Reliable & Intuitive Upper-Limb Prosthetic Control System *

Dhiyaa Al Jorf[1] and Firas Darwish[2]

[1]NYUAD, Computer Engineering, Abu Dhabi, UAE da2863@nyu.edu
[2]NYUAD, Computer Engineering, Abu Dhabi, UAE, fbd2014@nyu.edu

August 29, 2025

*This capstone project is supervised by Professor Mohamad Eid, Department of Electrical Engineering & Department of Computer Engineering, NYU Abu Dhabi.

Contents

Abstract

The physical design of upper-limb prosthetic devices has advanced rapidly to allow for very refined high Degree of Freedom (DoF) control. However, control strategies for these prostheses remain unsatisfactory—with long training periods and high rejection rates—preventing prosthetic users from achieving their devices' full potential. We identify low reliability and lack of intuitiveness to be the main drawbacks of current prosthetic control strategies, where reliability refers to the degree of success in object manipulation tasks, and intuitiveness refers to how naturally new users master control of their prosthetic. This ongoing project proposes an intuitive, reliable, transradial prosthetic control strategy. The real-time controller, used to control a virtual prosthetic, is split into three main modules: the computer vision module with a virtual head-mounted RGBD camera as input, an sEMG module controlled by an 8-channel armband, and a fusion module that determines the final prosthetic behavior. The computer vision module identifies grasp candidates and proposes rough grasps around them. The user intention is then captured by the sEMG module, and select gestures are highlighted to the user accordingly. Through the fusion module, the chosen rough grasp is adjusted based on the sEMG signals to successfully manipulate the grasp candidate. By relying less on sEMG control than other myoelectric prosthetic control strategies, the benefit is twofold: (1) The system achieves better reliability due to the high performance of vision based Grasp Pose Detection (GPD) models when compared to sEMG Hand Gesture Prediction (HGR) models. (2) The sEMG module needs less fine-tuning time to achieve satisfactory intention prediction. The intuitiveness of the system then stems from a combination of the low sEMG module fine-tuning time and the selective intention-based fusion module. Subjects were recruited and two baseline unimodal (i.e. vision-only and sEMG-only) strategies are developed to compare the proposed strategy's intuitiveness and reliability. Results show promise in achieving the desired reliability and intuitiveness through the fusion approach, with room for future work to expand on the proposed solution.

Keywords: Prosthetic Control; Rehabilitation Engineering; Human Computer Interface; Haptics and Haptic Interfaces, Medical Robots and Systems;

1 Problem Definition

1.1 Problem Analysis

1. Who has the problem?

   Limb difference either due to amputation or congenital malformations impacts an estimated 35–40 million people worldwide, and is expected to double by the middle of this century due to increased life expectancy (Organization, 2024). Unfortunately, prosthetic technology is insufficiently accessible or developed globally, where the cost is exorbitantly high for limited functionality. Regardless, the users with access to prostheses are often dissatisfied with the technology, leading to very high rejection rates. The problem is therefore one that impacts prosthetic users worldwide.

2. What does the problem seem to be?

   On a physical level, prosthetic devices have achieved impressive capabilities. However, these functions are locked behind a high cognitive load on the users, where, especially for upper-limb prostheses which need to be more generalized in use, the current existing prosthetic control systems place too much stress on their users. This stress results in rejections in high complexity prosthetics, and users end up settling either for less complex prosthetics (e.g. cosmetic prosthetics) or none at all. The problem therefore is the unusable interfaces of these prosthetic devices.

3. What are the features of a good prosthetic control system/interface then?

   There are two main features of a good prosthetic controller: reliability and intuitiveness.

   (a) Reliable Control: Reliability is a necessity for any system, where it refers to the system's ability to match performed output to the desired output. Prosthetics are not an exception, where reliability in control is essential to mitigate user frustration and the very possible chance of accidents occurring due to unpredicted prosthetic behavior.

   (b) Intuitive Control: Arguably, one of the most important elements of any human-usable system, where even the most advanced system can be rendered unusable if its interface is non-intuitive. Given the seamlessness of the human body, prostheses need to similarly integrate with the remainder of the body to offer more long-term utility.

4. What are the resources?

   Input-wise, there are multiple channels to be used for a prosthetic control system. A prosthetic controller's ultimate task is intention-prediction. There are multiple intention-encoding signals usable from the human body including but limited to EMG, FMG, EEG, vision, gaze-tracking, force-feedback, etc. On the market, there are many accessible devices for measuring these signals. The proposed system uses sEMG and vision signals picked up through the Mindrove 8-channel armband and virtual cameras in VR space respectively. The Meta Quest 3 VR headset is used in the proposed system, and the simulation is created within Unity.

5. When does the problem occur? Under what circumstances?

   Upon the completion of the learning stage, rejection is very unlikely due to the reduced cognitive load upon getting accustomed to the controller. That is to say, users are more likely to reject prostheses during the learning stage rather than later due to the increased cognitive load associated with learning new skills. This is critical as most users are not willing to restart the learning phase, and thus upon the rejection of high complexity prosthetics and completing the learning stage with simpler prostheses, users may never attempt to use higher complexity prostheses again.

6. Where does the problem occur?

   Prostheses are an extension of the user, and thus their use isn't localized to a singular location. As long as there are objects to interact with, a bad prosthetic control system would result in low user satisfaction and possible rejection.

7. How does the problem occur?

   More often than not, users are in a very emotional state during the prosthetic learning stage either due to the recent limb loss for the case of amputees or the fear that the prosthetic system may not be the solution they were looking for generally. Additionally, the cognitive load associated with a task is highest during the learning stage. This background stress level therefore disadvantages poor prosthetic control systems even further. Upon repeated failures, users may decide to terminate the learning stage. This often also involves the building of a strong emotional barrier for further learning.

8. What are the reasons commercial products are lacking?

   (a) In the commercial field, insurance constrains what innovations are worth investigation. Even when research suggests huge benefits to a certain approach, if the process would require new conversations with insurance to approve the changes, companies may find it more cost effective to scrap said approach.

   (b) Prosthetic users pay exorbitant amounts of money for their prostheses. With that much money on the line, users may be much less experimental with newer equipment and opt for simpler tried and tested systems.

1.2 Problem Clarification - Black Box Modeling

The black box models in the following figures were constructed based on the problem analysis. The goal of the proposed system is to consolidate the benefits of the vision-based and sEMG-based unimodal systems into one multimodal system that fulfills the criteria laid out in Section 3. The Black-box models of the unimodal and multimodal systems, as well as each component module are therefore presented and described in this section.

1.2.1 Unimodal Black Box Diagram

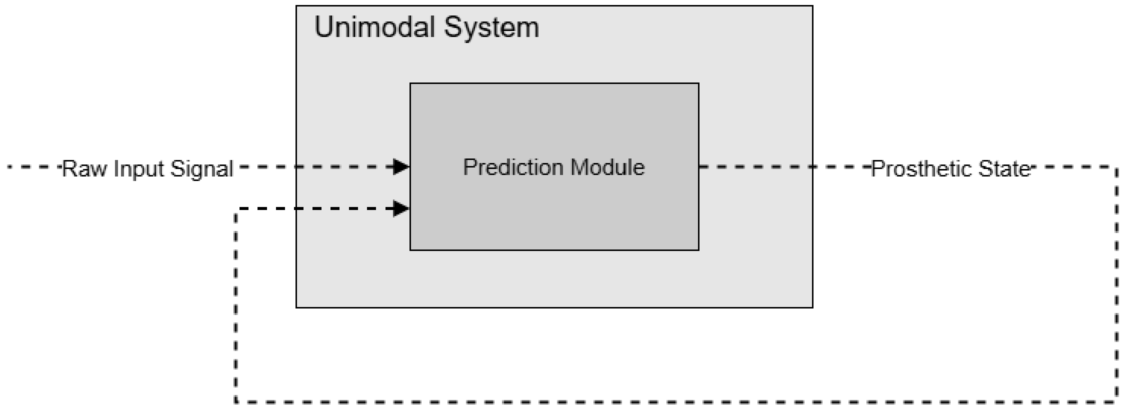

Figure 1: High-level black-box diagram of the proposed unimodal prosthetic control system

As shown in Figure 1, there would be a single input source processed by the singular prediction module to determine the system (prosthetic) state, which may feed back into the system. The prediction module is one of the sEMG or Vision modules described later.

1.2.2 Multimodal Black Box Diagram

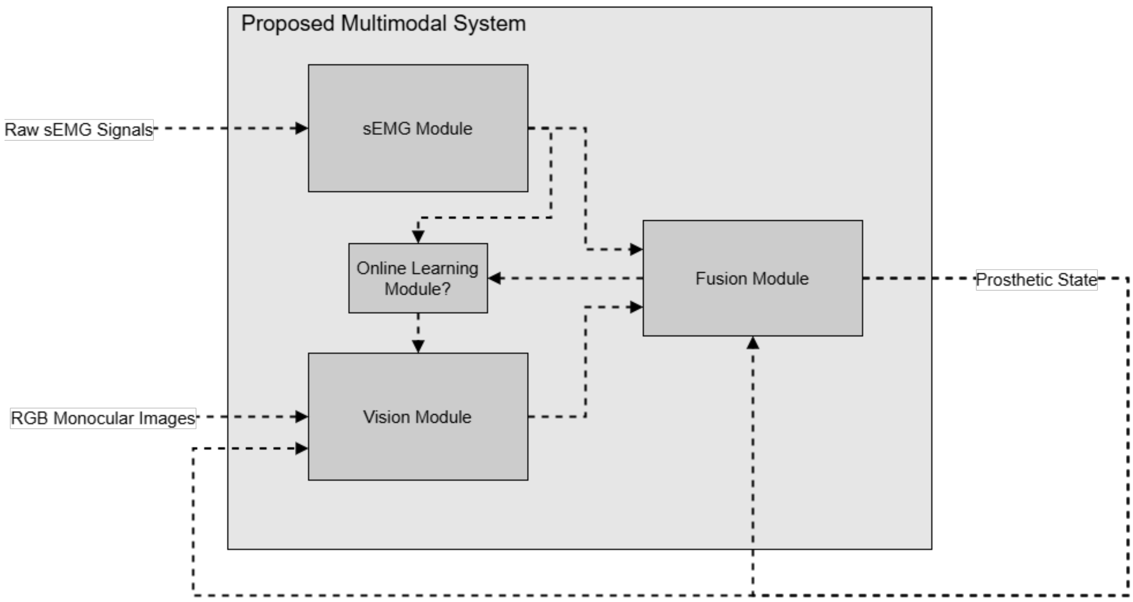

Figure 2: High-level full black-box diagram of the proposed multimodal prosthetic control system

As shown in Figure 2, the multimodal is an expansion of the unimodal system, where the two modules' outputs are fused together through a fusion module. Additionally, an online learning module is introduced to allow for better tuning of the vision module to the user's choices.

1.2.3 sEMG Module Black Box

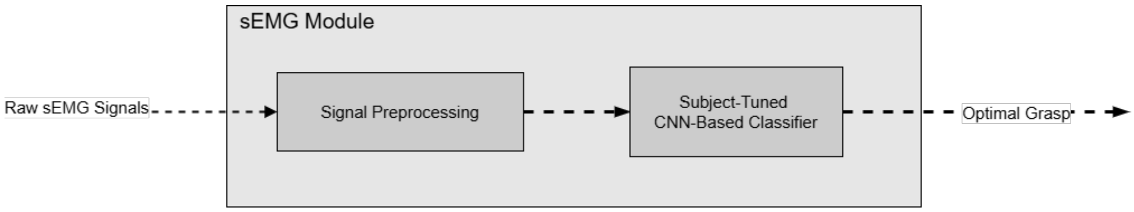

Figure 3: Black-box diagram of the proposed sEMG module

The sEMG module, shown in Figure 3, is a simple intention prediction that involves preprocessing the signal and running it through a fine-tuned classifier. The fine-tuning step is a standard procedure for sEMG-based prosthetic control systems, where subject-specific data is used to tune a pretrained general model.

1.2.4 Vision Module Black Box

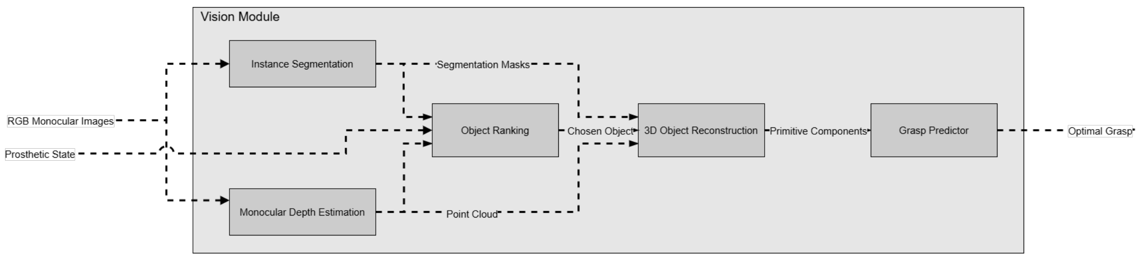

Figure 4: Black-box diagram of the proposed vision module

Compared to the sEMG module, the vision module, shown in Figure 4, is more complex, as it involves intention prediction through indirect user signals. Through instance segmentation, depth estimation, and knowledge of the current prosthetic state, a single object from the given RGB input image is ranked as the most likely to be grasped. Said object is then processed through a 3D object reconstruction module which reconstructs the object from 3D primitives used for the final grasp predictor. The grasp predictor is tunable based on the online learning module.

1.3 Problem Statement

- Initial Problem Statement:

  The prosthetic controller must be reliable and intuitive to ensure rejections due to cognitive load are unlikely.

- Refined Problem Statement:

  A new prosthetic learner should not reject the prosthetic within the 12-13 week typical training and physical therapy period. The prosthetic must be able to succeed at the user's intended task at a rate of 85 %, and the system must operate in real-time ($<$ 350 ms delay) (Roche *et al.*, 2014) (Russell and Bergmann, 2023). The system must learn during the training period such that the control becomes more autonomous as the system fits better to the user.

2 Design Constraints

2.1 Technical Constraints

1. Given that we are unable to build and design a physical prosthetic, we are limited to simulating the physical and embedded properties of the prosthetic in VR that is running on a machine with an Nvidia RTX 4060 GPU on an i7 CPU.

2. We are using the MindRove to collect sEMG data that is used for processing. Some of the technical constraints associated with this device is that it uses sparse sEMG with 8 channels (as opposed to high density sEMG sensors which can contain upwards of 64 channels). The performance of this sensor is also highly contingent on environmental factors (from sweat, hair, electrode shift, inconsistent placement).

3. To allow for seamless real-time control, the total delay of the hard real time components of the system must sum up to less than 350 ms. The soft real time systems which are not as critical in the final delay do not factor into this constraint.

2.2 Non-Technical Constraints

1. Prosthetic hardware itself incurs at a very high cost to begin with, not to mention the refusal of insurance companies to cover prosthetics they may deem to expensive for their features. Our solution therefore must not incur a huge additional cost to the total system including all additional hardware necessary.

2. VR simulations of prosthetic control may diverge in performance from their real world equivalent. The system therefore would not be able to perfectly reflect the predicted performance of the system in a real prosthetic control situation.

3. Prolonged wearing of an uncomfortable sensor suite is highly undesirable. The design must therefore select the components easiest to integrate into the usual prosthetic device without the need for extra bulky equipment.

4. Given the delay of conducting testing of this design on amputees, one major non-technical constraint is the fact that this system will be tested on able-bodied humans. Naturally, a consequence of this is that our evaluation of reliability and intuition may not directly map to the experiences of an amputee. At the same time, testing this device on able-bodied persons may cause challenges associated with that person instinctively relying on their arm (issues with proprioception, somatoception, and grasp strength).

5. The total budget of the project cannot exceed $3000 due to the maximum provided budget. However, to ensure the design doesn't significantly increase the cost of the already expensive prosthetics, the total budget shall not exceed $2000 (including all simulation costs which would not be transferred to the user).

3 Criteria for Design Evaluation & Testing

To evaluate the reliability and intuitiveness of the proposed design, we have established the following testing criteria:

- **sEMG Module Accuracy:** The sEMG module should be able to independently achieve 80% accuracy after fine-tuning to the given user.

- **Vision Module:** The Vision module is composed of mulitple sub-components. The goal is that the predicted optimal grasp should match with the user's intent 60% of the time in the general state before any online learning is implemented.

- **Fusion:** Upon fusing the modalities, the system needs achieve a total 85% success rate at each of the assigned tasks.

- **Online Learning Phase:** The online learning component needs to raise the vision module's agreement with the user intent to 80%.

- **System Variability:** Between different donnings and undonnings of the prosthetic, the system accuracy should not vary more than 5%.

- **Real-Time Control:** The hard-real time components of the controller need to amount to a total of 350 ms. The soft-real time components do not contribute to that total delay because a decision can be made based on the previous knowledge of the soft-real time information.

