HUMAN-ROBOT COOPERATION USING EEG SIGNALS WITH SELF-LEARNING

by

Sunny Arokia Swamy Bellary

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Electrical Engineering

Charlotte

2019

Approved by:

Dr. James M. Conrad

Dr. Andrew R. Willis

Dr. Thomas P. Weldon

ABSTRACT

SUNNY AROKIA SWAMY BELLARY. Human-robot Cooperation using EEG signals with Self-learning. (Under the direction of DR. JAMES M. CONRAD)

Able-bodied humans express and communicate their needs and ideas with each other through the exchange of verbal and non-verbal information. However, those with neuromuscular disorders, either inherited or acquired by other factors, have high needs and no practical way to communicate those necessities. Today, human interaction with machines and robots are common. Humans can control and interact with robots with relative ease. This communication is very difficult for people with neuromuscular disorders. Disabled people are often assisted by robots which have now become popular and more accessible, but still a struggle to communicate with the assistive robot. The purpose this research aims to build the communication bridge between assistive robots and disabled humans and provide them basic control abilities such as object selection tasks. This research focuses on Brain-Computer Interfaces (BCI) via Electroencephalography (EEG) based communication. EEG sensors were used because they are cost-effective and non-invasive devices. Error-Related Potentials (ErrP) signals, which are a sub-component of EEG signals are used since they are naturally occurring within the brain in response to an unexpected error. ErrP signals are used as the sole communication channel from human to robot. In this work, we showcase that the ErrP signals from the stimulated human brain are translated in real-time into binary control signals which can control PR2 robot by correcting its errors to perform a particular task. Here we consider the task to be the selection and sorting of household objects. A robot initially based on random selection decides upon the category of the objects. EEG signals are collected from a simulated human brain and are decoded in real time; if an ErrP is detected the robot immediately corrects its trajectory and classifies the object to another group, otherwise it continues in the planned trajectory. With this continuous process, the robot is being self-trained by the disabled for the object selection and classification task. With this work, the disabled will be able to communicate with the robot and train it without any external human aid.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. James M. Conrad for all his constant support towards completion of my thesis. His extensive knowledge and long experience allowed me to conduct this research in excellent conditions. I further thank Dr. Andrew R. Willis who guided me with his multi-disciplinary and abundance knowledge by providing endless advice and injected an ability in me to work on challenging tasks. I would also like to thank Dr. Thomas P. Weldon who trained me to gain ability to see the research in various forms and taught me the ways to present and document the research which plays a major role. I feel honored to have them as my committee members for my defense.

I thank my fellow labmates for providing me help in every possible way, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last 18 months. I would like to thank my mom and dad for their huge support, their prayers, financial sacrifice and their endless encouragement that gave me the inspiration and courage to pursue my dream of life. I would also like to thank Amrutha G.E who kept me on my path to success by providing heartfelt support and giving countless sacrifices. I would like to thank God for giving me the opportunity, physical & mental strength, good health, and ability to complete this work with his enormous blessings.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

1-D	1-dimensional
2-D	2-dimensional
3-D	3-dimensional
AEP	Auditory Evoked Potentials
AI	Artificial Intelligence
ALS	Amyotrophic Lateral Sclerosis
BCI	Brain Computer Interface
BFGS	Broyden–Fletcher–Goldfarb–Shanno
BMI	Brain Machine Interface
CAR	Common Average Reference
CNN	Convolutional Neural Network
CT	Computed Tomography
DeVAR	Desktop Vocational Assistant
DOF	Degrees of Freedom
ECG	Electrocardiogram
ECoG	Electrocorticography
EE	End Effector
EEG	Electroencephalography
El-E	Elevated Engagement
EMG	Electromyogram
EOG	Electrooculogram
ERN	Error Related Negativity

ERP	Event Related Potentials
ErrP	Error Related Potentials
fMRI	Functional Magnetic Resonance Imaging
GUI	Graphical User Interface
HRI	Human Robot Interaction
ICA	Independent Component Analysis
kNN	k-nearest neighbour
LDA	Linear Discriminant Analysis
MEG	Magnetoencephalography
ML	Machine Learning
MMI	Mind Machine Interface
MRI	Magnetic Resonance Imaging
NIRS	Near-Infrared Spectroscopy
NN	Neural Network
PC	Personal Computer
PR2	Personal Robot 2
ProVAR	Professional Vocational Assistant
RL	Reinforcement Learning
ROS	Robot Operating System
SARSA	State-Action-Reward-State-Action
SCARA	Selectively Compliant Articulated Robot
SGD	Stochastic Gradient Descent
SNR	Signal-to-Noise

SVM	Support Vector Machines
TFA	Time/frequency Analysis
VEP	Visual Evoked Potentials

CHAPTER 1: INTRODUCTION

Brain-computer interface (BCI), also referred to as Mind Machine Interface (MMI) or Brain Machine Interface (BMI), has been an active field of research for a number of decades. A majority of these interfaces being used in medical, security and authentication, diagnosis, robotics and in games and entertainment. A major component in BCI is the human interface, i.e., the signal acquisition from human brain. There are many signal acquisition approaches such as Functional Magnetic Resonance Imaging (fMRI), Near-Infrared Spectroscopy (NIRS), Magnetoencephalography (MEG), Electrocorticography (ECoG) and Electroencephalography (EEG) [1]. In this work, we concentrate on EEG signals, which are simply an electrical activity of the human brain.

Event-Related Potentials (ERPs) are the voltage fluctuations recorded over the scalp in response to an event or stimulus; in simpler terms, it is time-locked to an event. The procedure to measure ERPs is the same as EEG signals: non-invasive electrodes placed on the scalp to measure the electrical potentials generated by the brain. However, from the functional point of view, there is a fundamental difference between these two measures of brain function [2]. ERPs are classified based according to their latency and amplitude. Examples of ERPs are P50 wave, N100 wave, P200 wave, N200 wave, P300 wave, N400 wave, P600 wave, movement-related cortical potentials, contingent negative variation and post-imperative negative variation [3] which are discussed in detail in Chapter 3. Here we concentrate on the N200 wave which is a negative deflection peak about 200ms after the presentation of a stimulus. We refer to this characteristic pattern as Error Related Potentials (ErrPs) [4].

Human-robot Interaction (HRI) is a multidisciplinary research field with a focus on interactions between humans and robots. Assistive Robotics is one such HRI which is a branch of robotics that assists people with physical disabilities with physical interaction. Rehabilitation robots, mobility aiding robots, wheelchair robots and companion robots are some of the application of ongoing research in assistive robotics [5]. The areas where HRI is bringing revolution are elderly and aged, bio-feedback systems, schools and learning, medicine, entertainment, industry, space exploration, and military [6]. In this work, we concentrate on Human-robot Interaction using a closed-loop system with error-related potentials as feedback.

1.1 Motivation

Human communication is defined as the exchange of information verbally or non-verbally. There is a portion of the population who lose this ability to communicate due to various reasons such as accidents or birth disorders. There is an active area of research for the disabled. Researchers have built systems that can be controlled using joysticks, eye movements, head movements and many more. There is another condition of patients called locked-in syndrome [7] with complete loss of control over their voluntary muscles where patients are unable to speak and move but are conscious and can think and reason [8]. Some examples of which fall under this category which can result in severe motor paralysis are the stroke, severe cerebral palsy, motor neuron disease, amyotrophic lateral sclerosis, and encephalitis [9]. Our research focus is on patients who are affected by Locked-In Syndrome and to help physically challenged people to achieve greater independence by making technologies such as BCI/HRI accessible which provides an alternative communication bridge between a human brain and a robot/computer.

1.2 Objective of this Work

The purpose of this research is to develop novel methods that allow physically challenged to freely communicate with the devices around them and perform some of their everyday work without any human assistance. Our objective is to build the communication gap between the physically challenged and the technology. There is an active area of research going on to make physically challenged independent. Some of them are a brain-actuated wheelchair [10], robotic assistant for drinking [11], and robotic arm sorting objects [12]. Similarly, we present a solution such that disabled people will be able to train the robot to sort the objects and place them at the specified location where the subject wants the object to be placed. This builds as a feature for the assistive robots who work towards helping physically challenged.

1.3 Contribution

The main contribution of this research is to simulate a robot in the Robot Operating System (ROS) Gazebo virtual environment that is capable of learning to sort objects using the error related potentials elicited by the human brain in real time (simulating a human brain) as feedback. In order to achieve this objective, this research is subdivided into five parts:

1. Simulating the human brain like system such that it can generate human alike EEG data. Recorded EEG signals have been collected from researchers from different parts of the world working towards ErrPs.
2. Developing a novel classifier to classify the normal EEG vs ErrPs using various deep learning techniques with high classification rate
3. Demonstrating the applicability of error related potentials to robotic collaboration tasks.
4. Demonstrating the daily life tasks such as object sorting with the help of ErrPs as feedback for trajectory correction.
5. Developing a self learning algorithm that could learn on the go about the classification of objects in real time.
6. Integrating all the above individual models to a virtual prototype.

7. Evaluating its performance by conducting studies over various conditions such as:

    (a) comparing the effect of open loop and closed loop systems to know the importance of human feedback.

    (b) evaluating the classification accuracy of ErrP with respect to other existing models.

    (c) system performance results with the implementation of self learning i.e., to show how much a disabled has trained the robot in the object sorting task and how adaptable the robot is according to the user choice.

Novel tasks performed in this research are developing a classifier to classify the error related potentials with high accuracy and robot self-learning with human brain signals as feedback given the image and its class using online machine learning techniques.

1.4 Organization

This thesis is divided into two main parts. The first part contains background knowledge, the theory of the techniques used, literature review and summarizes the present state of the art methods relating to classification of error related potentials and their use in application such as a brain-computer interface. They are divided into many different chapters starting from assistive robots, BCI for assistive robots, machine learning algorithms used and robot motion planning. The basic knowledge and the different tools, algorithms, approaches, and techniques used in the thesis from these domains are clearly explained.

The second part is devoted to the actual implementation of the thesis with its design parameters and their utilization. The architecture used to accurately classify error-related potentials are used are detailed and the overall flow of the system. This part also contains the final results starting from EEG filtering to its classification, robot initializes pose to its trajectory correction and the applicability of the self-learning algorithm to the human-robot collaboration tasks with self-learning.

Chapter 2 contains various types of assistive robots and the methodologies used to interest with those robots as our interest is towards assistive robots for disabled with locked-in syndrome. Here, the focus relies on interaction using bio-signals and completing the chapter with some of the states of the art methods in the field on assistive robots specifically using bio-signals. Chapter 3 specifically focuses on the brain-computer interface for assistive robots. Basically, the conversion of brain signal and the tools such as EEGLAB used to analyze the brain signals are explained in detail. As this is continuous broad research, this chapter also summarizes the research from the first seen error related potential to the recent publications towards usage of error related potentials in robotic tasks. Chapter 4 concentrates on the concepts of deep learning and machine learning techniques used in this thesis. Convolutional neural networks (CNN) which are used to classify the error related potentials are explained in detail with every layer justification. Online machine learning which is the topic of current interest is elucidated and the algorithms used for this thesis are explained. Chapter 4 gives a heads up before going to the actual implementation by influencing on the robot kinematics and the Personal Robot 2 (PR2) robot which is used as an assistive robot. It also describes the Matrix Laboratory (MATLAB) robotics toolbox which plays a major role with the ROS communication and with some of the inverse kinematics problems.

Chapter 6 proposes the actual system framework by providing a complete overview of the system and the concepts from the theory used for actual implementation. It is divided into four sections starting with understanding the architecture used to classify the error related potentials, concepts used for trajectory planning, integration of error related potentials with PR2 robot and our novel method of self-learning algorithm with its design constraints. Chapter 7 consists of the results attained from the implementation in chapter 6. This is presented in two parts, one work with respect to error-related potentials classification and second is its use in trajectory correction and usage of self-learning. Chapter 8 concludes the thesis with some discussion and review of our work. It also presents future work that could be done upon the present work.

CHAPTER 2: ASSISTIVE ROBOTS

2.1 Introduction

According to the Oxford Dictionary, a robot is defined as a machine (programmable by a computer) capable of carrying out a complex series of actions automatically. In 1954, George Devol proposed the first programmable robot and coined the term Universal automation. Over the years many researchers presented their own versions of robots and its terminologies. The closer the human and robot get there was more risk to human beings.

Robots can be controlled either by an external device or within embedded control. Over the years many different types of robots have evolved which are classified on various grounds such as power source used, size of the robot, application in which it is used, mechanical structure, etc. One such application is assistive robots. Assistive robots are a branch of robotics that assists people with physical disabilities with physical interaction. The main goal is to bridge the gap between the capabilities of an individual with disabilities and the requirements of an activity the person wants to perform. These technologies are used to model specific activities of daily living such as dressing, hygiene, eating, communication or some work activities such as home management, educational, vocational, and care-of-others [13]. Figure 2.1 shows some of the examples of assistive robot technologies: a) Smart robot wheelchair, b) Feeding Robot, c) Bionic Arm, d) Assisted Walking, e) Hair Trim Robot and f) Vessel Wash.

Assistive robots mainly focus on aiding, motivating and assessing those in need, including and not limited to elderly, patients and individuals with disabilities. These robots are used to enhance mobility, to perform activities independently and improved communication [20]. Assistive robots are classified into two categories: (a)

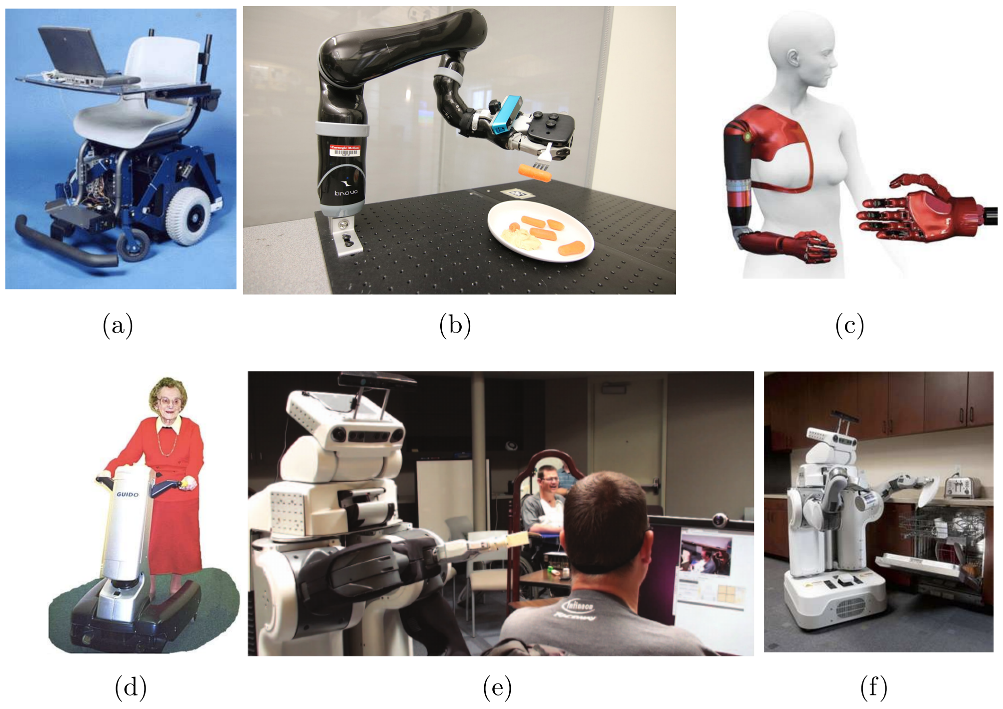


Figure 2.1: Examples of assistive robots: (a) Wheel Chair [14]; (b) Feeding Robot [15]; (c) Bionic Arm [16]; (d) Assistive Walking [17]; (e) Hair Trim [18]; and, (f) Vessel Wash [19]

noninteractive robots and (b) interactive robots. Surgical, rehabilitation and medication delivery robots are some of the noninteractive robots. Interactive robots are further classified into two types: (a) animal or creature-like, and (b) human-like [21]. A new study from the Georgia Institute of Technology has found that older and younger people have various preferences how their assistive robot should look like and have different minds about what robot should do. Participants were shown a combination of photos including human, robot and mixed human-robot faces and were asked to select the preference of their assistive robot appearance. Nearly 60% of older adults preferred a robot with human appearance [22]. This work mainly focuses on human-like interactive robots.

Assistive robotics is a rapidly growing research area. It is multidisciplinary research where researchers from various domain expertise such as robotics, engineering, computer science, psychology, anthropology, human-robot interaction, artificial intelligence combine together to build assistive technologies. Researchers working towards the development of assistive robot technologies focus to provide person-centered cognitive interventions to improve the quality of life of elderly adults. Their primary objective is to engage individuals in Human-Robot Interactions (HRI) in order to improve social, cognitive and effective functioning. Another group of people is working towards building robots that could sense and process the sensory information, and perform the tasks that benefit people with disabilities in their daily living activities. Activities of daily living include personal hygiene and grooming, housework, managing money, shopping, communicating, pet care, using technology, dressing, bathing and showering, food preparation and eating, working, playing, mobility, exercising, relaxing, education, reading, etc.

2.2 Human Robot Interaction

One of the major aspects of assistive robots is the communication between human and a robot. The field of study dedicated to design, develop and evaluate robotic systems for use by or with humans is known as Human-Robot Interaction (HRI). The goal of HRI is to define models that can build natural effective interaction between humans and robots by developing and designing a robot and its algorithm. In this field, research ranges from remote, teleoperated robots to pear-to-pear human-robot collaboration with anthropomorphic robots. A major study in this field is to understand how humans collaborate and interact with other humans such that behavior can be programmed or developed into robots using various techniques to give the human feel. With the advancement of artificial intelligence (AI) in robotics, the researchers are focusing towards the safest interaction between humans and robots as well as robot's social interaction and their goal is to build a spontaneously responding robots which develops easy communication with the robot through facial expressions, speech, and gestures. Another major aspect of HRI research is modeling the cognitive relationship between humans and robots. To build effective communication psychologists and robotic researchers need to come up with a common interest on both sides. With all these advancements and aspects, HRI is revolutionizing the world in different areas. Some of them are elderly and aged, bio-feedback systems, schools and learning, medicine, entertainment, industry, space exploration, industry, and military [6].

To have an effective interaction between humans and robots, communication plays a vital role. Communication between humans and robots may take several different forms but these forms are majorly influenced by their proximity i.e., whether the robot and human are in close proximity or not. Based on their communication proximity they are separated into two categories: (a) remote interaction - human and robots are not located at the same place or separated spatially or temporally and (b) proximate interaction - human and robot are located at the same place. Categorizing robots based on communication is useful to further classify the robots based on their application that require social interaction, physical manipulation or mobility. Robots which are teleoperated or having supervised control interacting remotely with a physical manipulator are referred to as telemanipulators or remote interaction robots. Mobile robots such as robot assistants are the best example of proximate interaction which includes physical interaction. Socially interactive robots are another subdomain of human-robot interaction including social, emotional and cognitive aspects of interaction which fall under the category of proximate interaction. There is a lot of research towards socially interactive robots or socially assistive robots which is out of the scope of our research hence we do not discuss further into this topic and our work focuses on proximate Interaction. Further interaction refers to proximate interaction [23].

Interaction between human and robot could be using different modalities like visual, physical, neurological and in many other ways. Bio-feedback based robots use bio-signals to build communication between humans and robots. Bio-signals consists of bio-electrical, bio-impedance, bio-acoustic, bio-optical signals. Electrical signals measured from the human body which generally originate either from neural or muscular activity is called bio-electrical signals. Some of them are:

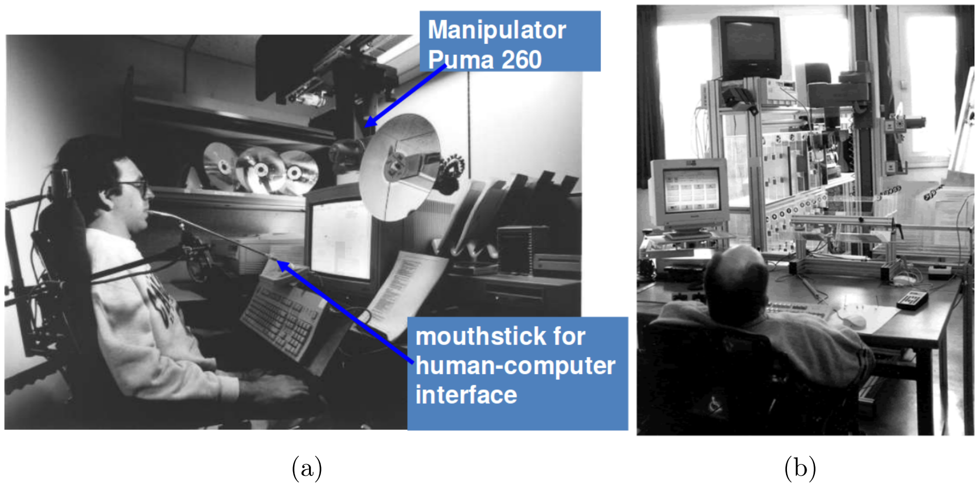
- *Electromyogram (EMG)* - electrical signals generated by skeletal muscles.
- *Electrocardiogram (ECG)* - signals that are generated from activity of human heart which are consequence of cardiac muscle depolarization and re-polarization during heartbeat.
- *Electrooculogram (EOG)* - signals generated due to the change of the corneo-retinal potential that exists between the front and back end of human eye.
- *Electroencephalogram (EEG)* - electrical signals generated due to electrical activity in human brain.

Every bio-signal has its own characteristic pattern and has various amplitudes and frequencies [24]. Our work is mainly focused on EEG signals and it is discussed in future chapters.

2.3 State of the Art Methods

Early researchers working towards building assistive robots focused mainly on the development of vocational robotic workstations. In the early 1990s, desktop vocational assistant (DeVAR) was developed at Stanford University. DeVAR was built for patients with high cervical spinal cord injuries to work independently. DeVAR was voice controlled small robotic arm which was mounted on an overhead track system to help disabled to perform daily tasks such as meal preparation, brushing teeth, shaving and feeding (shown in Figure 2.2) [25]. DeVAR followed by ProVAR which was a professional vocational assistant with force sensor and had personal computer-based interface [26]. Later researchers developed MASTER and RAID models which brought more independence for persons with paralysis but these devices were very complex and weren't economical.

Exact Dynamics came up with an assistive robotic manipulator mounted to the side of a wheelchair called MANUS which had 6 degrees of freedom. It was con-



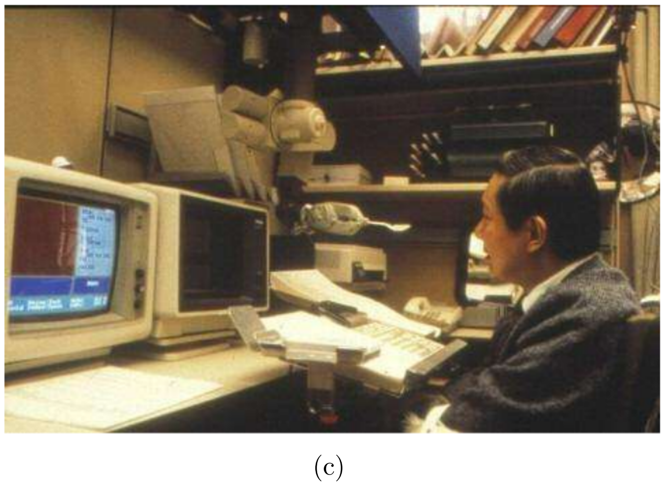
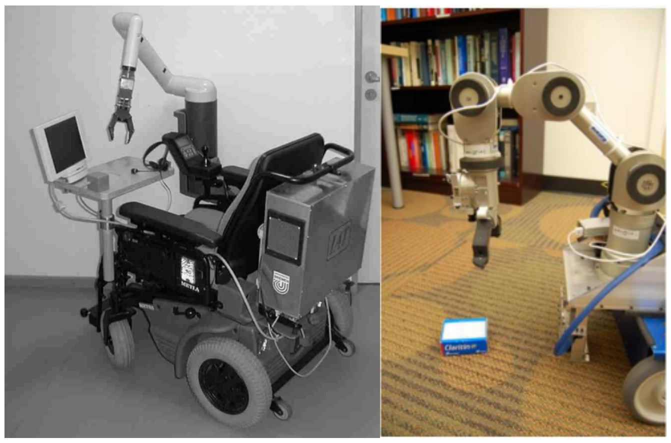
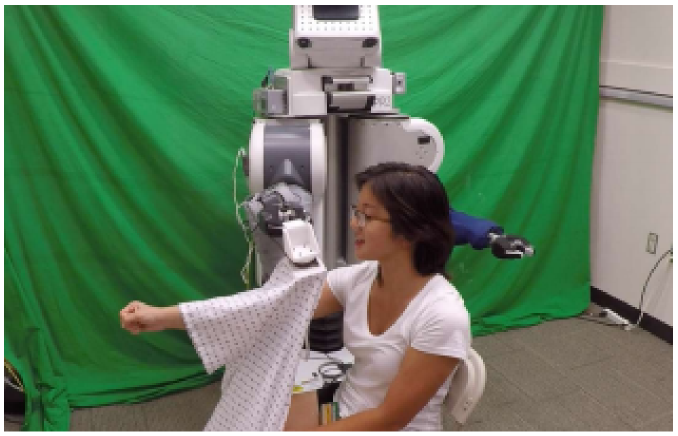


Figure 2.2: Early Assistive Robots: (a) DeVAR [25]; (c) MASTER/RAID Models [27]; (b) ProVAR [26]

trolled using a programmable personal computer (PC) or a joint control which had directional control modes. MANUS was made commercially available for people with upper-limb dysfunction or persons with high tetraplegia and neuromuscular disease. MANUS was used to perform daily tasks such as feeding, dressing and opening cabinet doors [27]. Over the years researchers from various parts of the world tried to add additional functionality to the MANUS. Researchers from the Delft University of Technology developed a software framework with various features integrated with different sensors. The integrated cameras into an existing system and implemented various computer vision algorithms that could help to retrieve objects more efficiently in challenging environments [28]. Another group of researchers from France developed a graphical interface and added ultrasound sensors to identify and locate objects to an unmanned mobile base. They enabled features such as path planning algorithms that allowed the robot to navigate autonomously to retrieve any objects [29]. Later another group came up with their own design of a small mobile robot with the goal of following a wheelchair user rather than mounting ARM to wheelchair [30].



(a) (b)



(c)

Figure 2.3: Assistive robots in early 20th century: (a) MANUS [27]; (b) El-E robot [31]; (c) Robot Assisted Dressing [32]

In 2008, the El-E (Elevated Engagement) robot was developed by Nguyen and Kemp at Georgia Tech to fetch objects from flat surfaces in the home environment. It had an arm mounted vertically on a small mobile robotic base. The user had to point a laser pointer towards the object, then the robot with the help of the camera and laser range finder sensors identifies the object, navigate to the object and grasps it and returns to the user [31]. Later over the years much of the development focused towards autonomous control of the robot using various sensors such as camera, LIDAR, laser, etc and vision-based robot control specifically for tasks such as robot feeding. Later, a wide range of research started few working towards developing methods for redundant degrees of freedom and a few others exploring manipulation related tasks. Researchers at Georgia Tech used PR2 (personal robot) for various takes. They used the PR2 robot for assistive feeding purposes. It had a web interface and was specialized in autonomous behaviors for scooping and feeding yogurt [33]. As a nursing robot, they developed robot-assisted dressing by understanding how much force should a robot put on the human body [32, 34] and also tracked the human pose while dressing [35]. Another group worked specifically on tasks related to pick and place objects. Deep learning models were used for incorporating the material knowledge such as its properties to the robot [36].

CHAPTER 3: BRAIN COMPUTER INTERFACE FOR ASSISTIVE ROBOTS

3.1 Introduction

A Brain-Computer Interface (BCI), sometimes called a direct neural interface or a brain-machine interface (BMI), is a direct and new communication channel interface between the human brain and a computer. The main objective of BCIs is to build assistive technologies that often are used to assist or repairing human cognitive or sensory-motor functions. Currently, the assistive robots used to acquire input to the computer are either a physical device or need mechanical interface by the user such as mouse, joystick or keyboard. Computer's feedback or output is given to audio or visual devices such as speaker and Graphical User Interface (GUI) interface as seen in the previous chapters. There were many successful applications with it. However, there are many limitations in terms of usability and accessibility of assistive robots. Specifically to some people who lost their fundamental ability to communicate because of accidents or neuromuscular disorders. Accessibility and usability of assistive robots become more challenging to people with complete loss over their voluntary muscles, most commonly known as locked-in syndrome patients. The main advantage of these patients is that they are not capable to speak or move but are conscious and are able to think and reason their surrounding events in a given environment. Neurological disorders or diseases such as stroke, server cerebral palsy, amyotrophic lateral sclerosis, motor neuron disease and encephalitis which could result in such severe motor paralysis [37]. Diseases of these kinds restrict humans to have effective communication with robots.

Early research had primarily focused on applications such as neuroprosthetics that aim at rehabilitating the hearing, movement or sight loss. Researchers have developed technologies by understanding brain signal pattern & thoughts and converting them to equivalent commands. In the context of BCI, thoughts refer to the computer-aided interpretation of neuronal activities of the user. Neuronal activities were recorded either at certain extremities of the human (arms, legs, etc.) or at the brain itself by analyzing brain waves. This was being developed to bridge the communication gap and providing more independence to the disabled.

Research path in BCIs started with Hans Berger who performed a systematic study and discovered the electrical activity of the human brain and developed the electroencephalography (EEG) in 1924. Later several researchers throughout the world used these concepts and applied to various fields. BCI research has contributed to various other fields of research and is being used in real time for various applications. Some of them are in medical, neuroergonomics and smart environment, neuromarketing and advertisement, educational and self-regulation, games and entertainment, and Security and authentication fields as shown in Figure 3.1 [38]. Our research focus is towards assistive robots so we discuss the research into it in the state of the art methods.

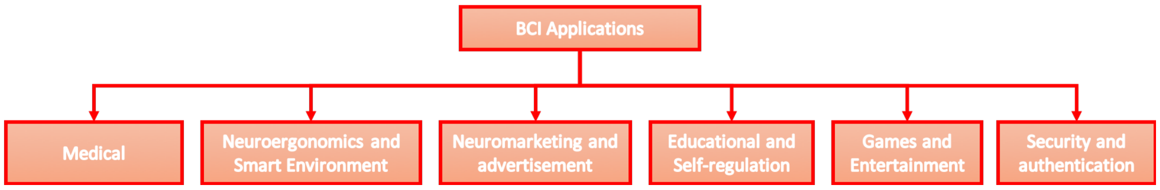

BCI Applications
Medical
Neuroergonomics and Smart Environment
Neuromarketing and advertisement
Educational and Self-regulation
Games and Entertainment
Security and authentication

Figure 3.1: BCI application fields

A classical BCI system generally consists of four major stages or components: data acquisition system, preprocessing the acquired signals, feature extraction process, classification of extracted features and finally sending the classification results to the device controller or control interface [8]. The basic functional block diagram of a simple BCI system for application smart wheelchair is shown in Figure 3.2. Each component is explained in detail in the following sections.

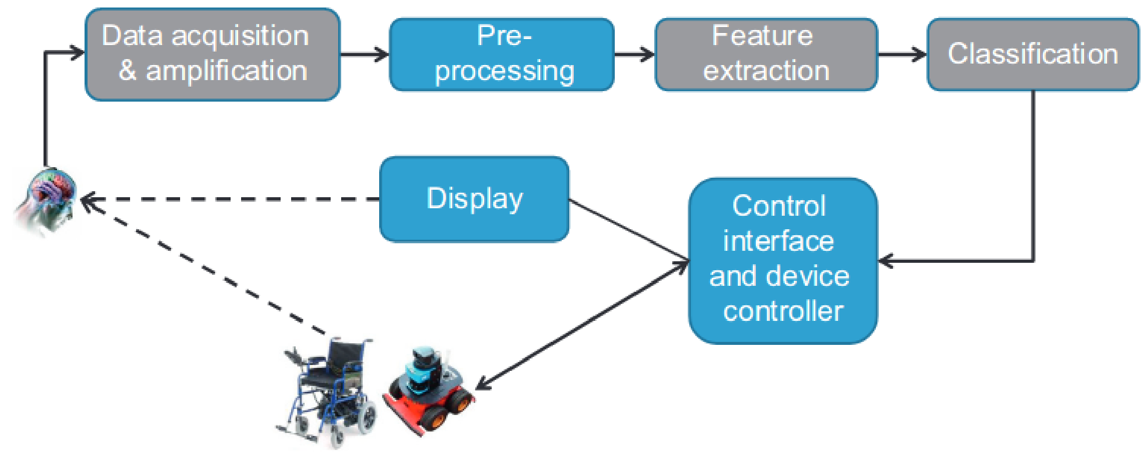

Data acquisition & amplification
Pre-processing
Feature extraction
Classification
Display
Control interface and device controller

Figure 3.2: Basic functional block diagram of a simple BCI [8]

3.2 Signal Acquisition

Signal acquisition plays a vital role in BCI. The signal recordings of the human brain can be either invasive approach or noninvasive. In an invasive approach the electrodes are directly implanted on the cortex by surgically whereas non-invasive methods collect the brain signals over the scalp or do not require surgery. Some of the signal acquisition techniques are: Functional Magnetic Resonance Imaging (fMRI), Near-Infrared Spectroscopy (NIRS), Magnetoencephalography (MEG), Electrocorticography (ECoG) and Electroencephalography (EEG) and have their own advantages and disadvantages [1]. EEG possesses many advantages over other techniques. They are:

- EEG based signal acquisition is cheaper when compared to other devices because the electrodes used for acquisition are relatively cheaper.
- The electrodes are widely available unlike other techniques which require a special type of equipment available only in high traffic hospitals.
- The whole system to record EEG is easily portable and can be used in mobility applications too.
- EEG has a very high temporal resolution in the order of milliseconds are

recorded at sampling rates between 250 and 2000 Hz for clinical and research purposes while other methods have a resolution in the order of seconds.

