1 2 9 0

UNIVERSIDADE Đ COIMBRA

Catarina Ribeiro Martins

A Human-Machine Interface Using Augmented Reality Glasses for Applications in Assistive Robotics

Dissertation supervised by Professor Doctor Urbano José Carreira Nunes and Doctor Luís Carlos Artur da Silva Garrote and submitted to the Electrical and Computer Engineering Department of the Faculty of Science and Technology of the University of Coimbra, in partial fulfillment of the requirements for the Master Degree in Electrical and Computer Engineering, specialization in Robotics, Control and Artificial Intelligence.

September of 2023

1290

UNIVERSIDADE Đ
COIMBRA

A Human-Machine Interface Using Augmented Reality Glasses for Applications in Assistive Robotics

Catarina Ribeiro Martins

Coimbra, September of 2023

1 2 9 0

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE Đ
COIMBRA

A Human-Machine Interface Using Augmented Reality Glasses for Applications in Assistive Robotics

Dissertation supervised by Professor Doctor Urbano José Carreira Nunes and Doctor Luís Carlos Artur da Silva Garrote and submitted to the Electrical and Computer Engineering Department of the Faculty of Science and Technology of the University of Coimbra, in partial fulfillment of the requirements for the Master Degree in Electrical and Computer Engineering, specialization in Robotics, Control and Artificial Intelligence.

Supervisor:
Prof. Dr. Urbano José Carreira Nunes

Co-Supervisor:
Dr. Luís Carlos Artur da Silva Garrote

Jury:
Prof. Dr. Paulo Jorge Carvalho Menezes
Prof. Dr. João Pedro de Almeida Barreto
Prof. Dr. Urbano José Carreira Nunes

Coimbra, September of 2023

Acknowledgments

I would like to express my deep appreciation and gratitude to my supervisor, Prof. Dr. Urbano Nunes, for their invaluable guidance, and for supplying me with every material needed for the development of this dissertation.

I am equally grateful to my co-supervisor, Dr. Luís Garrote, whose invaluable assistance was fundamental to completing this dissertation successfully. His expertise and mentorship were invaluable.

I wish to express my deep appreciation to my colleagues in the lab, especially to Miguel, Rute, and Perdiz. Their collaborative spirit was pivotal in this journey.

To my friends, who have been a constant source of encouragement and motivation, I offer my sincerest gratitude. Your support, both academically and personally, has meant the world to me.

To my aunt Isabel, who has consistently been there for me, offering support and encouragement, especially during challenging moments.

My gratitude extends to my siblings, Sérgio and Cristina, who have been a continuous source of inspiration, your belief in me has been a driving force.

Finally, to my parents, Ilda and António, words cannot express the depth of my gratitude. Your inexhaustible support, sacrifices, and encouragement have been the bedrock upon which I built this dissertation. Your love and belief in me have been my greatest motivators. Thank you for everything you have done.

Abstract

The use of Augmented Reality has grown significantly and has been applied in various fields. Simultaneously, assistive technology, such as Brain-Computer Interfaces, has made significant improvements in the quality of life of individuals with severe motor impairments.

In this dissertation, we explored the integration of Microsoft Hololens 2 and P300-based Brain-Computer Interface, for future applications in assistive robotics. The proposed Human-Machine Interface could identify relevant elements for the given scenario, such as doors, tables, or people, depending on the user's objective and the Brain-Computer Interface was utilized to determine the user's intentions.

Using YOLOv5 for real-time object detection and the SORT algorithm for object tracking, our system recognized and consistently tracked objects within the user's field of view. This precision was important for ensuring effective communication with the P300-based Brain-Computer Interface. By integrating the P300-based Brain-Computer Interface, users can interact with detected objects solely through their thoughts. The Brain-Computer Interface achieves this by interpreting neural signals associated with the discrimination of target and non-target objects, enabling hands-free selection and interaction with objects in the environment.

The experiment results showed potential as well as difficulties. Our integrated system showed the potential for hands-free interaction with the environment using the Brain-Computer Interface and Microsoft HoloLens. However, we encountered difficulties, possibly stemming from communication delays, when utilizing the Brain-Computer Interface and Microsoft HoloLens. In the future, strategies to improve accuracy and reduce latency between all modules need to be researched.