- **Testing Environments:** The testing environments needs to replicate realistic settings both indoors and outdoors. Each environment should involve set goals (e.g. bake a cake, fold origami, etc.) that can be evaluated quantitatively.

4 Deliverables Statement

- Hardware
  - Mindrove 8-channel sEMG armband
  - Meta Quest 3
- Software
  - Github repository:
    The Github repository should include all the code necessary to rebuild the system on similar hardware with step-by-step instructions on how to do so. Additionally, the training procedure code and instructions needed to perform it shall be present in the repository
  - Trained ML model weights for all relevant components:
    All the trained model weights need to be made available so retraining of the generalized models is not necessary.
- Class Requirements
  - Capstone Report and Poster
  - Video summary demonstration of the system's learning process over the training period
- Other
  - Live demonstrations of the system on trained subjects
    During the posters session, a live demonstration will be performed on the trained subjects. This demonstration should be available throughout the presentation period.

5 Conceptualization

5.1 Background research

5.1.1 Prosthetics

Limb difference either due to amputation or congenital malformations impacts an estimated 35–40 million people worldwide, and is expected to double by the middle of this century due to increased life expectancy. Prosthetics offer their users increased autonomy, self-determination, and participation in society, where they improve mobility, activity, and productivity and can provide access to education and work. However, prosthetic technology is insufficiently accessible or developed globally, where the cost is exorbitantly high for limited functionality. Additionally, even users with access to prostheses are often dissatisfied with the technology, leading to very high rejection rates (Piscitelli *et al.*, 2021). This proposal delineates a specific avenue of improvement for upper limb prosthetic technology to reduce prosthetic rejection rates and provide extra steps forwards towards a better future for people with limb difference worldwide (Piscitelli *et al.*, 2021).

There are 3 major classes of upper limb prostheses. Each class offers a different level of functionality and their price points can vary widely. We provide an overview of these types:

- Passive prosthetics: This class of upper limb prostheses allows for no active movement of any of the joints. They are lightweight devices that contain no motors or electronics and very few mechanical systems. Beyond simple push/pull tasks that can be performed to aid the intact arm with bimanual tasks, the purpose of these prostheses is largely cosmetic and to provide a natural look for users.

- Body-powered prosthetics: This type of prostheses is a terminal device that is operated typically by a harness and cable that is connected to the patient's shoulders. Movements of the upper arm, shoulder, and chest are captured by this system and transferred to the cable system. This can be used to operate a simple mechanism (such as opening/closing a hook or hand). While this offers more functionality than passive prostheses, this class of prostheses does not typically include motors or actuators. Furthermore, it can take some time for users to become accustomed to the mapping between upper arm movement and the prosthetic's mechanical response, as achieving intuitive control is a more difficult task with body-powered prosthetics.

- Myoelectric prosthetics: This class of prostheses is distinct from the other types in that they involve motors and batteries on board to power the movement of the device–opening up a vastly wider range of functionality and actions that can be performed by the wearer. These prostheses can mold to various high degrees of freedom (DoF) gestures and offer more intuitive control that corresponds with the wearer's intrinsic muscle activity. At the same time, the use of wearable electronics allows for the design of complex control algorithms that can address some of the insufficiencies in modern prosthetics. Myoelectric prosthetics can be further categorized into invasive and non-invasive myoelectric devices that we outline:
  - Non-invasive myoelectric prosthetics: These devices are designed to detect and use electrical signals from muscles without the need for surgical implants or invasive procedures. Non-invasive myoelectric prosthetics rely on surface electromyography (sEMG) sensors that are placed on the skin over the residual limb to capture muscle signals that can be decoded to detect the user's intention (through gesture classification or finger-specific regression).
  - Invasive myoelectric prosthetics: These devices are designed to detect and use electrical signals from muscle after invasive procedures to implant electrodes or sensors directly into the residual muscle or nerve tissue. As surface electromyography can suffer from various limitations arising from the growth of tissue, material deterioration, sweat, noisy signals, implanted electrodes provide less noisy signals (that can, at times, be amplified) to allow for improved gesture recognition; however, for many patients, invasive procedures would not be ideal or suitable, so the design of non-invasive myoelectric prosthetics has been an equally important goal for engineers and researchers alike.

5.1.2 Commercial Landscape

Table I: Summary of some available commercial upper limb myoelectric prosthetic devices and their features.

Parent Company	Product	sEMG	Vision
Open Bionics	Hero Arm	6 Gestures + 180 deg wrist rotation	No
Fillauer	Taska CX	23 Gestures	No
Ossur	iLimb Quantum	36 gestures	No
Psyonic	Ability Hand	32 gestures	No
Aether Biomedical	Zeus	12 Gestures	No

As mentioned earlier, the research landscape mostly comprises controlled experiments testing individual features rather than a complete control system. Furthermore, the commercial landscape is even further behind when it comes to control technologies. Table I summarizes the features of many upper-limb prosthetic devices. Practically none of the devices employ shared control using multiple different input modalities, though a few automate the process of holding grips to reduce the cognitive load associated with maintaining a specific muscle activity pattern. Practically no devices rely on regression control rather than gesture classification, and the number of gestures is lower than some of what can theoretically be achieved. With this in mind, this capstone project is unique in that it attempts to provide a viable commercializable solution that merges many different research directions to capitalize on the benefits of each to produce an intuitive and reliable upper limb prosthetic control system.

5.1.3 Prosthetic Control Strategies

There are many different control strategies within the domain of prosthetic control with varying degrees of reliability and intuitiveness. This section provides an overview of the most common of these approaches both in the research and commercial domains, weighing the advantages and disadvantages of each strategy:

- Gesture Classification:

  Prosthetic control can be framed as a classification task, where input sEMG patterns are mapped to predefined gestures in the prosthetic. There are two main subclasses within this approach: discrete actions performed separately and combined actions performed continuously. The former approach requires subjects to perform gestures starting from an initial "rest" state between the actions to facilitate the labeling and segmentation of data during model training, and is currently the most explored approach. While this approach boasts high offline recognition accuracy, online control during the transition stage exhibits less than ideal results as it ignores the switching between actions by necessitating a defined transitory rest state. The second less researched approach of combined actions performed continuously allows for more natural execution of different hand movements, combinations, or transitions, where "there is no strict requirement for subjects to return to the initial state between actions". This greatly increases the difficulty of the classification task, and current researchers are still exploring how to effectively recognize the movement intentions of different action combinations or transitions. Furthermore, given that sEMG signals have high inter-subject variability, gesture classification algorithms are trained on a subject-specific basis to achieve high accuracies (at times upwards of 97%) as compared to the more challenging task of creating generalized models or using transfer learning to finetune a generalized model on a subject-specific basis. While involving gesture classification techniques is an effective approach to account for users performing high DoF movements, the classification task often maps input sEMG patterns to a single gesture prediction from a set of movements; the outcome of this is a non-compliant invariant gesture that is performed identically without regard to the goal (e.g. grasping a particular object) unless other forms of control like regression or shared control are used.

- Regression - Simultaneous and Proportional Control:

  The second control paradigm attempts to predict the individual continuous states associated with the arm DoFs such as joint angles, positions, torques, or accelerations. This approach would allow for more intuitive and nuanced control for myoelectric prosthetics, though labeling the data requires additional components that can measure all the relevant hand states such as vision hand state recognition or force sensors. Other issues that need to be resolved are the limited number of DoF provided and the difficulty to train a subject-specific model on the prosthetic user especially if neither arm is complete, where there isn't really a way to measure the hand state.

- Shared Control:

  The shared control strategy, as the name suggests, involves the collaborative work of the controller with the user for reducing the cognitive fatigue associated with the complete reliance on the user. The shared control strategy involves semi-autonomous reliance on external modalities aside from the user's muscle activity to predict the intended action and perform the optimal gesture to execute it. For example, computer vision models can take scenes captured from glasses- or prosthetic-mounted cameras to discern possible objects of interest and the optimal grasp to carry the object. Other modalities such as eye tracking or pressure sensors at the tips of the prosthetic fingers can also be used in determining the user's intention. Studies have shown that this approach can successfully alleviate the cognitive load associated with using a prosthetic without losing performance quality or complete user autonomy.

5.1.4 Intention Prediction - sEMG & EEG

Intention prediction is a critical component of advanced prosthetic control systems, aiming to accurately interpret the user's desired movement and translate it into effective prosthetic actions. Both surface electromyography (sEMG) and electroencephalography (EEG) are widely studied modalities for this purpose, offering unique advantages in capturing the neural or muscular signals associated with motor intention.

Surface electromyography (sEMG) has become a prominent tool for intention prediction in prosthetics due to its ability to non-invasively capture muscle activation signals. These signals, originating from the residual limb in amputees or intact muscles, provide valuable insight into motor intention.

sEMG signals are typically acquired through electrodes placed on the skin, and can be collected using either sparse sEMG sensors or High-Density sEMG (HDsEMG) arrays. HDsEMG offers higher spatial resolution, allowing for more detailed mapping of muscle activity. The signal quality is influenced by factors such as skin impedance, electrode positioning, and muscle fatigue, necessitating preprocessing techniques like filtering, rectification, and normalization to enhance signal reliability.

Two primary approaches are used for sEMG-based intention prediction:

- Regression-Based Control: This approach focuses on mapping sEMG signals to continuous outputs such as joint angles, speed, or torque of the prosthetic device. Regression models are trained to capture the complex relationship between sEMG signals and desired prosthetic movements.

- Classification-Based Control: This approach aims to identify discrete gestures or movements from a predefined set by analyzing muscle activation patterns. Machine learning models, trained on labeled sEMG data, allow for the recognition of complex gestures and enable multi-degree-of-freedom control.

As for Electroencephalography (EEG), it offers a complementary approach by directly measuring brain activity associated with motor planning and execution. EEG signals, captured non-invasively using electrodes placed on the scalp, reflect neural activity in regions such as the motor cortex. These signals are particularly useful for capturing high-level intention in cases where muscle activity is limited or absent.

EEG-based systems rely on decoding patterns in neural oscillations, such as motor-related potentials or event-related desynchronization (ERD), to predict user intention. Similar to sEMG, EEG signals require extensive preprocessing to reduce noise and extract relevant features. Advanced algorithms, including deep learning models, are increasingly employed to improve the accuracy and robustness of intention prediction from EEG.

5.1.5 Vision-Based Prosthetic Control

Computer vision has emerged as a powerful modality for prosthetic control, leveraging advances in imaging and machine learning to enable intuitive and precise interactions. Unlike traditional bioelectric modalities like surface electromyography (sEMG) or electroencephalography (EEG), computer vision relies on visual data to interpret user intent and the surrounding environment, expanding the functional capabilities of prosthetic devices.

Computer vision systems in prosthetics use cameras or optical sensors to capture visual information, which is processed to infer the user's intended actions or adapt to environmental conditions. This modality is particularly valuable in scenarios where contextual awareness is critical, such as grasping objects of varying shapes and sizes or navigating complex environments.

Computer vision contributes to prosthetic control through several mechanisms:

- Object Recognition and Grasp Planning:
  - Vision-based systems enable the identification of objects in the user's vicinity. By recognizing the size, shape, and orientation of objects, these systems can assist in selecting the appropriate grasp type for a prosthetic hand.
  - Advanced algorithms, such as convolutional neural networks (CNNs), are widely used for object detection and classification, allowing prosthetic devices to autonomously adjust to diverse tasks.
- Gesture and Pose Recognition:
  - Computer vision can analyze body movements or hand gestures to interpret the user's intention. By tracking landmarks on the residual limb or intact hand, systems can infer desired actions and translate them into prosthetic movements.
  - Optical motion tracking, combined with deep learning models, has been employed to achieve high accuracy in gesture recognition, even under varying lighting or background conditions.
- Environment and Context Awareness:
  - Prosthetics equipped with computer vision can interpret the surrounding environment to enhance functionality. For example, vision-based navigation aids users in obstacle avoidance and pathfinding, critical for lower-limb prosthetics.
  - Context-aware systems can also integrate environmental cues to optimize control strategies, such as adjusting grip force based on the perceived fragility of an object.

5.1.6 sEMG/EEG and Vision Fusion

There are many benefits to multimodal prosthetic control systems that combine vision and intention-encoding biosignals like sEMG or EEG. Such systems leverage the complementary strengths of each modality: bioelectrical signals provide direct, real-time user input, while vision offers contextual awareness and adaptability. However, the design of these fusion strategies must carefully consider not only technical challenges but also the user's sense of autonomy and control.

Real-time processing of high-resolution visual data demands significant computational power, often necessitating optimization techniques or specialized hardware like GPUs. To address these challenges, fusion strategies may adopt a soft real-time reliance on vision modalities, where visual data is processed and updated less frequently than biosignals. This allows the bioelectrical signals (e.g., sEMG and EEG) to form the hard real-time component of the control system, ensuring immediate responsiveness while offloading computationally intensive tasks to the vision modality at a slower pace. This asynchronous update cycle can reduce control delay and improve the system's overall efficiency without compromising functionality.

Vision systems, while powerful, are susceptible to external factors such as variable lighting, occlusions, and environmental complexity. Biosignals like sEMG and EEG can provide redundancy in these situations, ensuring reliable operation even when visual data is partially or entirely unavailable. By fusing these modalities, the system can adapt dynamically to changing conditions, maintaining performance across diverse environments.

Beyond technical considerations, multimodal prosthetic systems must address the user's sense of agency and satisfaction. While fully autonomous systems that heavily rely on vision can minimize cognitive load by automating tasks, they may inadvertently diminish the user's feeling of control over their prosthetic device. For many users, the sense of direct involvement and control is not just a functional necessity but also a psychological factor that influences their acceptance and satisfaction with the prosthesis.

To strike this balance:

- Shared Control Strategies:
  - Instead of full autonomy, shared control models can allow the user to initiate and guide actions, while the system provides assistance or fine-tuning based on visual and environmental data. This approach ensures the user remains an active participant in the control process.
- Customizable Autonomy Levels:
  - Providing users with the ability to adjust the level of autonomy in their prosthetic system can enhance satisfaction. For example, in tasks requiring precision, the system might offer greater assistance, while simpler tasks might rely more heavily on direct user control.

5.1.7 Domain Incremental Online Learning

Domain Incremental Learning (DIL) is a subfield of continual learning where a model is trained sequentially on data from different domains (contexts or distributions). Unlike task-incremental learning, where task boundaries are explicitly labeled, domain incremental learning focuses on adapting to new data distributions without clear indications of the domain switch. This approach is especially relevant in online learning settings, where data arrives in a continuous stream.

Key Concepts:

- Domain Shift:
  - Refers to the change in the data distribution across different domains.
  - Examples include environmental changes (e.g., lighting in computer vision) or varying speaker accents in audio processing.
- Online Learning:
  - Involves updating the model incrementally with each new data point or small batch rather than retraining on the entire dataset.
  - The model needs to efficiently learn from new data without revisiting old data.
- Challenges:

- Catastrophic Forgetting: The model tends to forget knowledge from earlier domains when trained on new domains.
- Knowledge Transfer: Balancing adaptation to new domains while retaining general knowledge from previous ones.
- Efficiency: Online settings demand computational and memory efficiency.

Approaches to Domain Incremental Online Learning

- Regularization-Based Methods:
  - Penalize changes to important parameters to retain knowledge of earlier domains.
  - Examples: Elastic Weight Consolidation (EWC), Synaptic Intelligence (SI).
- Replay-Based Methods:
  - Store and replay a subset of data from earlier domains to reinforce prior knowledge.
  - Memory-efficient variants use generative models to synthesize previous data.
- Dynamic Architectures:
  - Expand or modify model architecture dynamically to accommodate new domains.
  - Examples: Progressive Neural Networks.
- Meta-Learning:
  - Leverages prior experience to quickly adapt to new domains with minimal forgetting.
  - Focuses on learning to learn across domains.

5.2 Concept Generation with Morphological Chart

Table II: Morphological chart for concept generation in the 3 main problems

Intention Prediction	Camera	Fusion Strategy
sEMG	Monocular	Online learning fusion
EEG	Stereo	Uncertainty-aware fusion
	360°	Immediate Feature Fusion

Using the morphological chart in Table II, the following concepts out of the 18 possible combinations were generated:

1. sEMG + Monocular + Online Learning Fusion (base)
2. sEMG + 360° + Immediate Feature Fusion
3. EEG + Stereo + Uncertainty-aware fusion
4. EEG + Monocular + Immediate Feature Fusion

5.3 Concept Selection with Decision Matrices & Pugh Chart

To select the best concept out of the 4 generated concepts, we went through a gradual decision process where decision matrices/criteria were generated for each sub-problem, and then a pugh chart was used to evaluate the 4 concepts against each other.

5.3.1 Intention Prediction Decision Matrix

For intention prediciton, the following criteria were taken into account in the decision:

1. Signal Variability:

   Prosthetics need to be regularly taken off and on again mutliple times within the same day, and the signal quality should not vary greatly between the many donning and undonnings of the prosthetic. This criterion evaluates how variable the signal for this situation.