- EEGs are non-invasive which doesn't require the subject to go through surgical methods unlike Electrocorticography to place electrodes on the surface of the brain.
- Process of recording EEG doesn't involve exposure to high-intensity magnetic fields or exposure to radioligands
- Studies related to the specific event can be conducted with simple paradigms.
- It is possible to visualize the EEG signal and understand what the signal is all about or what event it is related to.

Due to all these advantages, EEGs are considered as the best suited method of signal acquisition specifically to develop BCI systems.

3.2.1 10-20 System

One of the major challenges in EEG based data acquisition is the location of electrodes on the scalp. To overcome this challenge the International 10-20 System has come up with an internationally recognized method that describes the method and location where the electrodes to be placed on the scalp. These standards are widely used in the context of EEG study, polysomnograph sleep study or voluntary lab research. In general, the electrodes are placed on the scalp at 10% and 20% of the measured distance from the reference position including nasion, inion, left, and right preauricular [39], as shown in Figure 3.3. Therefore, this system is known as the 10-20 International System.

There are different standard electrode systems such as 16, 32, 64, 128 and so on which are used to record EEG signals. Based on the application in which they are used, researchers use different electrode systems. Typically researchers use an 64

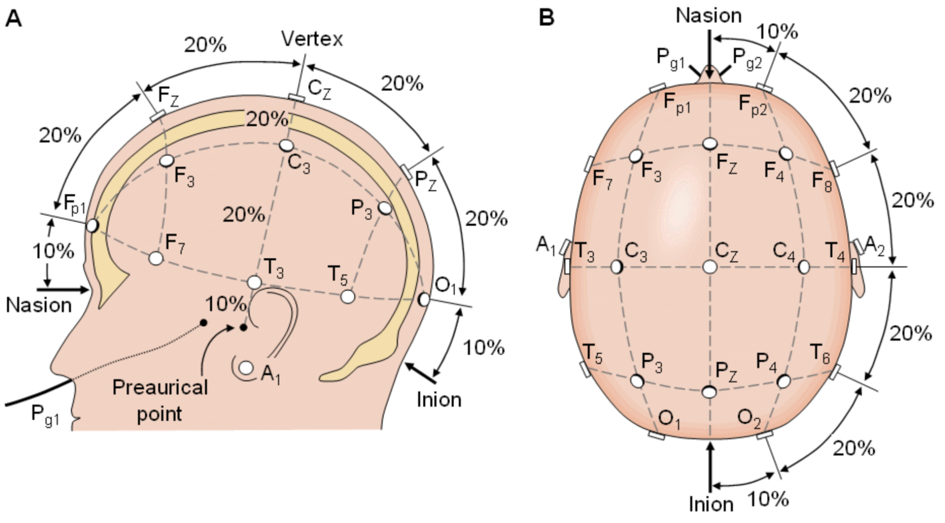


Figure 3.3: The International 10-20 system seen from (A) left and (B) above the head. A = Ear lobe, C = central, Pg = nasopharyngeal, P = parietal, F = frontal, Fp = frontal polar, O = occipital [39]

electrode system for developing applications related to BCIs. Figure 3.4 shows the 64 electrode system as per 10-20 International system.

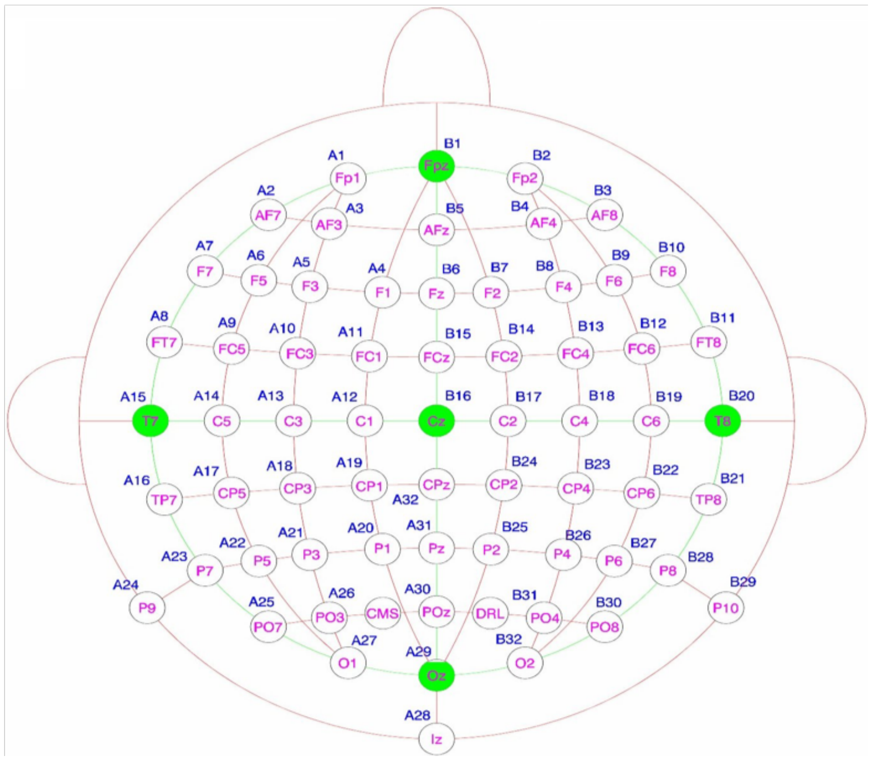


Figure 3.4: 10-20 International System for 64 electrodes [40]

3.2.2 Electroencephalogram

Electroencephalography (EEG) is a group of electrical potentials generated by neural activities in the human brain. The electrophysiological method to measure these electrical potentials are known as electroencephalography. As discussed in the previous section, EEG signals are obtained by placing electrodes on the scalp of the human brain. Typically an EEG of an healthy human adult is about 20-100 µV [41]. EEG is able to capture changes in brain activity on the range of milliseconds when compared to other devices whose time resolution between seconds and minutes. Hence, they are frequently used for most of the experimentation related to real time devices. Figure 3.5 shows an EEG acquired from a human brain when placing eight electrodes placed on the human scalp.

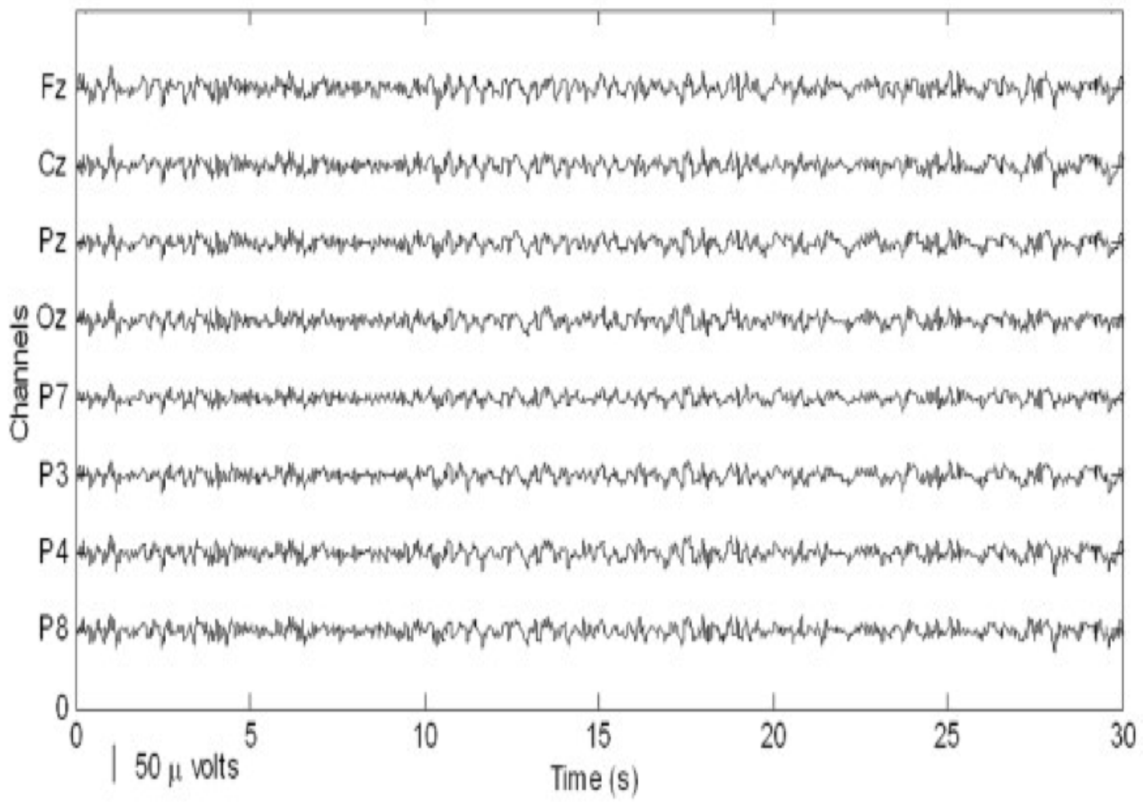


Figure 3.5: EEG from human brain

EEGs are generally used to diagnose epilepsy, sleep disorders, depth of anesthesia, coma, encephalopathies and brain death. For every diagnose method, different EEG patterns are generated by the human brain. For example, for epilepsy, the EEG signals are very abnormal and have multiple spikes in the data. EEGs are also used as a first-line method for diagnosis for tumors, strokes and other focal brain disorders before they undergo magnetic resonance imaging (MRI) and/or computed tomography (CT).

3.2.2.1 Event Related Potentials

EEG signals have many derivatives; one such is event-related potentials (ERP). ERP signals are generated by a population of neurons in response to a perceptual, cognitive or motor event, in opposition to spontaneous activity that reflects the brain activity related to volunteer self-paced tasks. These events/stimuli can be either visual, auditive or vibrotactile [42]. Evoked potentials are basically a type of ERP generated in response to the presentation of a stimulus. Based on the type of stimulus presented it is called visual evoked potentials (VEP) and auditory evoked potentials (AEP). Once the stimulus is presented and based on the reaction of patient or subject to stimuli results to various EEG traces. These ERP waveforms generated are categorized according to latency and amplitude after the presentation of stimuli such as P50 wave, N100 or N1 wave, P200 or P2 wave, N200 or N2 wave, N300, P300, N400, P600, movement-related cortical potentials, contingent negative variation, and post-imperative negative variation [3]. P300 and N200 waves play an important role in the field of BCI. There has been active research for decades over classification of P300 and its application in BCI [43] when compared to N200, which is relatively newer.

3.2.2.2 Error Related Potentials

In the late 1990s, reports were published of an event-related potential where ErrPs were recorded in focused attention condition by presenting visual stimuli with correct and incorrect trials [44]. Since then many researchers have reported the presence of ErrP components such as error-related negativity (ERN or Ne) and error-related positivity (Pe) in various tasks. Later, several studies reported that such event related to error elicit a different characteristic pattern over medial-frontal areas appearing between 200 and 300ms after an event has occurred [45]. The ErrPs are characterized by an initial positive peak occurring at about 200ms after an event followed by a massive negative deflection at about 200-250ms and a second positive peak at about 320ms [46] as shown in Figure 3.6. These waveforms are reported to have a similar pattern in tasks using different modalities [47–49] and maintained the almost identical when tested after several months [50]. Hence, the methods used to classify the signals can very well be generalized can be used in applications related to error processing.

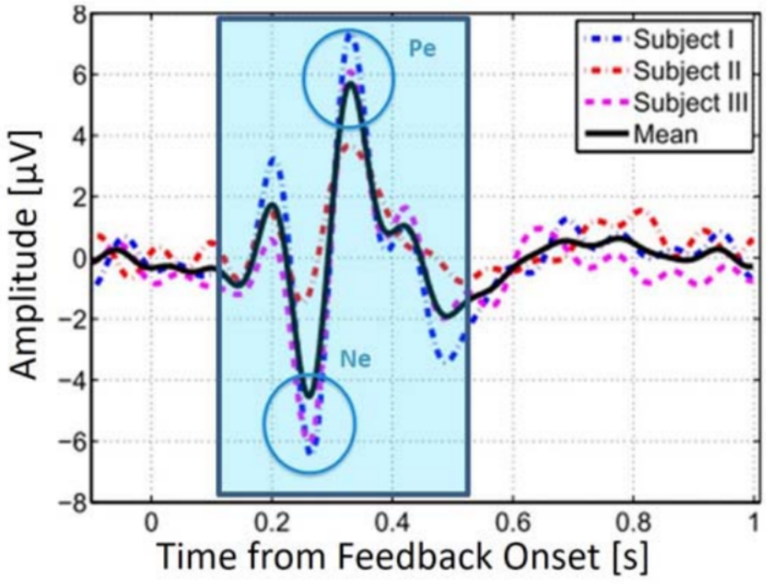


Figure 3.6: Visualization of Error related potential [45]

3.3 Preprocessing

EEG signals are recorded using non-invasive methods. Hence, the recorded output is a mixture of the signal of interest from the neural shooting potentials and some unknown and random noise added. These raw EEG signals will have very low amplitude and are prone to low or poor signal-to-noise (SNR) ratio. In order to overcome all these and obtain cleaner EEG signals, preprocessing is carried on the raw EEG which removes or filters out the unwanted components combined with the raw EEG. Thus it can reduce the computational load on the rest of the BCI components.

3.3.1 EEG Artifacts

EEGs are designed to record cerebral activity, but it contains the signals originated from the non-cerebral region which are basically called artifacts. EEG contains various artifacts such as:

- *Cardiac artifacts*: ECG artifacts are quite common ones and can be mistaken for spike activity. It is most common when the subject neck is short and are time-locked to cardiac contractions and are easily identified when matched with ECG signals.
- *Electrode artifacts*: artifacts related to electrode movement, poor contact, spontaneous discharging, reflection from skin cause the noise.
- *External device artifacts*: Numerous types of external devices produces EEG artifact and may do so through the electrical fields they generate or through mechanical effects on the body. The most common external artifact is due to the alternating current present in the electrical power supply.
- *Muscle artifacts*: Glossokinetic type of muscle artifacts caused by the potential difference between the base and the tip of the tongue. Minor tongue movements can cause EEG artifacts and eye movements can cause to ocular muscles.
- *Ocular artifacts*: Artifacts caused by eye blink and its movements which generates potentials add up to EEG artifacts.

3.3.2 Filtering

Pre-processing the raw EEG signals plays a very important role which increases the signal to noise ratio by removing unwanted artifacts [51]. Several groups working in the field of EEG filtering. Most of their approach involves subject specific or setup (hardware) specific, if taken care can improve the overall classification accuracy. Subject specific methods are to control the eye blink which reduces the ocular artifacts. However, there are other techniques of filtering such frequency-selective EEG filtering methods as well as spatial filtering techniques which are used to improve the EEG classification results

3.3.2.1 Spatial Filtering

Most commonly used spatial filtering methods in the field of BCI studies are Common average reference filter (CAR). CAR is one of the reference-free techniques that is not affected by problems associated with an actual physical reference [52]. In CAR, the common average is subtracted from the channel of interest. Basically, the assumption is that the electrodes are spaced equally over the head such that the mean voltage distribution equals zero. But in practice, it is not possible [53, 54]. Hence, CAR provides EEG recordings which is reference free. CAR is computed using the following formula:

$$V_i^{CAR} = V_i^{ER} - \frac{1}{n}\sum_{j=1}^{n} V_j^{ER} \tag{3.1}$$

where V_i^{ER} is the potential between the i^{th} electrode and the reference and n is the number of electrodes in the montage.

3.3.2.2 Spectral Filtering

Usage of appropriate spatial filters such as bandpass and lowpass can improve the accuracy and robustness of BMI by reducing the influence of the activities that lie out of the range of frequency of interest. Bandpass filters can reduce the effects of power line noise, ECG, EMG and other noises outside the frequency range of interest. More generally in EEG analysis higher order, Butterworth filter is used with a pass-band frequency of 1-10Hz because they are known to be relatively slow cortical potential [55]. To prevent aliasing, the cut off frequency is set above two times the frequency of preanalysis range.

3.3.3 Electrode Selection

Electrode selection plays a major role in preprocessing because of the following advantages [56]:

- EEG channel selection improves BMI performance by removing irrelevant channels.
- Enhances user convenience from the use of lesser channels.
- Decreases the number of features (dimensionality reduction).
- Removing of electrodes also removes the noise contamination.

Selection of electrode is not a trivial task and requires neurophysiologic knowledge about the area of the brain producing the signal of our interest. This can be performed by plotting spatiotemporal EEG maps [57].

3.4 Feature Extraction and Classification

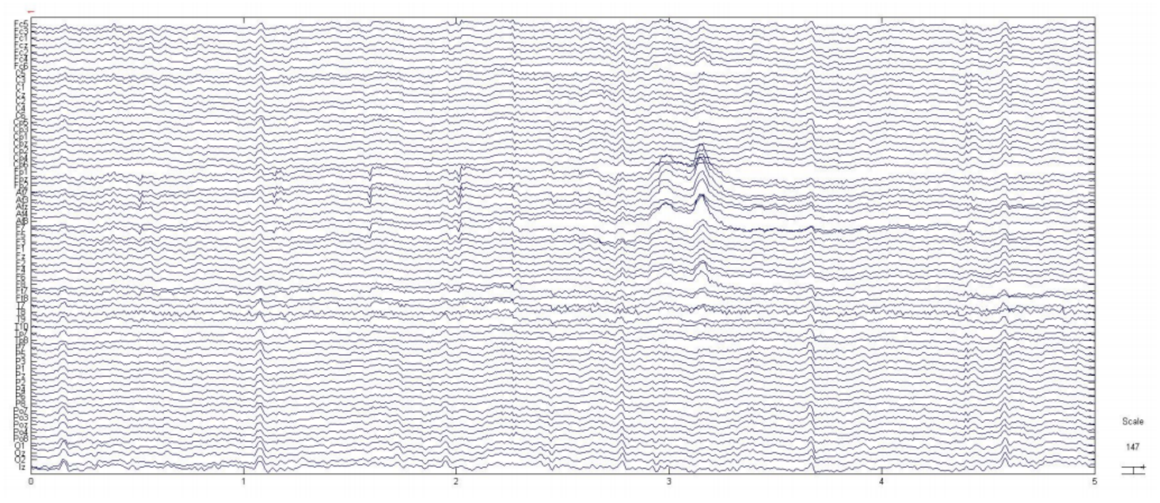
There have been several studies in past years intended to classify the ErrP signals using various classification algorithms. Errikos et. al [58] proposed a system which could extract features based on statistical measures from averaged ERP recording and to classify the signals, kNN and SVM were used. Similarly, Ricardo and Millan [59] used a Gaussian Classifier by assigning the same prior probability for both correct and error classes. A Bayesian filter was used as a formal probabilistic approach for classification of evoked error-related potentials and its performance was compared with a mixture of Gaussians [60]. Most of the commonly used classifiers were Gaussian Classifier, logistic regression, support vector machines, linear discriminant analysis, and its variations. Jacopo et. al performed the classification in time as well as frequency domain and have not found any notable results [49]. It is not a fair valuation to directly correlate the performance of these studies due to their distinction in their preprocessing techniques, feature selection, and performance metrics. However, their reported classification performances are between 70% and 80%.

In recent years, with the overwhelming performance of deep learning architectures, they are being used for classification of EEG signals. Hubert and Axel [43] first used CNNs (one of deep learning technique) for detection of P300 in oddball paradigms. Later, Robin et al [61] used Convolutional neural networks for decoding EEG signals which could classify left, right, feet and rest movements. Similar work was performed by Siavash et. al by using the parallel convolutional neural network and energy based features [62]. There is a little amount of work done in this towards the error related potential classification using deep learning approaches.

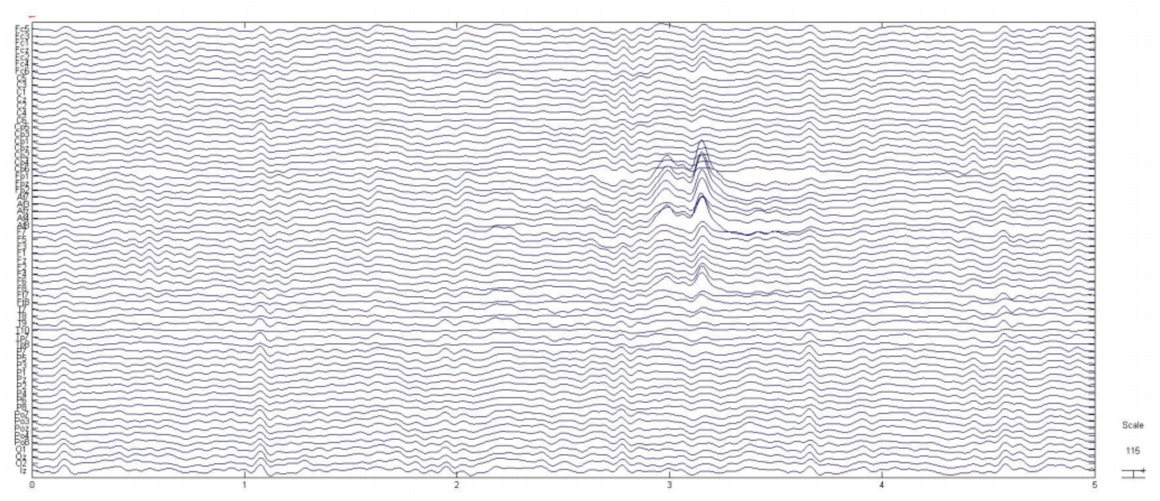
3.5 EEGLAB

EEGLAB created by Swartz Center for Computational Neuroscience, a research center of the Institute for Neural Computation at UCSD built an open software tool used with Matlab to focus on how EEG data can be used to understand the behaviour of different parts of human brain. EEGLAB is an interactive Matlab toolbox for processing continuious EEG signals. It is used to analyze and understand signals such as event related potentials, MEG and other electrophysiological signals. These signals are used to incorporate independent component analysis (ICA), time/frequency analysis (TFA), event related statistics and filtering artifacts from the raw signals. EEGLAB provides different visualization tools within that can be used to accurately visualize the signal with its potential maps. Figure 3.7 shows the filtered EEG data alongside its raw EEG data using EEGLAB.

EEGLAB is a graphical user interface (GUI) based tool which allows users to use ICA, TFA and other standard averaging methods. It also offers a structured programming environment for experienced users to store, access and manipulate the data and also to visualize the EEG potentials. Researchers around the world have developed various plugins for EEGLAB such as ERP plugin, BCILAB and many



(a)



(b)

Figure 3.7: EEG filtering using EEGLAB [63] (a) Raw EEG data; (b) Filtered EEG data

more. One of the important tool of EEGLAB is their ability to plot scalp maps given EEG signals at specific time as shown in figure 3.8. We used EEGLAB because it contains all these features discussed.

3.6 State of the Art and Applications

BCI is an vast research area and numerous researchers have come up with various solutions for different applications. Our focus is towards usage of bio-signals towards assistive robots. Initial works started in 2008 by Ferraz and Millan [64] who came up with the a simulated brain computer interaction that can generate error related potentials. They came up with an experimental protocol to elicit the naturally occurring error related potentials from human brain. Five healthy participants participated in their experiment and the researchers confirmed the existence of ErrP of same characteristic pattern which was found initially in late 1990s by Falkenstein [44]. To classify the ErrP, they used Gaussian classifier and achieved to have an average recognition rate of 75.8% for erroneous trails and 63.2% for correct trails [59]. While on the other side, Iturrate and team used P300 signal to control navigation of a wheel chair [10]. They used a virtual driving environment for the participants and the wheelchair was moved autonomously. On the basis of the map representation, the user concentrates on the location of the space to reach which is detected by their classifier and send to the wheelchair for navigating to that point. They claim that the offline-classification accuracy of P300 was as high as 94% and the commands were transmitted at the rate of two orders per minute. Later Iturrate's team came up with a different experimental protocol to elicit the error related potentials. Their intension was to control a robotic arm with ErrP considered as a reward signal. Their experiment had a robotic arm with 5 DOF performing correct/incorrect reaching tasks, while the system recorded the subject's brain activity. Support vector machine was used as a classifier and achieved a classification rate of 80%. Similar research was performed by Jiaxin and Zhang who used combination of EOG and EEGs to control a robot [65]. Zhang designed classifier to classify blink, gaze, wink, and frown. Linear discriminant analysis was used to classify the event related potentials. To verify the effectiveness of system they performed tests on two different robots: a humanoid robot NAO and a mobile robot Kobuki. They claim that the latency was very less which opens doors to real time implementation.

For the real world scenarios, robotic quadcopter was controlled in 3-D physical space using the EEG signals. This was performed based on P300 signals which was used to classify left vs right hand, both arms vs rest and constant forward velocity. These movements corresponds to the navigation of the quadcopter. Their study was much related to the response time of the system [66]. Similar research was conducted onto real car driving tasks which worked on the same principle as previous ones [67]. Over the years, we see that most of the work was either towards EOG signals or P300 signals a component of EEG signals and not much of research towards error related potentials in the filed of robotics.

In late 2016s, Ehrlich and Cheng, came up with a neuro-based method for detecting erroneous robot action. They used the concept of error related potentials in human robot interaction to find the robot mistakes. A humanoid robot was used such that it is gazing directly at the participant. The target stimuli were as a white rectangular of size 3x3 cm appearing either left, right or above the robot head. Once the arrow blink, the robot had to turn its head in that direction. Machine errors were introduced such that they can capture the erroneous signals. Linear discriminant analysis was used to classify the erroneous signals and found to have accuracy of 69.7±9.1% [68]. Similarly, various researchers came up with different applications using error related potentials. Some developed a BCI with visual cues system with motion imagination that could detect the error related potentials and are used to navigate a wheelchair [69]. Later, researchers from Massachusetts Institute of Technology came up with a real time system where baxter robot was used to perform a real time task. They proposed a feedback based system that corrects the robots mistakes in real time which performing tasks autonomously. The task was to classify the objects into two classes with human brain signal as feedback. To classify the error related potentials they used an elastic net and achieved average accuracy of 84%. They also introduced concept of secondary error which is occurred when the classification occurs. Their major contributions were to show the difference between closed-loop and open-loop system and to achieve the best system performance results [12]. With MIT's research the usage of error related potentials opened up the interest in researchers alongside used various ways to use it. One such was to use in a multi class object selection tasks where robotics arm moves over the object and the human subject elicits an automatic response based on which the robot understand which object needs to be picked up [70]. The work of these researchers contributes a lot to come up with a neuro-based solution for the object sorting task.

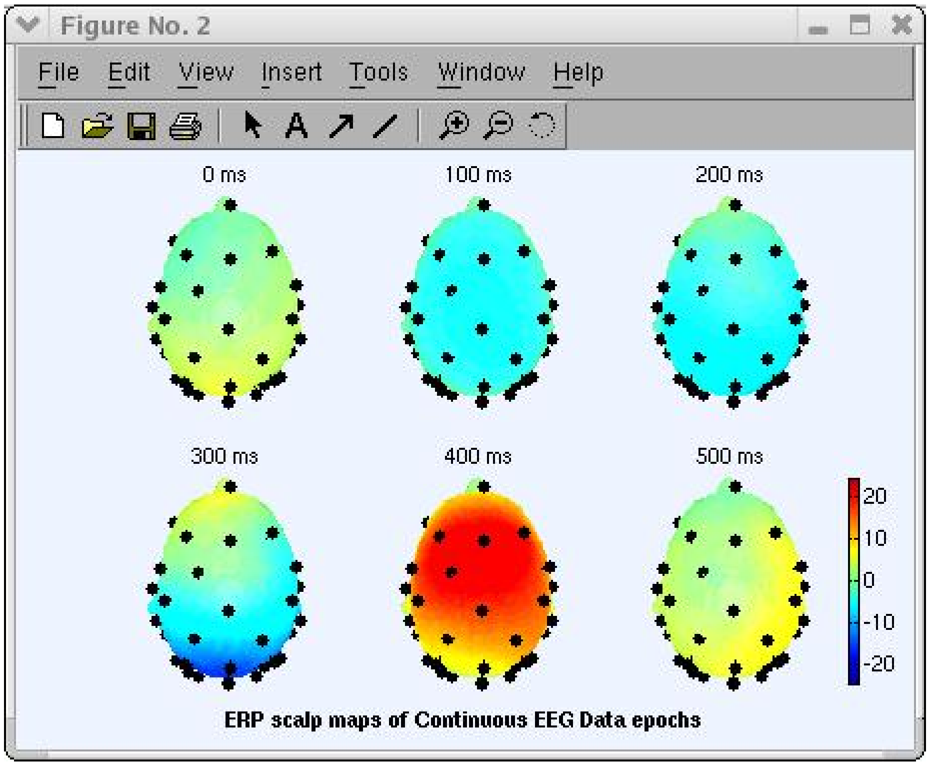


Figure No. 2
File Edit View Insert Tools Window Help
0 ms
100 ms
200 ms
300 ms
400 ms
500 ms
20
10
0
-10
-20
ERP scalp maps of Continuous EEG Data epochs

(a)

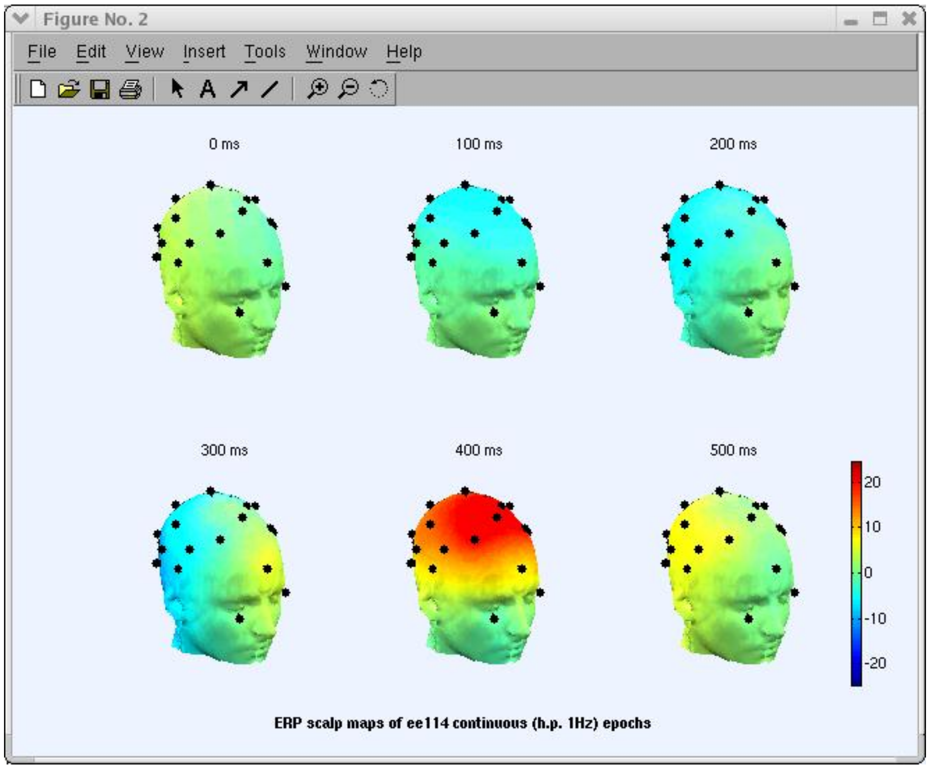


Figure No. 2
File Edit View Insert Tools Window Help
0 ms
100 ms
200 ms
300 ms
400 ms
500 ms
20
10
0
-10
-20
ERP scalp maps of ee114 continuous (h.p. 1Hz) epochs

(b)

Figure 3.8: ERP scalp maps of continuous eeg at specific time [63] (a) 2-D view; (b) 3-D view

CHAPTER 4: MACHINE LEARNING

Learning plays a vital role in any living creature. Especially humans expand their knowledge constantly by learning from their surrounding environment. learning basically is defined as the process of acquiring new or to modify existing knowledge, skills or behaviors. This learning happens in humans, animals, and some plants too [71]. Some researchers over the past few years were very interested in having machines to adapt this learning behavior by automatically learning from data. Until 1950s statistical methods were defined and refined which lead to deep learning models in 2010s but the name machine learning was coined by Arthur Samuel in 1959. The algorithms that computer systems use to perfect specific task effectively without any instructions but only relying on patterns and inference are called as machine learning (ML) which is a subset of Artificial Intelligence.

