data{file} = filtered_data;
    data_length = length(test_signal_struct.all_data{file});
    if mod(data_length, 100) ~= 0
        elements_to_remove = mod(data_length, 100);
        test_signal_struct.all_data{file} =
test_signal_struct.all_data{file}(1:data_length - elements_to_remove);
    end
   end
save('test_signal_struct.mat', 'test_signal_struct');

% feature extraction
features_test_signal = {};

for file = 1:length(test_signal_struct.all_data)
    current_data = test_signal_struct.all_data{file};
    len = length(current_data);
    temp_features = [];

    for startIdx = 1:100:len
        endIdx = min(startIdx + 99, len);
```

```
        segment = current_data(startIdx:endIdx);

        rms_feature = jfemg('rms', segment);
        var_feature = jfemg('var', segment);
        mav_feature = jfemg('mav', segment);
        sd_feature = jfemg('sd', segment);
        zc_feature = jfemg('zc', segment);
        iemg_feature = jfemg('iemg', segment);
        ssi_feature = jfemg('ssi', segment);
        wa_feature = jfemg('wa', segment);
        wl_feature = jfemg('wl', segment);
        ssc_feature = jfemg('ssc', segment);

        temp_features = [temp_features; [rms_feature, var_feature, mav_feature,
sd_feature, zc_feature, iemg_feature, ssi_feature, wa_feature, wl_feature,
ssc_feature]];
    end

    features_test_signal{file} = temp_features;
end
features_test_signal = vertcat(features_test_signal{:});
save("test_signal_Data.mat", "features_test_signal");


% Prediction by ANN model
load('trained_ANN_model.mat', 'net','tr');
load('test_signal_Data.mat', 'features_test_signal');

xTest = features_test_signal';
disp(['Size of the ANN input layer: ', num2str(net.inputs{1}.size)]);
disp(['Size of the test data: ', num2str(size(xTest))]);

yPred = net(xTest);

yPredClasses = vec2ind(yPred);
mouv=num2str(yPredClasses (1,1));

disp(['The identified movement is: ', mouv ]);

% Connect the arduino uno
arduino = arduino('COM8', 'Uno', 'Libraries', 'Servo');
s = servo(arduino, 'D9');
if mouv=="1"
    angle = 0.6 ;
end
if mouv== "2"
    angle = 0.4;
end
if mouv == "3"
    angle = 0.1;
end

writePosition(s, angle);
current_pos = readPosition(s);
current_pos = current_pos*180;
fprintf('Current motor position is %d degrees\n', current_pos);
pause(2);
```