2. Wear Comfort:

   Prolonged wearing of an uncomfortable sensor suite is highly undesirable. This criterion is therefore essential when evaluate the sensor used for intention prediction

3. Signal Quality:

   An essential factor is how easy it is to extract the features out of the sensed signal. This factor assesses the amount of preprocessing necessary to extract utility out of the signals.

4. Cost:

   Given the budget constraints of this project, the cheaper sensor would be ideal if the quality is sufficient.

Given that EMG ranks better on all these aspects than EEG, EMG is preferred over EEG in the final solution.

5.3.2 Camera Decision Matrix

We have three proposed camera options:

- A monocular camera is a single-lens camera that captures flat, 2D images or video from one perspective. It does not directly perceive depth and is unable to provide more than RGB values for each frame it captures.
- A stereo camera consists of two lenses, separated by a fixed distance (baseline), simulating human binocular vision. Using triangulation or disparity mapping, this system can compute depth and produce a 3D understanding of the scene.
- A 360° camera typically uses two or more ultra-wide-angle lenses (e.g., fisheye lenses) to capture a complete spherical view of the surroundings.

For evaluating the used camera, the following criteria were taken into account in the decision matrix:

1. Field of View (FoV):

   Objects not within the FoV of the camera would be inaccessible to the vision modality, and thus may limit the user's ability to interact with peripherally located objects.

2. Depth

   Having the depth information already present would save on the computational time taken to run depth estimation models to generate depth maps.

3. Ease of Integration (EoI):

   In a final usable design, the camera should seamlessly integrate into the system without any awkward work-arounds given the system would be worn for prolonged periods of time

4. Cost:

   Given the budget constraints of this project, the cheaper cameras would be ideal if the quality is sufficient.

Table III: Decision Matrix for camera concept selection with Monocular as base

Camera					
	FoV (1)	Depth (1)	EoI (3)	Cost (1)	Weighted Score
Monocular (base)	0	0	0	0	0
Stereo	0	+1	-1	-1	-3
360°	+1	-1	-2	-2	-8

Table IV: Decision Matrix for camera concept selection with Stereo as base

Camera					
	FoV (1)	Depth (1)	EoI (3)	Cost (1)	Weighted Score
Stereo (base)	0	0	0	0	0
Monocular	0	-1	+1	+1	+3
360°	+1	-2	-1	-1	-5

Table V: Decision Matrix for camera concept selection with 360° as base

Camera					
	FoV (1)	Depth (1)	EoI (3)	Cost (1)	Weighted Score
360° (base)	0	0	0	0	0
Monocular	-1	+1	+2	+2	+8
Stereo	-1	+2	+1	+1	+5

5.3.3 Fusion Strategy Decision Matrix

We have three proposed fusion strategies:

- Online learning fusion initiates gesture prediction using the best estimate that is provided by the vision modality, but provides the user with the full agency to alter the gesture using the EMG modality. Over time, the vision modality will adapt to the user's particular preferences for grasping certain objects derived from the EMG modality.
- Uncertainty-aware fusion relies on uncertainty quantification techniques to define how confident the system should be in either/both modalities and how dependent it should be on either for control at any point of time.
- Immediate Feature fusion defines a classical multi-modal network that takes both the vision and EMG modality as input and provides a singel gesture classification at discrete time steps.

For selecting the fusion strategy, the following criteria were taken into account in the decision matrix:

1. Adaptability to Users:

   Is there a component of the fusion strategy that varies from subject to subject to produce optimal performance for specific users?

2. Response time:

   Does the system allow for soft real-time for slower modalities or do all modalities need to be processed simulatenously?

3. User autonomy:

   Ideally, the user should be able to overrule the decision of the vision module if they so choose. Does the fusion strategy allow for that?

4. Generalizability to additional modalities:

   How different would the fusion component look if new modalities are added?

Table VI: Decision Matrix for fusion strategy concept selection with Online Learning Fusion as base

Fusion Strategy					
	Adaptability (3)	Response time (2)	User autonomy (2)	Generalizability (1)	Weighted Score
Online Learning Fusion (base)	0	0	0	0	0
Uncertainty-aware Fusion	-1	-1	-1	1	-6
Immediate Feature Fusion	-1	-1	-1	1	-6

Table VII: Decision Matrix for fusion strategy concept selection with Uncertainty-aware Fusion as base

Fusion Strategy					
	Adaptability (3)	Response time (2)	User autonomy (2)	Generalizability (1)	Weighted Score
Uncertainty-aware Fusion (base)	0	0	0	0	0
Online Learning Fusion	+1	+1	+1	-1	+6
Immediate Feature Fusion	0	0	0	0	0

Table VIII: Decision Matrix for fusion strategy concept selection with Immediate-Feature Fusion as base

Fusion Strategy					
	Adaptability (3)	Response time (2)	User autonomy (2)	Generalizability (1)	Weighted Score
Immediate Feature Fusion (base)	0	0	0	0	0
Uncertainty-aware Fusion	+1	+1	+1	-1	+6
Online Learning Fusion	0	0	0	0	0

5.3.4 Pugh Chart

Along with the decision matrices, a final decision is made through integrating these results into a pugh chart (Table IX) comparing the aforementioned concepts:

1. sEMG + Monocular + Online Learning Fusion (base)
2. sEMG + 360° + Immediate Feature Fusion
3. EEG + Stereo + Uncertainty-aware fusion
4. EEG + Monocular + Immediate Feature Fusion

We use a mix of the previous criteria to evaluate using the Pugh Chart, namely:

1. Signal Reliability & Quality
2. Comfort & Integration
3. Adaptability & User Autonomy
4. Depth / Env. Awareness
5. Cost

Table IX: Pugh Chart Comparison of Selected Concepts

	Criteria					Total
Concept	1	2	3	4	5	
1 (Base)	0	0	0	0	0	0
2	0	−	−	0	−	-3
3	−	−	−	+	−	-3
4	−	−	−	+	−	-3

As a result, all the concept selection charts point at the same solution: Concept 1 (sEMG + Monocular + Online Learning Fusion)

6 Modeling, Simulation, and Optimization Plan / Experimental Plan

6.1 Synthetic Data

In case any fine-tuning of pretrained vision models is necessary, BlenderProc2 can be utilized to generate high-quality synthetic datasets that closely mimic the characteristics of the target domain. This tool allows the creation of diverse, photorealistic data, including RGB images, depth maps, segmentation masks, and normal maps, which can be customized to specific tasks or environmental conditions. By leveraging BlenderProc2, datasets can be enriched with variations in object placement, lighting, textures, and camera perspectives, ensuring comprehensive coverage of scenarios the model might encounter. This synthetic data not only complements limited real-world datasets but also enhances the adaptability and robustness of pretrained models when fine-tuned for specialized tasks.

6.2 Model Complexity

While the proposed control system is tested on an Nvidia RTX 4060 with an Intel i7 12th generation CPU, the ultimate goal is to deploy the system on a significantly less powerful embedded system within a prosthetic arm. This introduces the challenge of reducing computational requirements without compromising the performance or reliability of the control system.

Upon achieving satisfactory results on the test hardware, several strategies will be implemented to optimize the model for deployment on resource-constrained devices. These strategies include but are not limited to:

- Pruning:

  This involves removing redundant or less critical parameters and connections from the neural network while preserving its overall performance. Structured pruning techniques, which remove entire neurons or layers, and unstructured pruning, which targets individual weights, will be explored to determine the best approach for the system's architecture.

- Knowledge Distillation:

  A smaller, simpler "student" model is trained to mimic the behavior of a larger, complex "teacher" model. This technique is particularly useful for transferring the knowledge of high-performance models into a lightweight model suitable for embedded devices.

- Quantization:

  This involves reducing the numerical precision of the model's parameters and operations (e.g., from 32-bit floating-point to 8-bit integers). Quantization-aware training will be utilized to ensure minimal loss in accuracy while significantly improving computational efficiency and reducing memory usage.

- Model Compression Libraries: Open-source libraries such as TensorRT or ONNX will be employed to further compress and optimize the models for deployment.

The impact of these optimizations will be evaluated through benchmarking on simulated low-power environments, ensuring the system maintains its real-time capabilities and accuracy while adhering to resource constraints.

6.3 Learning Speed

The system's ability to adapt to user-specific requirements during the training phase is critical for successful deployment. Strategies to accelerate the learning process will focus on minimizing user frustration and reducing the time needed to achieve satisfactory performance. Key considerations include:

- Incremental Online Learning:

  The system will incorporate an online learning module capable of adapting to the user's unique muscle activation patterns or preferences during use. By updating the model weights incrementally rather than requiring full retraining, learning speed can be improved significantly.

- Pretraining with Generalized Models:

  A generalized model will be pretrained on diverse datasets to provide a robust starting point. Fine-tuning this model with user-specific data will be quicker than training from scratch.

- Active Learning:

  The system will prioritize gathering informative data points during user interactions, ensuring that only the most relevant information is used for training. This reduces the overall dataset size needed for effective adaptation.

- Learning Rate Scheduling:

  Dynamic adjustment of the learning rate during training will ensure fast convergence in the early stages while maintaining stability as the model approaches optimal performance.

Simulated users and real-world experiments will validate these strategies, with metrics such as convergence time, user satisfaction, and task success rate guiding iterative improvements.

6.4 Real-time Control

Achieving real-time control is paramount for the proposed system, as delays in processing can compromise usability and safety. The real-time performance of the system will be assessed by measuring the total delay introduced by each component, including signal acquisition, preprocessing, intention prediction, and actuation. Strategies to ensure real-time operation include:

- Pipeline Optimization:

  The control pipeline will be profiled to identify bottlenecks. Critical components will be optimized using techniques such as multithreading or hardware acceleration.

- Asynchronous Processing:

  Non-critical tasks, such as vision processing, will be performed asynchronously to prioritize hard real-time components like sEMG signal processing and actuation.

- Latency-Aware Design:

  The fusion strategy will be designed to minimize the impact of soft real-time components on overall latency. For instance, heuristic updates from vision data can be applied at longer intervals without affecting immediate control.

- Transient Phase vs. Steady-State sEMG Signal Analysis:

  - The transient phase of sEMG signals, occurring at the onset of muscle activation, provides crucial information about the user's intention. By focusing on the transient phase, which exhibits sharp, identifiable signal changes, the system can detect intention faster than waiting for the steady-state phase, where the signal stabilizes.
  - Leveraging machine learning models trained to interpret transient phase data will allow the control system to predict movements almost instantaneously, reducing latency significantly.
  - While the steady-state phase may provide a more stable signal for prolonged actions, transient-phase analysis ensures that the system can react in real time, especially in scenarios requiring quick responses.

Experimental validation in a virtual environment and subsequent testing on hardware prototypes will verify the system's ability to maintain real-time responsiveness under practical conditions. Success criteria will include minimal task delay, smooth prosthetic operation, and user-reported satisfaction with control dynamics. Metrics such as time-to-response and prediction accuracy during transient phases will be key indicators of system performance.

6.5 Implemented Simulation and Testing

6.5.1 Vision Controller

The vision controller went through multiple phases of experimentation. Below is an outline of our experimentation along with some results for each tested component:

- Running ML models in Unity:

  To begin with, experimentation into different techniques of running ML inference for Unity to decide on all possible implementation pipelines.

  - Unity Sentis:

    Unity offers an internal solution to running ML models in ONNX format with support for GPU usage and matrix operations. The solution is limited however in the flexibility of pre- and post-processing the outputs of these models when compared to more developed python libraries. Additionally, the conversion to ONNX may be a problematic constraint given some models may use modules that aren't available on onnx.

- UDP connection to external python process:

  This solution is a more realistic implementation for an actual prosthetic device given the ability to offload the more intense computation away from the prosthetic arm itself. This pipeline was also implemented to

- Object Detection:

  Our first approach to identifying objects within the scene was through object detection and classification. We tested the zero-shot detection capabilities of YOLO11, as shown in Figure 5, within a simulated environment in Unity which doesn't perfectly match with its training set. The results were promising, but due to the limitation of object detection within a predetermined set of classes, object detection was ruled out as the tool of choice. This component was helpful nonetheless in setting up Unity Sentis.

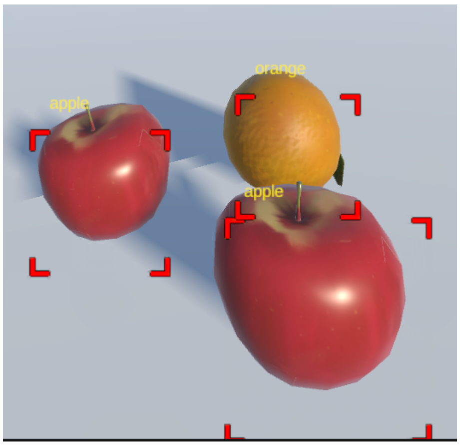

Figure 5: YOLO11 run on Unity Sentis for object recognition.

- Instance Segmentation:

  Upon deciding that object recognition is not the ideal tool for this project, we searched for zero-shot promptable instance segmentation models, to converge on Meta's Segment Anything Model 2 (SAM2). Through different prompts, either given as manually inputted points or bounding box regions or through an automatic segmentation procedure, SAM2 can produce different levels of segmentation even for the same object as shown in figure 6.

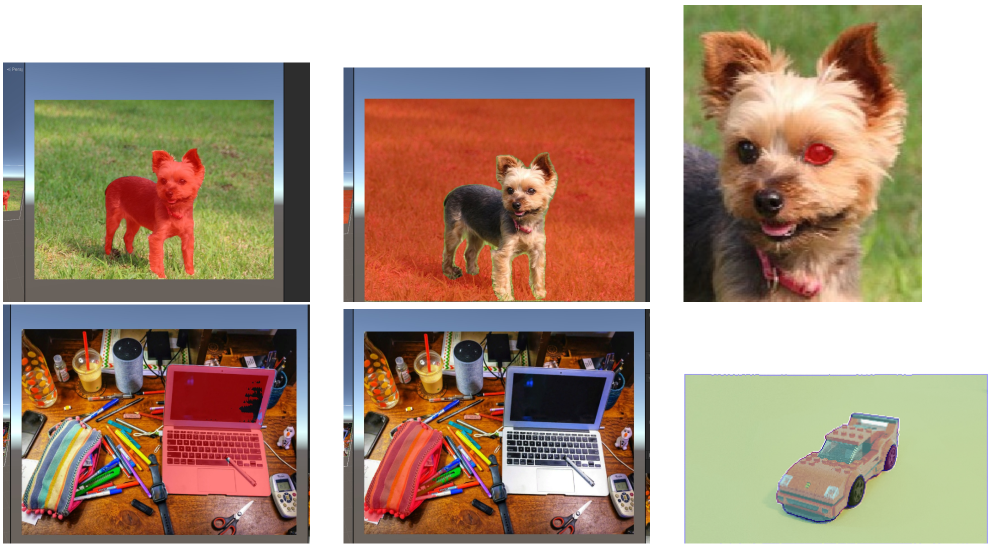

Figure 6: SAM2 capabilities demonstrated with different prompts including points, regions, and automatic segmentation. Ran both through Unity Sentis and python through UDP connection

- Depth Estimation:

For Monocular Depth Estimation (MDE), the DepthAnythingV2 model is used, which is trained on a mix of indoors and outdoors data. Below is an example of the output of DepthAnythingV2 on an image of a toy car:

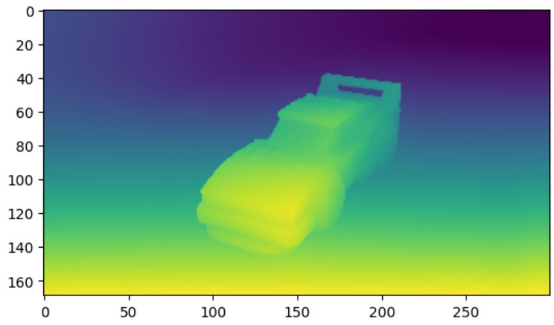

Figure 7: MDE using DepthAnythingV2

- 3D Object Reconstruction:

  Given the camera field of view, depth image, and segmentation mask of an object, a 3D point cloud can be reconstructed from the input RGB image, after which algorithms such as Random Sampling Consensus (RANSAC) or other fitting techniques can be used to decide the nearest primitive shape to an object and its geometric features (radius, height, width, etc). Below in Figure 8 is an example of the

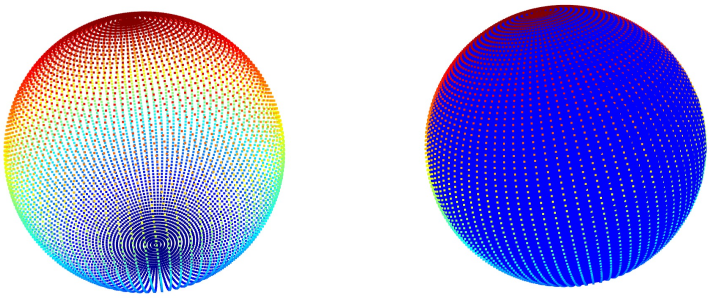

Figure 8: Reconstructing best fit sphere from spherical point cloud

- Entire Pipeline

  Aside from the final step of translating the reconstructed object into a gesture, the entire pipeline using UDP connection has been implemented, and snippets of each intermediary step are shown below in Figure 9. It is difficult to give an estimate of the total control delay within the current implemented pipeline as there are still some dependency issues causing a much slower performance than the sum of each individual component.