In ML, the main objective of any selected model is to generalize from its prior experience. To achieve this the ML algorithms build various mathematical models of the sample data. These sample data from which the algorithms generalize is known as training data. To verify the generalization of the algorithm/model, new and unseen data/tasks are performed known as test data. The whole idea is that the learner has to build a very generalized model such that it could predict accurately in new examples. To achieve this, there are various methods of model learning algorithms. Some of the well-known learning methods and the applications in which they are used is discussed below:

- *Supervised learning*: The applications in which the training data and its corresponding target data are known as supervised learning problems. Hence, the models under this category map an input to an output based on input-output pairs. Based on their output and the application in which it is used, these algorithms are classified as classification (discrete targets) and regression (continuous target) problems. Applications such as the character recognition in which each input vector is assigned to one of the finite discrete categories fall under classification problems [72]. Applications such as stock price prediction where the output consists of continuous variables fall under regression problems. Most widely used supervised learning algorithms are support vector machine, linear regression, logistic regression, naive Bayes, linear discriminant analysis, decision trees, neural networks.

- *Unsupervised learning*: The goal of unsupervised problems is to find the structure in the given input data (training examples) like discover similar groups or clustering of data points called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization [72]. The main idea is the algorithm is allowed to learn from data that has no labels or categories. Some of the most commonly used algorithms are k-means, mixture models, neural networks, Hebbian learning, Generative adversarial networks, expectation-maximization techniques.
- *Semi-supervised learning*: Semi-supervised learning is intermediate or combination of previous two learning methods. It uses a small amount of labeled data with a large amount of unlabeled data. Many researchers have come up with good predictions by giving less labeled data unlike in unsupervised with no labeled data. Most of the algorithms which fall under supervised and unsupervised are used in semi-supervised but the way of training it is bit different but approach is the same.
- *Reinforcement learning*: The problems of finding suitable actions to take in an environment by software agents in a given situation to maximize a reward. In

this case, the learning algorithms are not given the optimal output but the agent has to decide what to do to perform any given task. In contrast to supervised learning where the training data also has target values. These algorithms have a sequence of states and actions while interacting with their environment. The main focus here is to find the balance between exploration (system tries out new kinds of actions to find its effectiveness) and exploration (of the unexplored region). Some of the algorithms which come under RL are Q-learning, temporal difference learning, state-action-reward-state-action (SARSA).

With the rapid growth of machine learning algorithms in the domain of computer vision, speech and text analysis with better performance, there has been a interest in using machine learning in the field of robotics. Currently, ML algorithms are being applied in limited condition and enhancing the capabilities of industrial robots. Researchers are working towards reaching the full capacity of robotics and machine learning. The scope of AI in robotics are majorly impacting in four different areas of robotic process [73]: Vision - ML is used to detect unseen objects and recognize it to further perform some other task [74], Grasp - Using ML models robots are able to determine the best possible position and orientation to grasp objects [75], Motion Control - ML helps robots with dynamic interaction and obstacle avoidance [76], Data - ML helps robots to understand the physical and logistical data patterns to act productively.

4.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is a class of deep learning algorithms which has been very successful in numerous tasks such as image classification [77], speech recognition [78]. CNNs are known for their feature of shift invariant and translation invariant properties. CNNs were first developed by Yann LeCun in 1998 and named it LeNet5 because it was a five-layered architecture as shown in Figure 4.1 [79]. The technology and the computation power was available in limited amount to get CNNs into real world. Later in 2012 Alex Krizhevsky came up with AlexNet and competed in ImageNet Large Scale Visual Recognition Challenge and had outperforming results. His paper suggested that with an increase in depth of model the performance increase, which in turn is computationally expensive. To overcome that he proposed to use of graphics processing units (GPUs) during training. With that, CNNs grew exponentially and in three years researchers made headway from 8 layers of AlexNet to 152 layers of ResNet. Some of the well known CNN architectures are LeNet, AlexNet, ZFNet, VGGNet, GoogLeNet, and ResNet.

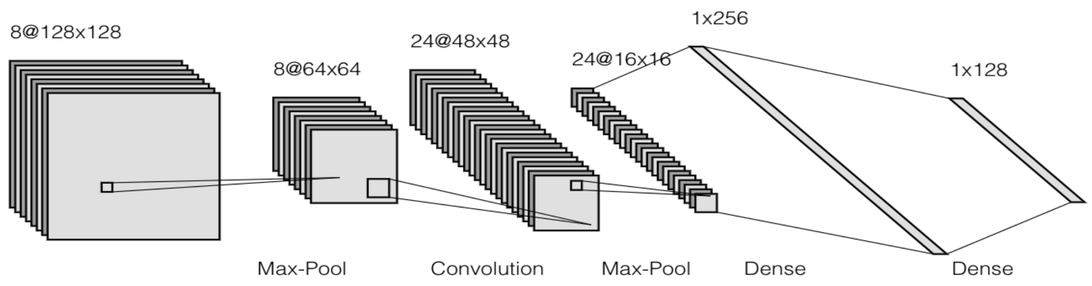


Figure 4.1: Visualization of deep CNN architecture (LeNet) [79]

In this current work, we use Convolutional Neural Networks (CNNs) for classification purposes. The advantage of using CNNs is its natures of end-to-end analysis and its ability to learn from raw data. Given input data CNNs can automatically detect the salient features with any human supervision. CNNs are computationally efficient as it uses special convolution and pooling processes and also performs parameter sharing. As per the review of ten-year update recent classification algorithms in BCI applications, CNNs have proved to be performed best even in EEG classification [80].

The pivotal component of CNNs is its ability to learn local patterns. The initial convolutional layers extract the low-level features from input data, and as we go much deeper into the network it learns more global and high-level features. CNNs can have multiple layers starting with one to two convolutional layer (shallow networks) to as much as 22 layers (deep networks) [81]. When compared to other networks, it is agreeably the most popular deep learning architecture. Typically ConvNets architecture is structured as a series of stages consisting of the following layers:

- *Convolutional Layer*: This layer extract features from the input data by applying sliding Convolutional filters whose step size is determined by strides. Conv layer retains the spatial relationship between the data by learning its features which are determined by its kernel size. Figure 4.2 shows convolution operation given 3x3 matrix kernel.

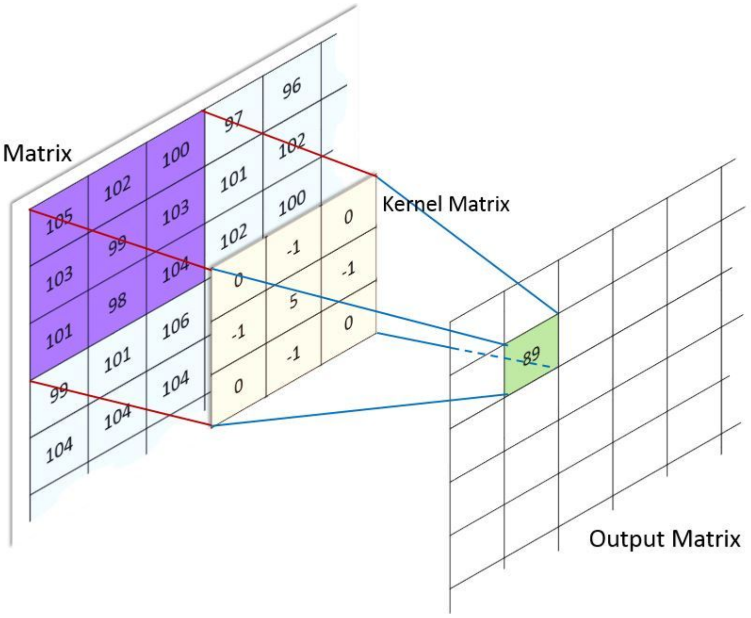


Figure 4.2: Example of Convolution Layer

- *Pooling Layer*: It represents an intermediate feature that can make the CNNs more translational variant. Basically, it downsamples the features by retaining the most important information by computing the maximum or average over a given neighborhood. Its size is determined by the pooling window and the number of steps to slide is given by strides. Figure 4.3 shows pooling layer with stride and pooling window of [2,2].

- *ReLU function*: Rectified linear unit (ReLU) is a nonlinear operation which accounts for nonlinearity by replacing all the negative values with zero. It in-

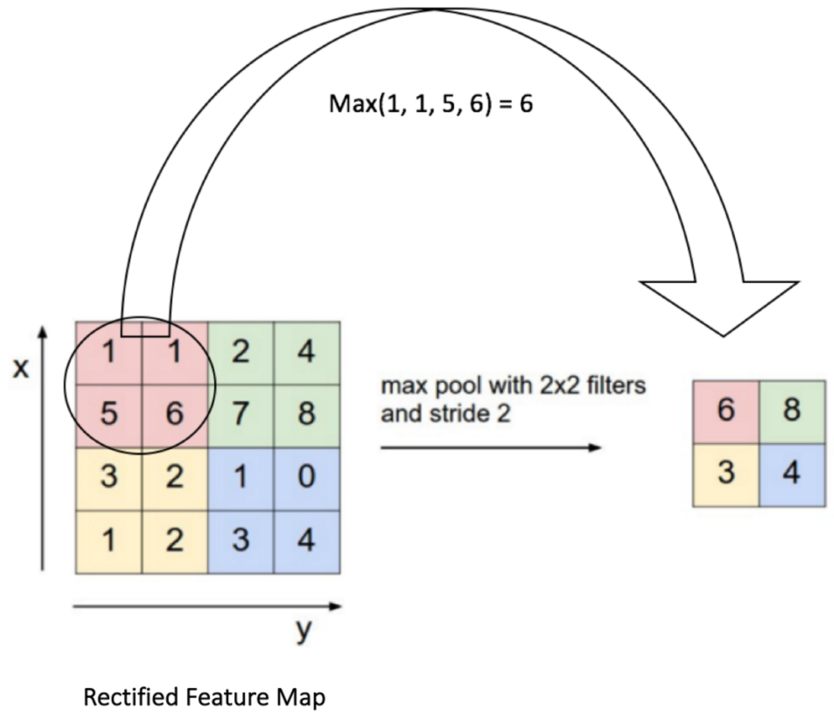


Figure 4.3: Example of Pooling Layer

creases the nonlinear properties of the network and individual functions without affecting the receptive fields of the convolution layer.

$$f(x) = \max(0, x) \tag{4.1}$$

The advantage of using ReLU activation is that it trains the network faster without serious penalty to generalization accuracy.

- *Fully Connected*: It is simply a multilayer perceptron in which every neuron in the previous layers is connected to every neuron on the next layer. This layer is used along with softmax for classifying the input data into various classes. The standard (unit) softmax function, $\sigma(z) : \mathbb{R}^{\mathbb{K}} \to \mathbb{R}^{\mathbb{K}}$ is defined by the formula,

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{4.2}$$

for j=1,....,K and z=(z_1,...z_K)$\in \mathbb{R}^{\mathbb{K}}$

- *Loss Layer*: This layer specifies how well the model is getting trained by penalizing the difference between the predicted output and true output which is the final layer of the network. Depending on the application various loss functions have been used. Generally, for classification problems, cross entropy loss is computed which involved multi-class with mutually exclusive classes. If there are K classes then the cross-entropy function uses 1-of-K coding scheme.

$$loss = -\sum_{i=1}^{N}\sum_{j=1}^{K} t_{ij} \ln y_{ij} \tag{4.3}$$

where N is the number of samples, K is the number of classes, t_{ij} is the indicator that the ith sample belongs to jth class, and y_{ij} is the output for sample i for class j, which in this case, is the value from the softmax function. That is, it is the probability that the network associates the ith input with class j.

In addition to these layers, there are few other techniques or layers which were introduced with the recent advancements in deep learning known as regularization techniques. Regularization is a process of introducing additional layers to the conventional ones to prevent overfitting. Regularization is applied when the objective functions are in ill-posed optimization problems. Some of the regularization methods are listed below:

- *Batch Normalization*: This layer normalizes the mini-batch training sets close to normal distribution. This technique is used to improve the performance and stability of the network. This is achieved by normalizing the input layer by adjusting and scaling its activation. In this work, we apply batch normalization before nonlinearity as recommended by the paper [82].
- *Dropout Layer*: This layer avoids network to overfit the data by randomly dropping units along with their connections during training with probability p [83]. This means those neurons are basically disabled from the connection. Figure 4.4 visualize the concept of drop with 50% probability for hidden layer. The probability of drop out is controlled by its hyperparameter p at every

training step. Disabling neurons performs better because dropout prevents the network from depending on a specific number of neurons and forces every neuron to work independently.

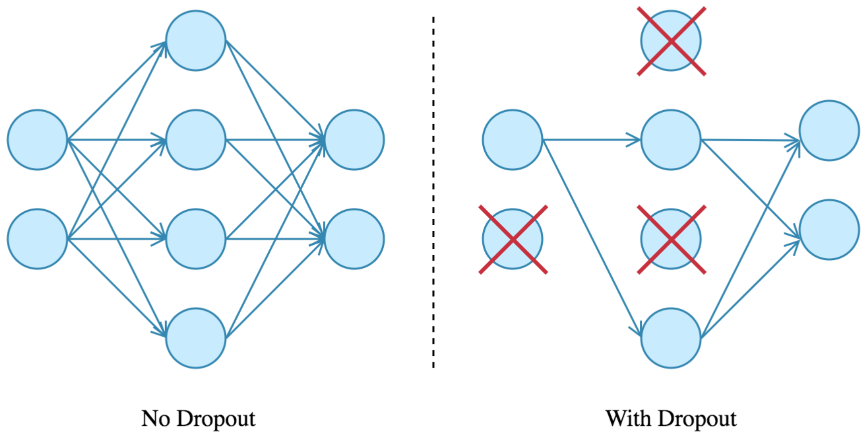


Figure 4.4: Visualize with and without Dropout layer [84]

4.2 Online Machine Learning

Online machine learning is a type of machine learning in which a sequence of questions is answered in sequential order. At each step, these answered data is further used to update the predictor such that the model could accurately predict for future questions. Online learning (refers to online machine learning) has become an important domain in machine learning that has interesting theoretical and practical applications. Online learning is most commonly used in applications where it is computationally not feasible to train data over the entire dataset and requires to use out-of-core algorithms. Online learning is data efficient - it sees data only once and it is no longer required to store, and adaptable - it makes no prior assumption about data distribution. The main advantage is as the data distribution changes either smoothly or rapidly, the model can adapt on-the-fly to keep trends in real-time. Hence it is used in situations where the model needs to dynamically adapt to new patterns like the stock prediction. The main disadvantage of having online learning algorithms is that they are prone to catastrophic interference which is a tendency of a network to completely forget the previously seen data upon seeing new information. To overcome these problems incremental learning approaches are used [85].

Online learning is performed with a sequence of cycles consecutively. At each cycle t the model is given an input x_t from dataset X, the model predicts its output with prior knowledge denoted by p_t. The actual correct answer is denoted as y_t which is taken from target data Y. The model suffers a loss of $l(p_t, y_t)$ which measures the deviation between the predicted and actual ones. The specific case of yes/no answers and predictions, namely $D = Y(0,1)$ is called online classification. In these cases, the loss function is nothing but the absolute distance between predicted output and actual output.

Some of the well-known algorithms used in online learning are Perceptron, Stochastic gradient descent, Naive Bayes classifier for classification problems. Figure 4.5 shows the block diagram of online learning and Algorithm 1 shows its implementation. Other than machine learning, online learning has been widely studied in several other research fields including game theory and information theory.

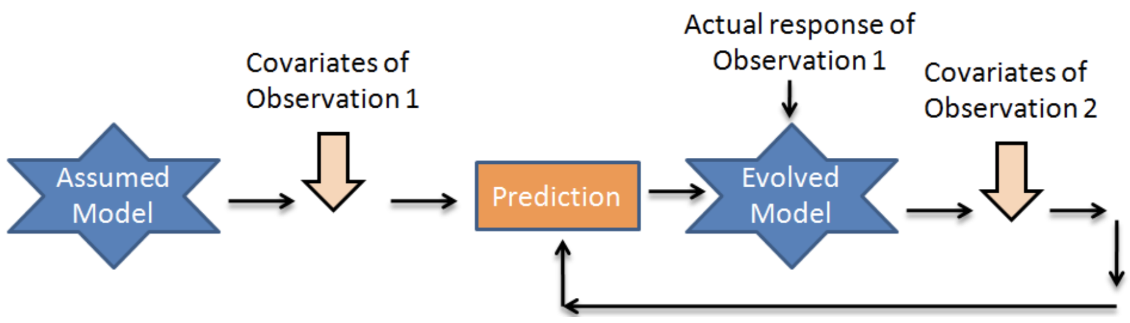

Assumed Model
Covariates of Observation 1
Prediction
Actual response of Observation 1
Evolved Model
Covariates of Observation 2

Figure 4.5: Visualization of Online Machine Learning [86]

Algorithm 1 Online Learning

```
1: for $i = 1, 2, \ldots$ do
2:     receive question $x_t \in \chi$
3:     predict $p_t \in D$
4:     receive true answer $y_t \in Y$
5:     suffer loss $l(p_t, y_t)$
6: end for
```

4.3 Stochastic Gradient Descent

Among the different online learning algorithms, one such algorithm which is widely used for the applications were data is available sequentially is Stochastic gradient descent often abbreviated as SGD. SGD is a type of gradient descent which is a first-order iterative optimization algorithm for finding the minimum of a function. Gradient descent is used to find local minima of any function, hence it is known as steepest descent. SGD is iterative method which is also used to optimize a objective function which is differentiable which is nothing but performing a stochastic a approximarion of gradient descent optimization. Initially in 1951 Herbert Robbins and Sutton Monro proposed an algorithm in their article title "A Stochastic Approximation Method" [87]. They presented a methodology for solving a root finding problem when the function is represented as an expected value. The main difference between a standard gradient descent and that of stochastic gradient descent is that, in gradient descent the samples are selected as a single group or the way they appear in the training set where as in SGD the samples are randomly selected or shuffled.

In simpler terms, SGD is used iteratively to make small adjustments to any type of model configuration (machine learning network in our case) to decrease the error of the network. Error function isn't any complicated one, it is rarely as simple as a parabola and more often with ups and downs like hills and valleys. As discussed, in general gradient descent looks for local minimum, if the gradient descent started from the left side of the graph as shown in figure 4.6, it would stop at the left valley irrespective of direction of travel from that point which is known as local minimum. However, for the entire graph there exists one unique point which is known as global minimum and SGD algorithm attempts to find the global minimum.

The problem of minimizing an objective function in any of the machine learning

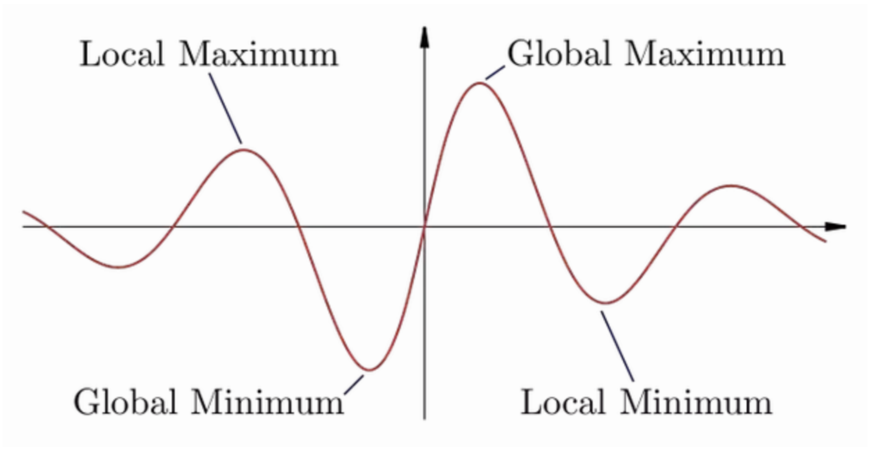


Figure 4.6: Global and local minimum of 1-D function [88]

has the form of the sum:

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w) \tag{4.4}$$

where the parameter w that minimizes $\mathbf{Q(\boldsymbol{w})}$ and is the function under evaluation. Each function Q_i is associated with the ith example or feature in the dataset which is normally used for training.

In gradient descent used for online processing, the actual gradient is approximated by gradients for each and every example data as the algorithm goes through every data in the dataset. The equation for weights updation is given as:

$$w := w - \eta \nabla Q_i(w) \tag{4.5}$$

SGD tries to find the global minimum by adjusting its configuration of the network at every training example rather than decreasing the error by finding its gradient for the entire dataset. By doing so, SGD decreases the error by approximating the gradient of one training example or minibatch samples. In general, many passes are perform in order to see the algorithm convergence. Once such approach is to shuffle the data for every pass to prevent cycles. Most common approach is to use an adaptive learning rate so that the algorithm converges. This method is very helpful in cases of wrong training example which will increase the error of the network, SGD updates the configuration such that it can get the right one in the future. However, this update might cost of predicting the other examples wrong and increasing the over all error of the network. Thus every iteration of the SGD doesn't improve the network as shown in Figure 4.7.

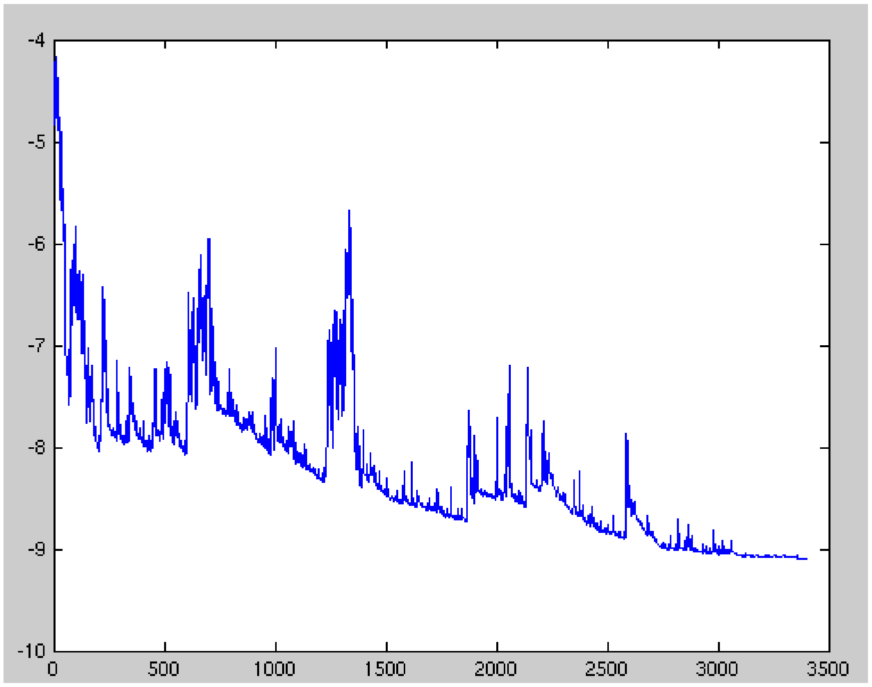


Figure 4.7: Fluctuations in the total objective function as gradient steps with respect to mini-batches are taken [89]

Another important aspect of using SGD is that it adjusts the network parameters such that if the model is struck at the local minimum, it makes a way to move out and approach towards global minimum. During this process, there are changes of increasing the error of the network, but this will make the algorithm reach the global minimum point. Advantages of using SGD is that it requires less memory to store the data as it sees one data at a time, it requires less computation when compared to true gradient descent and most of the times it stops at the global minimum.

CHAPTER 5: ROBOT MOTION PLANNING

As we saw in the previous chapter, robots are classified based on many different things. One such is the coordinate system in which they are normally described. Robots based on the coordinate system are classified as

- **Cartesian**: Robots defined by movement limited by three prismatic joints.
- **Cylindrical**: If one of the Cartesian robot's prismatic joints is exchanged for a revolute joint, a cylindrical robot is formed. A cylindrical robot's movement is defined by a cylindrical coordinate system.
- **Spherical**: Trading two prismatic joints for revolute joints forms a spherical robot. Spherical, or polar, robots are devices whose axes form a polar coordinate system.
- **Articulated**: Substituting a revolute joint for the final prismatic joint turns the arm into an articulated arm. Any robot whose arm has at least three rotary joints is considered to be an articulated robot.
- **SCARA**: SCARA (which stands for Selectively Compliant Articulated Robot Arm) is a specialty robot which has two parallel rotary joints to provide compliance in a plane. A third prismatic joint allows the arm to translate vertically. SCARA robots differ from articulated robots in that its workspace consists of two concentric cylinders.

Many different topologies are possible in which the links can be connected, most of the mechanisms are either a serial chain or fully parallel mechanisms. A serial chain is a system where all the objects or bodies are connected to two other objects, except the first and last ones which are connected to just one object. Whereas fully parallel is one in which two or more objects/bodies are connected to multiple joints. Our interest and requirement are towards serial chain mechanism robotic arm. For the application of assistance, researchers have used different types of robots like Baxter, NAO and PR2. For example, some researchers have used the robotic arm connected to the wheelchair base such that the robot can be a helping hand. Others have used a robotic arm over a mobile base that could navigate with the disabled. Here as our interest was towards the humanoid robots. Hence we use PR2 robot by Willow garage which is also known as personal robot 2.

5.1 PR2 Robot

The PR2 robot is a wheel based service robot with two 7 DOF arms, a tilting head, and a sliding joint. The PR2 robot can adjust the height of the arms and adds an extra degree of freedom to the system. It can navigate on the floor and rotate on its wheels around the z-axis providing three extra degrees of freedom to the system. Figure 5.1 shows the right arm specifications. Our vision of interest is towards arms and head of the robot.

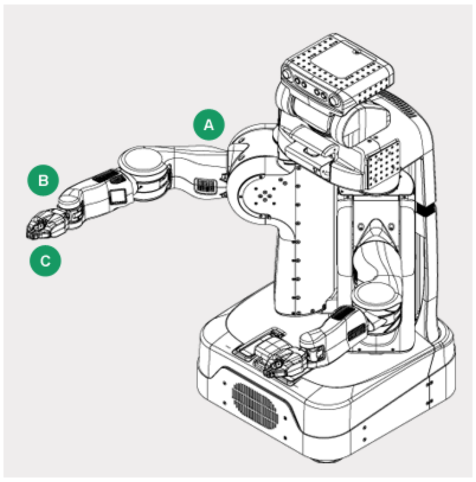


Figure 5.1: PR2 Arms DOF: (A) Arm: 4, (B) Wrist: 3, (C) Gripper: 1

PR2 robot is used for many applications and is categorized into three main categories as shown in figure 5.2.

- **Work Life**: From serving food to manufacturing where people spend most of their time doing repetitive tasks.
- **Personal Life**: From doing dishes at home to doing laundry.
- **Independence**: Providing assistance to people who need a human assistance.

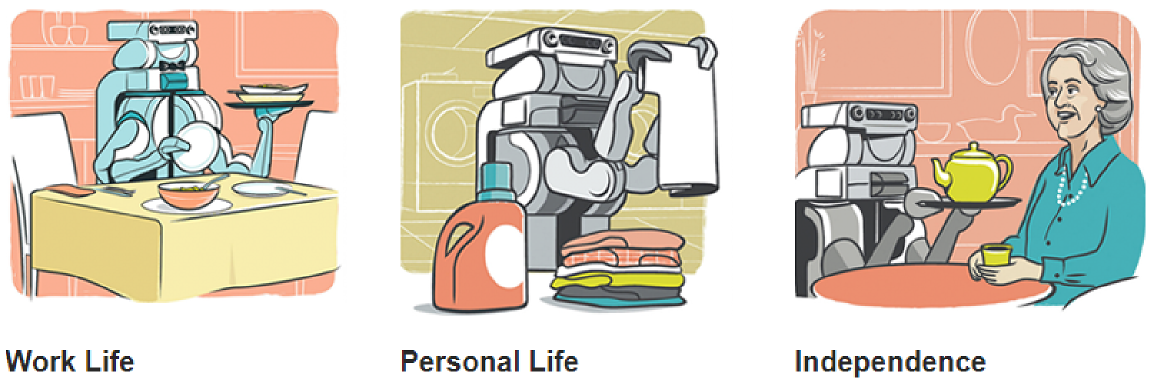


Figure 5.2: Possibilities for PR2 applications

5.2 Kinematics

Kinematics is the study of the motion of points, objects and group of objects without considering the forces that caused the motion. Often Kinematics is referred to as the "geometry of motion". In the robotics arm sense, Kinematics is used to describe the motion of systems composed on a multi-link system or joints. Kinematics is the most fundamental aspect of robot design, control, analysis, and simulation because their mechanisms are specifically for motion purposes. Kinematics defines the variables such as position, velocity, acceleration and higher derivatives of the variables. The important aspect of robot kinematics involves deriving equations. These equations are derived to construe an analytical relationship between the robot's joint parameters and end-effector. There are two ways to derive the equations of motion:

- **Forward Kinematics**: These equations are used to calculate the robot's end-effector position in the coordinate space given its joint angles.
- **Inverse Kinematics**: These equations are used to calculate the joint angles required to move end-effector to any specific location given the robot's end-effector location.

5.2.1 Forward Kinematics

Forward kinematics for a serial chain manipulator is to find its position and orientation of the end-effector relative to the base. It is calculated by providing the position of all joints and their values of all the geometric link parameters. In general, the end-effector's frame is referred to as a tool frame and other frames are referenced with this one. A more general expression of the forward kinematics problem is to find the relative position and orientation of any two designated members given the geometric structure of the manipulator and the values of a number of joint positions equal to the number of degrees of freedom of the mechanism. In practical, there arise many problems dealing with forwarding kinematics because the values i.e., joint positions are measured by sensors mounted over joints and it is fundamental to calculate the positions of the joint axes relative to the fixed reference frame [90].

In the real scenario, to solve the problems related to forward kinematics, the concept of transformation is used. The transformation is performed between the reference frame fixed in the end-effector and all other frames in the system. For serial chain manipulators, transformation is simpler when compared to parallel chain systems. Here basically we describe the position of the end-effector relative to the base which is obtained by concatenating transformations between frames fixed in adjacent links of the chain. Prior to the transformation, a homogeneous transformation matrix is formed which shows the geometric representation of a manipulator. The homogeneous transformation matrix is a 4x4 matrix that related the spatial displacement of the end-effector reference frame to the base reference frame [90].

5.2.1.1 Geometric Representation

To represent the geometry of a robotic manipulator Denavit and Hartenberg [91] introduced a fundamental convention which can help to conveniently define by attaching reference frames to each link and to maintain the consistency to adhere to a convention for locating the frames on the links. Instead of six parameters, only four parameters are used to locate a one reference frame to another. The four parameters being link length a_i, the link twist α_i, the joint offset d_i and the joint angle θ_i. The convention is applicable to manipulators having revolute and prismatic joints. In general for a six-degree-of-freedom serial chain manipulator composed of an articulated arm with no joint offsets and a spherical wrist, and neglecting the addition of tool and station frames, the transformation is given by

$$ {}^{0}\boldsymbol{T}_6 ={}^{0} \boldsymbol{T}_1^1\boldsymbol{T}_2^2\boldsymbol{T}_3^3\boldsymbol{T}_4^4\boldsymbol{T}_5^5\boldsymbol{T}_6 \tag{5.1}$$

5.2.2 Inverse Kinematics

Inverse kinematics for a serial chain manipulator is that given the end-effector's position and orientation relative to the base and geometric link parameters, the problem is to find the values of joint positions. As in forwarding kinematics, for inverse kinematics, the general statement is that to find the values of joint positions given their relative positions and orientations. Using this, all the joint positions can be calculated given the homogeneous transformation between the two members of interest [90].

For a six-degree-of-freedom serial chain manipulator composed of an articulated arm, the transformation is given by ${}^{0}\boldsymbol{T}_6$. From Equation 5.1 it is clear that for serial-chain manipulators the inverse kinematics problems require the solution of nonlinear sets of equations. For the standard example of six-degree-of-freedom serial chain manipulator, three of these equations relate to the position vector within the homogeneous transformation, and other three relate to the rotation matrix. Duffy in his book explains that there are many ways that no solution exist or multiple solution situations may arise. For the solution to exist, the desired position and orientation should lie in the workspace of the manipulator [92]. There are many ways of solving the solution for the given set of equations. Often two methods are used: Closed-Form solutions and Numerical methods. For situations where the solutions exist and cannot be presented in a closed loop form, numerical methods are used [90].

