

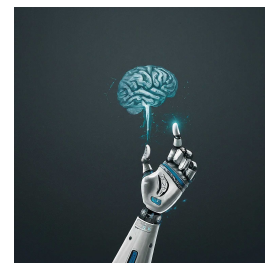
Mar 10, 2025

Version 2

🌐 Mind Controlled Bionic Arm with Sense of Touch [8 class version] V.2

DOI

<https://dx.doi.org/10.17504/protocols.io.n92ldr869g5b/v2>



Dhruva Shaw¹, Arittrabha Sengupta¹, Jay Baswaraj Khaple², Dr. Raam Dheep²

¹Creative Net; ²Lovely Professional University



Dhruva Shaw

Creative Net

Create & collaborate more with a free account

Edit and publish protocols, collaborate in communities, share insights through comments, and track progress with run records.

Create free account

OPEN  ACCESS



DOI: <https://dx.doi.org/10.17504/protocols.io.n92ldr869g5b/v2>

Protocol Citation: Dhruva Shaw, Arittrabha Sengupta, Jay Baswaraj Khaple, Dr. Raam Dheep 2025. Mind Controlled Bionic Arm with Sense of Touch [8 class version]. **protocols.io** <https://dx.doi.org/10.17504/protocols.io.n92ldr869g5b/v2> Version created by **Dhruva Shaw**

License: This is an open access protocol distributed under the terms of the **Creative Commons Attribution License**, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Protocol status: Working

We use this protocol and it's working

Created: March 08, 2025

Last Modified: March 10, 2025

Protocol Integer ID: 124019

Keywords: Bionic Arm, Robotics, Biotechnology, Mind Control, Prosthetics, mind controlled bionic arm with sense, mind controlled bionic arm, bionic arm, proposed bionic arm, effective bionic arm design, natural hand movement, tactile feedback, hand function in individual, integrated sensory feedback, restoring hand function, restoring complex hand function, sense of touch, tactile experience, servo motor control, touch sensor, offering sensory feedback, complex hand function, harnessing eeg signal, crucial for safe object manipulation, sensory feedback, touch, arm, using brain signal, pwm commands for servo motor control, gripper, controlled functionality, safe object manipulation, control mechanism, control potential, bionic technology, advancements in bionic technology, amputation, brain signal, paralysis, based control mechanism

Disclaimer

This is an ongoing project, so a lot of changes and variations in results are to be expected!

Abstract

Advancements in bionic technology are transforming the possibilities for restoring hand function in individuals with amputations or paralysis. This paper introduces a cost-effective bionic arm design that leverages mind-controlled functionality and integrates a sense of touch to replicate natural hand movements. The system utilizes a non-invasive EEG-based control mechanism, enabling users to operate the arm using brain signals processed into PWM commands for servo motor control of the bionic arm. Additionally, the design incorporates a touch sensor (tactile feedback) in the gripper, offering sensory feedback to enhance user safety and dexterity.

The proposed bionic arm prioritizes three essential features:

1. Integrated Sensory Feedback: Providing users with a tactile experience to mimic the sense of touch (signals directly going to the brain). This capability is crucial for safe object manipulation by arm and preventing injuries
2. Mind-Control Potential: Harnessing EEG signals for seamless, thought-driven operation.
3. Non-Invasive Nature: Ensuring user comfort by avoiding invasive surgical procedures.

This novel approach aims to deliver an intuitive, natural, and efficient solution for restoring complex hand functions.



Downloading of Datasets from Internet & Arranging them in a Proper Format

- 1 Please acquire the data associated with the following datasets::

Dataset

MILimbEEG: An EEG Signals Dataset based on Upper and Lower Limb

NAME

<https://data.mendeley.com/datasets/x8psbz3f6x/2>

LINK

Dataset

Supporting data for "EEG datasets for motor imagery brain comput

NAME

<https://gigadb.org/dataset/100295>

LINK

- 2 Extract it and keep the 2 datasets into 2 separate folder
In the **MILimbEEG dataset** run the following python file (it will sort the data into 8 different categories)
<https://github.com/Dhruvacube/Mind-Control-Bionic-Arm/tree/main/datasets/datasourceDatasets/MILimbEEG%20An%20EEG%20Signals%20Dataset%20based%20on%20Upper%20and%20Lower%20Limb%20Task%20During%20the%20Execution%20of%20Motor%20and%20Motorimagery%20Tasks>