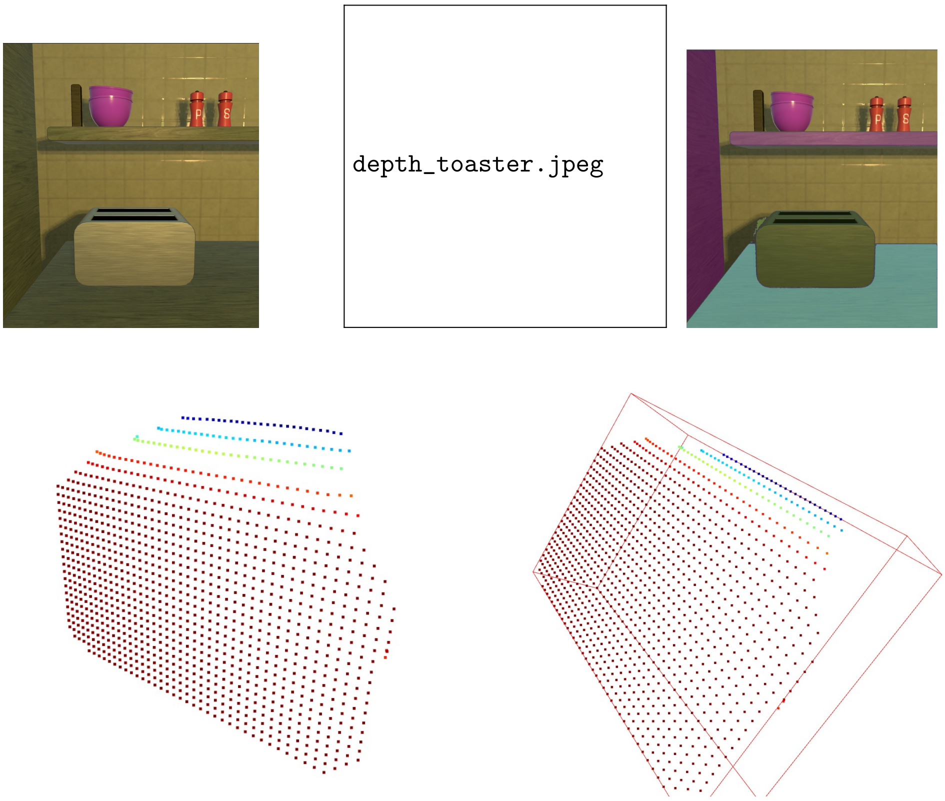

Figure 9: Entire pipeline implemented through python UDP connection receives raw RGB image from Unity and processes it into best fitting shape

6.5.2 sEMG Controller

We have designed and implemented several networks to begin testing the viability of EMG signals collected using the MindRove for gesture detection. Figure 11 presents a CNN architecture from (Loi *et al.*, 2022) for detecting 6 possible hand gesture outlined in Figure 10. Additionally, Figure 12 presents a BiLSTM architecture from (Loi *et al.*, 2022) for detecting joint angles from the sEMG signals. Both of these networks have been implemented and we describe their testing procedure next.
All the implementation is completed in Python using the the PyTorch machine learning framework for training. We leverage NYU's High-Performance Computing (Greene) services for training.

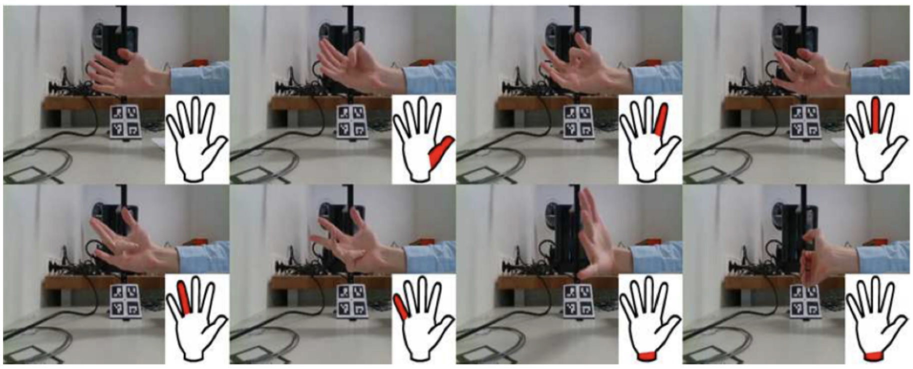

Figure 10: The target hand gestures predicted.

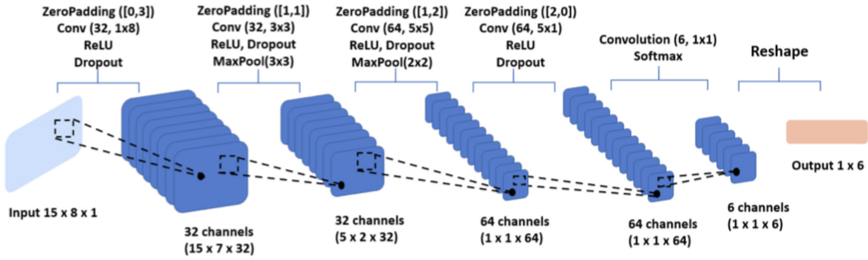

Figure 11: CNN architecture for classifying hand gesture from only sEMG signals.

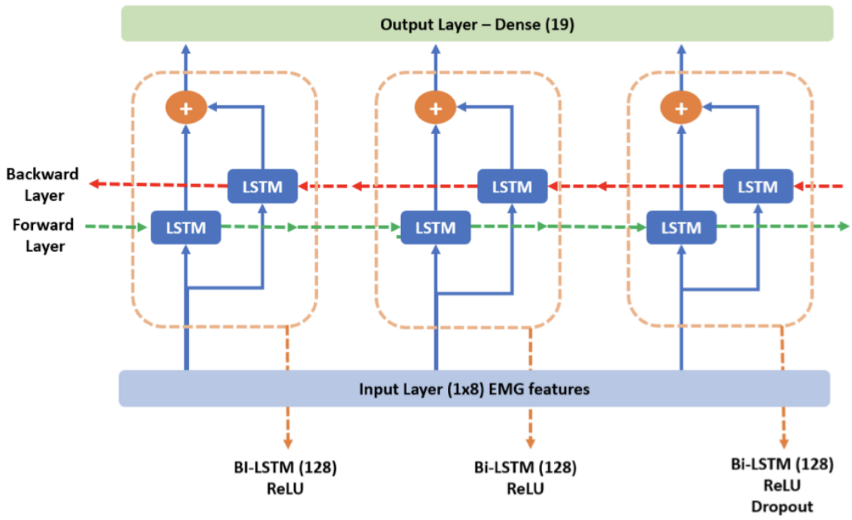

Figure 12: BiLSTM architecture for classifying joint angles from only sEMG signals.

(Köllőd *et al.*, 2022) provides a dataset of 6 subjects who performed the gestures in Figure 10. We test the CNN network described in Figure 11 on this dataset. We preprocess and filter the data as recommended in (Loi *et al.*, 2022) and (Köllőd *et al.*, 2022). We utilize three training approaches to test how this model performs when trained to perform well on one subject or a collection of subjects:

1. Subject-Specific Approach: In the subject-specific approach we train and test on a single subject. This is expected to produce models that have high testing accuracy as their training data was from the same distribution as testing data (same subject), which is indeed what we observe in Table X.

2. Generalized Approach: In the generalized approach, we pool all 6 subjects' training data together into one training set and we pool all 6 subjects' testing data together into one test set. We then train and test on those sets such that the CNN learns to recognize gestures from the set of subjects. While the performance of this model is expected to be poorer than subject-specific models, seeing as the testing data comes from the same distribution as the training data, the testing accuracy is still expected to be moderately high. Table XI matches those expectations where the performance is lower than each subject specific model but not by a huge margin.

3. Hidden Testing Subject (Leave-One-Out) Approach: In this approach we pool 5 subjects' training data together into one training set. We then train on this mixed training set and test on the 6th subject's testing data. This is classically referred to as the 'leave-one-out' approach. As expected, table XII shows that the performance of models trained in this fashion to have the poorest testing accuracy due to the out-of-distribution nature of the testing data. Completely hiding a subject's data is unrealistic in a practical setting, and usually some amount of callibration to the new subject's data is necessary to achieve acceptable performance. As shown in Figure 13, increasing the percentage of data coming from the new subject (right to left) results in better performance as expected.

Table X: Subject-Specific Training and Test Accuracy

Subject	Training Accuracy	Test Accuracy
0	98.3%	96.7%
1	96.4%	96.3%
2	99.1%	96.5%
3	97.5%	96.8%
4	98.0%	89.6%
5	94.4%	94.3%

Table XI: Generalized Training and Test Accuracy

Metric	Accuracy
Training Accuracy	92.6%
Test Accuracy	87.8%

Table XII: Training and Test Accuracy with Hidden Testing Subject (Leave-One-Out)

Testing Subject	Training Accuracy	Test Accuracy
0	94.3%	15.0%
1	85.7%	43.2%
2	93.3%	17.1%
3	87.2%	32.5%
4	95.7%	13.7%
5	96.1%	13.0%

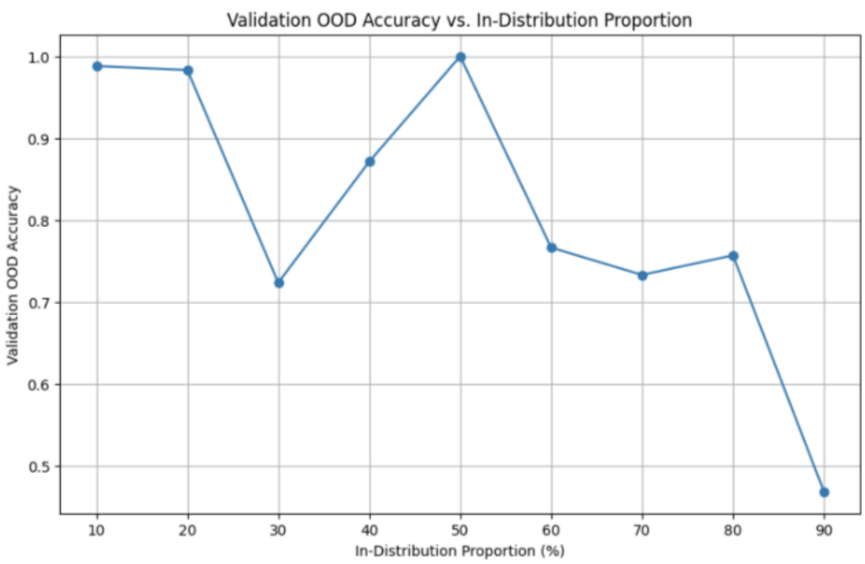

Figure 13: Validation Accuracy on Out-of-Distribution Subject against Proportion of Training Data that is In-Distribution

As for our own data collection, the pipeline for collecting our own data from the Mindrove armband has been set up in python through the Mindrove SDK. Alternatively, the data could be collected through the Mindrove Unity integration, though it is currently under maintenance due to recent errors. In the final system data could be captured either way, and inference can be performed through Unity Sentis or through a UDP connection to a python process.

7 Final Design Expected

The final design of the proposed prosthetic control system incorporates multimodal inputs, advanced processing pipelines, and real-world testing environments to ensure reliability and intuitiveness. The design follows the flowchart in Figure 14. Below, the design is broken into key components, each addressing specific functional goals.

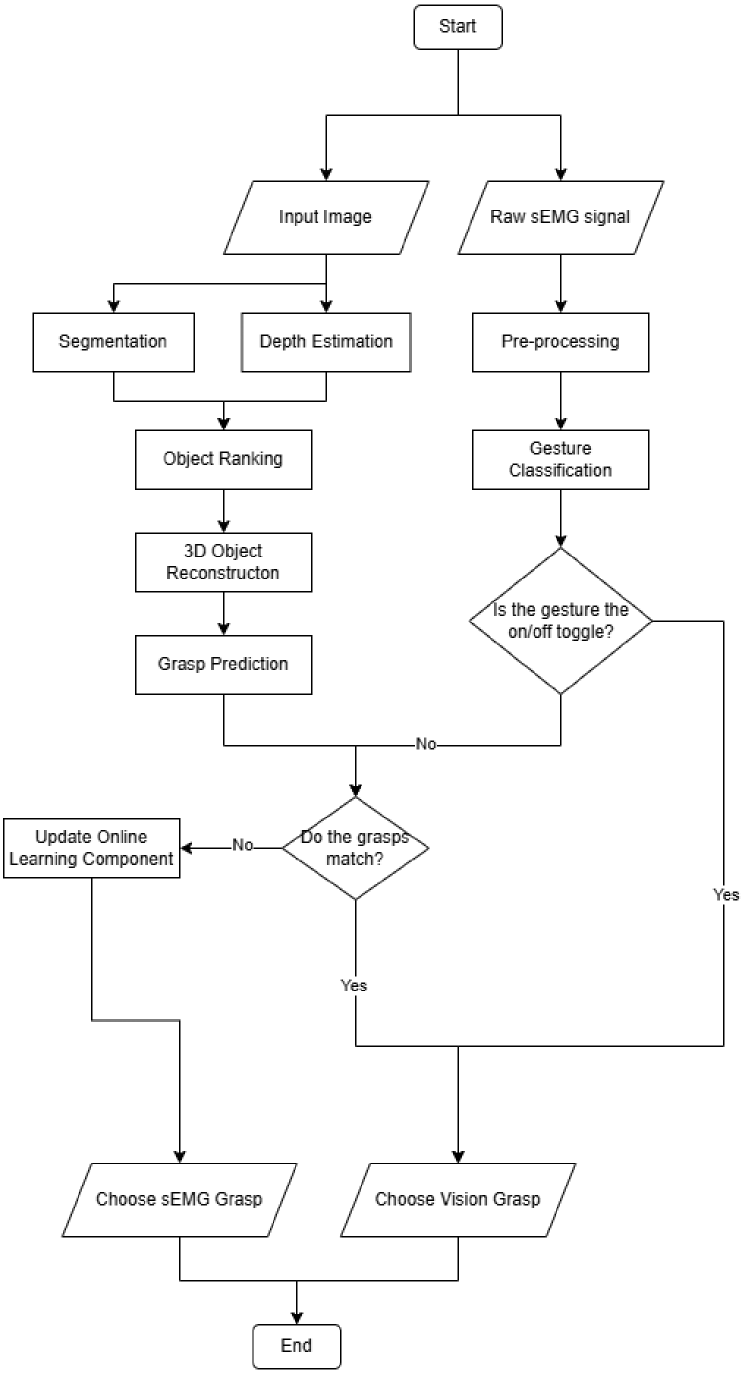

Figure 14: Proposed design flowchart.

7.1 Virtual Reality (VR) Simulation

Device: Meta Quest 3
Software: Unity Game Engine

The VR simulation environment replicates real-world scenarios to assess the system's performance across diverse contexts. Users will engage in daily interactions involving prosthetics within familiar settings, such as office spaces, kitchens, and gyms. These scenarios vary in terms of:

- Lighting Conditions: Evaluating system robustness under different levels of brightness and shadows.
- Object Interactions: Testing the grasping and manipulation of objects with varying sizes, textures, and complexities.
- Task Objectives: Clearly articulated goals for users (e.g., picking up utensils, typing, lifting weights) ensure task-specific assessment.

The VR testing suite is designed to provide extensive coverage of realistic conditions, offering quantitative and qualitative metrics for evaluation. The simulation aids in fine-tuning the system by identifying areas needing optimization and ensuring user adaptability and satisfaction.

7.2 sEMG Module for Intention Prediction

Hardware: Mindrove 8-channel sEMG armband **Approaches**: Classification and Regression

For intention prediction, the system leverages sEMG signals captured from the residual limb using the Mindrove armband. The design includes:

- Generalized Model Training:
  - The model is initially trained using datasets from publicly available sources and data collected by the team.
  - If available on time, the Mindrove Stargrip generalized gesture prediction model will be incorporated as a baseline.
- Subject-Specific Fine-Tuning:
  - Fine-tuning ensures adaptability to individual users, accounting for physiological variations such as muscle strength and residual limb size.
- Processing Pipeline:
  - Signal preprocessing steps, including filtering and normalization, follow guidelines from Mindrove's published studies.
  - A hybrid approach of classification (gesture prediction) and regression (continuous control of hand states) may be employed to maximize control versatility.