Among the different methods to solve the problem closed-form solutions are generally faster when compared to numerical solutions. But the closed-form solutions cannot be generalized and are specific to one single robot. In contrary, numerical methods are robot independent which means they can be applied to any kinematic structures. The only problem with numerical methods is they consume a lot of space and time to solve the set of equations and do not allow computation of all possible solutions. Closed-form solution cannot solve any numbers of nonzero geometric parameters. For such manipulator structures, the numerical methods are used. More often, the numerical methods are divided into categories of symbolic elimination methods, continuation methods, and iterative methods [93].

BFGS (Broyden–Fletcher–Goldfarb–Shanno) projection algorithm is used which is an iterative and gradient-based optimization technique and is used to solve unconstrained nonlinear optimization problems. BFGS method is a quasi-Newton method that is the type of hill-climbing optimization techniques which starts from an initial guess at the solution and tries to minimize the cost function. In this method, the second-derivative information is approximated using the gradients of the cost function from past iterations. It uses this second-derivative information in deciding the next step for the next iteration. While in this process if the algorithm converges to a configuration where the cost approaches close to zero or within the tolerance, it is assumed that the solution for the inverse kinematics problem is found. Nonetheless, if the solution is not found either for the combinations of initial guesses, the algorithm restarts with random restart mechanism. Random restart mechanism restarts the iterative search when it fails to find a configuration that achieves the desired pose of end-effector. Typically, random restarts are continued until the solution is found or the maximum time or iteration limit is exceeded [94].

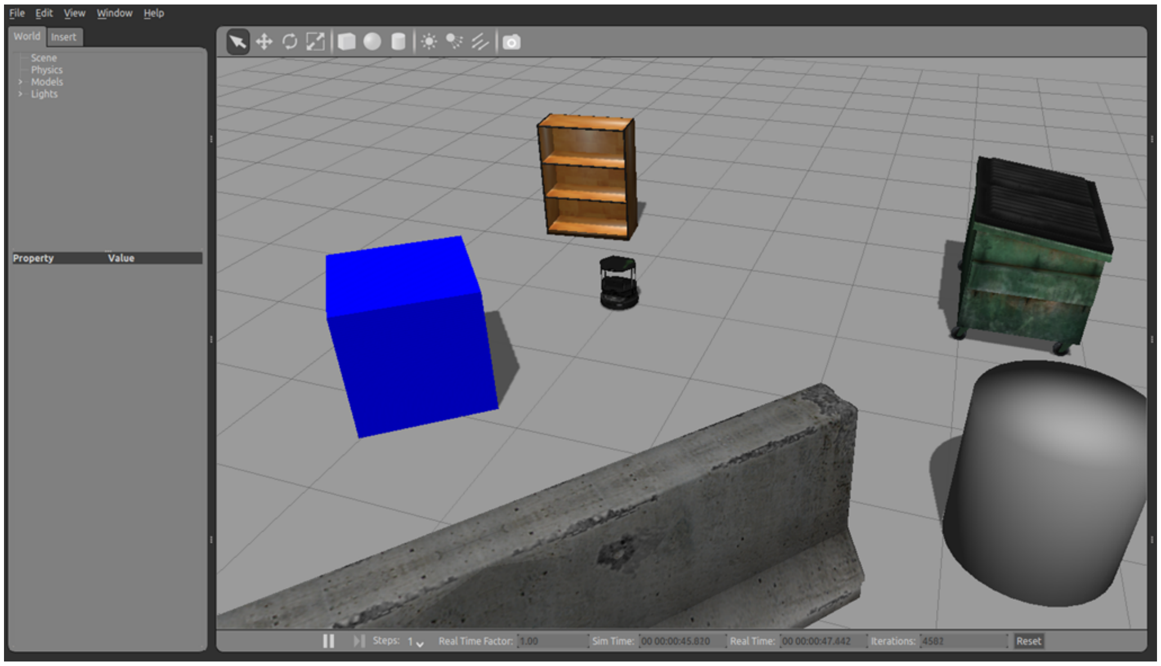
5.2.3 Instantaneous Kinematics

Instantaneous Kinematics deals with the change in motion or change in position of joints or end-effector [95]. Instantaneous Kinematics are of two types: forward instantaneous kinematics and inverse instantaneous kinematics. The forward instantaneous kinematics problem for a serial-chain manipulator is to find the total velocity of the end-effector, given the positions of all the joints. If the rate of motion for revolute joints is known as angular velocity and for prismatic joint are known as translational velocity. For any joint, the total velocity of the joint is the velocity of the origin of the reference frame fixed to it along with its angular velocity. Forward instantaneous kinematics always comes before forward kinematics is solved. The forward instantaneous kinematics problem is majorly used to analyze acceleration and to study its importance in dynamics. The inverse instantaneous kinematics problem for a serial-chain manipulator is to find the rates of motion of all joints, given the positions of all the joints and the total velocity of the end-effector. From the robotic point of view, inverse instantaneous kinematics problems are a very crucial problem. When controlling the motion of robotic arm which normally operates in the point-to-point mode, it is not only necessary to compute the final joint positions needed to assume the desired final hand position. More importantly, it also necessary to generate a smooth trajectory for motion between the initial and final positions. But the problem is that there is an infinitely possible trajectory between any two points. To solve these problems, there are many successful algorithms employed for inverse instantaneous kinematics problems. In our case, once the robot's joint angles are calculated using the inverse kinematics equations. Finally, to generate a motion profile, the Jacobian matrix is used to move the end-effector from the initial to the final location. Jacobian matrix helps to define a relationship between the robot's joint parameters and the end-effector velocities as shown in equation 5.2.

$$J(x_1, \dots x_n) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{5.2}$$

5.3 MATLAB Robotics Toolbox

MATLAB's robotics system toolbox provides various inbuilt algorithms for developing autonomous applications. It also provides hardware connectivity for these applications. Some of them are aerial vehicles, ground vehicles, manipulators and humanoid robots. This toolbox provides algorithms relating to path planning and path following for differential drive robots. It also has algorithms inbuilt such as scan matching, obstacle avoidance, and state estimations. Especially for robots such as PR2 which is a manipulator robot, this toolbox includes algorithms for inverse kinematics, kinematic constraints, and dynamics using a rigid body tree representation. More importantly, it provides an interface between MATLAB and the robot operating system (ROS) which is very useful to test and verify applications. Some of ROS enabled applications being either real ROS-enabled robots or robot simulators such as Gazebo. Figure 5.3 shows the virtual simulator gazebo with turtlebot and real-world objects.



Steps: 1 Real Time Factor: 1.00 Sim Time: 00 00:00:45.820 Real Time: 00 00:00:47.442 Iterations: 4582

Figure 5.3: Gazebo simulator

CHAPTER 6: SYSTEM FRAMEWORK

6.1 Overview of the System

Assistive robots are used for doing the daily life activities such as cleaning, trimming, feeding and many more as discussed in the previous chapters. But all these tasks are carried out autonomously or self-decided by the robot. However, there are tasks such as sorting objects, relaxing (either listening to music, watching a movie) can be very tedious to a robot to understand what human needs at that moment. Researchers have come up with solutions such as emotion recognition of the disabled but when it comes to the object sorting kind of tasks, every individual categorizes objects based on his/her interests and not every configuration be pre-trained to the robot.

Our main goal is to bridge the communication between the disabled and the assistive robot such that the daily activities can be performed as per the choice and interest of disabled. Here we demonstrate object sorting and classification performed by an assistive robot with the supervision of disabled i.e., by sending feedback to the robot with human brain signals. Figure 6.1 provides a complete overview of the research. Objects used in daily life such as coke can, cube, tools, etc are kept on the table in front of the assistive robot. Here PR2 is used as an assistive robot. A disabled person is made to sit in front of the robot such that he/she visually inspects the actions performed by PR2. EEG electrodes are placed on to the human scalp and their EEG signals are being continuously monitored. The two bowls on the left and right side of the robot are considered to be two categories in which the objects will be sorted.

The whole process is broken into different steps given below:

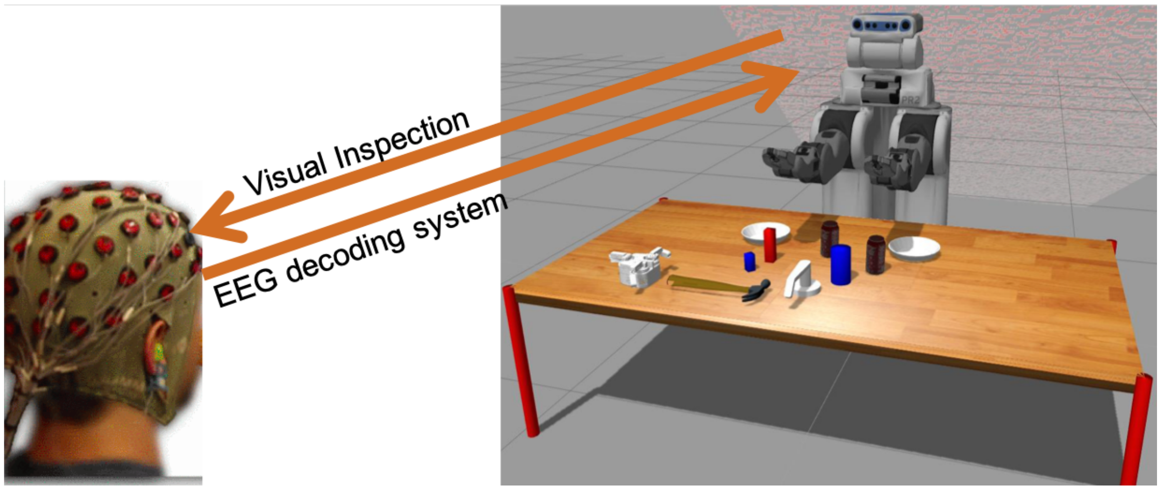


Figure 6.1: Overview of the system

1. The robot randomly selects the object to be picked from the objects kept on the table visually using its head camera.

2. Once the object is selected, the robot plans its trajectory to pick the object from the robot's initial position to the object's position and executes the plan. Trajectory planning is done with respect to the position of end-effector.

3. After robot picks up the object, the goal location needs to be decided i.e., to decide upon the category it belongs to. This is done using the prediction results from the self-learning algorithm. In the initial stages as the robot isn't aware of the category of objects, it randomly classifies based on its minimum knowledge.

4. Two goal locations are the positions of bowls kept on either of the sides of the robot.

5. Once the goal location is set. The robot plans its trajectory from the picked object to its goal in discrete steps. It samples the whole trajectory into a discrete number of steps such that it can verify its action with a human by monitoring his/her brain signals.

6. As the planning is done to its final location, but robot discretizes the planned trajectory space into discrete 3-D locations such that after each location it

reaches it can verify whether error potential is received or not.

7. If the error signal is received then robot changes its goal location and replanning work is taken place. If an error isn't received the robot continue the onto the planned trajectory until it reaches its goal position.
8. Once the trajectory planning is executed, the robot self learns the object's target using the online machine learning algorithm such that if the robot encounters the same object in future, its initial classification would be correct.

To visually understand the research, the process flow is shown in Figure 6.2.

The work is divided into three parts: (1) Error-related potential (ErrP) classification, (2) pick and place an object, and (3) robot self-learning. In the first part, the EEG signal collected from the human brain is analyzed, processed and classified into binary feedback i.e., if it is error related potential (ErrP) or not. The second part consists of trajectory planning from an initial position to object position, plan execution, picking the object, trajectory planning from the current object position to its goal position and finally placing the object. Last part is all about the robot self-learning. As the robot knows which objects belong to which category or group, it trains its online-learner such that it could classify correctly if the same object is encountered in the future.

6.2 ErrP Classification

Generally, raw EEG are collected from the human brain. But in our research, the data was collected from researchers working towards error-related brain signals and was preprocessed. In order to classify the signals well known deep learning model such as a convolutional neural network (CNN) is used. Here we consider two class problem which means the classifier classifies the data whether it belongs to ErrP which means error signal or non-ErrP which means normal EEG signal. The detailed block diagram of the first part i.e., classification of error related potential is explained

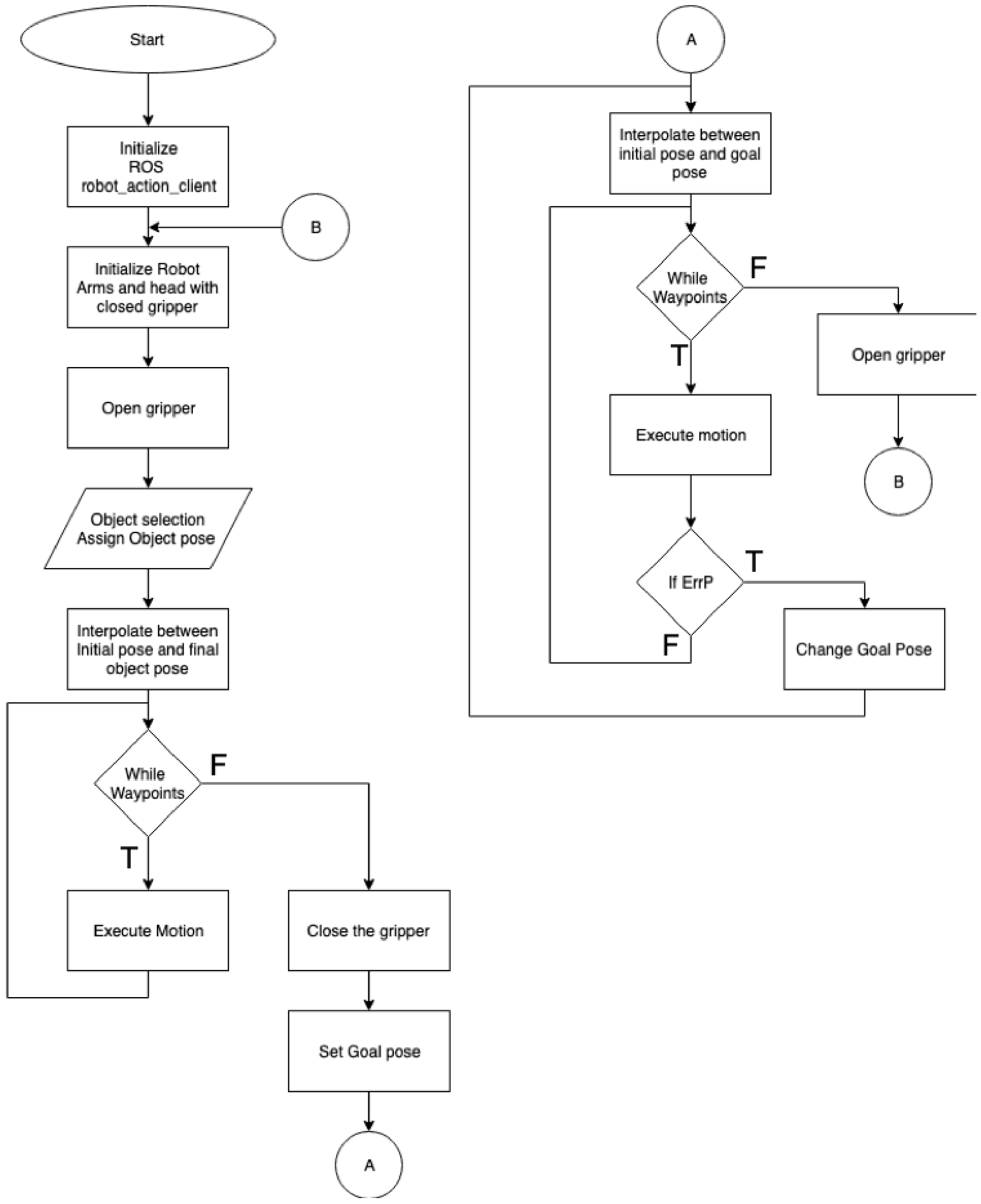

Start
Initialize ROS robot_action_client
B
Initialize Robot Arms and head with closed gripper
Open gripper
Object selection Assign Object pose
Interpolate between Initial pose and final object pose
While Waypoints
F
T
Execute Motion
Close the gripper
Set Goal pose
A
A
Interpolate between initial pose and goal pose
While Waypoints
F
T
Open gripper
Execute motion
B
If ErrP
T
F
Change Goal Pose

Figure 6.2: Flow chart of the system

in Figure 6.3.

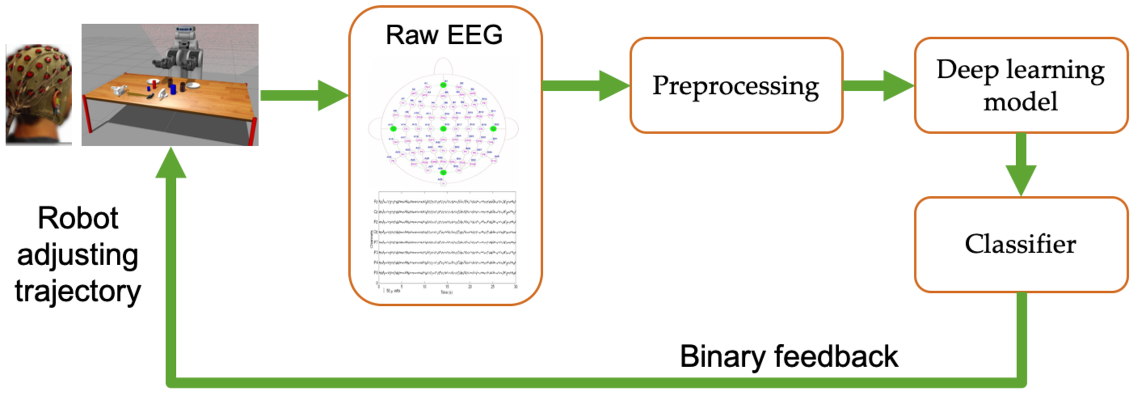


Figure 6.3: Detailed block diagram of the system

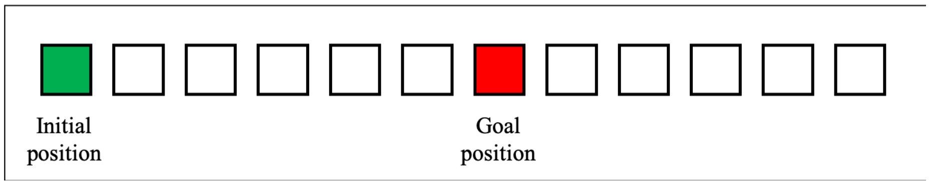
6.2.1 Dataset

The EEG signals for this study were collected from publicly available dataset [59]. Ricardo et al. studied the existence of error related potentials (ErrP) and its feasibility to decode them in the single trail with the interest towards the brain-computer interface. This dataset corresponds to an experiment conducted by researchers from the Swiss Federal Institute of Technology in Lausanne in order to study ErrPs [96]. ErrP were elicited when the user monitors the behavior of an external device upon which the subject had no control.

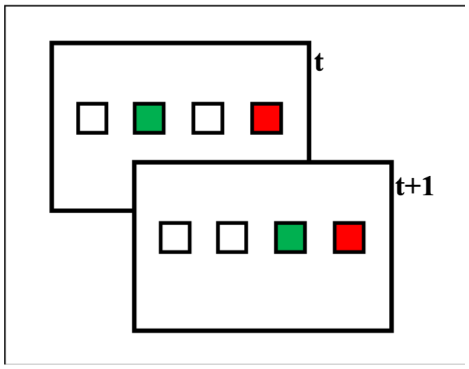
6.2.1.1 Experimental protocol

Subjects were made to sit in front of a computer screen. The computer screen had a moving cursor was in green and the target location was displayed in red as shown in Figure 6.4(a). The cursor's working area had a total of 20 locations along the middle horizontal plane of the computer monitor and subjects were made to sit such that their center is with the center of the screen. At each time step t the cursor either moves some step towards its right or left horizontally. Depending on the target's location, if the current cursor's position is moving towards target location at next time step as shown in Figure 6.4(b) then no ErrP is generated as it is towards the goal/target position. If the cursor moves towards the opposite direction as shown in

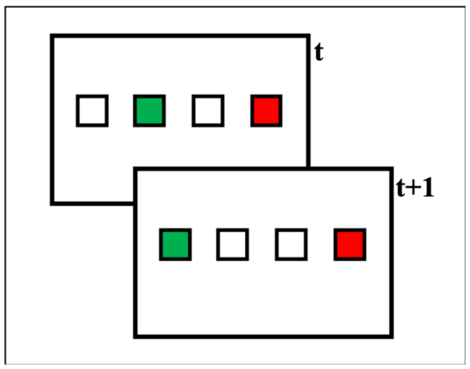
Figure 6.4(c) then ErrP is automatically generated from the human brain. During the entire experiment, the user has no control over the cursor's movement and is asked to only monitor the performance of the agent. In order to generate erroneous action, at each time step the probability of cursor going in the wrong direction was 20% i.e., going in the opposite direction that of the target location.



(a)



(b)



(c)

Figure 6.4: Experiment Protocol (a) Experimental setup; (b) ErrP not received; (c) ErrP received

6.2.1.2 About the dataset

This dataset contains recordings of six subjects (mean age 27.83 ± 2.23) performed two sessions of recordings separated by several days as shown in Table 6.1. Each experimental session consisted of 10 blocks of 3 minutes each.

Table 6.1: Difference in days between recording sessions for each subject

Subject	1	2	3	4	5	6
Days	51	50	54	211	628	643

EEGs were recorded at full DC at a sampling rate of 512 Hz for all the 6 participants using the Biosemi ActiveTwo system. 64 electrodes were placed on subject's scalp according to the 10/20 International system as shown in Figure 6.5.

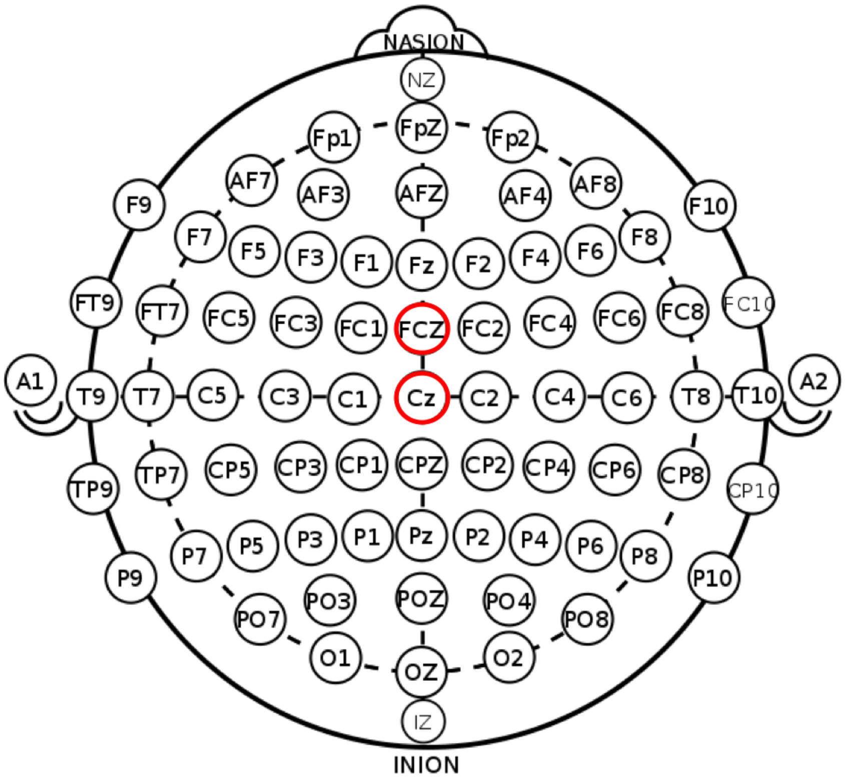


Figure 6.5: Standard 10-20 International System for 64 electrodes

6.2.2 EEG Pre-processing

The raw data is collected and analyzed visually. Since the EEG data is noisy and has poor signal to noise ratio, filtering was performed. Raw EEG data were spatially filtered using common average reference filter as performed by many researchers working towards EEG signals. Later, the signals were spectral filtered using bandpass filtering. The lower cutoff and higher cutoff frequencies were set to 1 Hz and 10 Hz respectively. Both spatial and spectral filtering was performed using EEGLAB [63].

Since we know that not all parts of the brain respond the same to every event, we had to decide upon the electrodes which contribute the most towards eliciting the error related potentials. Spatiotemporal maps were plotted using EEGLAB. Electrodes were selected based on visual inspection of the topographical representation of the averaged ErrP as shown in Figure 6.6. As proposed by Holroyd and Coles most of the variation was found to be at the region of Anterior Cingulate Cortex (ACC) [97]. Hence, out of 64 electrodes, only data from two electrodes were selected 'Cz' and 'FCz', as shown in red circles in Figure 6.5.

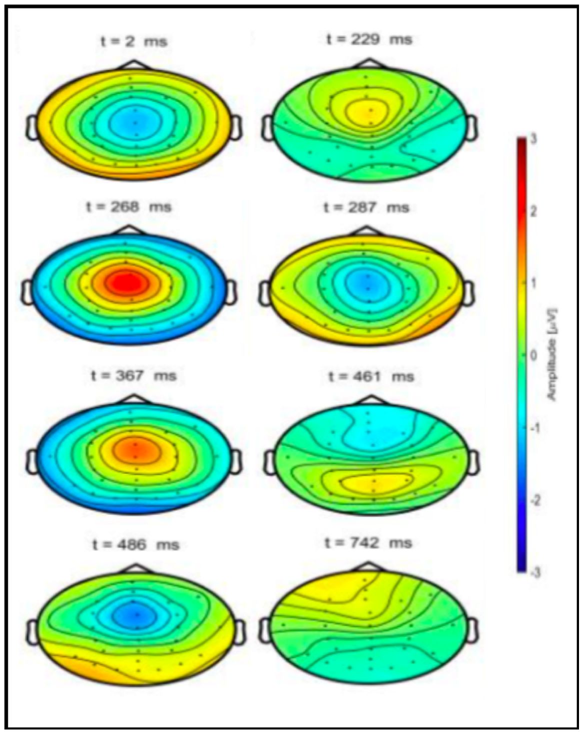


Figure 6.6: Topographical maps visualization of scalp activity [98]

6.2.3 Input Data Representation

The input data is represented as a 2D array with the number of time steps as the width of the array and number of electrodes as their height. Each training example was of size 2x512 and we had a total of 1322 samples of ErrP data and rest were normal data and their corresponding labels were marked as binary values. Due to the unequal number of samples in each class, there arose an imbalanced dataset problem known as between-class imbalance [99]. In order to overcome this problem, we had two possible approaches:

1. Randomly reject a few samples from the class which has more class and make it an equal number of samples as other class. The problem here is we lose a lot of

data and information and our model is not well trained by rejecting the data.

2. Replicate the class 1 data equal to the size of another class, where the problem of overfitting.

We choose the latter approach and to tackle the problem of overfitting we use regularization methods such as drop out layer. Finally, the dataset was divided into three groups: training(70%), validation(10%) and testing(20%).

6.2.4 Training

The training was done using a deep convolutional neural networks as proposed by Sunny and Conrad [100]. CNN had a five layered architecture. Each layer consists of a convolutional layer, pooling layer and ReLU activation function except for the first layer. CNNs were used because of the following advantages:

- CNNs performs end to end analysis which means it learns. directly from raw data, and need not perform the feature extraction
- Pivotal component of CNNs is their ability to learn from local patterns.
- The initial convolutional layers extract the low-level features from input data, and as we go much deeper into the network it learns more global and high-level features.
- CNNs scale well to large datasets and can exploit hierarchical structure in natural signals.

Two different architectures were trained, one with the traditional layers (hereafter known as CNN1) and the other with additional layers such as batch normalization and drop out layer (hereafter known as CNN2). The basic architecture is described in Table 6.2. For all the layers no padding was used and the stride was set to be [1 1].

Table 6.2: deep Convolutional neural network Architecture

Layer	**stage**	**kernel**	**num of filters**
Input	-	-	0
Convolution	1	2x64	16
Convolution	2	1x64	32
ReLU activation	2		0
Max Pooling	2	1x2	0
Convolution	3	1x32	32
ReLU activation	3		0
Max Pooling	3	1x2	0
Convolution	4	1x16	64
ReLU activation	4		0
Max Pooling	4	1x2	0
Fully Connected	5	2	2
Softmax activation	5		0
Total Learnable Parameter			239938

Supervised training is performed using stochastic gradient descent with momentum to minimize the difference between actual and predicted outputs with fixed learning rate schedule. We optimized our validation accuracy for changes in the learning rate, the number of epochs, momentum, batch size, and the best-performed ones was selected as shown in Table 6.3.

Table 6.3: Training Parameters

Parameter	**Value**
Optimization	Stochastic Gradient Descent with momentum
momentum	0.9
Initial learning rate	0.001
L2 regularization	0.00001
Number of epochs	1000
Batch size	512

6.3 Robot Trajectory Planning

The core part and most important part is the trajectory planning for a robot. It plans the trajectory of motion of the end gripper over a 3-D coordinate space. The trajectory planning section is further divided into three subdivision: (1) Initialization, (2) Trajectory Planning, and (3) Motion execution

6.3.1 Initialization

The PR2 Robot is initialized for the following things:

- To connect to the ROS network from MATLAB.
- To create action clients for robot arms and head.
- Send commands to robot arms & head such that the robot initializes its position.

ROS initialization starts the global ROS node with a default MATLAB and tries to connect to a ROS master running on localhost or another computer whose IP address is used to specify the master. Some of the prerequisites are that in order to communicate with a ROS network, ROS node needs to be connected to the ROS master. Here we use two different computers, one running gazebo with the PR2 robot model and other computer running MATLAB which acts like master node and ROS master respectively.

Before sending commands i.e., joint trajectories to the PR2 arms, which are controlled via ROS actions. They need to be initialized with all the joints linked to that action client. Normally, joint trajectories are manually specified for each arm and sent as separate goal messages via separate action clients. Once the ROS action client is created rosactionclient object and goal message are returned. Wait for the client to connect to the ROS action server. The arm joint trajectory is set by creating the rosaction client to the 7 joints of control: shoulder pan joint, shoulder lift joint, upper arm roll joint, forearm roll joint, wrist flex joint, wrist roll joint for both right and left hand. Figure 6.7 shows the links and its names in PR2 for the right arm

Both arms are moved such that the robot's arms doesn't cover the region of interest and movement of the head is such that the camera can completely view the objects kept on the table.

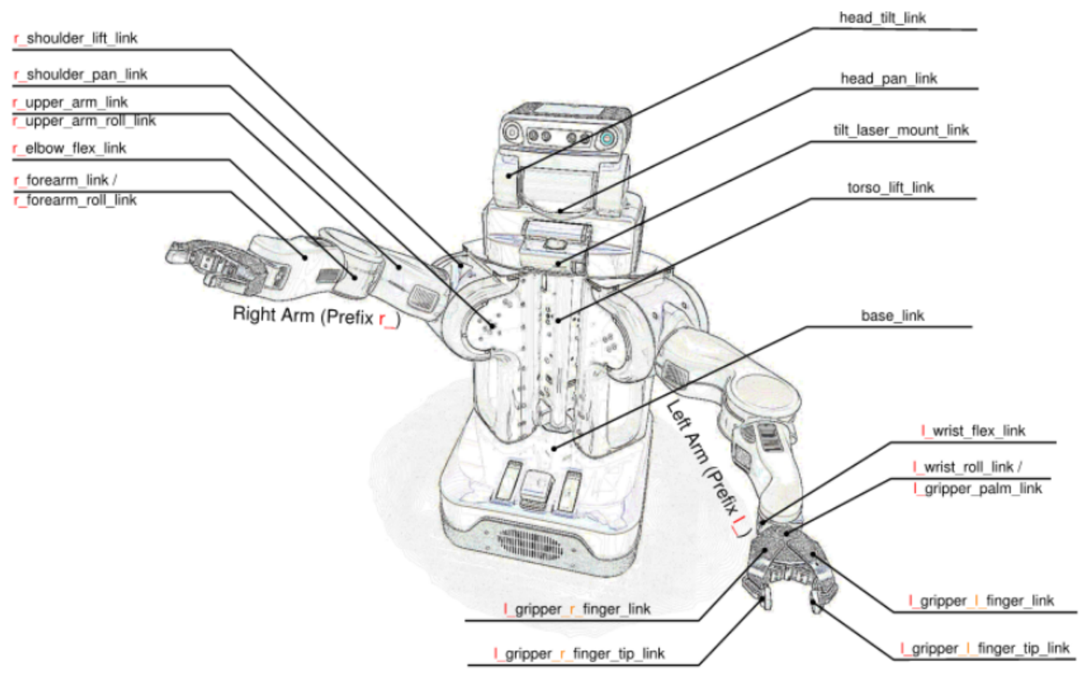


Figure 6.7: PR2's link with its naming conventions [101]

6.3.2 Trajectory Planning

Trajectory planning is performed by calculating the inverse kinematics for an End-effector position. In our case the end effector being robot's right gripper. This is done by calculating a trajectory for joints based on the gripper's positions. MATLAB's *robotics.InverseKinematics* class was used to calculate all the required joint positions, which can be further sent as a trajectory goal message via the action client. A sequence of end-effector poses over a period of time is called a trajectory. Generally, to define the robot parameters, its configuration and to visualize, *robotics.RigidBodyTree* class is used which is available in MATLAB.