Command

MindLimbEEG data sorter in python (Windows 11)

```
from pathlib import Path
import pandas as pd

task_dict = {1: 'BEO', 2: 'CLH', 3: 'CRH', 4: 'DLF', 5: 'PLF', 6:
'DRF', 7: 'PRF', 8: 'Rest'}
task_var_dict = {1: 0, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0, 8: 0}
folders_list = [Path(f'./S{i}') for i in range(1,60+1)]

for j in range(60):
    for k in [i for i in folders_list[j].glob('*.csv')]:
        for h in k.parts[1]:
            if h in ('M', 'I'):
                next_index: int = (k.parts[1].index('M' if h == 'M'
else 'I')) + 1
                task_var_dict[int(k.parts[1][next_index])] += 1

                df = pd.read_csv(k)
                df = df.drop(df.columns[0], axis=1)
                df.to_csv(f'./{task_dict[int(k.parts[1]
[next_index])]}_{task_dict[int(k.parts[1]
[next_index])]}_{task_var_dict[int(k.parts[1][next_index])]}_csv',
index=False, header=False)

print(f'BEO: {task_var_dict[1]}\nCLH: {task_var_dict[2]}\nCRH:
{task_var_dict[3]}\nDLF: {task_var_dict[4]}\nPLF:
{task_var_dict[5]}\nDRF: {task_var_dict[6]}\nPRF:
{task_var_dict[7]}\nRest: {task_var_dict[8]}')
```

Categories this python file sorts into:

- Recording a Baseline with Eyes Open (BEO) without any task command: only once at the beginning of each run.
- Closing Left Hand (CLH): five times per run.
- Closing Right Hand (CRH): five times per run.
- Dorsal flexion of Left Foot (DLF): five times per run.
- Plantar flexion of Left Foot (PLF): five times per run.
- Dorsal flexion of Right Foot (DRF): five times per run.



- Plantar flexion of Right Foot (PRF): five times per run.
- Resting in between tasks (Rest): after each task.

Now, similarly goto the **GigadbEEG dataset** folder and run the following python file:

<https://github.com/Dhruvacube/Mind-Control-Bionic-Arm/tree/main/datasets/datasourceDatasets/GigaDb>

Command

GigaDB class Sorter (Windows 11)

```
# Data to extract
# - movement_left
# - movement_right
# - imagery_left
# - imagery_right
# - rest

# next is nose
# - noise
# It is a 5x1 cell array in matlab

import scipy.io
import pandas as pd

for i in range(1, 53):
    mat = scipy.io.loadmat(f's{i}.mat')
    mat_eeg = mat['eeg']

    for j in range(0, 5):
        df_noise = pd.DataFrame(mat_eeg['noise'][0,0][j,0])
        df_noise.to_csv(f'noise/noise_{i}_{j}.csv', index=False,
header=False)
        print(f'noise/noise_{i}_{j}.csv')

        df_movement_left = pd.DataFrame(mat_eeg['movement_left'][0, 0])
        # df_movement_left =
df_movement_left.drop(labels=df_movement_left.columns[0], axis=1)

        df_movement_right = pd.DataFrame(mat_eeg['movement_right'][0, 0])
        # df_movement_right =
df_movement_right.drop(df_movement_right.columns[0], axis=1)

        df_imagery_left = pd.DataFrame(mat_eeg['imagery_left'][0, 0])
        # df_imagery_left =
df_imagery_left.drop(df_imagery_left.columns[0], axis=1)

        df_imagery_right = pd.DataFrame(mat_eeg['imagery_right'][0, 0])
        # df_imagery_right =
df_imagery_right.drop(df_imagery_right.columns[0], axis=1)
```

```
df_rest = pd.DataFrame(mat_eeg['rest'][0, 0])
# df_rest = df_rest.drop(df_rest.columns[0], axis=1)

df_movement_left.to_csv(f'movement_left/movement_left_{i}.csv',
index=False, header=False)
print(f'movement_left/movement_left_{i}.csv')

df_movement_right.to_csv(f'movement_right/movement_right_{i}.csv',
index=False, header=False)
print(f'movement_right/movement_right_{i}.csv')

df_imagery_left.to_csv(f'imagery_left/imagery_left_{i}.csv',
index=False, header=False)
print(f'imagery_left/imagery_left_{i}.csv')

df_imagery_right.to_csv(f'imagery_right/imagery_right_{i}.csv',
index=False, header=False)
print(f'imagery_right/imagery_right_{i}.csv')

df_rest.to_csv(f'rest/rest_{i}.csv', index=False, header=False)
print(f'rest/rest_{i}.csv')
```

Note

In GigaDB for this case we will take only the Rest class data

In this **GigaDB dataset** take the segregated **Rest data** and mixed with the **MindLimb Rest data**.

Conversion to Audio Files (single channels) from CSV Files

- 3 In the sorted data run the following python file to get it segregated into different channels and convert it into **audio file in .wav format**

Command

CSV to Single Channel Audio File Converter (Windows 11)

```
import pandas as pd
import soundfile as sf
import os

fs = 125 # sampling frequency
destinationFolder = "./audioFiles/"
sourceFolder = "./orderedDatasets/"

for root, dirs, files in os.walk(sourceFolder):
    if len(files) > 0:
        classCurrent = root[len(sourceFolder):]
        try:
            os.makedirs(destinationFolder + classCurrent)
        except Exception as e:
            pass
        for file in files:
            df: pd.DataFrame =
pd.read_csv(sourceFolder+classCurrent+'/'+file, header=None)
            columns = list(df.columns)
            for column in columns:
                sf.write(destinationFolder + classCurrent + '/' +
file[:-4] + '_' + str(column) + '.wav', df[column], fs)
```

Training of Model (Transfer Learning from Yamnet)

- 4 Now open up the **MATLAB** and run the following file to train the **yamnet** for our *specific use case* and then test it and generate the confusion matrix.