Figure 15: MindRove 8-channel EMG Armband

7.3 Vision Module

The vision system provides contextual awareness and enhances grasp planning through a sophisticated pipeline:

1. Input Acquisition:
   (a) A monocular virtual camera in the VR environment captures real-time video frames.
2. Object Segmentation:
   (a) The Segment Anything Model (SAM) is utilized for zero-shot instance segmentation, detecting objects of interest in the scene.
3. Depth Mapping:
   (a) Monocular depth estimation generates a depth map of the environment to locate objects spatially.
4. Probabilistic Ranking:
   (a) Objects are ranked based on factors including (but not limited to):
      i. Centrality in the field of view.
      ii. Proximity to the camera.
      iii. Arm motion trajectory.
5. The highest-ranked object is selected for further processing.
6. 3D Object Reconstruction:
   (a) Using single-view reconstruction techniques like RANSAC, a 3D model of the selected object is created based on the depth map and segmentation mask.
7. Gesture Mapping:
   (a) The object's primitive shape, size, and position inform the predicted grasp. These parameters are user-tunable through the online learning module, linking 3D model data to sEMG-driven gesture labels.

7.4 Fusion Strategy

To balance user autonomy and system assistance, the fusion strategy prioritizes the following:

- sEMG Priority: Muscle-driven predictions take precedence over vision-derived heuristics to preserve user autonomy and enable adjustments to grasp strategies.
- Semi-Autonomous Design:
  - The vision system aids decision-making but defers to the user for final control.
  - This shared control reduces cognitive load while maintaining a sense of control for the user.
- Online Learning:
  - Through domain incremental online learning approaches, the vision modality will use the sEMG as ground truth for what the user deems the optimal grasp and adjust its predictor accordingly.

8 Implementation Details

8.1 Discussion of Implementation Success

8.1.1 sEMG Module

For sEMG-based classification systems, we evaluated some of the successes and limitations of the approach outlined in 6, and explored alternate datasets and strategies for training our sEMG controller.

While (Köllőd *et al.*, 2022) is one of the only datasets available that is entirely collected with Mindrove (and therefore matches the hardware conditions we will be using in our implementation), the dataset suffers from several issues that limit its efficacy in different training set ups

1. **Signal Quality** We could see in XII that, depending on the subject that was being evaluated/tested on, a model trained on the remaining subjects could entirely underperform (as evidenced by the 43.2% test accuracy vs 13.0% depending on the testing subject).
2. **Limited Gestures** The target hand gestures that were performed and accounted for in the dataset are displayed in 10. Relying on a dataset that uses these one DoF gestures limits the practicality of any control algorithm produced by relying on it is difficult to use these models on predicting more complicated gestures (grasps, for example) that will be used to interact with different objects.
3. **Limited Subjects** This dataset was trained on 6 subjects, which limits its application to building a generalizable feature extractor that we can rely on for downstream classification on a larger set of gestures.

Consequently, we turned our eyes to the Ninapro DB1 dataset (Atzori *et al.*, 2015). This dataset is often used in sEMG hand gesture recognition literature for how reliable and comprehensive it is. Ninapro DB1 is often used because:

1. **Large Subject Pool:** The Ninapro DB1 dataset includes data from 27 subjects, providing a diverse range of muscle signals for robust analysis and generalization.
2. **Extensive Gesture Selection:** The dataset offers 52 different hand and wrist gestures, allowing for comprehensive testing of machine learning models in myoelectric control applications.
3. **High-Quality EMG Data:** Surface electromyography (sEMG) signals are recorded using 10 high-precision electrodes, ensuring reliable and accurate data collection for research and development.

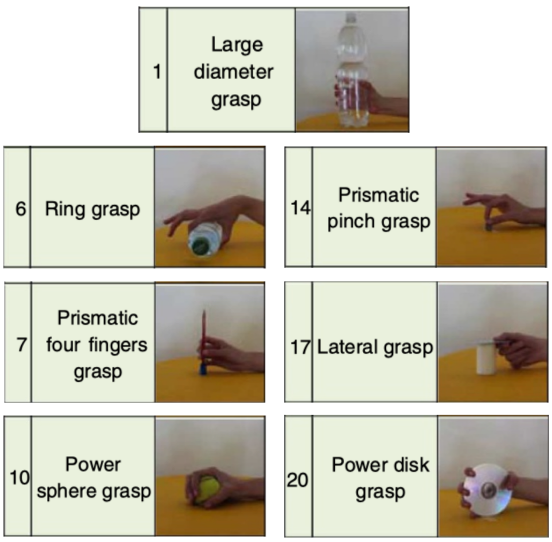

Figure 16: Ninapro Subset of Available Gestures Selected

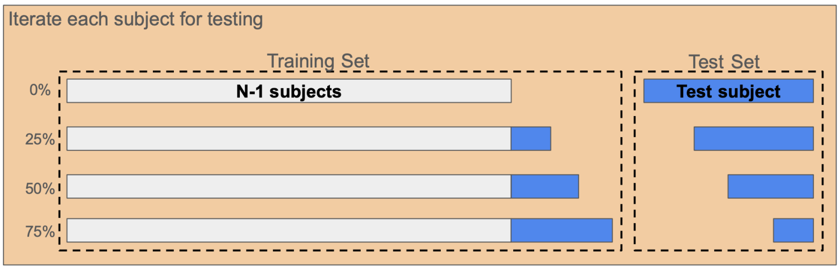

Figure 17: Ninapro Training Scheme

We decided to test out the same CNN model described in 6 on data from Ninapro DB1 that we collect and organize as follows:

1. **Subset of Subjects** From the 27 total subjects, we select the 19 subject that we can reliably use in training (as they are right-handed males).

2. **Subset of Gestures** From the 3 exercise sets and 52 different hand movements (gestures) that could be predicted, we select the gestures in 16. As opposed to the gestures in 10, these gestures are more practical for grasping and interacting with various objects in a physical space.

3. **Preprocessing** The raw surface electromyography (sEMG) data undergoes multiple pre-processing and validation steps to ensure valid signal processing and generalization across subjects. The main steps of the pipeline are as follows:

   (a) **Low-Pass Filtering:** A Butterworth low-pass filter (cutoff frequency of 1 Hz) is applied to smooth the signal and reduce high-frequency noise.

   (b) **Sliding Window Segmentation:** The EMG signals are segmented into overlapping windows of length T_w, calculated as:

   $$T_w = \frac{\text{window length in ms} \times f_s}{1000}$$

   where f_s is the sampling frequency (100 Hz). The step size for the overlapping windows is defined as:

   $$\text{Step Size} = T_w \times (1 - \text{overlap percentage}/100)$$

   Each segmented window is stored as a structured data record containing:

   - Stimulus ID
   - Repetition ID
   - EMG window data

   (c) **Subject-Wise Data Splitting:** The dataset is split into:

   - **Training Set:** Comprising data from $N - 1$ subjects.
   - **Test Set:** Comprising data from the single left-out subject.

   (d) **Min-Max Normalization:** To ensure consistency across subjects, all EMG windows are normalized using min-max scaling:

   $$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

   where X_{min} and X_{max} are computed from the training set only, and the same scaling is applied to the test subject.

4. **Training Scheme** We utilize the following modification on the classical Hidden Testing Subject (Leave-One-Out) Set-Up:

   - We cycle through the dataset, selecting one subject at a time for testing while using the remaining subjects for training.
   - A proportion of the test subject's data is incorporated into the training set to simulate varying levels of subject inclusion.
     - The proportions considered are: 0%, 25%, 50%, and 75%.

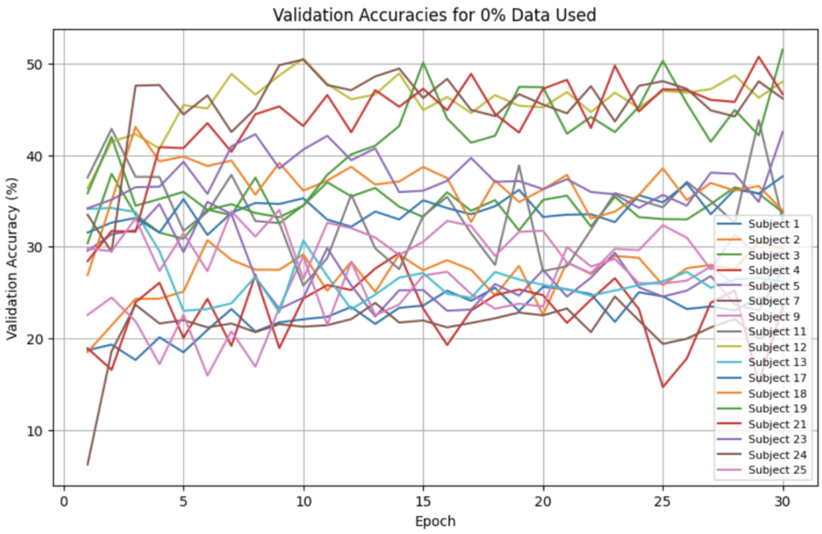

Figure 18: Ninapro Validation Results when 0% of Hidden Subject is Involved in Training

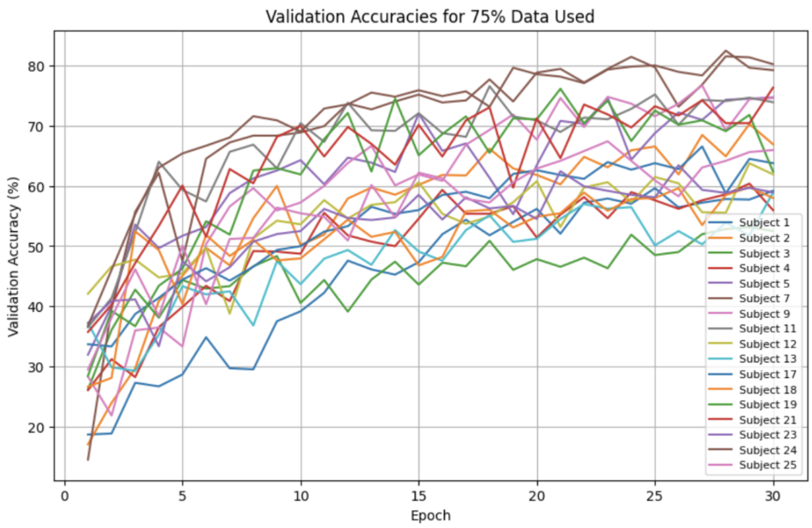

Figure 19: Ninapro Validation Results when 100% of Hidden Subject is Involved in Training

Proportion of Test Subject's Data in Training	Repetitions to Calibrate	Accuracy Range
0%	0	25-50%
25%	2-3	35-70%
50%	5	50-75%
75%	7-8	50-80%

Table XIII: Effect of training proportion on calibration and accuracy.

Naturally, we see in our results that, the greater the level of subject inclusion, the better the performance of our classifier on that hidden subject (which pointed us to the proportion of data needed to effectively finetune a model to a new, unseen subject). 18 indicates how, across all subjects, including 0% of the hidden subject's data in training leads to mediocre performance when testing on that hidden subject.

19 highlights the improved performance in testing that arises from including 75% of the hidden subject's data in testing (and this trend is alike across all subjects). These results are confirmed in table XIII. This investigation allowed us to determine that Ninapro is still a repetition-hungry dataset that requires a substantial amount of repetition from a hidden subject before it is effectively finetuned to that subject. To have to use 5+ repetitions to achieve a sub-optimal accuracy of 50-80% is no better than utilizing our own subject-specific model.

To pivot towards training and using our own subject-specific model, we developed a pipeline for recording data directly from the Mindrove that we can use for training. The recording script was implemented to perform the following:

1. **System Initialization:** The script initializes a connection with the MindRove board, prepares the session, and starts data streaming.
2. **Gesture Execution Protocol:**
   - A predefined set of hand gestures is performed.
   - Each gesture is executed multiple times (repetitions).
   - A rest period is included between different gestures to prevent fatigue and ensure clean signal acquisition.
3. **Countdown and Data Collection:** Before each repetition, a countdown is provided to prepare the user. The system then records EMG signals while the gesture is being performed.
4. **Data Handling and Storage:**
   - The script captures the real-time EMG signal data from the MindRove board.
   - The recorded signals are segmented into individual repetitions for each gesture.
   - The data is structured and stored in a dictionary format for future processing.
5. **Session Management and Cleanup:** Upon completion, or if interrupted, the script properly stops the data stream and releases system resources.
6. **Saving Processed Data:** The collected EMG data is saved as a NumPy binary file for further analysis, enabling easy retrieval and post-processing.

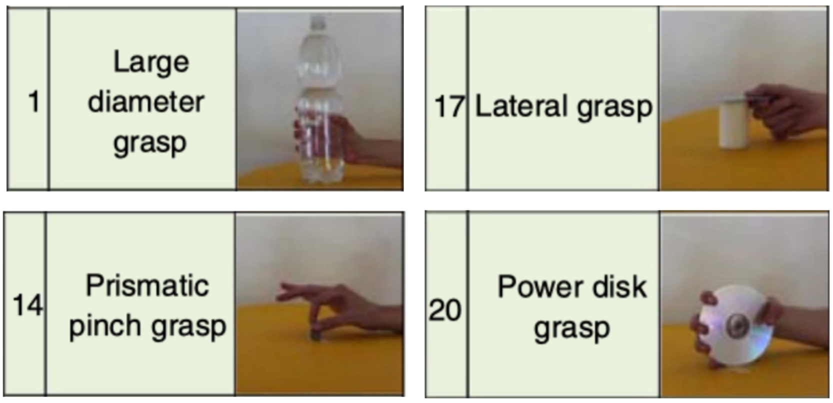

Figure 20: 4 Gestures Used for Self-Recording with Mindrove

On 3 occasions, we used the aforementioned recording script to record ourselves (Firas) performing the 4 gestures described in 20. Using this script multiple times allowed us to investigate whether a model trained on one recording will generalize immediately to a new recording. Across all 3 recordings, we ensured that the placement of the Mindrove on the arm was consistent and that the same tightness of the strap was maintained.

To evaluate whether using this self-recorded data to train subject-specific models will lead to better performing models that can point to the reliability and efficacy of subject-specific models for use in our pipeline, we designed the following training and testing set up:

1. **Architecture** We adopt an architecture similar to that in 11 with the following adjustments:
   (a) **Convolutional Blocks:** The model comprises four sequential convolutional blocks, each containing:
      - A 2D convolutional layer with increasing filter sizes across layers.
      - A ReLU activation function to introduce non-linearity.
      - A dropout layer with a probability of 0.2 to reduce overfitting.
      - A max-pooling layer to reduce spatial dimensions.

(b) **Detailed Block Structure:**

- **Block 1:** Convolution with 8 filters, kernel size of (3×3), followed by ReLU, dropout, and max-pooling (2×1).
- **Block 2:** Convolution with 16 filters, kernel size of (3×3), followed by ReLU, dropout, and max-pooling (2×1).
- **Block 3:** Convolution with 32 filters, kernel size of (3×3), followed by ReLU, dropout, and max-pooling (2×1).
- **Block 4:** Convolution with 64 filters, kernel size of (3×3), followed by ReLU, dropout, and max-pooling (3×2).

(c) **Fully Connected Layer:** After feature extraction, the output is flattened and passed through a dense layer with 512 input features and a final output size corresponding to the number of classes (4, in this case).

2. **Preprocessing** The following preprocessing steps were applied to the EMG data:

(a) **Sliding Window Segmentation:** EMG signals were divided into overlapping segments using a window size of 150 ms, corresponding to 75 collected data points. Each window had an overlap of 60% to ensure continuity in feature extraction.

(b) **Conversion to Real Values:** The raw EMG values were converted to real voltages by multiplying each value with the least significant bit (LSB) conversion factor:

$$\text{LSB} = 0.0045 \times 10^{-6} V$$

(c) **Root Mean Square (RMS) Calculation:** The RMS values of the EMG signals were computed for every 5 samples using a step size of 1. RMS calculation helps to quantify the signal's energy and smooth out variations.

(d) **Anti-Aliasing Filtering:** A first-order low-pass Butterworth filter with a cutoff frequency of 50 Hz was applied to the RMS-transformed EMG data to remove high-frequency noise and prevent aliasing.

(e) **Min-Max Normalization:** The EMG data was normalized using min-max scaling to ensure consistency across different recording sessions. The normalization process was performed using the min and max values of the RMS-rectified EMG signals.

i. Normalization was applied separately to each EMG channel.

3. **Training and Testing Scheme** The training and testing scheme was designed to evaluate the performance of a subject-specific model under different conditions. The following approach was used:

- **Repetition Splitting:**
  - Repetitions **1, 3, 5** were used for **training**.
  - Repetitions **2, 4** were used for **testing**.
- **Preprocessing Considerations:**
  - Min-max normalization was applied using the minimum and maximum values computed from the training set for each channel.
- **Training and Testing Scheme:**

  (a) **Single-Recording Training and Testing:** In this setup, the model was trained on the training set of one recording and tested on the testing set of that same recording. This provided a **baseline** evaluation, demonstrating how well a subject-specific model trained on could perform when tested under identical setup conditions.

  (b) **Cross-Recording Generalization:** In this setup, the model was trained using the training data from **two different recordings** and tested on the **third recording**. The training and testing sets were rotated across the three available recordings. This experiment aimed to assess whether data collected in one recording session would match the distribution of data collected from another recording (according to predictive model performance).