To achieve the proper trajectory planning we perform the following steps

- Collect the robot's current state from the ROS network (master) and assign it to the *robotics.RigidBodyTree* object to work with the robot.
- Defining the goal pose i.e., 3-D coordinates of the end-effector.

- Use inverse kinematics to calculate joint positions from goal end-effector poses. Its goal is to calculate the joint angles for the PR2 arm that places the gripper in the desired pose.
- Send the trajectory of joint positions to the ROS action server to command the actual PR2 robot.

6.3.3 Motion Execution

The desired cartesian trajectory is sampled into various key points. This is generally done such that the robot's joint movements don't reach its maximum limit. The end-effector moves along the trajectory. The motion is executed as follows:

- Open the gripper.
- move the end-effector from the current pose to the first waypoints. Here the number of waypoints depends on how much we sampled given the start and end goal.
- After reaching all the goal points, the robot moves towards the center position of the object with its gripper open.
- The robot closes the gripper and grabs the object.
- Now the end-effector starts moving towards the goal location. Even this trajectory is sampled into any numbers of waypoints.
- Once the robot reaches its goal point it opens the gripper and releases the can.
- The robot goes back to its initial position to execute the next motion.

Algorithm 2 shows the complete algorithm from the robot's initial position to picking the object, and going back to the initial position by placing the object. The robot performs pick and place objects from the object's location to its goal location.

6.4 Integration

In this section, we investigate integrating brain-computer interface with robotics. It is well known that the systems with feedback perform much better when compared with systems that are open loop ones i.e., without any feedback. Generally, we show how the trajectory correction happens when a feedback signal is given using error related potential to the robot. The Algorithm 2 is used as a base and ErrP signal related is added to it.

While the robot interpolates the waypoints and finds the intermediate points, a function is called at that point which checks whether error related potential is generated or not. If generated then the goal point is changed and the interpolation is between the current position and the new goal point. With this, the robot is able to correct its trajectory and change its goal position to classify the object. There are 15% chances that the classification of error related potentials might go wrong but with the way we implement for every waypoint, the algorithm checks if error related potential is generated or not which increases the probability overtime to get the trajectory correction to happen perfectly.

6.5 Self-learning

With the concepts discussed above the robot is very well able to determine the location of the object to be placed but when the robot encounters the same object, it does the same mistake of wrongly classifying or randomly classifying the data. To overcome this, the robot needs to self learn the object that needs to placed in specific class or categories. The robot needs to learn on the go while it is working in real time and to achieve this performance, online machine learning algorithms are used. Once the robot picks an object, the machine learning algorithm predicts its label, if it is in the first iteration the labeling would be random. As the robot keeps picking various objects its labeling becomes accurate by updating its weights over time.

Algorithm 2 Pick and place objects

```
1: ROS initialization and defining rosaction objects
2: Initialize Robot arms and head to given position
3: for $i = 1$ to number of objects do
4:     Open the gripper;
5:     for $j = 1$ to number of waypoints $-1$ do
6:         Initial point = waypoint[j]
7:         Goal point = waypoints[j+1]
8:         Interpolate keypoints between initial point and goal point given N
9:         Calculate the joint position for each EE keypoints using IK
10:        Extract their joint positions
11:        define time points corresponding to each way point
12:        Estimate their joint velocity trajectory numerically
13:        Send the estimated pose values to the robot's right arm
14:        Execute the motion command
15:    end for
16:    Close the gripper to grab the object
17:    for $j = 1$ to number of waypoints $-1$ do
18:        Initial point = waypoint[j]
19:        Goal point = waypoints[j+1]
20:        Interpolate keypoints between initial point and goal point given N
21:        Calculate the joint position for each EE keypoints using IK
22:        Extract their joint positions
23:        define time points corresponding to each way point
24:        Estimate their joint velocity trajectory numerically
25:        Send the estimated pose values to the robot's right arm
26:        Execute the motion command
27:    end for
28:    Open the gripper;
29:    Goes back to its initial position
30: end for
```

As the data is available sequentially which means only one image is seen at one and needs to be processed, we use a stochastic gradient descent algorithm to train and learn how to predict the data accurately. As we are using Gazebo software for the robotic simulation and due to the unavailability of objects of our interest in the gazebo. To overcome this, images from the various dataset are collected and accumulated.

6.5.1 Dataset

There have been many studies and numerous surveys conducted by researchers working towards rehabilitation and orthotics to understand the needs of potential users [102]. Their main interest was towards understanding which of the daily tasks should the robotic assistive manipulator prioritize as per the user needs. In order to study this, researchers conducted multiple surveys asking people with various disabilities specifically locked-in syndrome patients and their clinical personals who assist them. Among the various tasks which generally any assistive robot can perform, object retrieval i.e., object fetch and retrieval was persistently found to be given high priority tasks. Other tasks such as personal hygiene, entertainment, eating, and drinking also has also been prioritized and have received significant mention [103]. Researchers from Georgia Tech have put efforts into understanding the needs of motor-impaired patients with amyotrophic lateral sclerosis (ALS). They provided patients caregivers with cameras and notepads to document when objects were dropped or were otherwise unreachable in daily life which confirmed the importance of robotic retrieval. The survey was conducted on 25 patients through in-person interviews. The study was concluded by listing 43 object classes for robotic retrieval [104]. The list is presented in Table 6.4.

The images mentioned in the table above were not available in the gazebo such that the processing could be done in real time. Hence, we came up with other solution of using objects from the standard dataset. There are many datasets available with respect to images, but we were interested in objects that specifically that has objects used in daily life activities. Hence, data was collected from Robot Learning Lab, Cornell University who worked with the problem of grasping novel objects [75]. Sample of the images in the dataset is shown in Figure 6.8.

Out of 43 objects from the listing of ALS patients for robotic retrieval, only 15 objects were found from the grasping novel objects database. Hence, we used only 15

Table 6.4: Prioritized list of object classes [104]

Rank	**Object class**	**Rating mean**	**Rating Stddev.**
1	TV Remote	6.64	0.57
2	Medicine Pill	6.36	1.55
3	Cordless Phone	6.28	1.31
4	Prescription Bottle	6.08	1.31
4	Fork	6.08	1.12
6	Glasses	6.00	1.53
7	Toothbrush	5.96	1.81
8	Spoon	5.92	1.19
9	Cell Phone	5.88	1.69
10	Tooth Paste	5.72	1.84
10	Book	5.72	1.46
12	Hand Towel	5.72	1.46
13	Mail	5.60	1.98
14	Cup/Mug	5.56	1.76
15	Soap	5.44	2.08
16	Disposable bottle	5.40	1.66
17	Shoe	5.36	1.98
17	Dish Bowl	5.36	1.66
19	Keys	5.28	2.28
20	Dish Plate	5.24	1.85
21	Pen/ Pencil	5.04	2.13
22	Table Knife	4.96	1.95
22	Credit Card	4.96	2.37
24	Medicine Box	4.88	1.88
24	Bill	4.88	2.26
26	Straw	4.80	2.22
26	Magazine	4.80	2.02
28	Plastic Container	4.72	2.16
29	Newspaper	4.60	2.16
29	Non-disposable bottle	4.60	2.00
31	Pants	4.53	2.47
31	Shirt	4.53	2.47
33	Wallet	4.48	2.33
34	Small Pillow	4.44	2.08
35	Socks	4.40	2.08
36	Hairbrush	4.36	2.46
37	Can	4.32	2.08
38	Coin	4.16	2.51
39	Walking Cane	3.72	2.47
40	Wrist Watch	3.52	2.35
41	Scissors	3.40	2.33
42	Purse / Handbag	2.84	2.29
43	Lighter	2.04	1.99

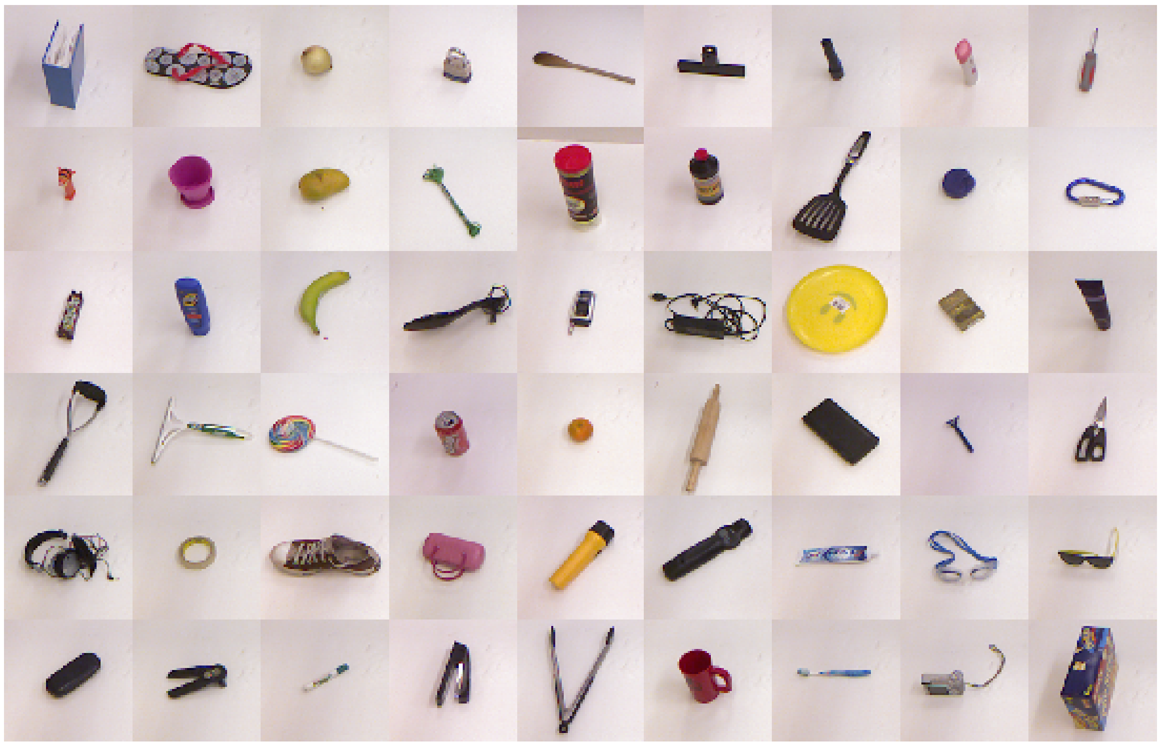


Figure 6.8: A small sample of the images included learning to grasp dataset [75]

objects to train the online self-learning algorithm. The objects used listed alphabetically are a book, can, cell phone, cup, dish bowl, glasses, hairbrush, keys, pen, scissors, shoe, spoon, toothbrush, toothpaste, and TV remote. Various combinations of these images were categorized into two categories. In practice, the way these objects are combined is completely based upon the need or interest of the disabled person.

6.5.2 Algorithm

Stochastic gradient descent (SGD) algorithm is implemented to train the network one image at a time. The standard gradient descent algorithm updates the parameters θ of the objective $J(\theta)$ as,

$$\theta = \theta - \alpha \nabla_\theta E[J(\theta)] \tag{6.1}$$

where the expectation is approximated by calculating the cost and gradient over the entire dataset whereas SGD computes this in a very simple way by updating and computing the gradient of the parameters using either fewer training examples or just one training example. With this kind of implementation, it reduces the variance in parameter update and can lead to more stable convergence. The new update is given by,

$$\theta = \theta - \alpha \nabla_\theta J\left(\theta; x^{(i)}, y^{(i)}\right) \tag{6.2}$$

with a pair $(x^{(i)}, y^{(i)}$ from the training set and α is the learning rate [105]. Learning rate α is typically smaller when compared to the standard gradient descent because of more variability in the update. Implementation of the stochastic gradient descent algorithm in steps is given below:

- Initialize the parameters w and η.
- Until approximate minimum of the objective function is achieved repeat the following steps:
  - Perform random shuffling the example data in the training set Randomly.

```
– For $i$=1,2,...,$n$, do:
    * $w := w - \eta \nabla Q_i(w)$
```

CHAPTER 7: RESULTS

This chapter summarizes the results obtained. The results are presented into three different experiment sections. The first section being to work related to accurately classifying the error related potentials. This section covers the visualization of collected data to the classification results. The next section describes the core implementation i.e., robot trajectory planning. This section deals with everything from how a robot successfully performs pick and place objects, plan the trajectory using inverse kinematics and motion execution traversing the trajectory. This section is further divided into two subparts i.e., with ErrP integration and without integration. The third section describes our novel implementation of online learning to the robotic retrieval of objects and learning to classify the objects. The category in which the images were presented or classified as two classes are well described and the results obtained with that categorization are shown.

7.1 ErrP Classification

7.1.1 Raw EEG

The goal of the first experiment is to investigate the use of error potentials to improve the performance of BCI. To achieve this we understand how to accurately classify the error related potentials given raw EEG. As discussed in the previous chapter, the dataset was collected from an online database. The raw data was loaded into EEGLAB using MATLAB and visualized as shown in Figure 7.1. For visualization purpose the DC offset was removed from the data because every electrode data was in different range and plotting to one specific scale was not possible. But every electrode is with different range of voltage potentials. The data was sampled at the rate of 512

Hz and the data in Figure 7.1 was of subject 01 session 01. The session number and subject number are as per the Table 6.1.

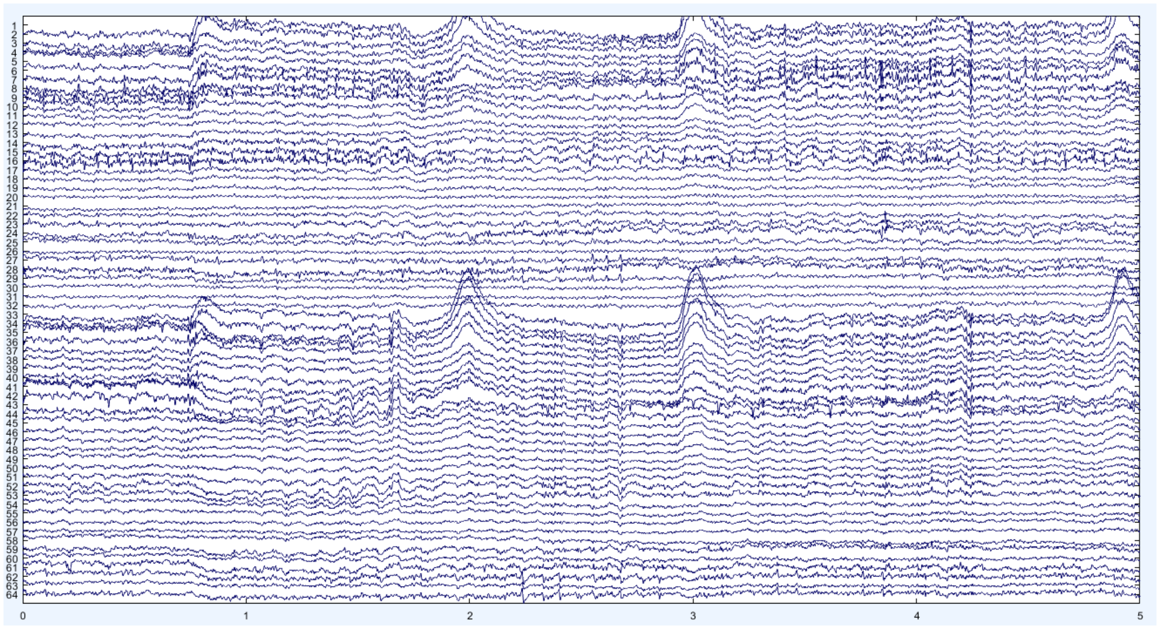


Figure 7.1: Raw EEG from the database [59]

7.1.2 EEG filtering

As the data was noisy, raw data from all six subjects from two different sessions were spatially filtered using the common average filter and spectrally filtered using FIR filter (bandpass) with a lower cut off frequency of 1Hz and higher cut off frequency as 10 Hz. The order of the filter is set to its default value 1690. Bode plot or Frequency response of FIR filter is as shown in Figure 7.2. From the frequency response curve, we see that the signal is attenuated at lower frequencies until the frequency reaches the lower cut-off point. The output continues at the maximum gain until it reaches the upper cut-off point attenuating any higher frequency signals.

The preprocessed data after filtering with the above filter description is shown in Figure 7.3. We see that most of the high-frequency components (various artifacts) present in the Raw EEG (as shown in Figure 7.1) are filtered and a smooth data curve is obtained with its DC component removed.

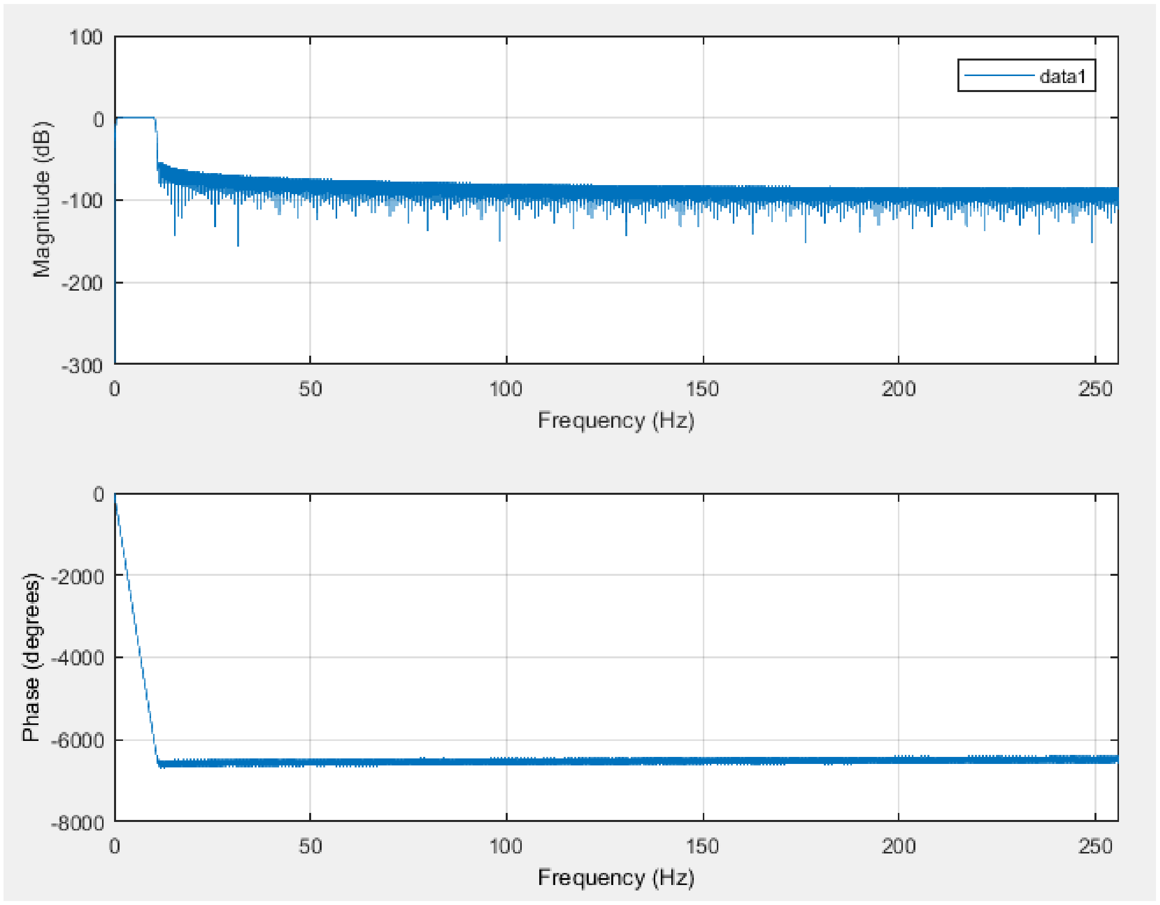


Figure 7.2: Bode plot or Frequency response of FIR filter

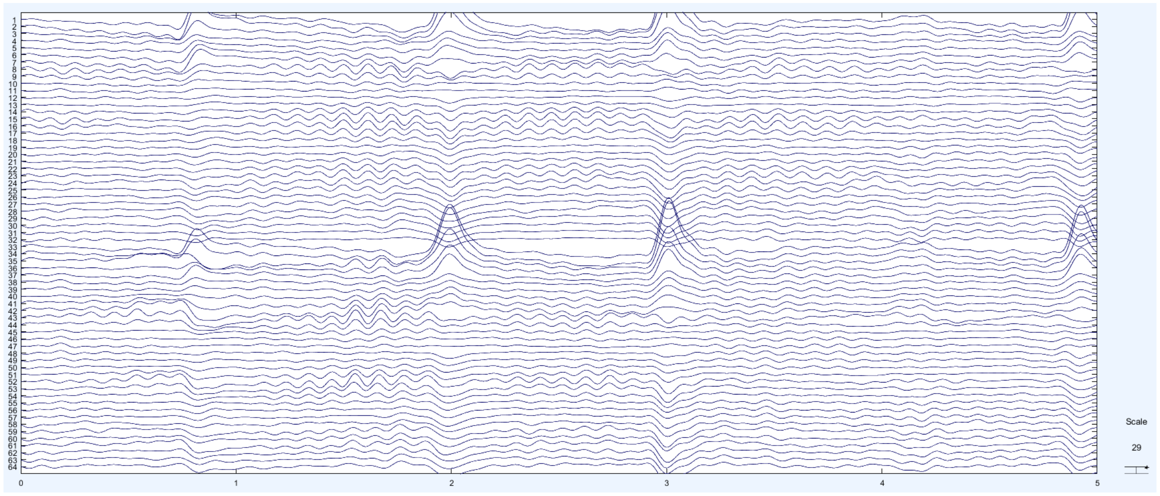


Figure 7.3: Filter EEG data

The event labels for error related potential are mentioned in the header file which is along with the database. Using the header file corresponding data was stacked into two different class. One class contains all the signals having generated when an error has occurred thus having error related potentials and other class containing normal EEG data. Based on this the data was labeled as a binary class either 1 or 0.

7.1.3 Electrode selection

Our data was collected using 64 electrodes. Not all parts of the brain responds the same to every event. Most of the studies suggest that error-related potentials are detected in a scalp area covered by electrode FCz and its neighbors but it is important to determine the best electrodes for classification of error related potentials [46]. To do so we plot topographical maps of the data which plots the scalp potentials at the given time. This can be used to see which part of the brain reacts to the error event at the time points as of a characteristic ErrP signal. We plot the scalp potentials for time 2 ms, 220 ms, 229 ms, 268 ms, 287 ms, 367 ms, 461 ms, 486 ms, and 742 ms as shown in Figure 7.4.

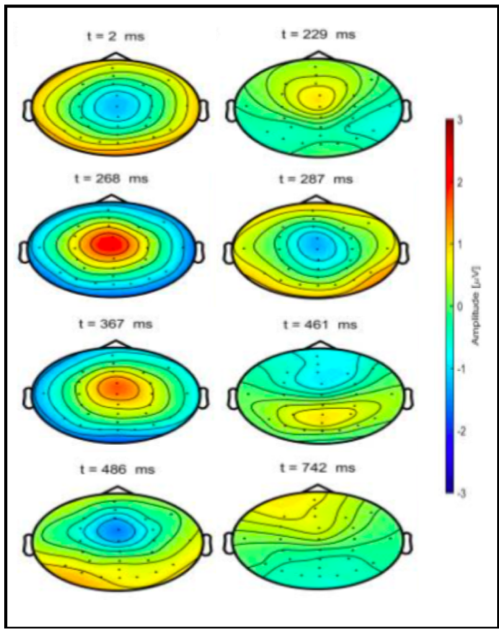


Figure 7.4: Topographical maps visualization of scalp activity [98]

We see that most of the variation is at the region of Anterior Cingulate Cortex (ACC) which have also proved by many other studies. Hence we select two electrodes from the front central area of the human brain. Electrodes 'FCz' and 'Cz' were selected as suggested by various studies.

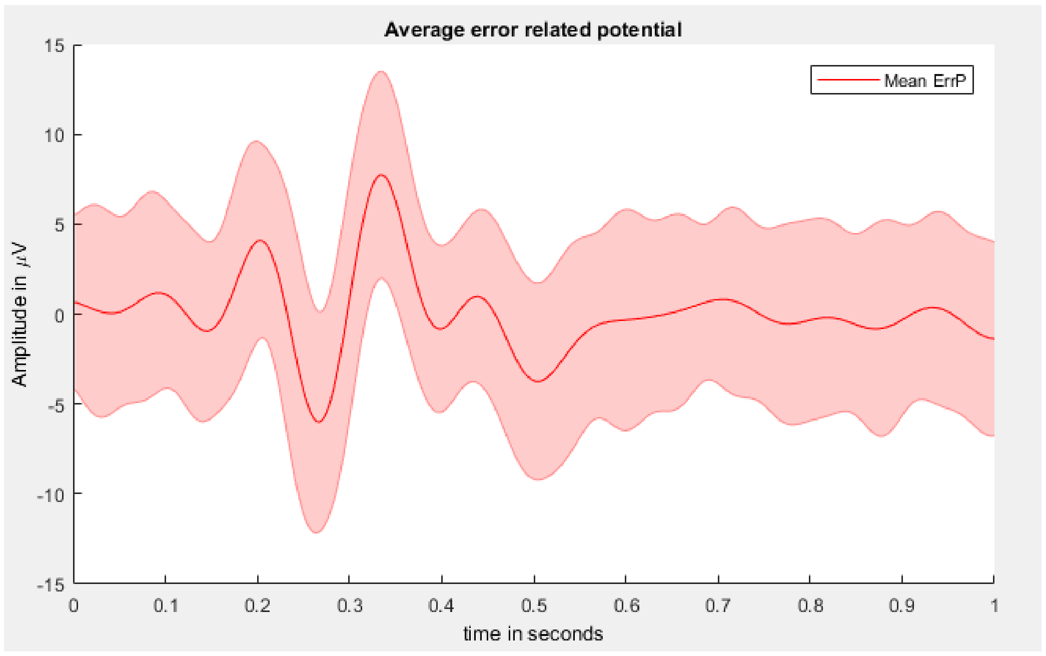
7.1.4 Grand Averages

Figure 7.5(a) and Figure 7.5(b) shows the averages of error for channel FCz and Cz respectively. The Ne and Pe component discussed it can be clearly seen in these figures. As discussed the ErrPs are characterized by an initial positive peak occurring at about 200ms after an event followed by a massive negative deflection at about 200-250ms and a second positive peak at about 320ms which can be clearly seen in the figure. The same results match with early ErrP taken by Falkenstein et al. [44]. Additionally, a positive peak appears after 400ms after the stimulus is present and a negative peak after 500ms, but all the main components of ErrP are present at the same time. The negative peak appears 250ms after the presentation of the stimulus. This means any human to process the visual feedback it takes so much amount of time delay.

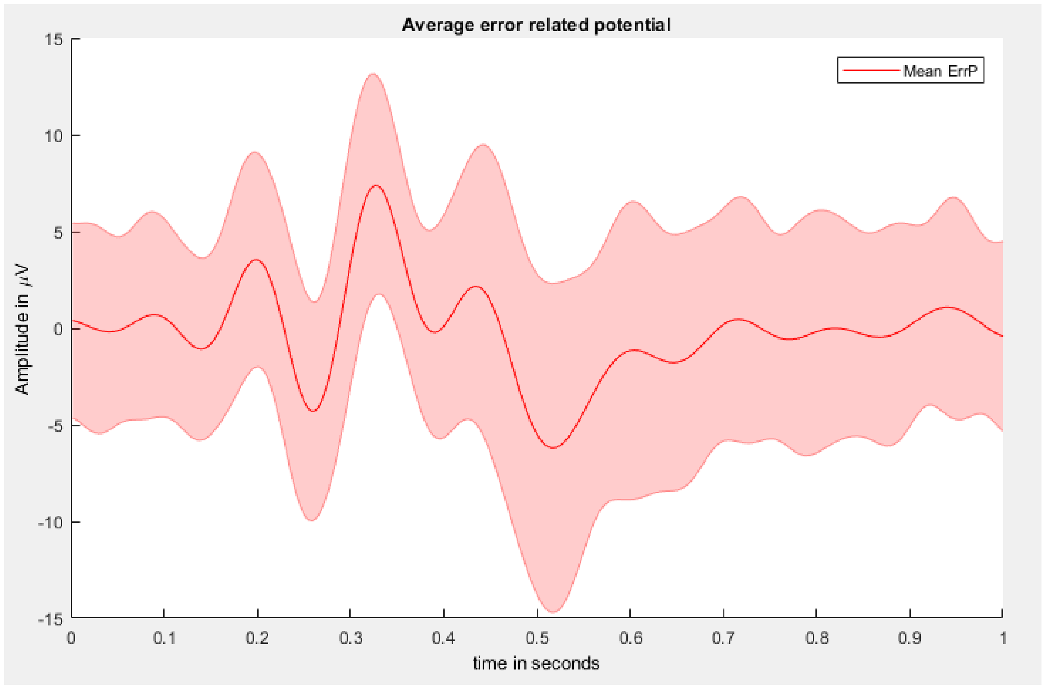
The dataset was collected in two sessions. Figure 7.5 and Figure 7.6 shows the averaged error-related potentials recorded in two different sessions separated by number of days shown in Table 6.1 respectively. We see that the characteristic pattern looks similar by showing the Ne and Pe component clearly as discussed above. This behavior of error related potential is very advantageous to develop a general classifier. Researchers have proved that with different types of stimulus still, the error related potential remains the same having its characteristic pattern.

7.1.5 Classification

The input data for the network is represented as a 2D array with the number of time steps as the width of the array and number of electrodes as their height. Hence one training example was of size 2x512 and a total of 1322 samples of ErrP data and rest were normal data. The data which was divided into two classes were concatenated.