In our research, we studied the integration of Augmented Reality, object detection, and Brain-Computer Interfaces based on event-related potentials for future applications in assistive robotics. Although we were able to integrate this technology, we encountered some challenges in integrating Brain-Computer Interface and Microsoft HoloLens, particularly with communication delays. Despite these obstacles, we successfully reduced delays in image acquisition, however, addressing latency in the Brain connection is a complex task that needs to be researched in the future.

Keywords: HMI, Microsoft Hololens 2, BCI, Object Detection, Multi-Object Tracking, Assistive Robotics, SORT, YOLOv5

Resumo

O uso da Realidade Aumentada cresceu significativamente e tem sido aplicado em várias áreas. Ao mesmo tempo, a tecnologia assistiva, especificamente as Interfaces Cérebro-Computador, avançou significativamente na melhoria da qualidade de vida de pessoas com graves deficiências motoras.

Nesta dissertação, criámos uma interface que pode ser combinada com uma Interface Cérebro-Computador para uso em situações de assistência. A Interface Humano-Máquina proposta pôde identificar elementos relevantes para o cenário dado, como portas, mesas ou pessoas, dependendo do objectivo do utilizador, e a Interface Cérebro-Computador determinou as intenções do utilizador.

Utilizando o YOLOv5 para deteção de objetos em tempo real e o algoritmo SORT para o seguimento de objetos, o nosso sistema reconheceu e seguiu consistentemente objetos no campo de visão do utilizador. Esta precisão foi importante para garantir uma comunicação eficaz com a Interface Cérebro-Computador baseada no potencial relacionada a eventos P300. Ao integrar a Interface Cérebro-Computador baseada em P300, os utilizadores podem interagir com objetos detetados apenas através dos seus pensamentos. A Interface Cérebro-Computador alcança isto ao interpretar sinais neuronais associados à discriminação entre objetos-alvo e objetos não-alvo, permitindo a seleção e interação sem usar as mãos com objetos no ambiente.

Os resultados experimentais revelaram-se promissores, no entanto, também enfrentamos algumas dificuldades. O nosso sistema integrado demonstrou potencial para interação sem o uso das mãos com o ambiente, utilizando a Interface Cérebro-Computador e o Microsoft HoloLens. No entanto, encontrámos dificuldades, possivelmente relacionadas com atrasos na comunicação, ao utilizar a Interface Cérebro-Computador e o Microsoft HoloLens. Alcançar detecções precisas, na Interface Cérebro-Computador, foi difícil, requerendo melhorias para um melhor desempenho.

Na nossa investigação, estudámos a integração da Realidade Aumentada, deteção de objetos e Interface Cérebro-Computador baseados no potencial relacionado a eventos P300, para futuras aplicações em robótica assistiva. Embora tenhamos conseguido integrar esta tecnologia, enfrentámos desafios na integração do Interface Cérebro-Computador e do Microsoft HoloLens, nomeadamente relativos a atrasos na comunicação. Apesar destes obstáculos, conseguimos reduzir os atrasos na aquisição de imagens, no entanto, abordar a latência na ligação Interface Cérebro-Computador é uma tarefa complexa que precisa de ser melhorada no futuro.

Palavras-chave: HMI, Microsoft Hololens, BCI, Detecção de Objectos, Rastreamento de vários Objectos, Robótica Assistida, SORT, YOLOv5

"Success is stumbling from failure to failure with no loss of enthusiasm."
Winston S. Churchill

Contents

List of Acronyms

AR Augmented Reality

BCI Brain-Computer Interface

BCW Brain-Controlled Wheelchairs

CNN Convolutional Neural Networks

DEVIS Dynamic Environment-based Visual Interface System

DVI Dynamic Visual Interface

EEG Electroencephalographic

ERP Event-Related Potential

FPN Feature Pyramid Network

HMI Human-Machine Interface

MOT Multiple Object Tracking

NMS Non-Maximum Suppression

RPN Region Proposal Network

SORT Simple Online Real-time Tracking

SOT Single Object Tracking

SSD Single Shot Multibox Detector

TCP Transmission Control Protocol

UDP User Datagram Protocol

YOLO You Only Look Once

List of Figures

List of Tables

1

Introduction

This chapter introduces the motivation behind the development of this work, along with the goals and essential contributions.