Observation	Interpretation
Classification test scores within a recording (isolating 2nd and 4th repetition) and testing on them are great (85-95% range).	The CNN model appears to be a sufficient learner for the task when trained and tested on the same recording. This suggests that within-session variability is minimal, and the model effectively captures subject-specific EMG patterns.
Classification test scores across different recordings are poor (40-60% accuracy, depending on preprocessing steps selected).	There is likely a calibration issue between recordings, meaning preprocessing steps do not generalize well across different sessions. This suggests that EMG signal characteristics shift between recordings, making direct model transferability difficult.

Table XIV: Comparison of classification performance within a single recording vs. across multiple recordings.

Given the challenges described in XIV, it is clear that the preprocessing approach we used limits the applicability of trained models outside the data distribution of the recording they were trained on. We turned to the software development kit and tools provided by the manufacturers of Mindrove to find a better preprocessing approach suited for inferencing (with new recordings). As explained on Mindrove's Github `https://github.com/MindRove/NaviFlame`, Naviflame is "a middle-sized AI system for detecting hand and finger motions from wearable sensor data". The "NaviFlame implements a pipeline for recording, fine-tuning, and performing real-time inference of gesture-based inputs using a MindRove device. The system incorporates signal processing, Deep Learning based feature extraction, MLP-based classification, and real-time visualization". We had core successes working with NaviFlame's implementation that help orient our capstone's future direction and approach:

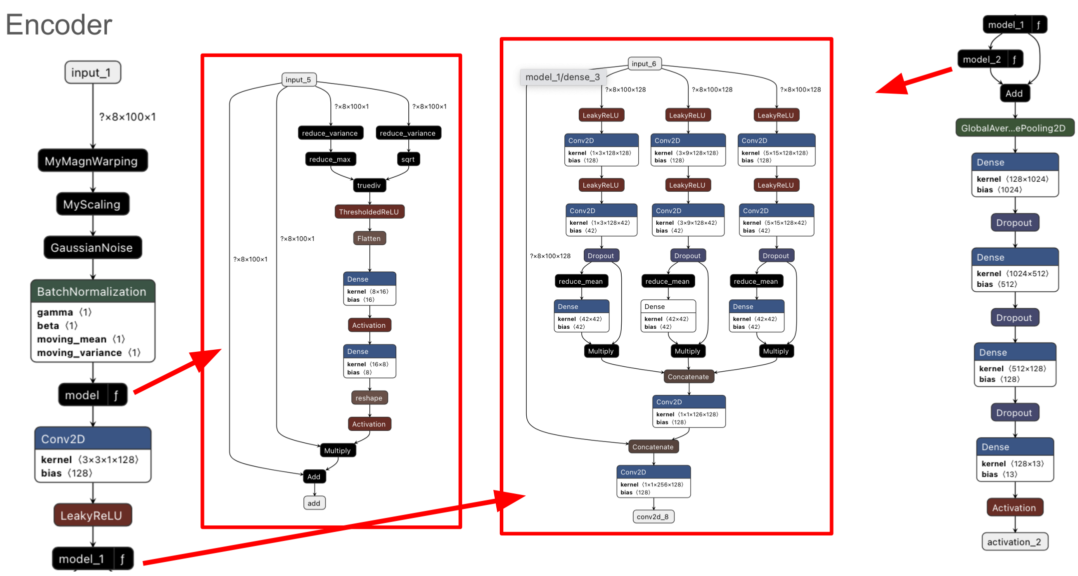

Figure 21: Encoder architecture of Naviflame model.

1. **Preprocessing Approach** The preprocessing steps undertaken in NaviFlame are, as shown in Figure 21:

   (a) **Filtering: Recursive Processing**

   - The signal is passed through a sequence of recursive filters applied on a channel-by-channel basis.
   - These filters include:

i. **High-pass filter** to remove DC bias (cutoff frequency: 4.5 Hz).
ii. **Notch filter** to remove power line interference (cutoff: 50 Hz).
iii. **Low-pass filter** to remove high-frequency noise (cutoff: 100 Hz).

- The filtering process for each sample is recursive, meaning that the output of one step is used as the input to the next, ensuring continuous noise suppression. Each channel of the EMG data undergoes recursive filtering. The filtering process can be expressed as follows:
  - **Biquad Filter Equations:**

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] - b_1 y[n-1] - b_2 y[n-2] \quad (1)$$

    where:
    * $x[n]$ is the input signal.
    * $y[n]$ is the filtered output.
    * a_0, a_1, a_2, b_1, b_2 are filter coefficients computed based on the filter type.
  - **Notch Filter for Power Line Noise Removal:**

$$H(f) = \frac{1 - 2\cos(2\pi f_0) z^{-1} + z^{-2}}{1 - 2\cos(2\pi f_0) r z^{-1} + r^2 z^{-2}} \quad (2)$$

    where $f_0 = 50$ Hz and r is the notch filter tuning parameter.

(b) **Scaling: Normalization using StandardScaler:**

- The extracted features from the feature extractor (described below) are normalized using a precomputed mean and standard deviation (from the training set used for finetuning).
- The transformation is given by:

$$X_{norm} = \frac{X - \mu}{\sigma} \quad (3)$$

where μ and σ are the mean and standard deviation of the training data.

2. **Feature Extractor** The feature extractor model is a deep neural network designed for electromyography (EMG) gesture classification. It is composed of multiple layers, including convolutional layers, normalization layers, dense layers, and dropout layers. This model was trained on data from 12 subjects performing multiple gestures, making it a strong foundation model for further fine-tuning on new users or datasets. The model consists of the following layers:

- **Gaussian Noise Layer:** Introduces controlled noise to the input data to improve generalization and robustness. This technique achieves this by:
  - **Improving generalization:** The model learns to extract features that are invariant to small variations in the input.
  - **Mitigating overfitting:** Prevents the model from memorizing noise present in the training data.
  - **Enhancing performance on real-world data:** Improves adaptability when faced with sensor drift or subject variations.
- **Convolutional Layer (conv2d):** Extracts spatial and temporal features from the EMG signals.
- **Batch Normalization:** Normalizes activations to stabilize training and speed up convergence.
- **Leaky ReLU Activation:** Introduces non-linearity while allowing small negative gradients.
- **Add Layer (add_1):** Implements residual connections to prevent vanishing gradients.
- **Global Average Pooling (global_average_pooling2d):** Reduces spatial dimensions, maintaining important features while lowering computational complexity.
- **Dropout Layers (dropout_6, dropout_7, dropout_8):** Regularization technique to prevent overfitting.

   - **Fully Connected Layers (dense_8, dense_9, dense_10, dense_11):** Higher-level feature representations leading to final gesture classification.

3. **Transfer Learning and Prediction** The feature extractor, trained on 12 subjects performing multiple gestures, is leveraged as a foundational model. Instead of retraining the entire network from scratch for new users, transfer learning is applied by:

   - **Freezing initial layers of the feature extractor:** These layers capture general EMG signal representations and do not need to be re-learned for every new user.
   - **Using the feature extractor up to a specific layer (e.g., `dense_8`):** The outputs from this layer are high-level feature representations.
   - **Adding a new Multi-Layer Perceptron (MLP) classifier:** A new MLP with fully connected layers is added on top of the extracted features.
   - **Fine-tuning only the MLP using recorded data:** The MLP classifier is trained using the new user's recorded data while keeping the lower layers frozen.

   In real-time inference, predictions are made on multiple overlapping windows of EMG data. Each window generates a probability distribution over gesture classes. To obtain a robust final prediction, *majority voting based on softmax probabilities* is used.

Given the robustness of the NaviFlame development kit, we look to fuse several of its elements into our final implementation and work, primarily its feature extractor as a base foundation model that works well for extracting features from mindrove-based sEMG windows as well as its preprocessing script that applies recursive filtering that is more scalable and robust for inference.

8.1.2 Vision Module

For vision-based classification systems, there are a few main approaches. The initial approach detailed in Section 7, Final Design Expected, is the approach of fitting a convex hull around the object. Due to issues outlined in the following section, this approach was then scrapped for the alternative Grasp Pose Detection (GPD) approach. The implementation of both approaches is described in detail below:

Fit a Convex Hull:

Fitting a convex hull involves collecting point cloud data on the object, segmenting said point cloud, fitting the nearest convex 3D shape, and mapping each shape into different grasps. This approach is supposed to run in real-time, providing the best grasp prediction for the most likely object. Ranking the objects as likely or unlikely grasp candidates can depend on a variety of conditions, such as hand proximity to the object, centrality of the object in the field of view, the user's gaze, etc. Figure 9 shows the steps of the implemented pipeline.

Grasp Pose Detection

Grasp Pose Detection (GPD) is a technique used in control robotic grippers, where it provides a set of candidate gestures, ranks the gestures based on certain metrics that gauge the likelihood a proposed gesture would succeed, select one of those gestures, and execute it. Given the similarity in the domains, our project borrows the first two steps of the pipeline in implementing a version of GPD that works on prosthetic control, where the user has full authority to select which gesture they want (there is no automatic selection and execution of best gestures). This approach requires much more time per iteration to run due to the higher complexity of the task, where the raw point cloud is provided and all possible grasps are generated based on a variety of conditions. In our approach, we tested multiple models and settled with FGC-Net GPD. Figure 22 provides an overview of the pipeline in the GPD approach.

Improvements

Ideally, to drive the cost down, onnly an RGB camera would be essential for the vision module. After experimenting, it was deciding that incorporating a depth camera at this stage is essential, and after fine-tuning depth models to a synthetic dataset, it may then be possible to return to the use of a simple RGB camera.

8.1.3 Fusion Module

As outlined in Section 7, the fusion strategy plays an important role in creating an intuitive and reliable control system. The initial design involved a continual learning module that would use the sEMG predictions as ground truth to align the vision module with the sEMG module. This

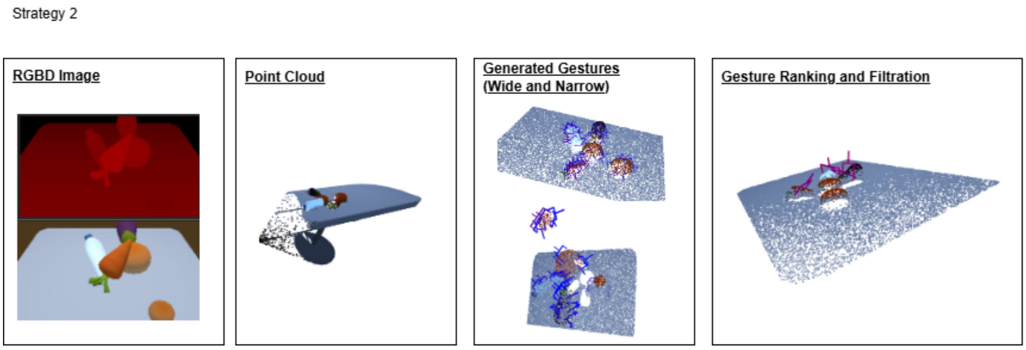

Figure 22: GPD Pipeline. Left to Right: RGBD image from Unity transformed into point cloud, two types of grasps generated in all possible positions, and then ranking of the proposed gestures. These gestures are then selectively displayed to the user in the environment depending on hand trajectory towards the objects.

assumed the initial approach with the Vision Module (fitting a convex hull). After discovering the issues caused by that approach, outlined in the next section, the fusion approach was adjusted to a version that would work best with the GPD approach. Figure 23 provides a clear overview of the fusion strategy and the steps involved in executing a gesture.

As for possible future improvements, finding a way to still incorporate an online learning module that trains based on the grasp success would greatly benefit the system. This could be through incorporating sensors to detect the grasp success (e.g. Force sensors to detect if an object slips after the grasp is attempted), that way this information may be used to improve the grasping strategy.

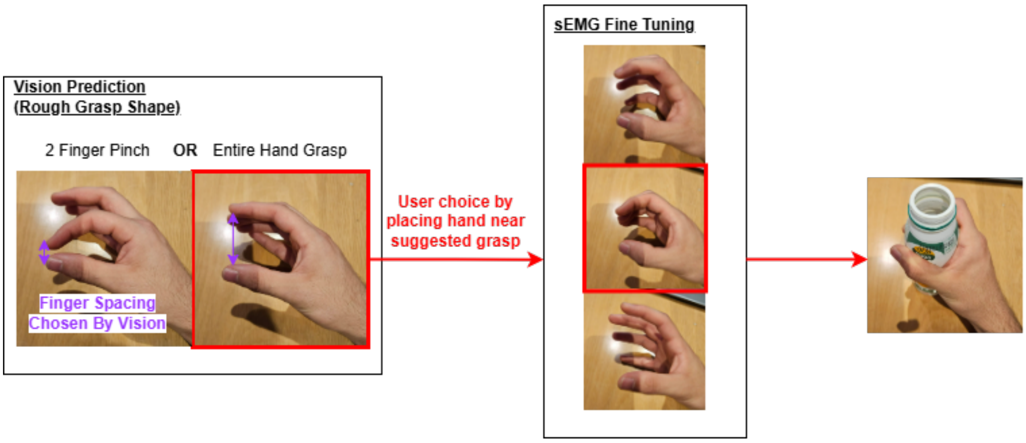

Figure 23: Pipeline for the fusion module. The vision module provides a set of full-hand (wide) and two finger (narrow) grasps with different finger spacing. User picks the gesture by approaching the grasp with their hand.

8.2 Issues Faced

8.2.1 sEMG Module

Throughout the development of our sEMG-based classification system, several challenges emerged, impacting dataset selection, preprocessing techniques, model training strategies, and classification performance. These challenges are categorized based on the datasets and methodologies employed.

Challenges with the Mindrove Dataset (Köllőd *et al.*, 2022) was initially considered due to its compatibility with our hardware setup. However, it presented several limitations:

- **Signal Quality Variability:** The classification performance varied significantly across different subjects, as seen in Table XII. Models trained on a subset of subjects struggled to generalize, with test accuracies ranging from 43.2% to as low as 13.0%, indicating considerable inter-subject variability.

- **Limited Gesture Set:** The dataset contained only simple one degree-of-freedom (DoF) gestures (10), limiting its applicability for real-world tasks that involve complex grasping and object manipulation.
- **Limited Subject Pool:** With only 6 subjects, the dataset lacked the diversity required for training a generalizable model, making it unsuitable for downstream classification on a broader population.

Due to these constraints, we explored alternative datasets with richer gesture representations and larger subject pools.

Challenges with the Ninapro DB1 Dataset The Ninapro DB1 dataset (Atzori *et al.*, 2015) was selected for its large number of subjects and diverse gesture set. However, several challenges arose:

- **High Repetition Requirements:**
  - The dataset required multiple repetitions per subject to achieve reasonable classification performance.
  - As seen in Table XIII, classification accuracy improved with increased subject inclusion, yet 50% inclusion only resulted in 50-75% accuracy.
  - The need for multiple repetitions made real-time applications impractical and motivated us to use subject-specific training on self-recorded data.

Given these limitations, we explored the feasibility of recording subject-specific datasets directly from the Mindrove device.

Challenges with Self-Recorded Mindrove Data To improve classification performance, we recorded our own dataset using the Mindrove device. However, this introduced new challenges:

- **Cross-Session Generalization Issues:**
  - Classification performance within a single session was high (85-95%), demonstrating the CNN model's ability to learn subject-specific EMG patterns.
  - However, cross-session generalization was poor (40-60% accuracy), as shown in Table XIV, due to sensor placement variations, muscle fatigue, and execution differences.
- **Preprocessing Pipeline Limitations:**
  - Standard preprocessing techniques (min-max normalization, RMS filtering, low-pass filtering) failed to normalize inter-session signal variations effectively.
  - The inconsistency in preprocessing limited model transferability to new recording sessions.
- **Recording Setup Sensitivity:**
  - Despite maintaining consistent electrode placement and strap tightness, slight variations resulted in inconsistent signal quality.
  - This raised concerns about the reliability of subject-specific models outside of controlled conditions.

Challenges with NaviFlame-Based Preprocessing and Feature Extraction To address cross-session variability, we explored Mindrove's NaviFlame pipeline, which implemented advanced filtering and feature extraction. While it provided improvements, it also introduced new challenges:

- **Reliance on Proprietary Software:**
  - NaviFlame is inherently a Mindrove software development kit, limiting flexibility in modifying the pipeline.
  - Some preprocessing and training steps were opaque (for instance, the feature extractor's layers were hidden and had to be reverse engineered and context regarding training the feature extractor (number of subjects, gestures used) were hidden and had to be discovered by contacting the developers directly), restricting experimentation with alternative filtering and feature extraction techniques.

8.2.2 Vision Module

There were many issues encountered along the way with both implementations of the vision modules. A common issue faced throughout this projects is the lack of complete documentation for many of the used components, as well as environmental incompatibilities. This was resolved by a careful choice between different versions of dependancies, as well as operating over WSL (Windows Subsystem for Linux) to create the most stable and replicable environment.