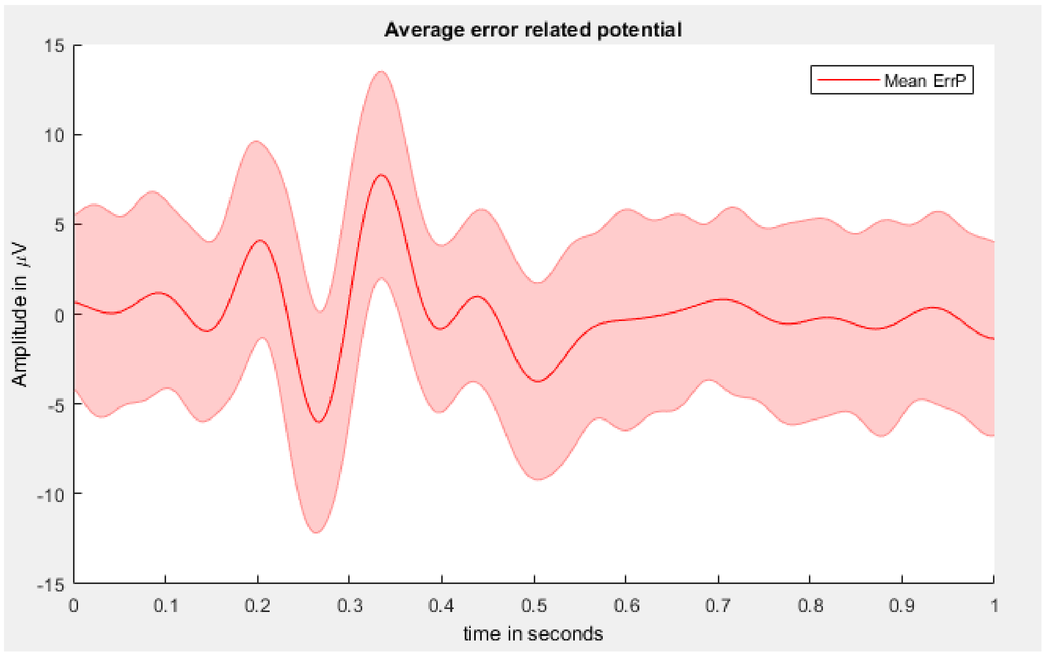
(a)



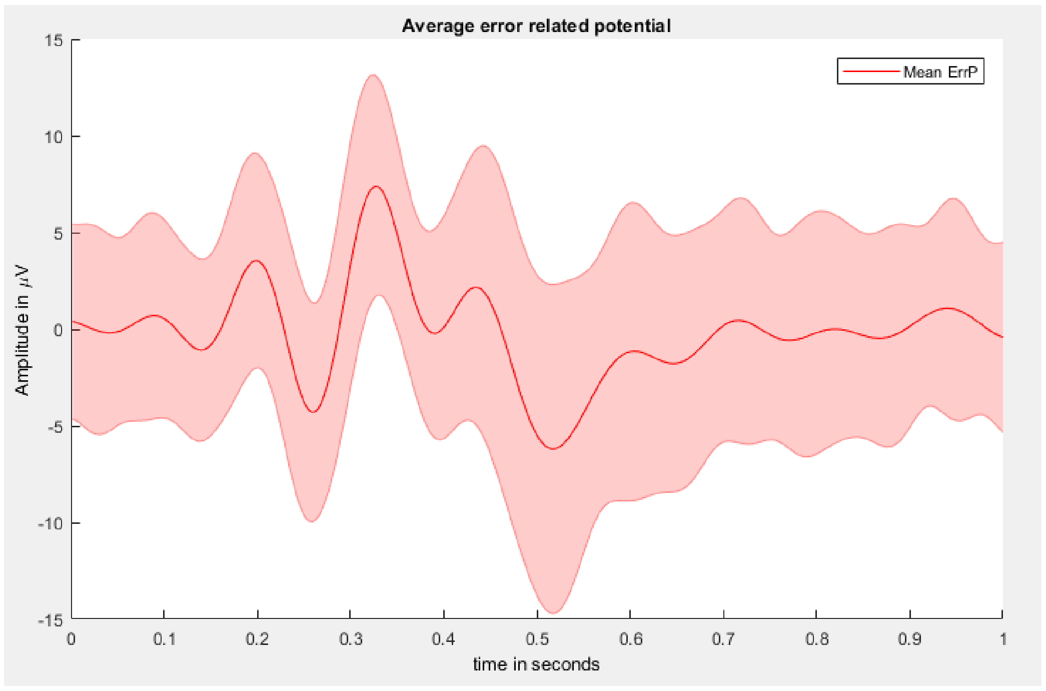
(b)

Figure 7.5: Averaged ErrP for Subject 1 and Session 1: (a) Electrode:'FCz'; (b) Electrode:'Cz'

The data were divided into three groups: (1) 70% for training, (2) 5% for validation, and (3) 25% for testing. The data was shuffled before training so that the training data



(a)



(b)

Figure 7.6: Averaged ErrP for Subject 1 and Session 2: (a) Electrode:'FCz'; (b) Electrode:'Cz'

does not obtain the entire mini-batches of highly correlated examples. For classifier, we used a deep convolutional neural network of 5 layer architecture as shown in

Table 6.2 with the hyperparameters setting as shown in Table 6.3. In this study two different architectures were studies, first was base architecture (CNN1) and second being modified (CNN2). The difference between CNN1 and CNN2 are that in CNN2, regularization methods such as batch normalization and drop out layers are used whereas CNN1 was simple architecture with standard layers of a convolutional neural network. The network was trained for all the subjects of session 1 and tested over individual subject data over session 2 and vice-versa. The classification results are shown in Table 7.1 with an overall classification rate of 82.4% for CNN1 and 86.1% for both the test scenarios. We observed that with additional layers the network (CNN2) was able to get a train with less number of epochs and perform much better with an accuracy improvement of about 4%.

Table 7.1: Classification Results of Individual subjects

Subjects	**Train Sess1 & Test Sess2**		**Train Sess2 & Test Sess1**	
	CNN1	*CNN2*	*CNN1*	*CNN2*
1	86.62%	87.62%	77.42%	85.35%
2	80.52%	80.13%	79.2%	82.77%
3	78.84%	79.40%	81.32%	80.01%
4	71.81%	78.46%	78.64%	83.28%
5	73.34%	70.92%	77.63%	78.59%
6	72.13%	78.76%	70.83%	76.46%

We also tested the network performance for cross subjects of the same sessions. Subjects 1,2,3 were taken for training and subjects 4,5,6 for testing of session 1 and vice-versa. We see that the network performs better irrespective of subjects from the same or cross sessions. The comparison table are shown in Table 7.2 and Table 7.3.

Table 7.2: Cross subjects classification of both architectures for Session 1

	Subjects 1 2 3	Subjects 4 5 6
CNN1	78.88%	75.09%
CNN2	81.62%	77.48%

Table 7.3: Cross Subjects classification of both architectures for Session 2

	Subjects 1 2 3	Subjects 4 5 6
CNN1	85.22%	79.39%
CNN2	82.34%	78.32%

7.2 Robot Motion Planning

This section deals with the results of robotic arm motion. It deals with robot setting up its initial position for the task, planning and executing the object pick, and planning the return trajectory and executing the motion to its corresponding class (in our case the object bowl). This section is subdivided into three parts: (1) Robot initialization, (2) Robot trajectory planning and execution, and (3) Robot integration with brain signals

7.2.1 Robot Initialization

The robot itself has an initial position while loading the robot to gazebo as shown in Figure 7.7. We initialize the robot position as per our requirements i.e., our robot's arms should not coincide with the workspace and they are spread wide outside. The initialization is performed in the following order: (1) right arm, (2) left arm and (3) head. Each arm has 7 degrees of freedom hence we specify the position of every joint in the robotic arm and execute the motion. The right arm and left arm initialization in four steps are shown in Figure 7.8 and Figure 7.9 respectively. PR2 head has two joints and they are initialized such that the table is completely in the robot's field of vision. The robot's head positioning in four steps is shown in Figure 7.10 and its corresponding vision from the robot's head camera is shown in Figure 7.12. Once the initialization is performed the robot's current state is shown in Figure 7.11. Figure 7.13 shows the visualization of the robot with the given joint position vector before initialization, right arm movement, left arm movement and final initialized position.

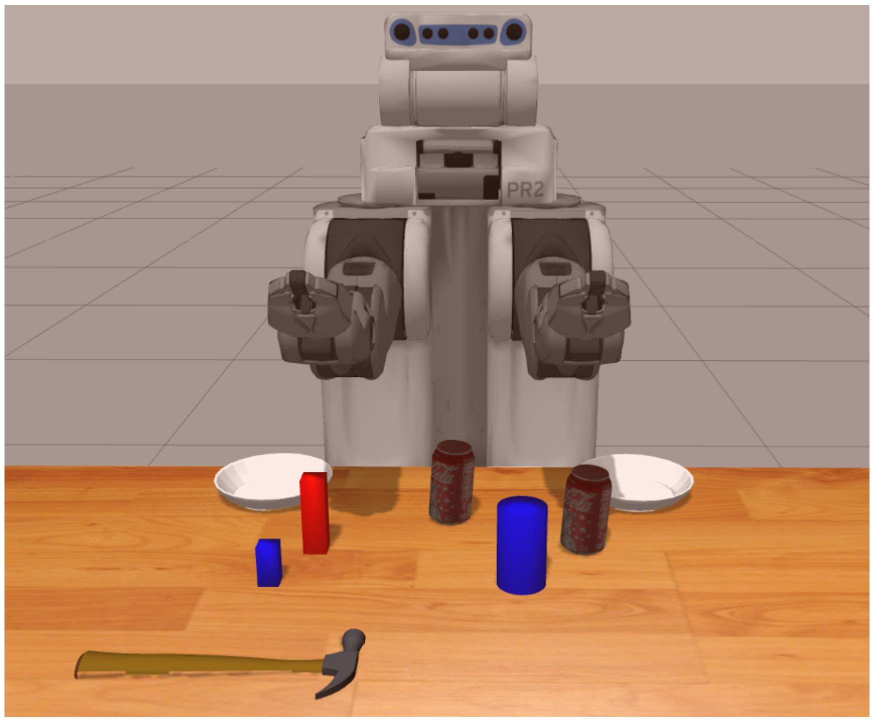


Figure 7.7: Robot before setting it to its initial position

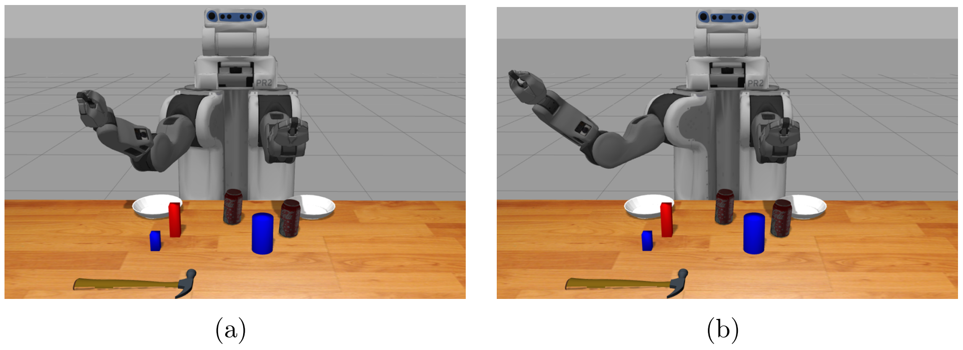


Figure 7.8: Right arm positioning of robot in four steps

(a) (b) (c) (d)

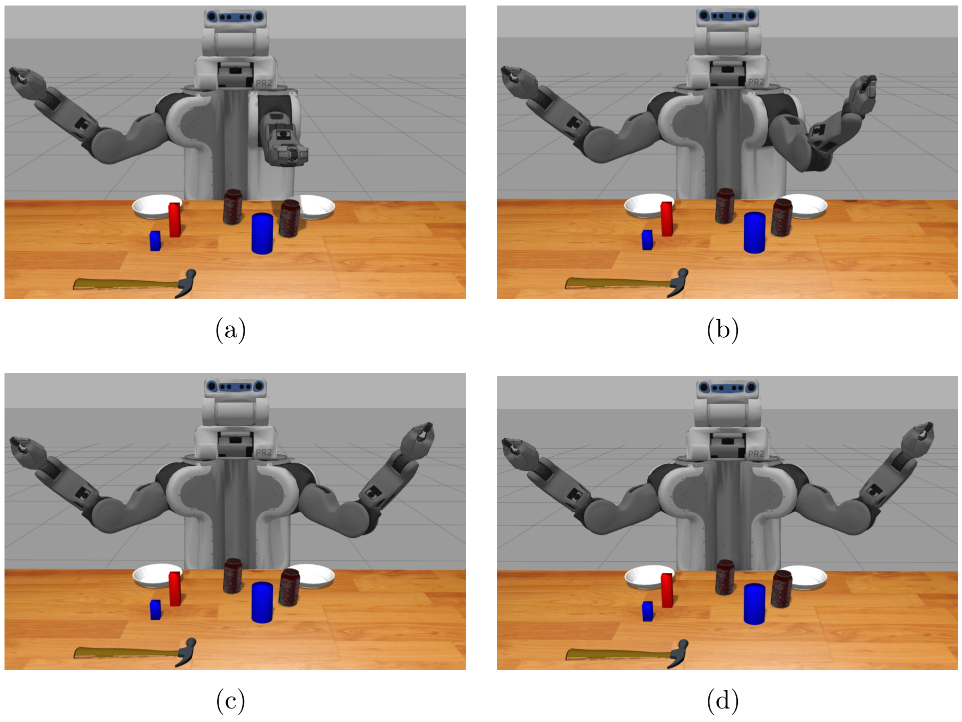


Figure 7.9: Left arm positioning of robot in four steps

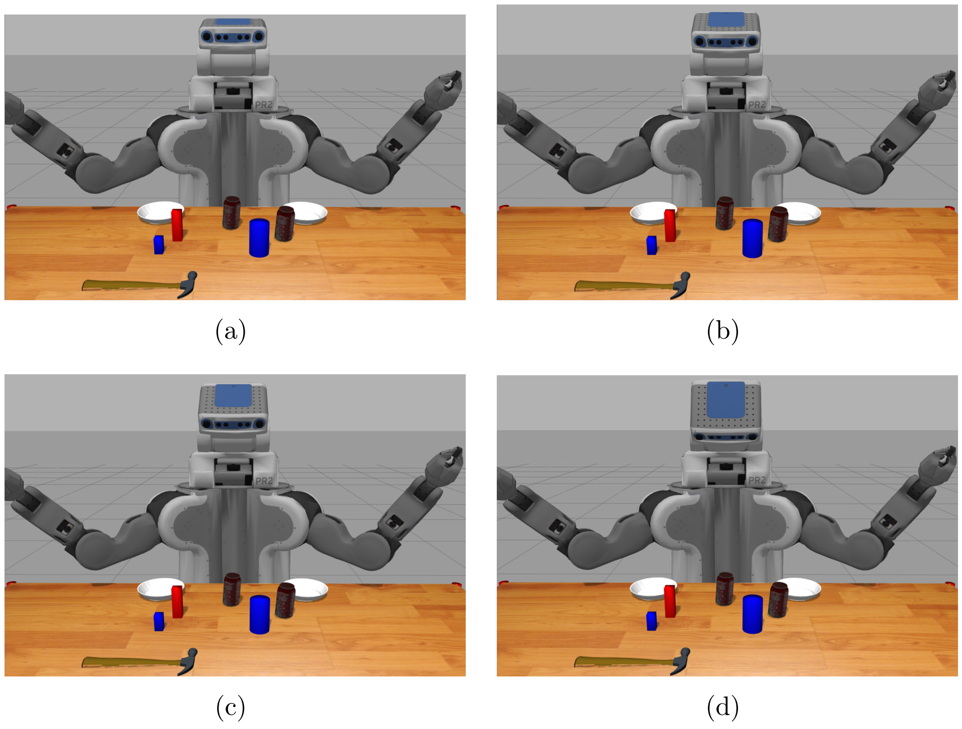


Figure 7.10: Head positioning of robot in four steps

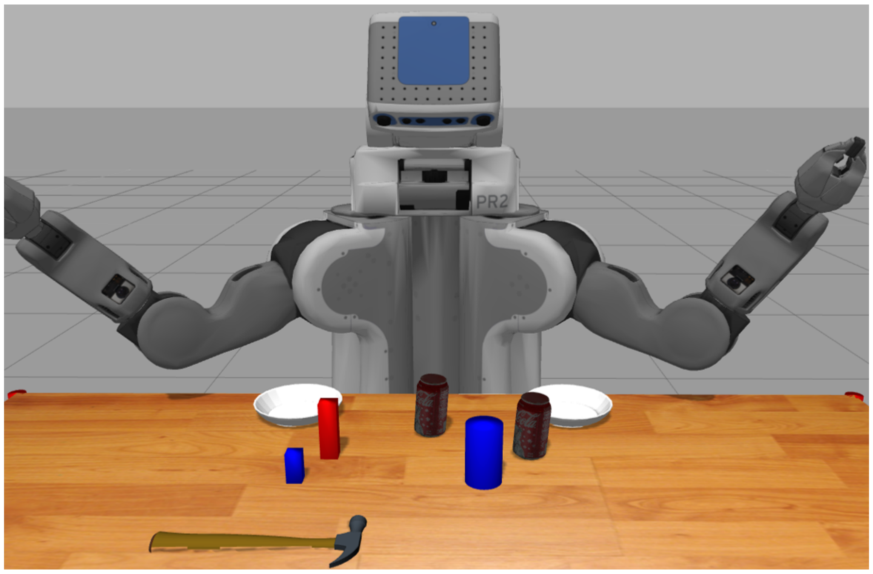


Figure 7.11: Robot position initialized to perform task

7.2.2 Robot Trajectory Planning

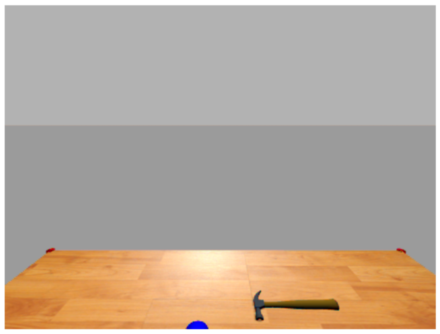
Robot trajectory planning is required to perform any motion. Our task is to pick and place from the object's initial position to its corresponding class. Once the robot is initialized with its position, to perform the object pick and place task we follow the following steps: (1) Open the gripper, (2) Pick the object, (3) Grasp the object, (4) Place the object in their corresponding class. All the steps are explained in the following sections.

7.2.2.1 Open the Gripper

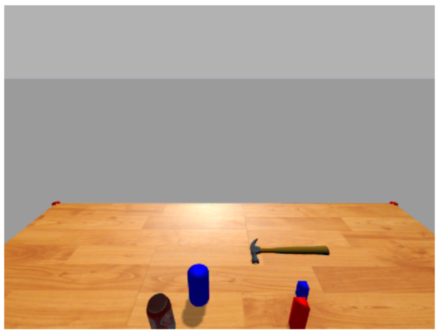
The motion execution is performed with the robot's gripper opened. To open the robot gripper, gripper position and the effort required to open is to be mentioned before calling gripper command topic. Here we send position as 0.1 and effort as -1. The robot opens its right arm gripper as shown in Figure 7.14.

7.2.2.2 Pick the Object

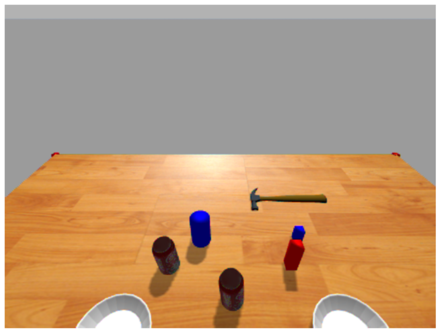
In order to pick the object, the object's initial position was predefined to the robot as per the global coordinate system. The object's initial position is required because



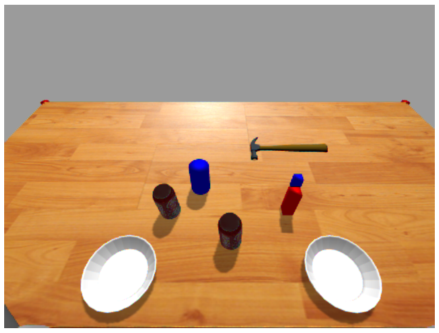
(a)



(b)



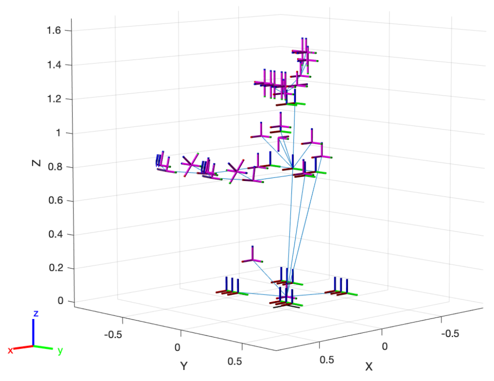
(c)



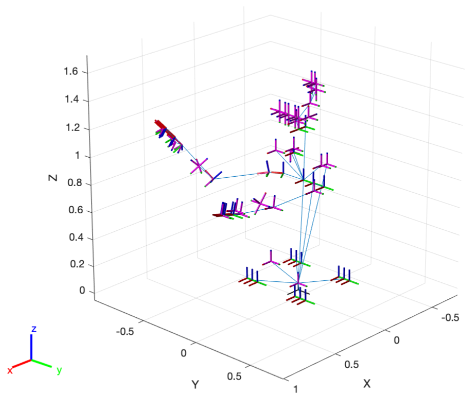
(d)

Figure 7.12: Visual positioning of robot in four steps viewed from its head camera

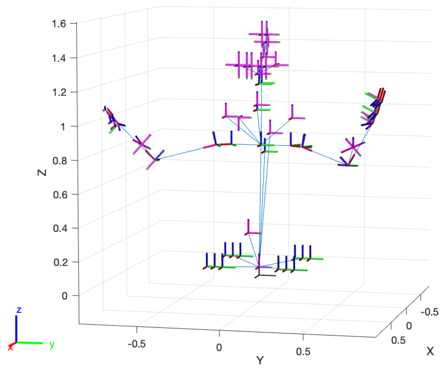
the robot can plan its trajectory motion from its current state to the final state (object's position). Instead of planning the trajectory from initial to the final position, we break the trajectory into five waypoints before it could grasp to make sure the robot's joints don't reach its extreme limit. Every waypoint is linearly interpolated into the 3D cartesian plane. Total of ten points is interpolated between one-way point to others. The trajectory taken from the robot's initial position to the first-way point is shown in Figure 7.15(a). Similarly, the planning waypoints until the robot reach the grasp position is shown in the remaining images of Figure 7.15. To execute the move, the velocities at which each joint has to move is calculated using the Jacobian matrix by setting the maximum time from one-way point to another to 3 seconds and



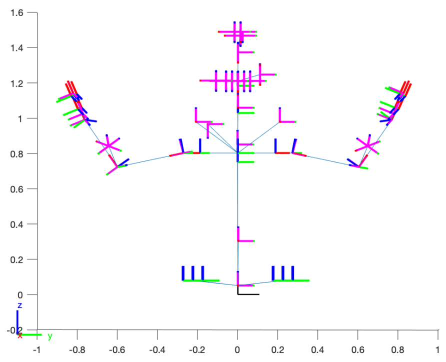
(a)



(b)



(c)



(d)

Figure 7.13: Joints of robot while performing motion (a) Current position before initialization; (b) Right arm movement; (c) Left arm movement; (d) Initialized position (front view)

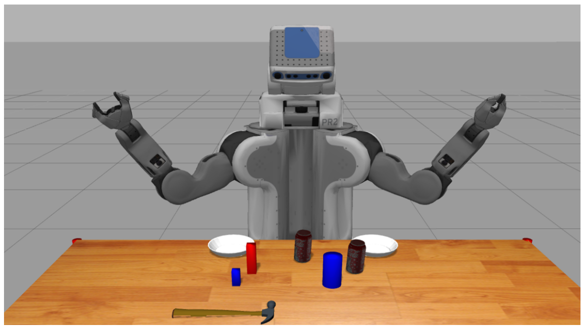


Figure 7.14: Robot Opening its gripper

performing linear in time. Using interpolated position data, the distance between the sub waypoint is found and its velocity is determined. The robot's motion execution is shown in Figure 7.16 and its view from the robot's vision camera is shown in Figure 7.17.

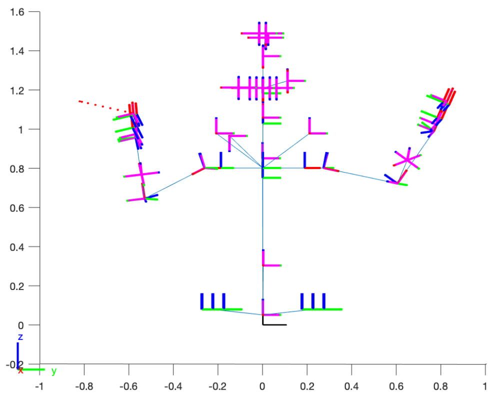
7.2.2.3 Grasp the Object

The robot closes its gripper in order to grasp the object. The robot's gripper is given a command with its position and maximum effort which is given to be 0.0 and 1000 respectively in order to successfully grasp the object. Figure 7.18 shows the robot grasping object and its corresponding camera view is shown in Figure 7.19.

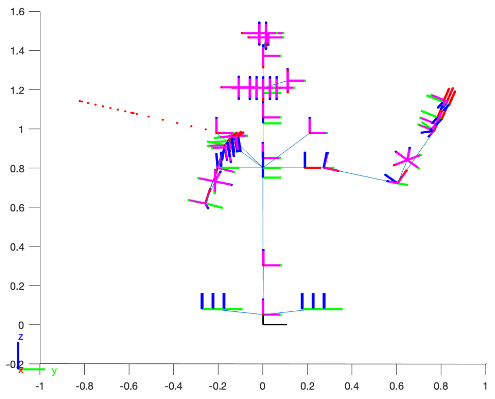
7.2.2.4 Place the Object

Once the robot grasps the object there are two different possibilities where the object could be placed. The possibilities are two white bowls placed on the table on either side of the robot. As we have two categories in which the data is sorted, hence the trajectory planning is performed either towards bowl kept on robot's right arm side or the bowl which is on the robot's left-hand side. Both implementations are shown separately. For each of the implementation, there are two waypoints so that the robot's joints don't reach its maximum limit. The trajectory is linearly interpolated between two waypoints to 10 intermediate waypoints called as sub waypoints. The trajectory plan for various waypoints from picking the object to placing the object in the target placed on the right side of the robot is shown in Figure 7.20. Similarly, the trajectory plan for various waypoints from picking the object to placing the object in the target placed on the left side of the robot is shown in Figure 7.23.

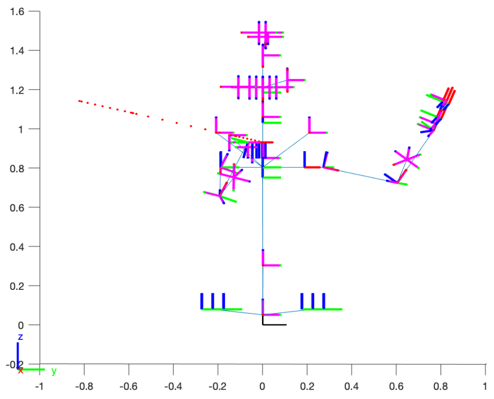
To execute the motion plan, the robot has to determine the velocity at which the joints need to be moved. We set a constraint that any motion between two waypoints or from an initial position to the waypoint to be performed in 3 seconds. We know the position sub waypoints based on the interpolation results. Using those results we find the Jacobian matrix and perform the motion execution with the calculated values. The motion execution from picking the object to placing towards the right



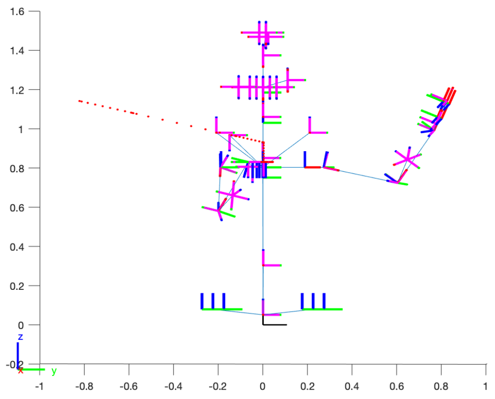
(a)



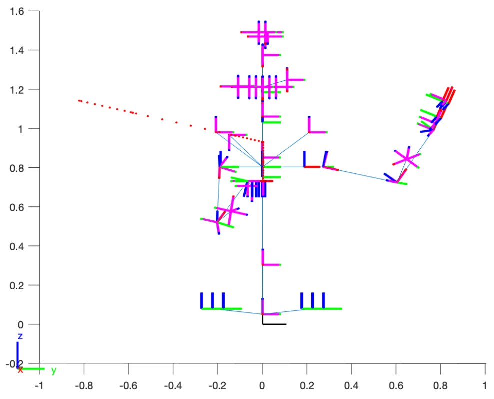
(b)



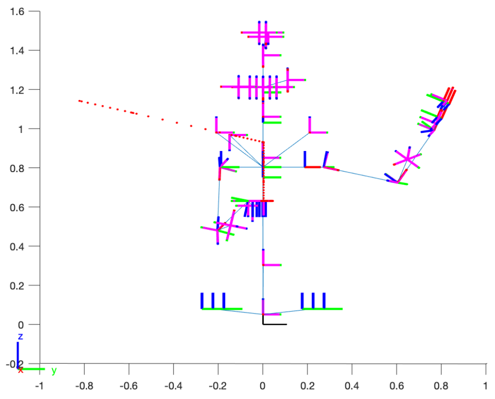
(c)



(d)



(e)



(f)

Figure 7.15: Trajectory Planning : Robot Reaching to the object and picking it along its way points

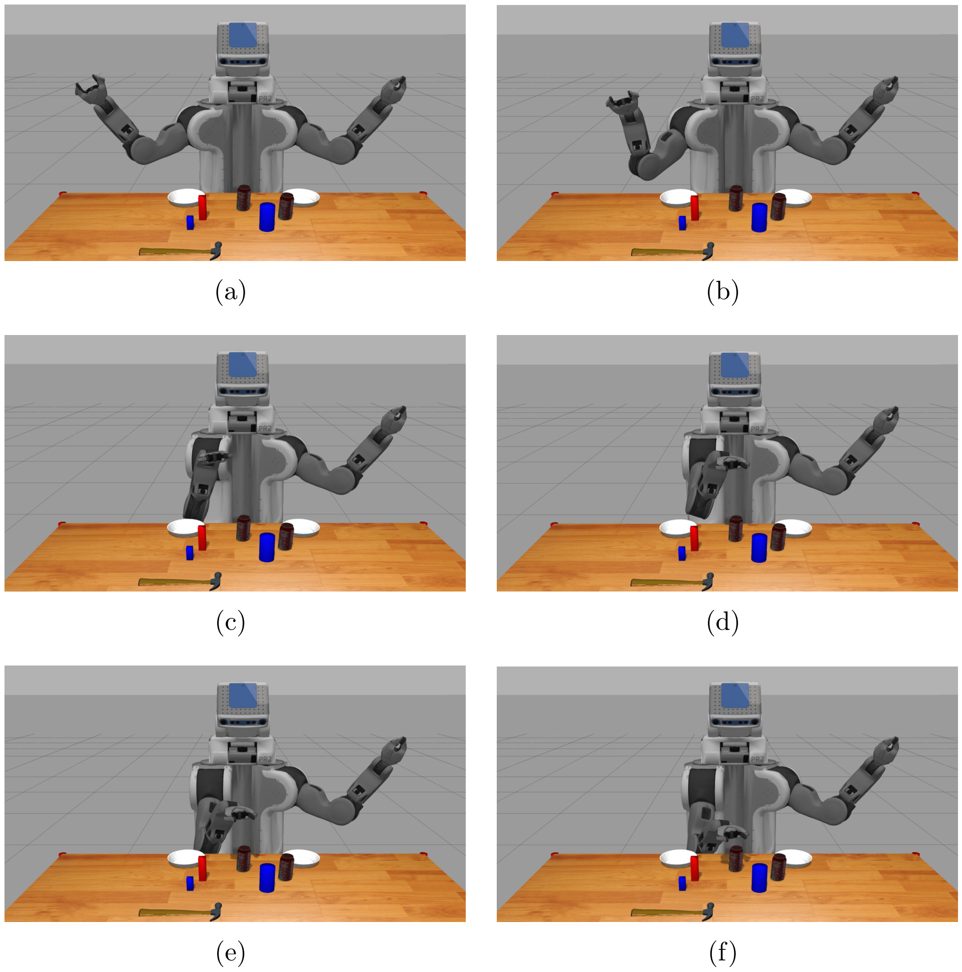


Figure 7.16: Motion Execution: Robot Reaching to the object and picking it along its way points

and left the side of the robot are shown in Figure 7.21 and Figure 7.24 respectively. The motion execution's corresponding robotic vision view is shown in Figure 7.17 and Figure 7.22 for right and left side movements respectively.

7.3 Integration

In this section, we integrate the two results obtained previously. The concept of trajectory correction with feedback from the human brain is shown in this section. The classification result obtained from the CNN classifier is binary i.e, either to change the trajectory or not. Our algorithm at every waypoint checks for the occurrence of error related potential. If there exists any ErrP signal then the object's goal position

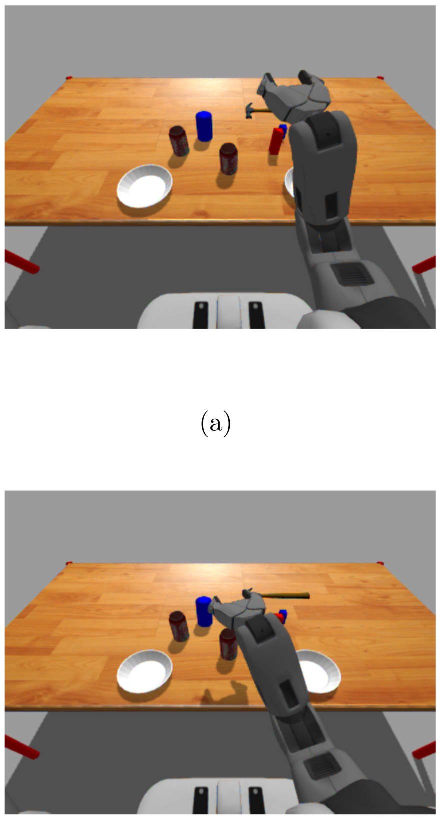

(a) (b) (c) (d)

Figure 7.17: Robotic camera view: Robot Reaching to the object and picking it along its way points

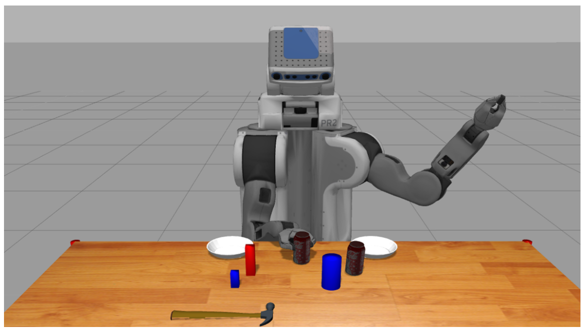


Figure 7.18: Grasp the object

changes which in turn changes the planning and execution of the object motion. This occurs only when the robot has picked the object and tries to place in the category in which it believes it should be placed and is wrong according to the user. If there isn't

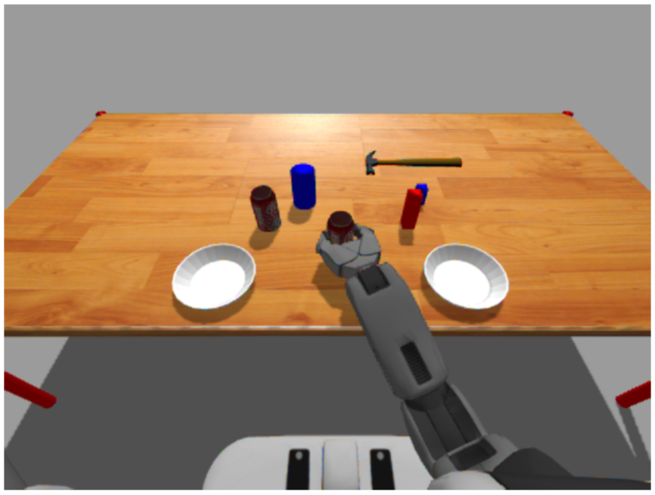


Figure 7.19: Grasp the object viewing from robot's vision camera

any ErrP signal then the robot performs the normal motion. The motion planning and execution for normal operation are the same as shown in the previous section but the trajectory correction when ErrP is received is shown in the next section.