Take the supporting files from here: <https://github.com/Dhruvacube/Mind-Control-Bionic-Arm/>

Command

Model Training and Testing (Transfer Learning from Yamnet) (Windows 11)

```
%start from here
addpath(fullfile('yamnet'))
fs = 125; %since sampled at 125Hz

adsSource = audioDatastore("D:\projects\Research\Mind Control Bionic
Arm\datasets\audioFiles\","IncludeSubfolders=true,LabelSource="folderna
mes",FileExtensions=[".wav"]);
[adsTrain,adsValidation,adsTest] =
splitEachLabel(adsSource,0.7,0.2,0.1,"randomized");

trainLabels = adsTrain.Labels;
classNames = unique(adsTrain.Labels);
numClasses = numel(classNames);
testLabels = adsTest.Labels;

net = audioPretrainedNetwork("yamnet",NumClasses=numClasses);

% Extract features using YAMNet
adsTrain = transform(adsTrain,@audioPreprocess, "IncludeInfo",true);
adsValidation = transform(adsValidation,@audioPreprocess,
"IncludeInfo",true);
adsTest = transform(adsTest,@audioPreprocess, "IncludeInfo",true);

miniBatchSize = 128;
validationFrequency = floor(numel(trainLabels)/miniBatchSize);
options = trainingOptions('sgdm', ...
    InitialLearnRate=3e-4, ...
    MaxEpochs=2, ...
    MiniBatchSize=miniBatchSize, ...
    Shuffle="every-epoch", ...
    Plots="training-progress", ...
    Metrics="accuracy", ...
    Verbose=false, ...
    LearnRateSchedule="exponential", ...
    ValidationData=adsValidation, ...
    ValidationFrequency=validationFrequency, ...
    ExecutionEnvironment="parallel-auto");
```

```
net = trainnet(adsTrain,net,"crossentropy",options);

YTest = minibatchpredict(net,adsTest);
YTestFinal = scores2label(YTest,classNames);
plotconfusion(testLabels,YTestFinal);

[C,order] = confusionmat(testLabels,YTestFinal);
stats = statsOfMeasure(C, 1);
```

Confusion Matrix									
Output Class	BEO	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	CLH	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	CRH	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	DLF	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	DRF	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	PLF	192 1.6%	958 8.0%	960 8.1%	960 8.1%	960 8.1%	960 8.1%	5954 50.0%	8.1% 91.9%
	PRF	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
	Rest	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	NaN% NaN%
		0.0% 100%	0.0% 100%	0.0% 100%	0.0% 100%	0.0% 100%	100% 0.0%	0.0% 100%	8.1% 91.9%
		BEO	CLH	CRH	DLF	DRF	PLF	PRF	Rest
		Target Class							

Confusion Matrix of the Trained Model



Protocol references

Citation

Victor Asanza , Daniel Montoya , Leandro Leonardo Lorente-Leyva , Diego Hernán Peluffo-Ordóñez , Kléber González
(2023)
. MILimbEEG: An EEG Signals Dataset based on Upper and Lower Limb Task During the Execution of Motor and Motorimagery Tasks.
Mendeley Data.

<https://doi.org/10.17632/x8psbz3f6x.2>

LINK

Citation

Cho H; Ahn M; Ahn S; Kwon M; Jun SC (2017)
. Supporting data for "EEG datasets for motor imagery brain computer interface". GigaDB Dataset.

<https://doi.org/10.5524/100295>

LINK

Citation

(2024). Transfer Learning with Pretrained Audio Networks. Mathworks Help Center.

<https://in.mathworks.com/help/audio/ug/transfer-learning-with-pretrained-audio-networks.html>^{LINK}



Citation

(2024). yamnetPreprocess. Mathworks Help Center.

<https://in.mathworks.com/help/audio/ref/yamnetpreprocess.html>^{LINK}

Citation

(2024). audioDatastore. Mathworks Help Center.

<https://in.mathworks.com/help/audio/ref/audiodatastore.html>^{LINK}

Citation

Google, Dan Ellis, Manoj Plakal (2020). YAMNet. GitHub.

<https://github.com/tensorflow/models/tree/master/research/audioset/yamnet>^{LINK}

Citation

Jon Hamilton (2021). Scientists Bring The Sense Of Touch To A Robotic Arm. NPR.

<https://www.npr.org/sections/health-shots/2021/05/20/998725924/a-sense-of-touch-boosts-speed-accuracy-of-mind-controlled-robotic>

^{LINK}



Citation

Almabrok Essa, Hari Kotte (2021)

. Brain Signals Analysis Based Deep Learning Methods: Recent advances in the study of non-invasive brain signals. arXiv.

<https://doi.org/10.48550/arXiv.2201.04229>

LINK