Presented below is an outline of issues particular to each strategy, why they happen, and how they were resolved:

Fit a Convex Hull

- The cursor can never be reliable unless it is fine tuned to the particular user in the said particular situation, or if we have full awareness of the scene which the prosthetic would not have given it's a partially observable problem.
  - There isn't really a perfect solution for this. Multiple options were attempted like using a bounding box rather than a point, using a ray projection from the cursor onto a canvas slightly farther away from the user, etc. but none to the reliability we aimed for. This is one of the major reasons this approach was substituted
- While it would've been ideal if it was real-time, this strategy still took a bit too long, with the lite versions of the models still running at around 2-5 fps.
  - The trade-off would be lower accuracy through switching to lower resolution images, which would beat the purpose of this approach.
- Over UDP, packets tended to drop once the image exceeded a certain resolution, placing a cap on the maximum accuracy we can get out of our segmentation and depth detection.
  - Sending information over mutliple packets helped increase the limit, though the 5th packet onwards would often be dropped. We could implement TCP, but the cap on the image resolution was more restricted by the aformentioned speed issue rather than this communication problem.

Grasp Pose Detection

- Similar issues with maximum fps of the system according to the resolution because of UDP but not as crucial since the GPD models do not require a perfect point cloud.
- Similarly, we faced issues with the
- The point cloud was warped due to perspective. Unity is really bad at giving true depth using the built in depth shaders.
  - This was resolved by switching to sending true depth instead using a custom depth camera script.
- Another issue resolved by out custom depth script was the different scales used in Unity and by the GPD model, where the returned grasps were not mappped to the corresponding location in Unity.
- Not real-time anymore due to the amount of time it takes to run an iteration. The illusion of real-time needed to be preserved though.
  - Resolved by hand distance aware hiding and unhiding of gestures as well as only re-prompting the GPD model whenever 70 % of the gestures are not in line of sight.

8.2.3 Fusion Module

The main issues faced during fusion were the many changes happening in the previous two modules. Upon resolving them, the fusion strategy became more straightforward, where the only major issue was choosing the most intuitive set of gestures for both modules such that the user felt total ownership over the prosthetic hands operation. This was done by choosing few, very distinct, simple gestures and grasps for both modules.

8.3 Changes During Implementation

8.3.1 sEMG Module

Given the challenges encountered with session-to-session variability in our self-recorded dataset, we will now adopt NaviFlame's preprocessing approach. This methodology has demonstrated the most consistent performance across different recording sessions, mitigating the issues of sensor placement sensitivity and inter-session signal drift.

Utilizing NaviFlame's Feature Extractor for Transfer Learning In addition to its robust preprocessing pipeline, we will leverage NaviFlame's pre-trained feature extractor as the foundational model for our classification system. This decision is driven by several factors:

- **Pre-trained on a Large Subject Pool:** The feature extractor has been trained on data from a substantial number of subjects, all using the Mindrove device. This ensures that the extracted features are well-suited for our hardware conditions.
- **Diverse Gesture Representation:** The training dataset includes a broad range of hand gestures, improving the model's ability to generalize across different movement patterns.
- **Gaussian Noise Augmentation:** A key advantage of the feature extractor is its use of Gaussian noise in the input layer. This technique enhances generalization by making the model robust to small variations in signal amplitude and electrode placement, reducing the risk of overfitting to session-specific conditions.

Given these strengths, we will fine-tune the pre-trained feature extractor using a lightweight Multi-Layer Perceptron (MLP) classifier. The MLP will be adapted to our specific set of desired gestures while preserving the robust feature representations learned from a larger dataset.

Maximally Distant Gesture Mapping for Improved Discrimination To improve the reliability of gesture classification, we will explore a *maximally distant gesture selection strategy* during training. Instead of training on gestures that are closely related and difficult to differentiate, we will:

- Select a subset of gestures for training that are highly distinct from each other in terms of muscle activation patterns.
- Ensure that these gestures are easily distinguishable in the training phase, allowing the model to learn strong, separable feature representations.
- Establish a mapping from these learned gestures to more subtly different gestures during inference, reducing ambiguity in real-world usage.

This approach aims to enhance classification robustness by maximizing inter-class variance in the learned feature space, making it easier to generalize to finer gesture distinctions during real-time deployment.

Redefining the Role of sEMG: From Gesture Correction to Fine-Tuning A fundamental shift in our approach is redefining the role of sEMG in our control system. Previously, sEMG was viewed as a *corrector* to refine and adjust vision-based predictions. However, we will now position sEMG as a *fine-tuning mechanism* that enhances vision-based gesture predictions rather than overriding them.

- Vision-based tracking will provide an initial prediction of the intended gesture and associated movement.
- sEMG signals will serve as a fine-tuner, refining the vision-based predictions to offer more precise movement execution.
- This paradigm shift reduces reliance on EMG as the sole classifier, leveraging it instead as a secondary, enhancing modality that adjusts based on subtle muscle activations.

By combining vision and sEMG in a complementary manner, we aim to improve real-time gesture classification accuracy while minimizing the impact of signal inconsistencies inherent to EMG-based systems.

8.3.2 Vision Module

Originally, the design would utilize fitting a convex hull on a generated point cloud to select the most useful grasp for an object. This process would involve generating a point cloud, segmenting the point cloud, and fitting the most suitable shape on that partial point cloud. Any lack of performance at any stage of this process would result in poor performance in the final chosen grasps. Due to the reasons outlined earlier, Issues Faced, it was decided to scrap this original strategy and substitute with the more reliable GPD with a depth camera. This approach offered more in both reliability and intuitiveness.

8.3.3 Fusion Module

The fusion module decide what the final grasp would look like based on the input sEMG and vision predictions. Originally, the module would always give priority to the sEMG regardless of the vision's recommendation if the user chooses so. This correction is then used to further train the vision classifier using a suitable domain incremental online learning approach. This approach would've operated smoothly with convex hull fitting and direct sEMG grasp mappings. Once those two components changed, it was imperative that the fusion module adjusts accordingly. After many different design iterations, we settled on the fusion module outlined earlier, where computer vision creates the base gesture and sEMG predictiosn operate as a bunch of knobs to adjust parameters of the grasp.

8.4 Initial Results

Throughout our experimentation with different datasets and training strategies for sEMG-only control, we conducted multiple evaluations to assess model performance under various conditions. The key findings from our testing phases are summarized below.

Evaluation of the Mindrove Dataset Initial experiments with (Köllőd *et al.*, 2022) highlighted significant limitations in signal consistency and generalizability:

- **Inter-Subject Performance Variability:** When training on a subset of subjects and testing on a different subject, classification accuracy was highly inconsistent. As shown in Table XII, accuracy ranged from 43.2% down to as low as 13.0%, demonstrating the dataset's inability to generalize across users.
- **Gesture Set Limitations:** The dataset's small set of one-degree-of-freedom (DoF) gestures (10) restricted the practicality of trained models, as they struggled to classify more complex, real-world gestures.
- **Small Subject Pool:** With only 6 subjects available, the dataset did not provide enough variability to build a generalizable feature extractor, making it less effective for downstream classification.

Due to these constraints, we transitioned to a larger and more diverse dataset to improve performance.

Evaluation of the Ninapro DB1 Dataset To overcome the limitations of the Mindrove dataset, we trained and tested models using the Ninapro DB1 dataset (Atzori *et al.*, 2015). While it provided a larger subject pool and a broader gesture set, new challenges emerged:

- **Subject Generalization Challenges:**
  - Using a leave-one-out subject evaluation approach, models trained on $N-1$ subjects and tested on a hidden subject exhibited poor generalization, achieving an accuracy range of only 25-50% (Figure 18).
  - This demonstrated that even with a large dataset, inter-subject variability remained a significant challenge.
- **Impact of Subject Inclusion in Training:**
  - Performance improved when portions of the hidden subject's data were incorporated into training.

  - As shown in Table XIII:
    - Training with 25% of the subject's data improved accuracy to 35-70%.
    - Increasing to 50% resulted in 50-75% accuracy.
    - Using 75% of the subject's data achieved up to 80% accuracy (Figure 19).
  - These results indicated that substantial subject-specific data was required to achieve reliable classification, making real-world deployment less practical.

- **Repetition Dependency:**
  - The dataset required multiple gesture repetitions to achieve stable performance.
  - The necessity of 5+ repetitions to reach only 50-80% accuracy suggested that a subject-specific approach may be more practical than relying on pre-trained models with limited generalization.

The high dependence on subject-specific training led us to explore whether recording and training our own dataset could offer more consistent performance.

Evaluation of Self-Recorded Mindrove Data To overcome the need for extensive subject-specific data in pre-existing datasets, we recorded our own dataset using the Mindrove device and conducted multiple tests:

- **Within-Session Performance:**
  - When training and testing within the same recording session, classification accuracy was significantly higher (85-95%), as seen in Table XIV.
  - This indicated that the CNN model effectively learned subject-specific EMG patterns within a single session.
- **Cross-Session Performance Drop:**
  - When testing across different recording sessions, classification accuracy dropped significantly to 40-60%.
  - This highlighted the issue of session-to-session variability, suggesting that preprocessing techniques were not adequately normalizing signal shifts.
- **Preprocessing Limitations:**
  - Despite applying min-max normalization, RMS filtering, and low-pass filtering, results were inconsistent across different recordings.
  - These findings indicated that additional signal calibration or domain adaptation techniques were necessary to maintain classification reliability across different sessions.

These results guided our decision to transition towards a feature extraction-based approach using NaviFlame's pre-trained model, with a focus on improving consistency across different recording sessions.

As for the vision and fusion module, with preliminary tests conducted to assess the feel of the system. Snippets of the implementation are shown in Figure 24. There are a few known issues/possible improvements outlined below:

- GPD
  - The model can be trained further on objects with which we will be testing the system.
  - Including more types of base grasps.
- Grasps
  - Deciding how grasping is initiated rather than a simple trigger.
  - Best way to calculate the success of a grasp in VR leads to different success rates.
  - Selective displaying of grasps taking into account more than proximity of prosthetic to object.
- Fusion

  - Deciding the optimal sliders requires a lot of experimentation
  - Incremental rather than binary sliders
- Environment
  - Using a more realistic environment for training and testing

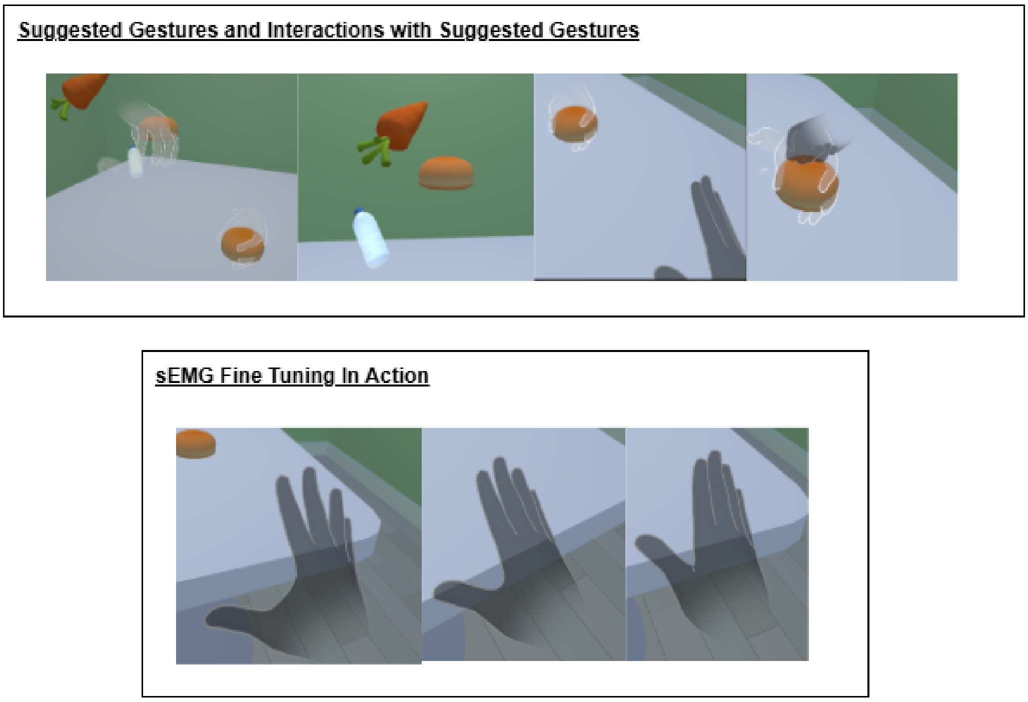

Figure 24: Some implementation snippets showing the utility of the implementation

9 Ethics

Developing an upper-limb prosthetic control system involves a range of ethical considerations centered on accessibility, usability, and potential psychosocial impacts on users. This section addresses the primary ethical concerns and presents solutions to ensure the design aligns with the highest ethical standards for technology intended to assist individuals with limb differences.

- Accessibility and Equity of Technology
  - Challenge: Prosthetic technology is often prohibitively expensive, limiting access for many individuals who would benefit from it. Additionally, geographical disparities can make it difficult for people in less developed regions to access high-quality prosthetic solutions.
  - Solution: To promote equitable access, the proposed system aims to use cost-effective components, such as the Mindrove armband and Meta Quest 3, chosen for their balance of functionality and affordability. Additionally, by relying on open-source software and providing clear documentation, the project supports scalability and replication in various settings, potentially lowering costs and increasing accessibility in diverse locations.
- User Autonomy and Control
  - Challenge: A key ethical concern is ensuring that the prosthetic does not override or misinterpret user intentions, as lack of control over a prosthetic device could negatively affect the user's sense of autonomy and lead to frustration or safety issues.
  - Solution: To address this, the system will undergo rigorous testing to ensure reliable and precise interpretation of user commands. The sEMG controller generally gets precedence over the vision module to preserve the user autonomy

- User Safety and Psychological Well-being
  - Challenge: Adapting to a new prosthetic can be psychologically taxing, particularly if the device is difficult to control or frequently malfunctions. Poor design could lead to frustration, emotional distress, and even device rejection.
  - Solution: To minimize user frustration, the system focuses on intuitive and low-cognitive-load control methods.Providing users with thorough training and support during the learning phase is essential to mitigate stress. Additionally, the VR-based testing environment will replicate real-world scenarios, allowing users to practice and adapt comfortably in a controlled setting before transitioning to more complex tasks.
- Sustainability and Long-term Support
  - Challenge: Many users depend on their prosthetic devices for daily functionality, so it's ethically important to consider the longevity and maintainability of the system. Devices that quickly become outdated or unsupported may negatively impact users who rely on them.
  - Solution: By employing widely supported hardware and open-source software, this project seeks to ensure that users can receive updates and find technical support well after initial deployment. Furthermore, documentation provided with the system will be detailed enough to facilitate troubleshooting and maintenance, supporting long-term usability.

10 Did the Design Meet the Requirements

10.1 Criteria for Testing

Originally, to evaluate the reliability and intuitiveness of the proposed design, we had established the following testing criteria:

- **sEMG Module Accuracy:** The sEMG module should be able to independently achieve 80% accuracy after fine-tuning to the given user.
- **Vision Module:** The Vision module is composed of mulitple sub-components. The goal is that the predicted optimal grasp should match with the user's intent 60% of the time in the general state before any online learning is implemented.
- **Fusion:** Upon fusing the modalities, the system needs achieve a total 85% success rate at each of the assigned tasks.
- **Online Learning Phase:** The online learning component needs to raise the vision module's agreement with the user intent to 80%.
- **System Variability:** Between different donnings and undonnings of the prosthetic, the system accuracy should not vary more than 5%.
- **Real-Time Control:** The hard-real time components of the controller need to amount to a total of 350 ms. The soft-real time components do not contribute to that total delay because a decision can be made based on the previous knowledge of the soft-real time information.
- **Testing Environments:** The testing environments needs to replicate realistic settings both indoors and outdoors. Each environment should involve set goals (e.g. bake a cake, fold origami, etc.) that can be evaluated quantitatively.

The criteria were adjusted as the design changed significantly, and testing will be done through two main avenues: Quantitatively and Qualitatively. Four subjects were recruited to test the system, and those results are provided in this report.