7.3.1 Trajectory Correction

We see our classification results shows that it is possible to accurately classify the ErrP signal with a probability of 85%. There are still 15% chances of the signal getting misclassified. The concept of multiple waypoints is introduced such that if the verification of ErrP signal is performed at each waypoint then the probability of misclassification is reduced. The trajectory correction is shown in Figure 7.26, its motion execution is shown in Figure 7.27, and its camera view is shown in Figure 7.28.

7.4 Self Learning

Self-learning is the novel work in this research. By the end of the previous section, the robot knows which object needs to place in which class. Basically, it is able to perform the pick and place object task using human feedback about the class of

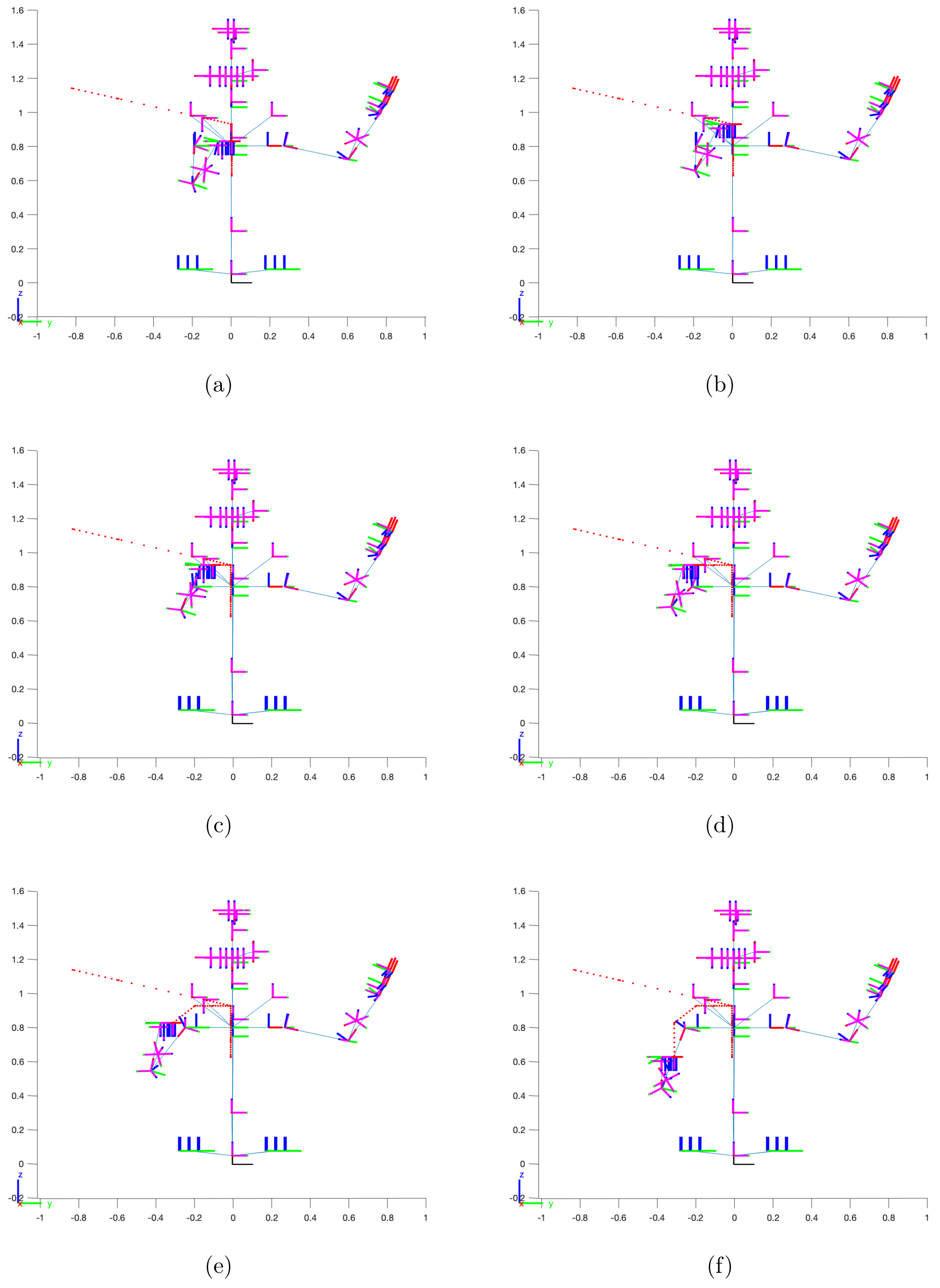


Figure 7.20: Trajectory Planning: Robot planning its movement from grasp position to the target placed right side of robot

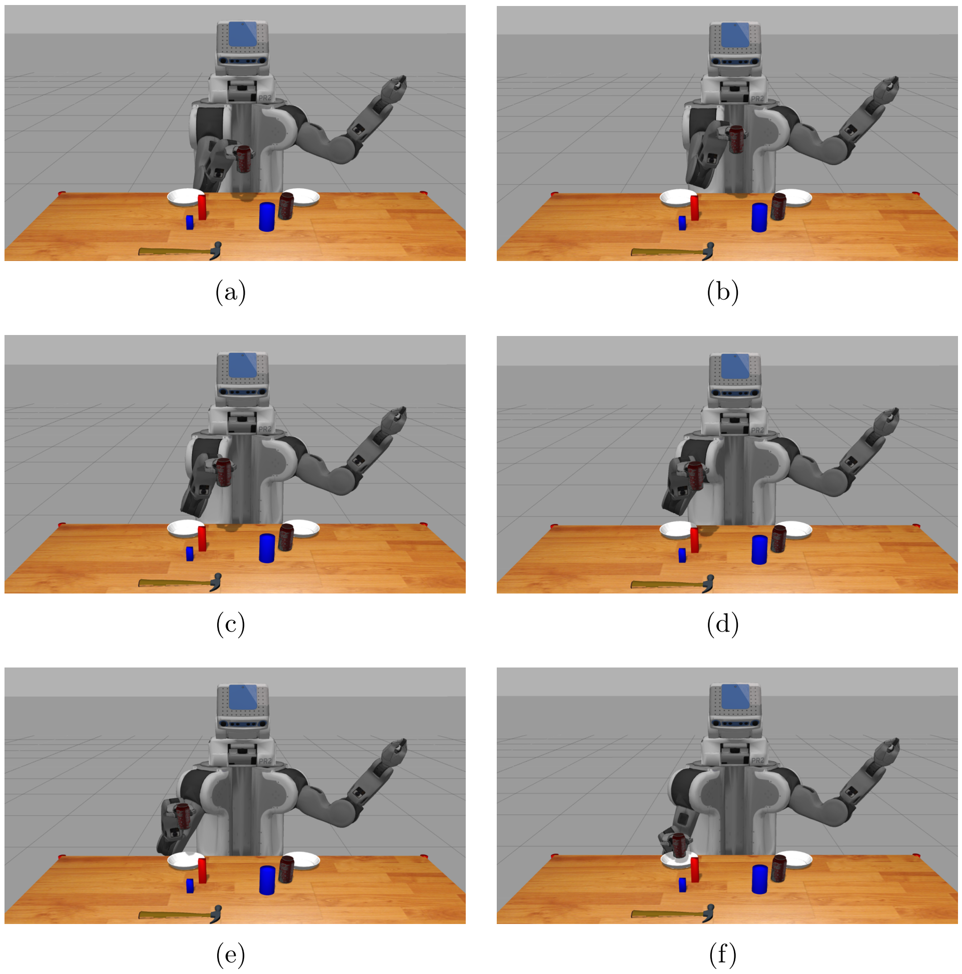
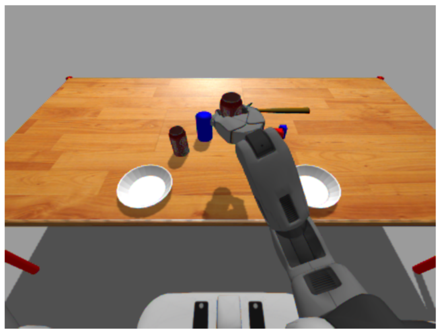


Figure 7.21: Motion execution: Robot planning its movement from grasp position to the target placed right side of robot

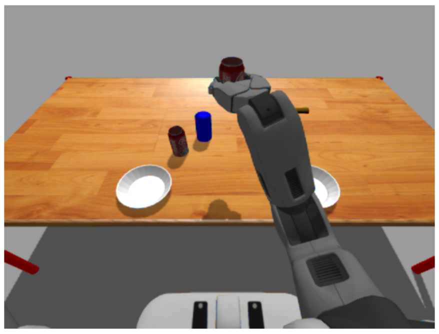
the object. Now the robot's task is to self learn the object's class based on human preference. The beauty of human thinking is, every individual thinks or sorts objects in his/her own manner. Hence present two different cases of how humans want the robot to classify objects and is discussed in the following section.

7.4.1 Dataset sorting

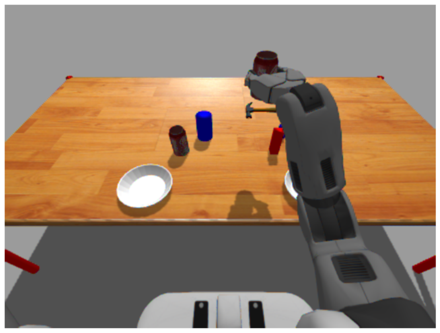
In order to perform the self-learning algorithm over the objects, the more important things are the object's database. Object database was collected from the online available dataset as discussed in the previous chapter. The following objects were related to our work and were picked as shown in Figure 7.29.



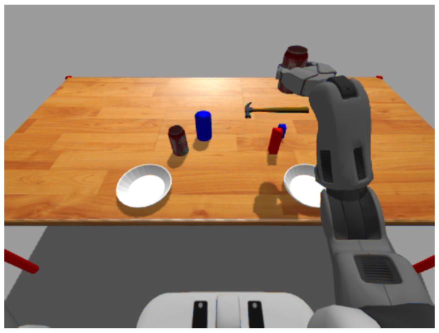
(a)



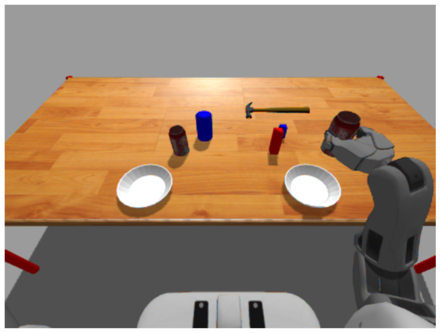
(b)



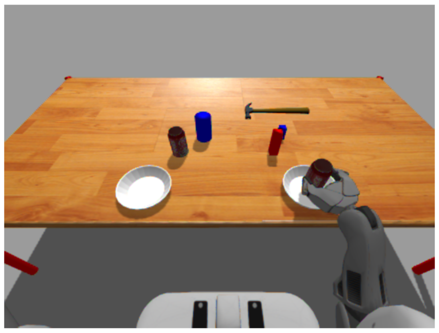
(c)



(d)



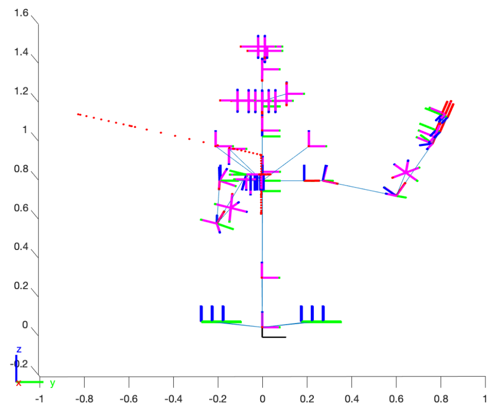
(e)



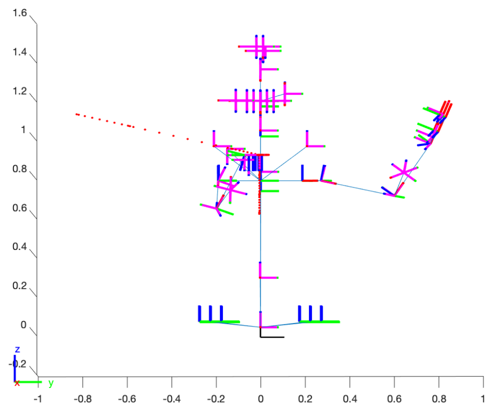
(f)

Figure 7.22: Robot's camera view: Robot planning its movement from grasp position to the target placed right side of robot

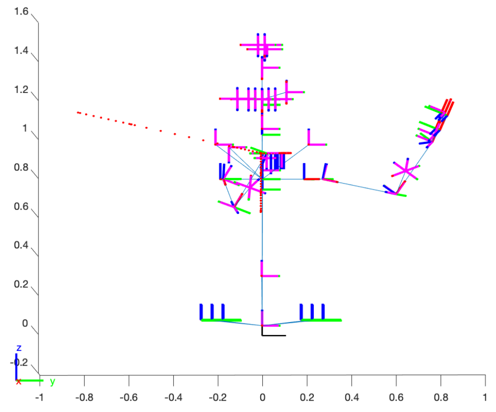
The images are grouped into two cases as shown in Table 7.4. In case 1 objects such as a book, can, cell phone, TV remote, keys are grouped as study table related



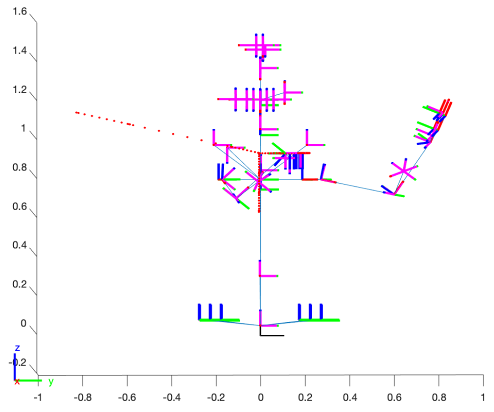
(a)



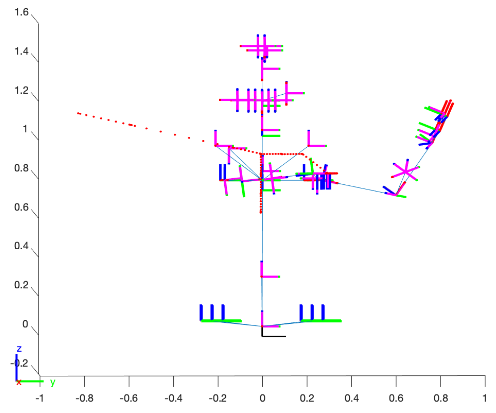
(b)



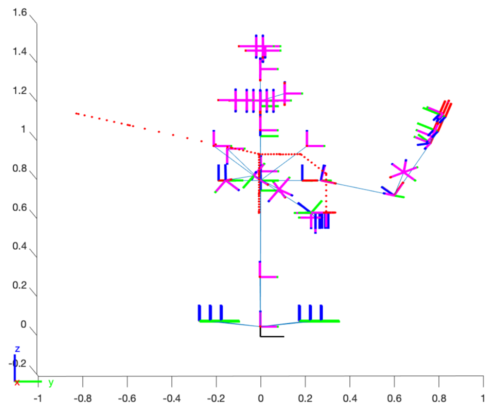
(c)



(d)



(e)



(f)

Figure 7.23: Trajectory Planning: Robot planning its movement from grasp position to the target placed left side of robot

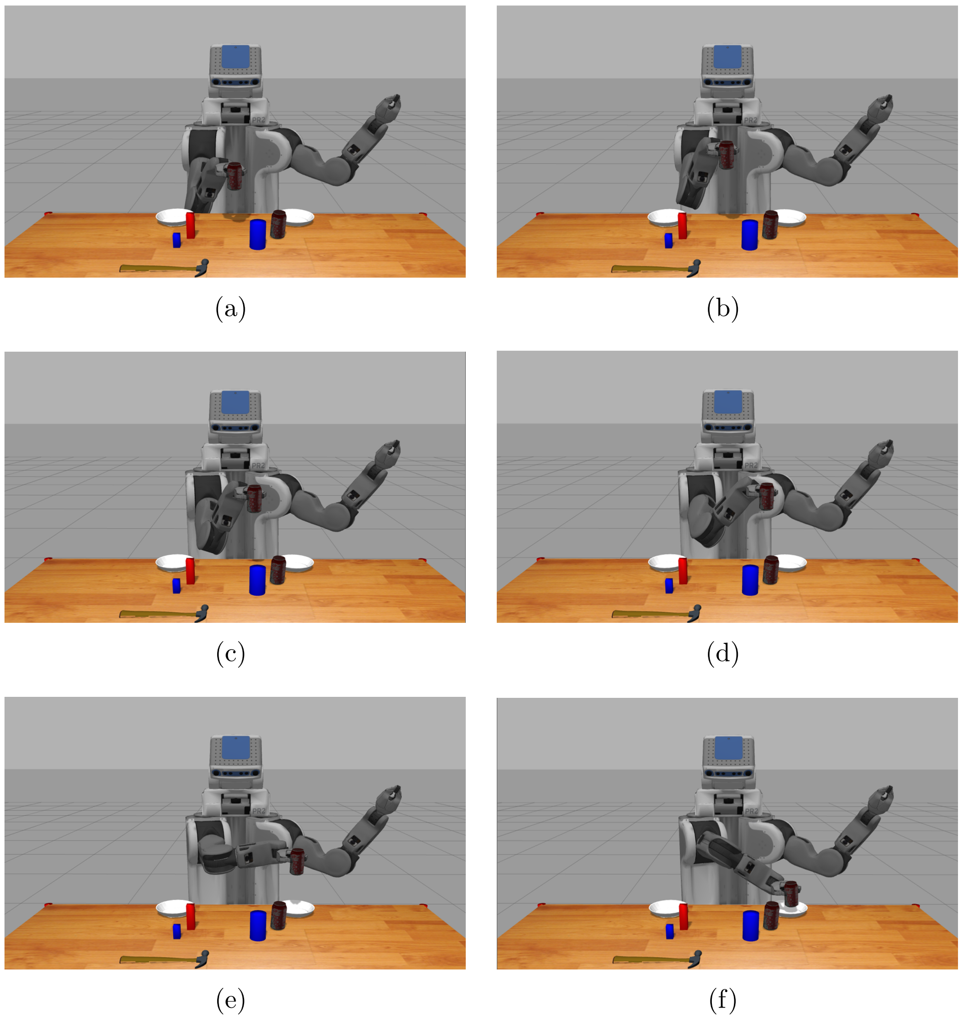
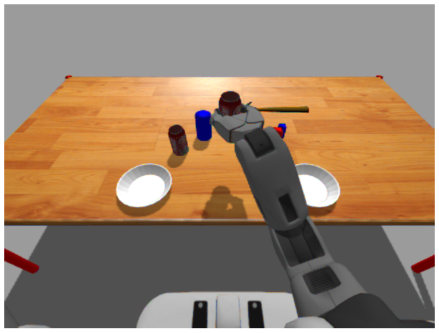


Figure 7.24: Motion execution: Robot planning its movement from grasp position to the target placed left side of robot

objects and objects such as toothpaste, toothbrush, scissors and hairbrush as bathroom related objects. In case 2 objects such as can, cup, dish bowl, spoon, TV remote are considered as objects kept in the kitchen and objects such as shoe and glasses to be related to dressing. We present two different results showing the robot can learn any pattern which human expects it to learn.

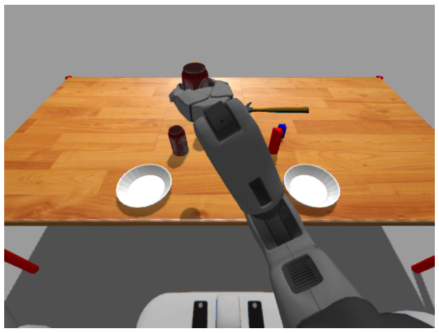
7.4.2 Stochastic gradient descent

Stochastic gradient descent is applied using sklearn's SGD classifier with its default values. The model is trained for case 1 where the objects are sorted based on whether

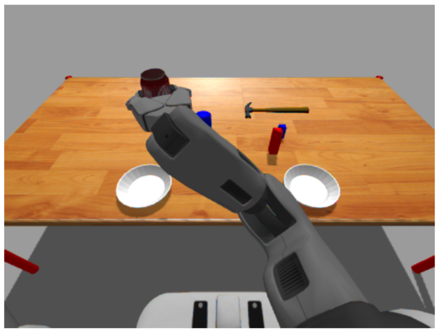


(a)

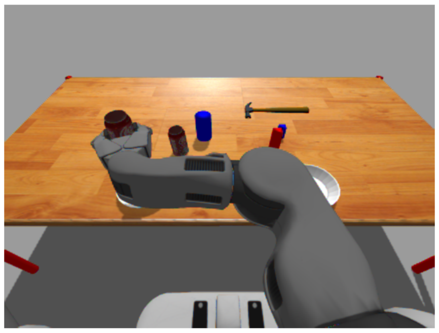
(b)



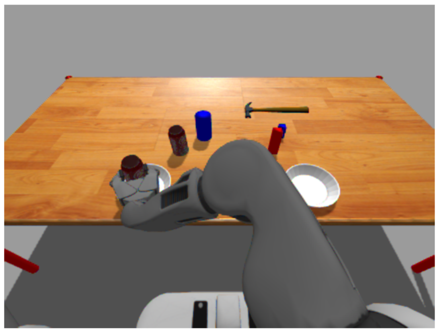
(c)



(d)



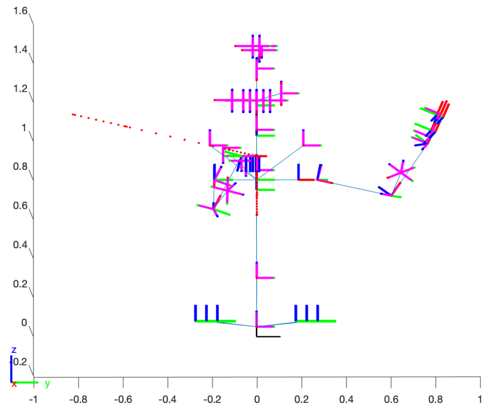
(e)



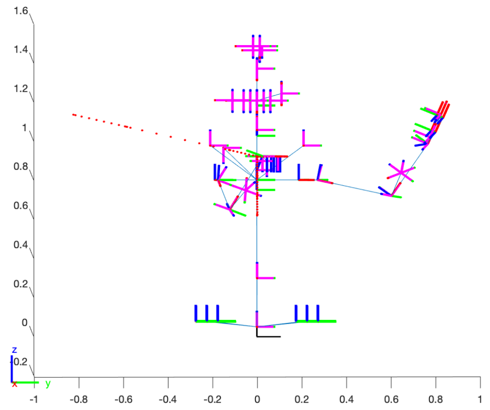
(f)

Figure 7.25: Robot's camera view: Robot planning its movement from grasp position to the target placed left side of robot

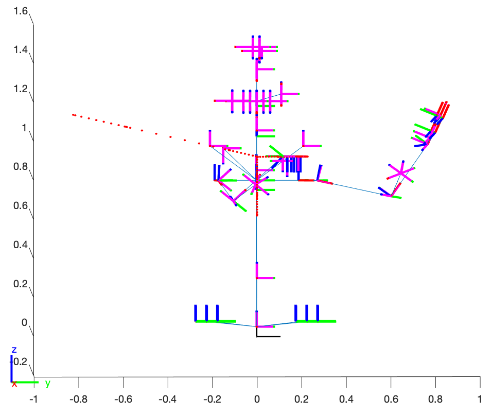
it belongs to the category of objects related to table or category of objects related to the bathroom. The result obtained is shown in Figure 7.30. The figure shows how



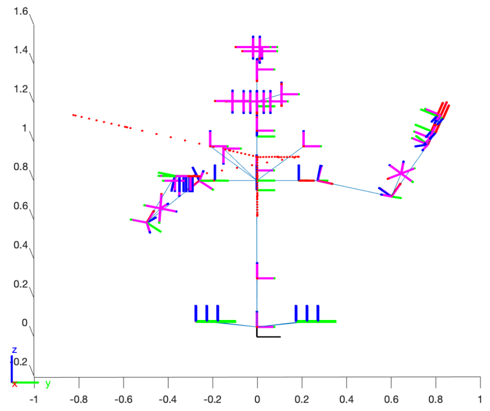
(a)



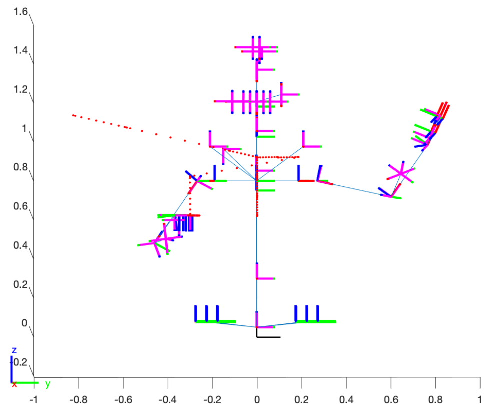
(b)



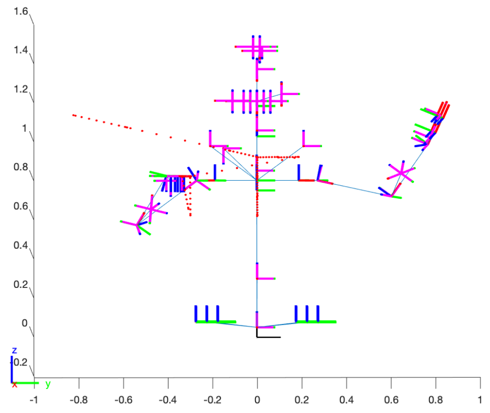
(c)



(d)

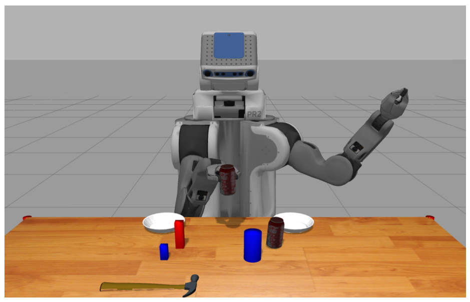


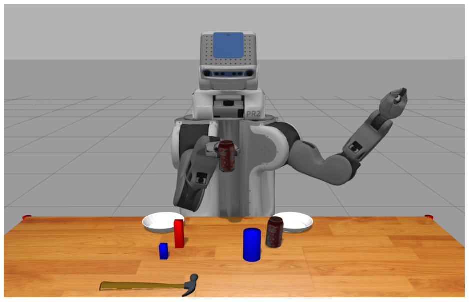
(e)

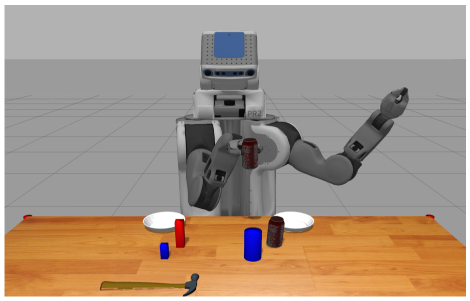


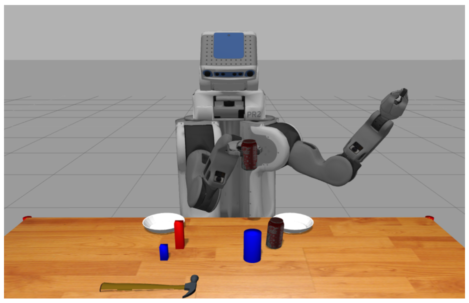
(f)

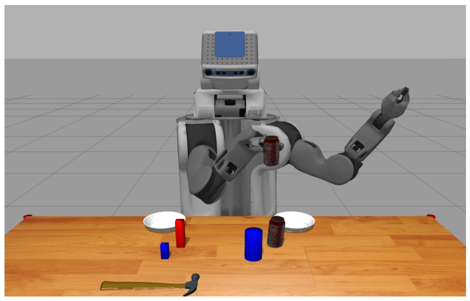
Figure 7.26: Trajectory Planning: Change of trajectory while the robot tries to place the object to its left side when it should be to its right side

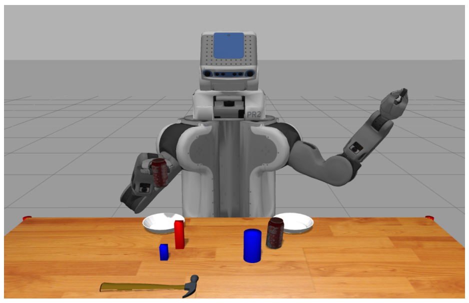

(a)

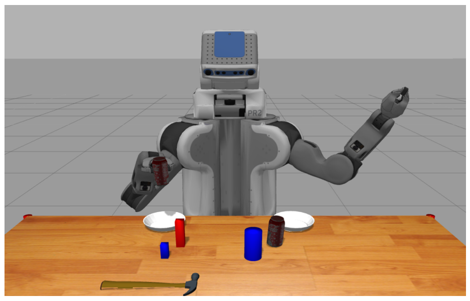

(b)


(c)


(d)


(e)


(f)


(g)

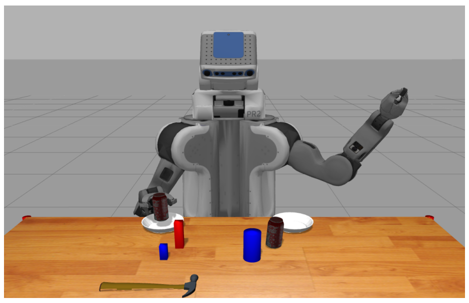

(h)

Figure 7.27: Motion execution: Change of trajectory while the robot tries to place the object to its left side when it should be to its right side

the robot is able to learn to classify the objects over the epochs. After a few sets of epochs, the robot is pretty much able to achieve a classification rate of 100%.

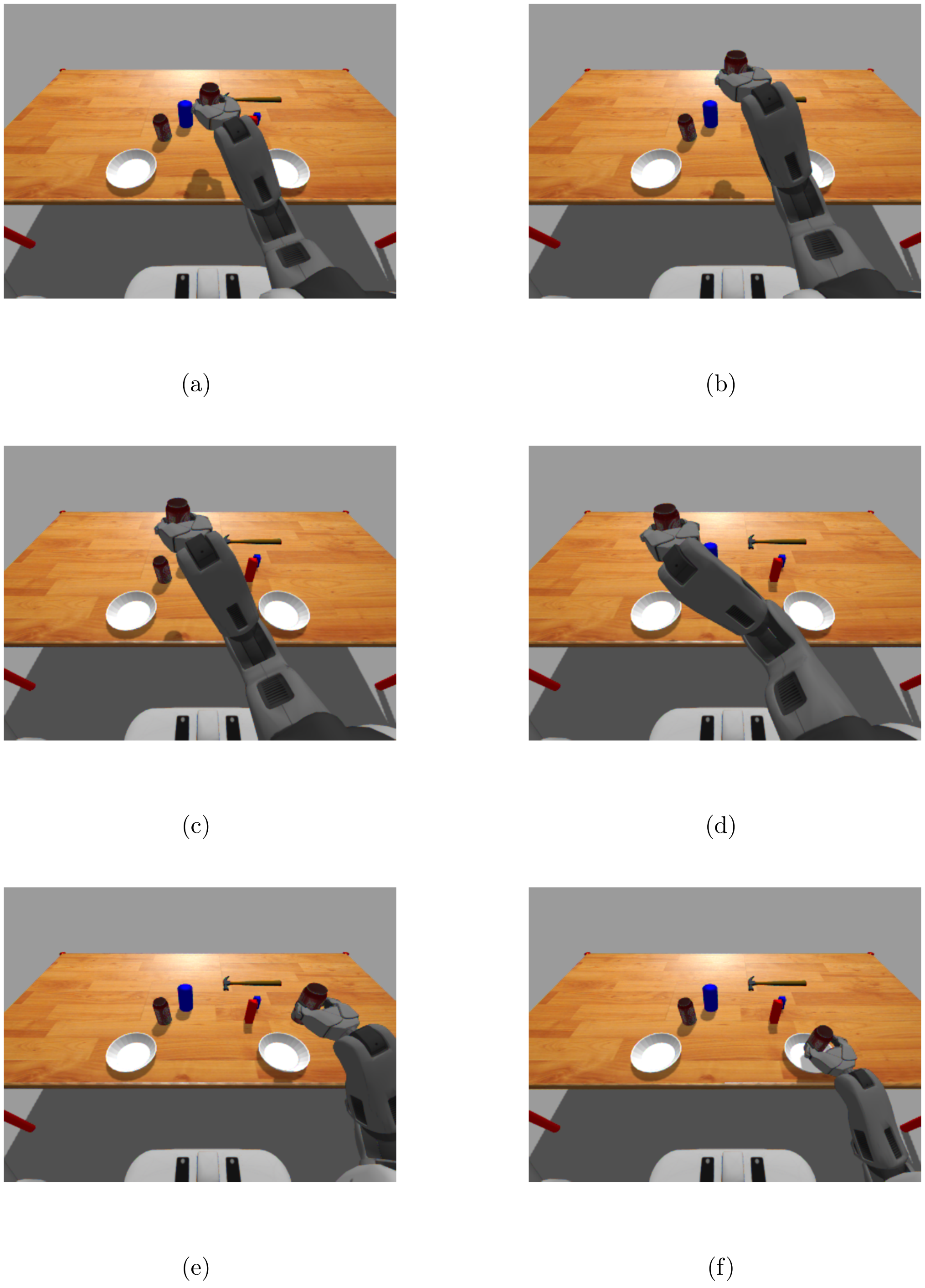


Figure 7.28: Robot's camera view: Change of trajectory while the robot tries to place the object to its left side when it should be to its right side

Similarly, the model is also trained for Case 1 where the objects are sorted based on whether it belongs to the category of objects related to Kitchen or category of

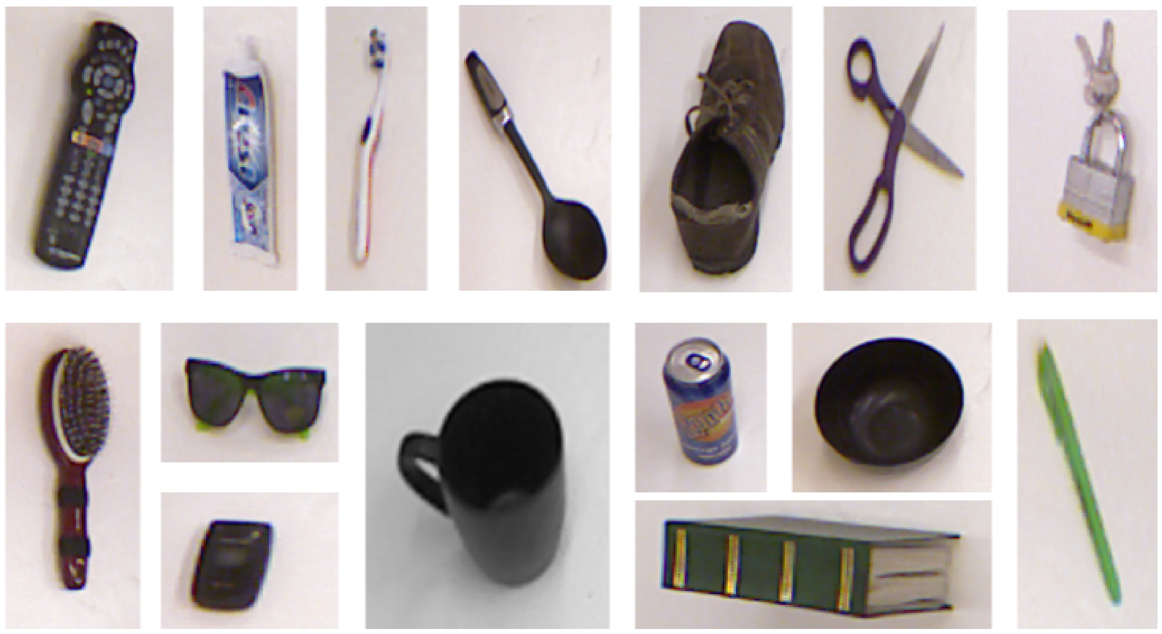


Figure 7.29: Images used to train self learning algorithm

Table 7.4: Categorizing objects based on preference of two different users

USER 1		**USER 2**	
Category 1	***Category 2***	***Category 1***	***Category 2***
Book	Toothpaste	Can	Shoe
Can	Toothbrush	Cup	Glasses
Cell phone	Scissors	Dish bowl	
TV remote	Hairbrush	Spoon	
Keys		TV remote	
Pen			

objects related to clothing. The result obtained is shown in Figure 7.31. The figure shows how the robot is able to learn to classify the objects over the epochs. After a few sets of epochs, the robot is pretty much able to achieve a classification rate of 100%.

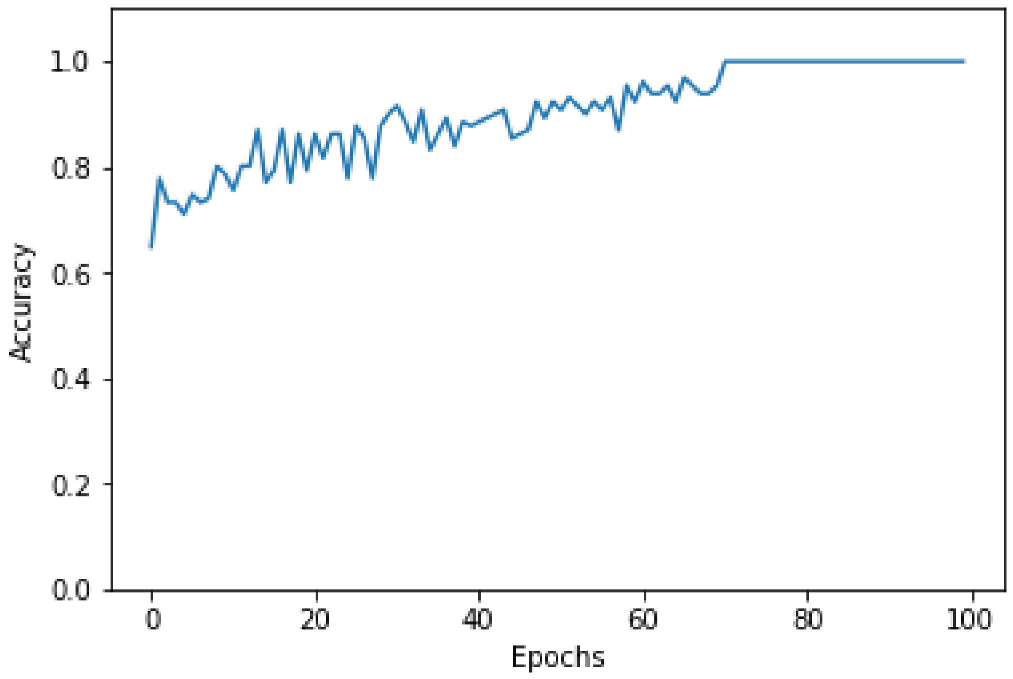


Figure 7.30: Object classified according to user 1

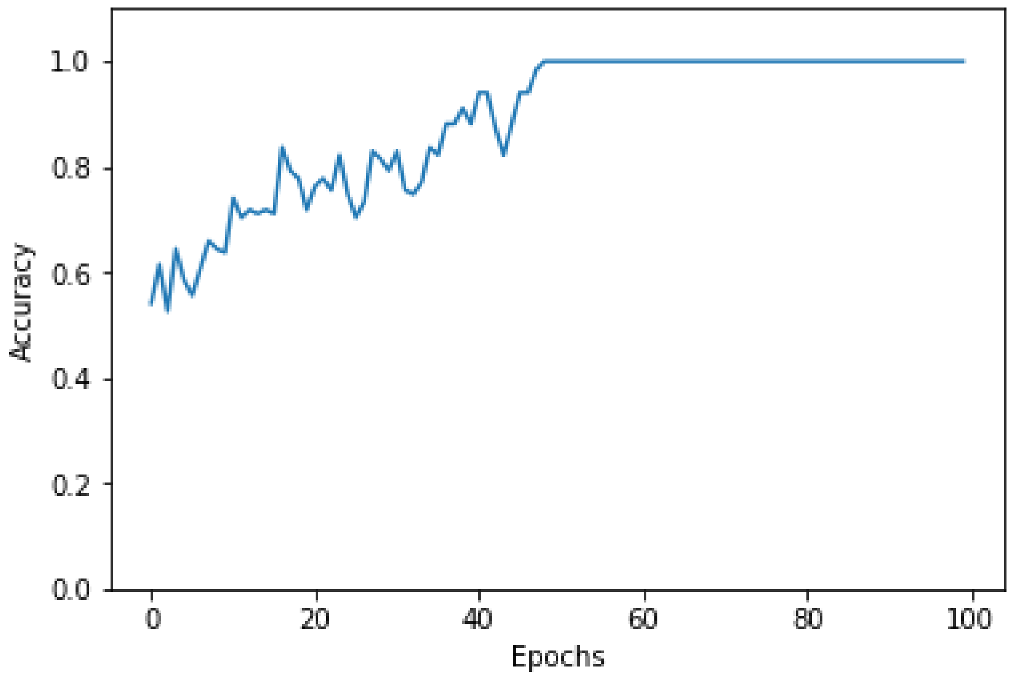


Figure 7.31: Object classified according to user 2

CHAPTER 8: CONCLUSION

This research has showcased the problems associated with communication between locked-in syndrome patients and an assistive robot. It showed a novel approach to bridge the communication gap by enabling communication through brain signal. This was demonstrated by showing disabled persons training a robot using his/her EEG signals to perform daily household tasks such as object retrieval. The task and the objects to be retrieved were selected based on the user requirements presented by ALS patients. The work focused on three major concepts: (1) Developing a novel classifier to accurately classify error-related potentials, (2) Established a system that can autonomously pick and place objects placed on table and (3) Developing an online learning algorithm such that the robot can self learn the class or category of the object it is about to pick.

8.1 ErrP classification

The dataset for error-related potentials was collected and analyzed. The occurrence of ErrP characteristic wave was verified by plotting its grand average and its spatiotemporal maps for electrode selection. We developed a classifier using one of the deep learning algorithm known as a convolutional neural network for classifying of error related potentials. Our proposed method outperformed the previous methods with a lot of difference in their classification accuracy. The network developed by adding batch normalization and drop out layer performed very well when compared to a prior network without any regularization layers. We showed that the CNNs can accurately predict with cross subjects and cross sessions with a difference in the number of weeks. The major concern in CNNs include that they may output false predictions with high confidence rates, may require a large amount of data, may take longer to train than simpler models and involve a large number of hyper-parameters such as the number of layers, type of activation functions, etc.

8.2 Pick and Place objects

The robot pick and place object were performed autonomously. PR2 robot was used as an assistive robot to perform tasks. Using ROS and gazebo we simulated a model like real environment with objects placed on the table. PR2 robot has 7 joints and in order to perform trajectory planning concepts of inverse kinematics and instantaneous inverse kinematics were used for position and velocity respectively. The motion was performed in three steps: (1) Motion from the current position to object position, (2) Grasp the object, and (3) Motion from object's location to its target location. Integration of robot pick and place with human brain signal as feedback was shown which shows the importance of the closed-loop system. Using the human feedback the results for trajectory correction was presented.

8.3 Self-learning

As the robot knows which object belongs to which group based on human feedback, and the data is available in the sequential order, concepts of online machine learning were used for the robot to develop a self-learning capability. The self-learning capability to the robot was a novel implementation and was done using stochastic gradient descent. Objects were collected from an online image database which was as per the choice of ALS patients. 15 objects were classified into two groups based on various criterion's. We show that as the number of epochs or cycles increases the robot is able to learn the class of the object and is able to accurately predict when it sees the same or similar object in the future. We show that the robot can learn as per the choice of human.

8.4 Future Work

This work opens up many possibilities for future work starting from improving the accuracy of the ErrP classifier to improving the self-learning capability of the robot. Some of the possible future work is mentioned below:

- Collecting the dataset of ErrP from various researchers around the world and work towards building a generalized classifier and use the concepts of transfer learning.
- Real-time decoding of EEG signals.
- Other tasks such as personal hygiene, entertainment, eating, and drinking also have also been prioritized and the robot needs to be trained for these tasks too.
- ErrP based control can be used in self-driving tasks as well as autonomous robot navigation.
- In our current research we discussed just two class data, the research can be extended for multiple class and objects in multiple environments.
- Various algorithms can be tried for robotic self-learning keeping in mind about the accuracy of the system. The best model is selected the one which can learn quickly the object selecting tasks.
- Integration of all these modules such as Pick and place objects, ErrP classifier and self-learning into a real robot to perform tasks in real time.

REFERENCES

[1] T. C. Major and J. M. Conrad, "A survey of brain computer interfaces and their applications," in *IEEE SoutheastCon 2014*, pp. 1–8, March 2014.

[2] J. D. Kropotov, "Chapter 1.6 - event-related potentials," in *Functional Neuromarkers for Psychiatry* (J. D. Kropotov, ed.), pp. 59 – 78, San Diego: Academic Press, 2016.

[3] S. S. Sur and V. Sinha, "Event-related potential: An overview," *Industrial Psychiatry Journal*, vol. 18, no. 1, pp. 70–73, 2009.

[4] S. Luck and E. Kappenman, *The Oxford Handbook of Event-Related Potential Components.* Oxford Library of Psychology, Oxford University Press, USA, 2012.

[5] D. Feil-Seifer and M. J. Mataric, "Defining socially assistive robotics," in *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pp. 465–468, June 2005.

[6] B. Chandrasekaran and J. M. Conrad, "Human-robot collaboration: A survey," in *SoutheastCon 2015*, pp. 1–8, April 2015.

[7] M. G. R. Patterson, "Locked-in syndrome: A review of 139 cases," *Stroke; a journal of cerebral circulation*, vol. 17, pp. 758–64, 07 1986.

[8] V. Gandhi, "Chapter 1 - introduction," in *Brain-Computer Interfacing for Assistive Robotics* (V. Gandhi, ed.), pp. 1 – 6, San Diego: Academic Press, 2015.

[9] A. Kübler, B. Kotchoubey, J. Kaiser, and N. Birbaumer, "Brain-computer communication: Unlocking the locked in," *Psychological Bulletin*, vol. 127, pp. 358–375, 5 2001.

[10] I. Iturrate, J. M. Antelis, A. Kubler, and J. Minguez, "A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation," *IEEE Transactions on Robotics*, vol. 25, pp. 614–627, June 2009.

[11] I. K. S. Schroer, "An autonomous robotic assistant for drinking," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015, pp. 6482–6487, 06 2015.

[12] A. F. Salazar-Gomez, J. DelPreto, S. Gil, F. H. Guenther, and D. Rus, "Correcting robot mistakes in real time using EEG signals," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6570–6577, May 2017.

[13] P. Encarnação and A. Cook, *Robotic assistive technologies: Principles and practice.* CRC Press, 2017.

[14] H. Yanco, “Wheelesley Development of a Robotic Wheelchair System.” http://www.cs.uml.edu/robots/wheelesley/wheelesley.html, 2012.

[15] CMU, “Personal and Assistive Robotics.” https://www.ri.cmu.edu/research/personal-assistive-robotics/.

[16] S. Adee, “DARPA’s prosthetic arm gives amputees new hope.” https://spectrum.ieee.org/robotics/medical-robots/winner-the-revolution-will-be-prosthetized/2, 2009.

[17] A. J. Rentschler, R. A. Cooper, B. Blasch, and M. L. Boninger, “Intelligent walkers for the elderly: Performance and safety testing of va-pamaid robotic walker,” *Journal of rehabilitation research and development*, vol. 40, no. 5, pp. 423–432, 2003.

[18] T. HORNYAK, “PR2 robot helps quadriplegic man shave himself.” https://www.cnet.com/news/pr2-robot-helps-quadriplegic-man-shave-himself/, 2011. [Online; posted July 14, 2011].

[19] T. L. Chen, M. Ciocarlie, S. Cousins, P. M. Grice, K. Hawkins, K. Hsiao, C. C. Kemp, C.-H. King, D. A. Lazewatsky, A. E. Leeper, *et al.*, “Robots for humanity: using assistive robotics to empower people with disabilities,” *IEEE Robotics & Automation Magazine*, vol. 20, no. 1, pp. 30–39, 2013.

[20] S. W. Brose, D. J. Weber, B. A. Salatin, G. G. Grindle, H. Wang, J. J. Vazquez, and R. A. Cooper, “The role of assistive robotics in the lives of persons with disability,” *American Journal of Physical Medicine & Rehabilitation*, vol. 89, no. 6, pp. 509–521, 2010.

[21] G. Nejat, Y. Sun, and M. Nies, “Assistive robots in health care settings,” *Home Health Care Management & Practice*, vol. 21, no. 3, pp. 177–187, 2009.

[22] A. Prakash, “Putting a Face on a Robot.” https://www.news.gatech.edu/2013/10/01/putting-face-robot, 2013.

[23] K. Kanda, “Human-Robot Interaction.” http://humanrobotinteraction.org/1-introduction/, 2012. [Online; posted 8th-Feb-2012].

[24] A. Cohen, “Biomedical signals: Origin and dynamic characteristics; frequency-domain analysis,” in *The biomedical engineering handbook*, vol. 2, pp. 951–74, CRC Press, 2000.

[25] H. Van der Loos, J. Hammel, and L. Leifer, “Task-specific assessment of robot effectiveness: enhancing the independence of quadriplegics in the workplace,” in *Fifth International Conference on Advanced Robotics’ Robots in Unstructured Environments*, pp. 905–909, IEEE, 1991.

[26] H. M. Van der Loos, J. J. Wagner, N. Smaby, K. Chang, O. Madrigal, L. J. Leifer, and O. Khatib, "ProVAR assistive robot system architecture," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 1, pp. 741–746, IEEE, 1999.

[27] B. Driessen, H. Evers, and J. v Woerden, "MANUS a wheelchair-mounted rehabilitation robot," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of engineering in medicine*, vol. 215, no. 3, pp. 285–290, 2001.

[28] H. A. Tijsma, F. Liefhebber, and J. L. Herder, "Evaluation of new user interface features for the MANUS robot arm," in *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pp. 258–263, IEEE, 2005.

[29] M. Ghorbel, M. Haariz, B. Grandjean, and M. Mokhtari, "Toward a generic human machine interface for assistive robots: The AMOR project," in *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pp. 168–172, IEEE, 2005.

[30] M. Trabelsi, N. Aitoufroukh, and S. Lelandais, "Improvements of object grabbing method by using color images and neural networks classification," in *IECON 2006-32nd Annual Conference on IEEE Industrial Electronics*, pp. 3922–3927, IEEE, 2006.

[31] H. Nguyen, C. Anderson, A. Trevor, A. Jain, Z. Xu, and C. C. Kemp, "El-E: An assistive robot that fetches objects from flat surfaces," in *Robotic helpers, int. conf. on human-robot interaction*, 2008.

[32] Z. Erickson, H. M. Clever, G. Turk, C. K. Liu, and C. C. Kemp, "Deep haptic model predictive control for robot-assisted dressing," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8, IEEE, 2018.

[33] D. Park, Y. K. Kim, Z. M. Erickson, and C. C. Kemp, "Towards assistive feeding with a general-purpose mobile manipulator," *arXiv preprint arXiv:1605.07996*, 2016.

[34] A. Clegg, W. Yu, J. Tan, C. C. Kemp, G. Turk, and C. K. Liu, "Learning human behaviors for robot-assisted dressing," *arXiv preprint arXiv:1709.07033*, 2017.

[35] Z. Erickson, M. Collier, A. Kapusta, and C. C. Kemp, "Tracking human pose during robot-assisted dressing using single-axis capacitive proximity sensing," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2245–2252, 2018.

[36] Z. Erickson, S. Chernova, and C. C. Kemp, "Semi-supervised haptic material recognition for robots using generative adversarial networks," *arXiv preprint arXiv:1707.02796*, 2017.

[37] A. Kübler, B. Kotchoubey, J. Kaiser, J. R. Wolpaw, and N. Birbaumer, "Brain–computer communication: Unlocking the locked in.," *Psychological bulletin*, vol. 127, no. 3, p. 358, 2001.

[38] S. N. Abdulkader, A. Atia, and M.-S. M. Mostafa, "Brain computer interfacing: Applications and challenges," *Egyptian Informatics Journal*, vol. 16, no. 2, pp. 213 – 230, 2015.

[39] F. Sharbrough, "American electroencephalographic society guidelines for standard electrode position nomenclature," *J clin Neurophysiol*, vol. 8, pp. 200–202, 1991.

[40] R. Oostenveld and P. Praamstra, "The five percent electrode system for high-resolution EEG and ERP measurements," *Clinical neurophysiology*, vol. 112, no. 4, pp. 713–719, 2001.

[41] E. Niedermeyer and F. L. da Silva, *Electroencephalography: basic principles, clinical applications, and related fields.* Lippincott Williams & Wilkins, 2005.

[42] E. Lopez-Larraz, M. Creatura, I. Iturrate, L. Montesano, and J. Minguez, "EEG single-trial classification of visual, auditive and vibratory feedback potentials in brain-computer interfaces," *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4231–4234, 2011.

[43] H. Cecotti and A. Graser, "Convolutional neural networks for p300 detection with application to brain-computer interfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 433–445, March 2011.

[44] H. J. Falkenstein M, Hohnsbein J and L. Blanke, "Effects of crossmodal divided attention on late ERP components. ii. error processing in choice reaction tasks," *Electroencephalography and Clinical Neurophysiology*, vol. 78, no. 6, pp. 447–455, 1991.

[45] D. M. Olvet and G. Hajcak, "The error-related negativity (ERN) and psychopathology: toward an endophenotype.," *Clinical psychology review*, vol. 28 8, pp. 1343–54, 2008.

[46] R. Chavarriaga, A. Sobolewski, and J. d. R. Millán, "Errare machinale est: the use of error-related potentials in brain-machine interfaces," *Frontiers in Neuroscience*, vol. 8, p. 208, 2014.

[47] M. Lehne, K. Ihme, A. Brouwer, J. B. F. van Erp, and T. O. Zander, "Error-related EEG patterns during tactile human-machine interaction," in *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*, pp. 1–9, Sept 2009.

[48] R. Chavarriaga, I. Iturrate, Q. Wannebroucq, and J. d. R. Millán, "Decoding fast-paced error-related potentials in monitoring protocols," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1111–1114, Aug 2015.

[49] J. Tessadori, L. Schiatti, G. Barresi, and L. S. Mattos, "Does tactile feedback enhance single-trial detection of error-related EEG potentials?," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1417–1422, Oct 2017.

[50] P. W. Ferrez and J. del R. Millan, "Error-related EEG potentials generated during simulated brain–computer interaction," *IEEE Transactions on Biomedical Engineering*, vol. 55, pp. 923–929, March 2008.

[51] T. C. Major and J. M. Conrad, "The effects of pre-filtering and individualizing components for electroencephalography neural network classification," in *SoutheastCon 2017*, pp. 1–6, March 2017.

[52] D. J. McFarland, L. M. McCane, S. V. David, and J. R. Wolpaw, "Spatial filter selection for EEG-based communication," *Electroencephalography and clinical Neurophysiology*, vol. 103, no. 3, pp. 386–394, 1997.

[53] M. J. Alhaddad, "Common average reference (CAR) improves p300 speller," *International Journal of Engineering and Technology*, vol. 2, no. 3, p. 21, 2012.

[54] S. H. F. Syam, H. Lakany, R. Ahmad, and B. A. Conway, "Comparing common average referencing to laplacian referencing in detecting imagination and intention of movement for brain computer interface," in *MATEC Web of Conferences*, vol. 140, 2017.

[55] W. J. Gehring, B. Goss, M. G. Coles, D. E. Meyer, and E. Donchin, "A neural system for error detection and compensation," *Psychological science*, vol. 4, no. 6, pp. 385–390, 1993.

[56] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, "Optimizing EEG channel selection by regularized spatial filtering and multi band signal decomposition," in *IASTED Int. Conf. Biomedical Engineering*, pp. 86–90, 2010.

[57] T. Tanaka and M. Arvaneh, *Signal processing and machine learning for brain-machine interfaces.* Institution of Engineering & Technology, 2018.

[58] E. M. Ventouras, P. Asvestas, I. Karanasiou, and G. K. Matsopoulos, "Classification of error-related negativity (ERN) and positivity (Pe) potentials using knn and support vector machines," *Computers in Biology and Medicine*, vol. 41, no. 2, pp. 98 – 109, 2011.

[59] R. Chavarriaga and J. d. R. Millan, "Learning from EEG error-related potentials in noninvasive brain-computer interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, pp. 381–388, Aug 2010.

[60] J. Bollon, R. Chavarriaga, J. del R. Millan, and P. Bessiere, "EEG error-related potentials detection with a bayesian filter," in *2009 4th International IEEE/EMBS Conference on Neural Engineering*, pp. 702–705, April 2009.

[61] R. Schirrmeister, L. Gemein, K. Eggensperger, F. Hutter, and T. Ball, "Deep learning with convolutional neural networks for decoding and visualization of EEG pathology," in *2017 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, pp. 1–7, Dec 2017.

[62] S. Sakhavi, C. Guan, and S. Yan, "Parallel convolutional-linear neural network for motor imagery classification," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, pp. 2736–2740, Aug 2015.

[63] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9 – 21, 2004.

[64] P. W. Ferrez and J. d. R. Millán, "Error-related EEG potentials generated during simulated brain–computer interaction," *IEEE transactions on biomedical engineering*, vol. 55, no. 3, pp. 923–929, 2008.

[65] J. Ma, Y. Zhang, A. Cichocki, and F. Matsuno, "A novel EOG/EEG hybrid human–machine interface adopting eye movements and ERPs: Application to robot control," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 3, pp. 876–889, 2015.

[66] K. LaFleur, K. Cassady, A. Doud, K. Shades, E. Rogin, and B. He, "Quadcopter control in three-dimensional space using a noninvasive motor imagery-based brain–computer interface," *Journal of neural engineering*, vol. 10, no. 4, p. 046003, 2013.

[67] H. Zhang, R. Chavarriaga, Z. Khaliliardali, L. Gheorghe, I. Iturrate, and J. d R Millán, "EEG-based decoding of error-related brain activity in a real-world driving task," *Journal of neural engineering*, vol. 12, no. 6, p. 066028, 2015.

[68] S. Ehrlich and G. Cheng, "A neuro-based method for detecting context-dependent erroneous robot action," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 477–482, IEEE, 2016.

[69] Y. Zhang, W. Chen, J. Zhang, and J. Wang, "Extracting error-related potentials from motion imagination EEG in noninvasive brain-computer interface," in *2017 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pp. 768–773, IEEE, 2017.

[70] H. Kolkhorst, M. Tangermann, and W. Burgard, "Guess what i attend: Interface-free object selection using brain signals," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7111–7116, IEEE, 2018.

[71] R. Karban, *Plant sensing and communication.* University of Chicago Press, 2015.

[72] C. M. Bishop, *Pattern recognition and machine learning.* springer, 2006.

[73] "Applying Artificial Intelligence and Machine Learning in Robotics." https://www.robotics.org/blog-article.cfm/Applying-Artificial-Intelligence-and-Machine-Learning-in-Robotics/103, 2018. [Online; Posted on 26th June 2018].

[74] Y. Chang, P. Chung, and H. Lin, "Deep learning for object identification in ROS-based mobile robots," in *2018 IEEE International Conference on Applied System Invention (ICASI)*, pp. 66–69, April 2018.

[75] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.

[76] V. M. Aparanji, U. V. Wali, and R. Aparna, "Robotic motion control using machine learning techniques," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, pp. 1241–1245, April 2017.

[77] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pp. 253–256, May 2010.

[78] T. N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A. rahman Mohamed, G. Dahl, and B. Ramabhadran, "Deep convolutional neural networks for large-scale speech tasks," *Neural Networks*, vol. 64, pp. 39 – 48, 2015. Special Issue on Deep Learning of Representations.

[79] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[80] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update," *Journal of neural engineering*, vol. 15, no. 3, p. 031005, 2018.

[81] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[82] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[83] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[84] A. Dertat, "Applied Deep Learning - Part 4: Convolutional Neural Networks." https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2, 2017.

[85] E. Hazan *et al.*, "Introduction to online convex optimization," *Foundations and Trends in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[86] T. Srivastava, "Introduction to Online Machine Learning: Simplified." https://www.analyticsvidhya.com/blog/2015/01/introduction-online-machine-learning-simplified-2/, 2015.

[87] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[88] "Stochastic Gradient Descent." https://deepai.org/machine-learning-glossary-and-terms/stochastic-gradient-descent, 2017. [Online].

[89] L. Bottou, "Online algorithms and stochastic approximations," in *Online Learning and Neural Networks* (D. Saad, ed.), Cambridge, UK: Cambridge University Press, 1998. revised, oct 2012.

[90] B. Siciliano and O. Khatib, *Springer handbook of robotics.* Springer, 2016.

[91] J. Denavit, "A kinematic notation for low pair mechanisms based on matrices," *ASME J. Appl. Mech.*, vol. 22, pp. 215–221, 1955.

[92] J. Duffy, *Analysis of mechanisms and robot manipulators.* Edward Arnold London, 1980.

[93] J. Nocedal and S. Wright, *Numerical optimization.* Springer Science & Business Media, 2006.

[94] J. Zhao and N. I. Badler, "Inverse kinematics positioning using nonlinear programming for highly articulated figures," *ACM Transactions on Graphics (TOG)*, vol. 13, no. 4, pp. 313–336, 1994.

[95] L. Han and J. C. Trinkle, "The instantaneous kinematics of manipulation," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 3, pp. 1944–1949 vol.3, May 1998.

[96] R. Chavarriaga, A. Sobolewski, and J. d. R. Millán, "Errare machinale est: the use of error-related potentials in brain-machine interfaces," *Frontiers in neuroscience*, vol. 8, p. 208, 2014.

[97] C. B. Holroyd and M. G. Coles, "The neural basis of human error processing: reinforcement learning, dopamine, and the error-related negativity.," *Psychological review*, vol. 109, no. 4, p. 679, 2002.

[98] M. Spüler and C. Niethammer, "Error-related potentials during continuous feedback: using EEG to detect errors of different type and severity," in *Front. Hum. Neurosci.*, 2015.

[99] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, pp. 1263–1284, Sept 2009.

[100] S. A. S. Bellary and J. M. Conrad, "Classification of error related potentials using convolutional neural networks," in *9th International Conference on Cloud Computing, Data Science and Engineering, Noida, India*, January 2019.

[101] W. Garage, "PR2 user manual," 2012.

[102] T. Rahman, S. Stroud, R. Ramanathan, M. Alexander, R. Alexander, R. Seliktar, and W. Harwin, "Consumer criteria for an arm orthosis," *Technology and Disability*, vol. 5, no. 2, 1996.

[103] C. A. Stanger, C. Anglin, W. S. Harwin, and D. P. Romilly, "Devices for assisting manipulation: a summary of user task priorities," *IEEE Transactions on rehabilitation Engineering*, vol. 2, no. 4, pp. 256–265, 1994.

[104] Y. S. Choi, T. Deyle, T. Chen, J. D. Glass, and C. C. Kemp, "A list of household objects for robotic retrieval prioritized by people with ALS," in *2009 IEEE International Conference on Rehabilitation Robotics*, pp. 510–517, IEEE, 2009.

[105] "Optimization: Stochastic Gradient Descent, Supervised learning and Optimization." http://deeplearning.stanford.edu/tutorial/supervised/Optimization StochasticGradientDescent/. [Deep Learning Tutorial - Stanford University].