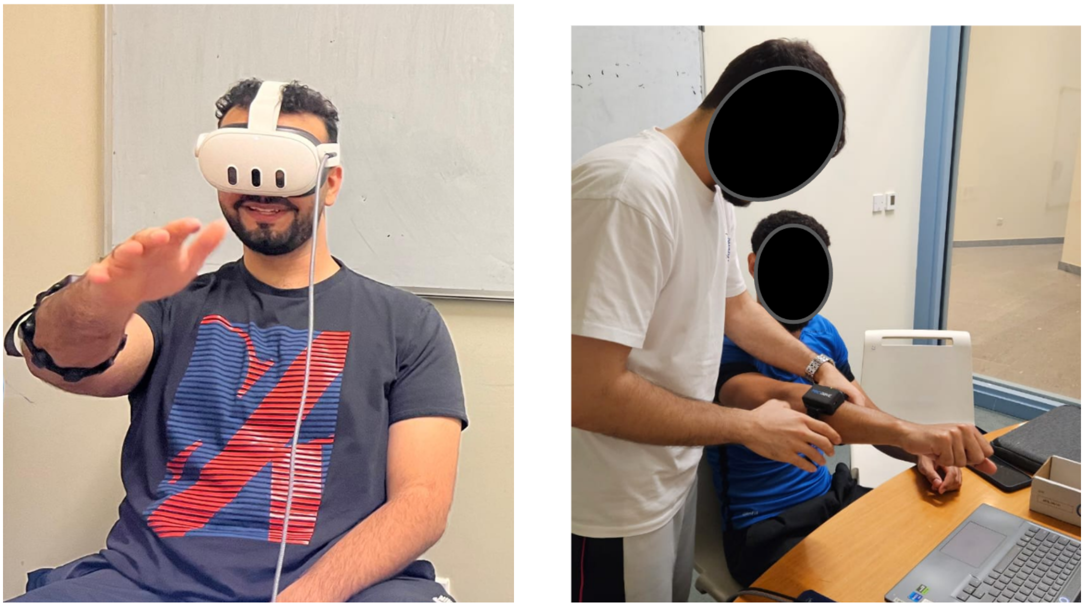

Figure 25: Test subjects going through the simulation.

10.1.1 Quantitative Testing

The testing procedure involved 4 prosthetic controllers presented according to the inverse latin square selection procedure:

1. Baseline (Hand tracking)
2. Vision only
3. sEMG only
4. Fusion

Subjects were placed in a playground period to attune to each controller for as long as needed. A physics based interactions system was developed to balance between optimization and accuracy.

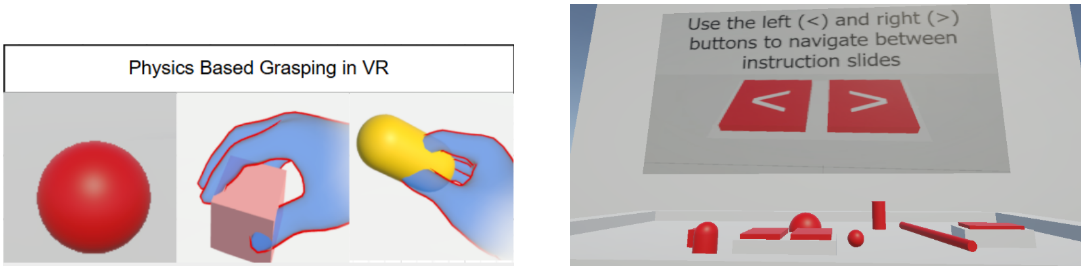

Figure 26: (Left) description of physics system used in the simulation. (Right) Playground tasks with instructions

After each playground session, 3 trials of the basket task were conducted per controller. The task involved grasping all objects from one bin and placing them in the other bin as fast as possible with as few grasps as possible. The number of failed grasps as well as the time taken is recorded to evaluate the reliability of the model.

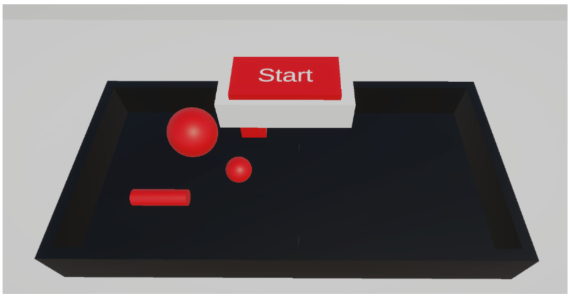

Figure 27: Basket task

10.1.2 Qualitative Testing

To assess the perceived intuitiveness and reliability of the system, a Likert scale questionnaire was asked after the completion of the experiment assessing:

1. Perceived Ease of Use (1 = Very Difficult, 7 = Very Easy)
2. Intuitiveness (1 = Not Aligned at all, 7 = Perfectly Aligned)
3. Reliability (1 = Very Unreliable, 7 = Very Reliable)
4. Speed of Response (1 = Very Slow, 7 = Very Fast)
5. Mental Effort (1 = Very Low Effort, 7 = Very High Effort)
6. Comfort & Fatigue (1 = Not Fatiguing at All, 7 = Extremely Fatiguing)
7. Learning Curve (1 = Required a Lot of Practice, 7 = Felt Comfortable Almost Immediately)
8. Sense of Control or Agency (1 = No Sense of Control, 7 = Strong Sense of Control)
9. Task Performance (1 = Very Dissatisfied, 7 = Very Satisfied)

Additionally, a set of fixed open-ended questions and some specific to each user's concerns were asked to get a complete picture of the controller's reliability and intutitiveness.

10.2 Test Data

Below is the summary of the test quantitative and qualitative data generated from the users:

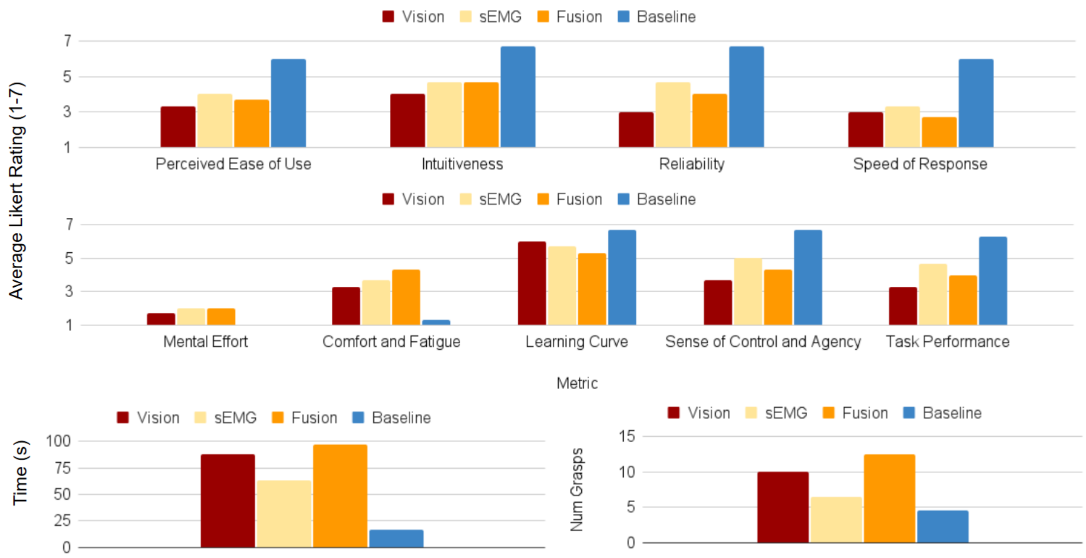

Figure 28: Average Qualitative and Quantitative results

10.3 Discussion the testing data

The preliminary test data indicates that the proposed design shows strong potential in meeting its intended requirements. Participants reported an increased sense of agency when using sEMG-based control, both in the standalone sEMG mode and in the Fusion controller configuration. This suggests that the system supports intuitive control as intended. Additionally, users found it easier to remember a smaller number of gestures, which validates the decision to use sEMG selectively in the Fusion design and confirms a reduction in cognitive load. Importantly, the subjective sense of reliability closely aligned with the observed quantitative performance metrics, further affirming the consistency of the system's behavior. However, several areas for improvement were identified through testing. The Vision controller's full capability was underutilized due to the simplicity of the basket task, limiting the assessment of its advantages. Moreover, delays caused by physics simulation lags, particularly in collision resolution, negatively impacted perceived usability—most notably in the Vision and Fusion controllers. The Mindrove armband also exhibited hardware issues, specifically rapid battery depletion, which affected signal quality over time. Finally, while subjects were encouraged to explore the environment, they preferred to engage directly with the task, suggesting that task design should be streamlined to match user preferences. These insights point to specific refinements in both hardware and simulation fidelity required to fully meet the design objectives.

11 Bill of Materials

Table XV: Bill of materials so far.

Part	Webpage	Price (USD)	Quantity	Total
Meta Quest 3	Amazon Link	472.63	1	472.63
Mindrove 8 channel armband	Purchase link	729.00	1	729.00

12 Project Management

12.1 Work Breakdown Structure

Table XVI: Project WBS with Durations and Planned Dates

Primary Tasks	Sub-Tasks	Duration (days)	Planned Dates	
			Start	End
	0.1 Begin Project	7	2024-09-01	2024-09-07
1 Determine Prosthetic User Needs	1.1 Map out prosthetic rejection causes	10	2024-09-08	2024-09-17
	1.2 Research commercial products	10	2024-09-18	2024-09-27
	1.3 Create a hierarchical list of user needs	8	2024-09-28	2024-10-05
	1.4 Revise problem statement	5	2024-10-06	2024-10-10
2 Research Prosthetic Literature	2.1 Identify used input signals	8	2024-10-11	2024-10-18
	2.2 Research reliable control approaches	10	2024-10-19	2024-10-28
	2.3 Research intuitive control	8	2024-10-29	2024-11-05
	2.4 Select promising concepts	7	2024-11-06	2024-11-12
3 Plan Implementation	3.1 Look into possible simulation environments	8	2024-11-13	2024-11-20
	3.2 Find devices & update budget	10	2024-11-21	2024-11-30
	3.3 Find (or plan generation) of datasets	8	2024-12-01	2024-12-08
	3.4 Find relevant pretrained models	7	2024-12-09	2024-12-15
4 Prototyping	4.1 Purchase components	10	2024-12-16	2024-12-25
	4.2 Set up software tools	8	2024-12-26	2025-01-02
	4.3 Implement modules with test data	15	2025-01-03	2025-01-17
	4.4 Integrate modules together	12	2025-01-18	2025-01-29
5 Testing Prototype	5.1 Develop testing criteria	7	2025-01-30	2025-02-05
	5.2 Perform unit tests	10	2025-02-06	2025-02-15
	5.3 Perform Integration tests	10	2025-02-16	2025-02-25
	5.4 Identify improvements to be made	7	2025-02-26	2025-03-04
6 Build Final System	6.1 Implement any improvements from prototype	10	2025-03-05	2025-03-14
	6.2 Test and refine final system	8	2025-03-15	2025-03-22
7 Document & Report	7.1 Document design process	10	2025-03-23	2025-04-01
	7.2 Refine analysis and proposed solution	10	2025-04-02	2025-04-11
	7.3 Prepare final report	10	2025-04-12	2025-04-21
	7.4 Prepare poster	19	2025-04-22	2025-05-10

12.2 Design Structure Matrix

Table XVII: DSM of project

Task	0.1	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	4.1	4.2	4.3	4.4	5.1	5.2	5.3	5.4	6.1	6.2	7.1	7.2	7.3	7.4
0.1 Begin Project	0.1																										
1.1 Map out prosthetic rejection causes		1.1																									
1.2 Research commercial products			1.2																								
1.3 Create a hierarchical list of user needs		x	x	1.3																							
1.4 Revise problem statement				x	1.4																						
2.1 Identify used input signals						2.1	x	x	x																		
2.2 Research reliable control approaches						x	2.2	x	x																		
2.3 Research intuitive control						x	x	2.3	x																		
2.4 Select promising concepts						x	x	x	2.4																		
3.1 Look into possible simulation environments										3.1	x																
3.2 Find devices & update budget										x	3.2		x														
3.3 Find (or plan generation) of datasets												3.3															
3.4 Find relevant pretrained models											x		3.4														
4.1 Purchase components										x				4.1													
4.2 Set up software tools															4.2												
4.3 Implement modules with test data												x				4.3											
4.4 Integrate modules together																	4.4										
5.1 Develop testing criteria																		5.1									
5.2 Perform unit tests																		x	5.2								
Perform Integration tests																		x		5.3							
5.4 Identify improvements to be made																			x	x	5.4						
6.1 Implement any improvements from prototype																					x	6.1					
6.2 Test and refine final system																							6.2				
7.1 Document design process																								7.1			
7.2 Refine analysis and proposed solution																									7.2		
7.3 Prepare final repotr																										7.3	
7.4 Prepare poster																											7.4

12.3 Critical Path

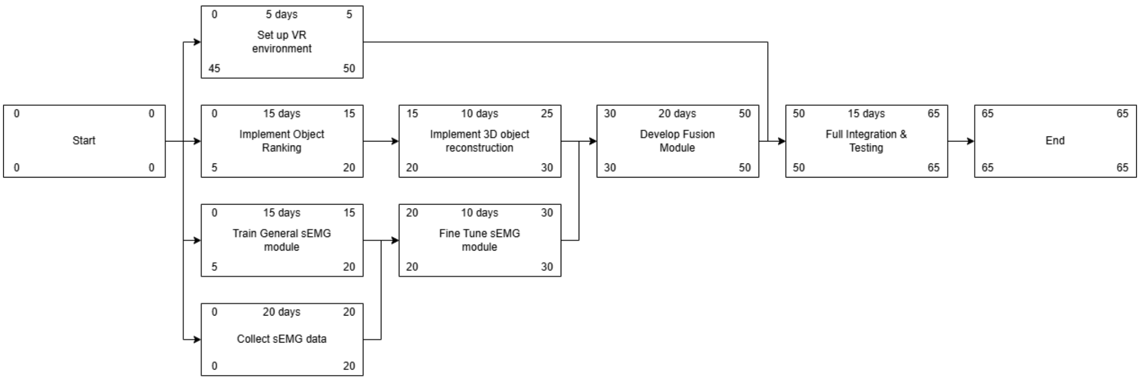

Figure 29: Project critical path identified to be 65 days

12.4 Gantt Chart

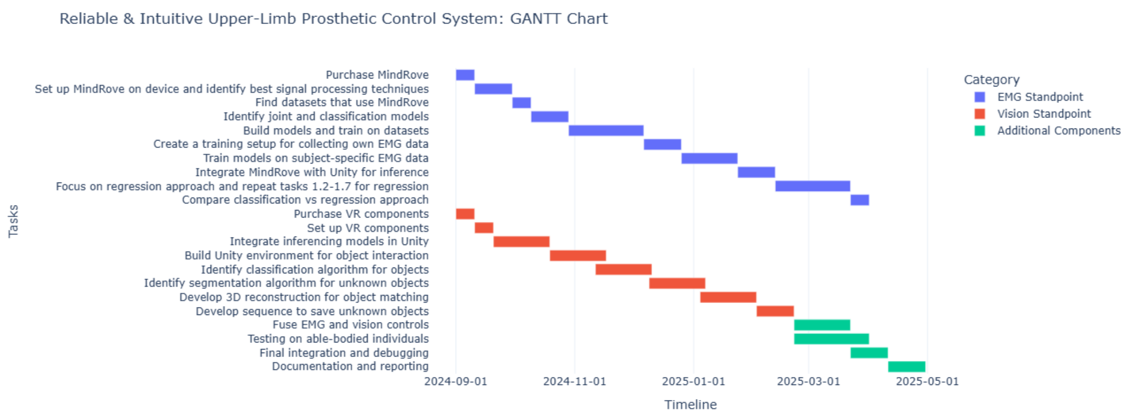

Figure 30: Project Gantt Chart

References

Atzori, Manfredo *et al.*, (2015). "Characterization of a Benchmark Database for Myoelectric Movement Classification", *Transactions on Neural Systems and Rehabilitation Engineering*, Vol. 23 No. 1. (In press), pp. 73–83. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2014.2328495. **available at**: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6825822.

Köllőd, Csaba Márton *et al.*, (2022). "Classification of Semi-Automated Labeled MindRove Armband Recorded EMG Data", *2022 IEEE 22nd International Symposium on Computational Intelligence and Informatics and 8th IEEE International Conference on Recent Achievements in Mechatronics, Automation, Computer Science and Robotics (CINTI-MACRo)*, pp. 000381–000386. DOI: 10.1109/CINTI-MACRo57952.2022.10029540.

Loi, Iliana *et al.*, (2022). "Proportional Myoelectric Control in a Virtual Reality Environment", *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, pp. 1–5. DOI: 10.1109/IVMSP54334.2022.9816252.

Organization, World Health (2024). *WHO standards for prosthetics and orthotics*, Accessed: 2024-11-22. **available at**: https://www.who.int/publications/i/item/9789241512480.

Piscitelli, Daniele *et al.*, (2021). "Prosthesis rejection in individuals with limb amputation: a narrative review with respect to rehabilitation", *Rivista di Psichiatria*, Vol. 56 No. 4, pp. 175–181. ISSN: 2038-2502. DOI: 10.1708/3654.36344. **available at**: http://dx.doi.org/10.1708/3654.36344.

Roche, A. D. *et al.*, (2014). "Prosthetic Myoelectric Control Strategies: A clinical perspective", *Current Surgery Reports*, Vol. 2 No. 3. DOI: 10.1007/s40137-013-0044-8.

Russell, J. and Bergmann, J. H. (2023). "Real-time intent sensing for assistive devices with implications for minimising maintenance", *Prosthesis*, Vol. 5 No. 2, pp. 453–466. DOI: 10.3390/prosthesis5020031.