1.1 Context and Motivation

Augmented reality (AR) has been growing in the past few years and is expected to continue to grow. In 2022 AR was valued at 38.56 billion dollars and is expected to continue to grow at a compound annual growth rate (CAGR) of 39.8% from 2023 to 2030 [2]. AR technology has applications in various fields, including medical training, interior design, modeling, education, entertainment, and retail industry.

There are many people around the world who face significant difficulties in their daily lives due to severe motor impairments. These individuals often face limitations in their autonomy and mobility, restricting their ability to interact and navigate in their environments. To manage this crucial necessity, the area of assistive technology has witnessed remarkable advancements, particularly in the domain of Brain-Computer Interfaces (BCIs). BCIs offer hope for improving the quality of life for those with severe motor disabilities. Traditional control interfaces, such as joysticks or button switches, have proven to be inadequate for those with limited physical mobility. In response to this critical issue, brain-actuated systems, like brain-actuated wheelchairs, have gained importance within the area of assistive technology, to promote independence and elevate the quality of life for users of assistive platforms [3].

Previous research at the University of Coimbra's Institute of Systems and Robotics introduced DEVIS (Dynamic Environment-based Visual Interface System) [4], a system designed to improve the user experience of a brain-actuated wheelchair. DEVIS features a Dynamic Visual Interface, scene analysis, and a BCI for navigation target selection. The study demonstrated real-time functionality and high accuracy in target selection using the BCI.

The goal of this dissertation is to develop an interface that will be integrated with a BCI, with applications in assistive contexts, such as to aid users with severe motor disabilities in driving robotic wheelchairs. The Human-Machine Interface (HMI) must detect elements of interest for the aforementioned context, such as doors, tables, or humans (for aid requests or interaction). A BCI paradigm will be used to determine the user's intent and goal. In the future, the robotic wheelchair should be able to drive the user towards their intended destination, guaranteeing the user's comfort and safety. Recent research has shifted its focus from static and/or predefined interfaces towards more immersive and dynamic interfaces (some using the Microsoft HoloLens [5]).

During the development of this dissertation, BCI used AR technology to interact with both physical and virtual worlds and explore new methods of displaying feedback to the user. This is important for users to perceive and control their brain activity or shape their communication intentions.

To achieve these objectives, in this dissertation, we used YOLOv5 [6] as a Real-Time Object Detection algorithm and SORT [7] as a Multi-Object Tracking algorithm.

1.2 Proposed Framework

Our research centers on a multi-step process. Initially, we acquire images using the Microsoft Hololens 2, followed by applying an object detector to detect specific elements of interest within the image. These identified elements are then transmitted to the Hololens 2 to be displayed to the user (Part 1). Concurrently, the same information is forwarded to a Brain-Computer Interface (BCI), which plays an important role in object selection (Part 2). It is important to note that while our work contains the image acquisition and BCI integration aspects, the guidance of a wheelchair toward the chosen target object, which represents the third part of the system, does not make part of our research.

The proposed framework, depicted in Fig. 1.1 combines real-time object detection through YOLOv5, multi-object tracking using SORT, and integration with the Hololens 2 augmented reality platform. This framework enhances the autonomy and quality of life of individuals with motor disabilities by facilitating efficient interactions with their environment.

1.3 Objectives and Key Contributions

The proposed study objectives are listed below in chronological order of execution:

1. **Development of the TCP/IP connection between the Computer and the Hololens**
   Establishing an efficient TCP/IP connection between the computer and the HoloLens was an important aspect of our research. Our goal was to optimize timing, enabling seamless and rapid communication between the two devices.

2. **Hololens Camera image acquisition**
   One of the critical challenges encountered during image acquisition was minimizing delays. Many techniques were explored and optimized to guarantee real-time image capture from the Hololens camera while maintaining low latency.

3. **Integration of YOLOv5s**
   We evaluated various object detectors to determine which one best suited our research. Initially, we tested YOLOv3, then transitioned to YOLOv5 and explored different model

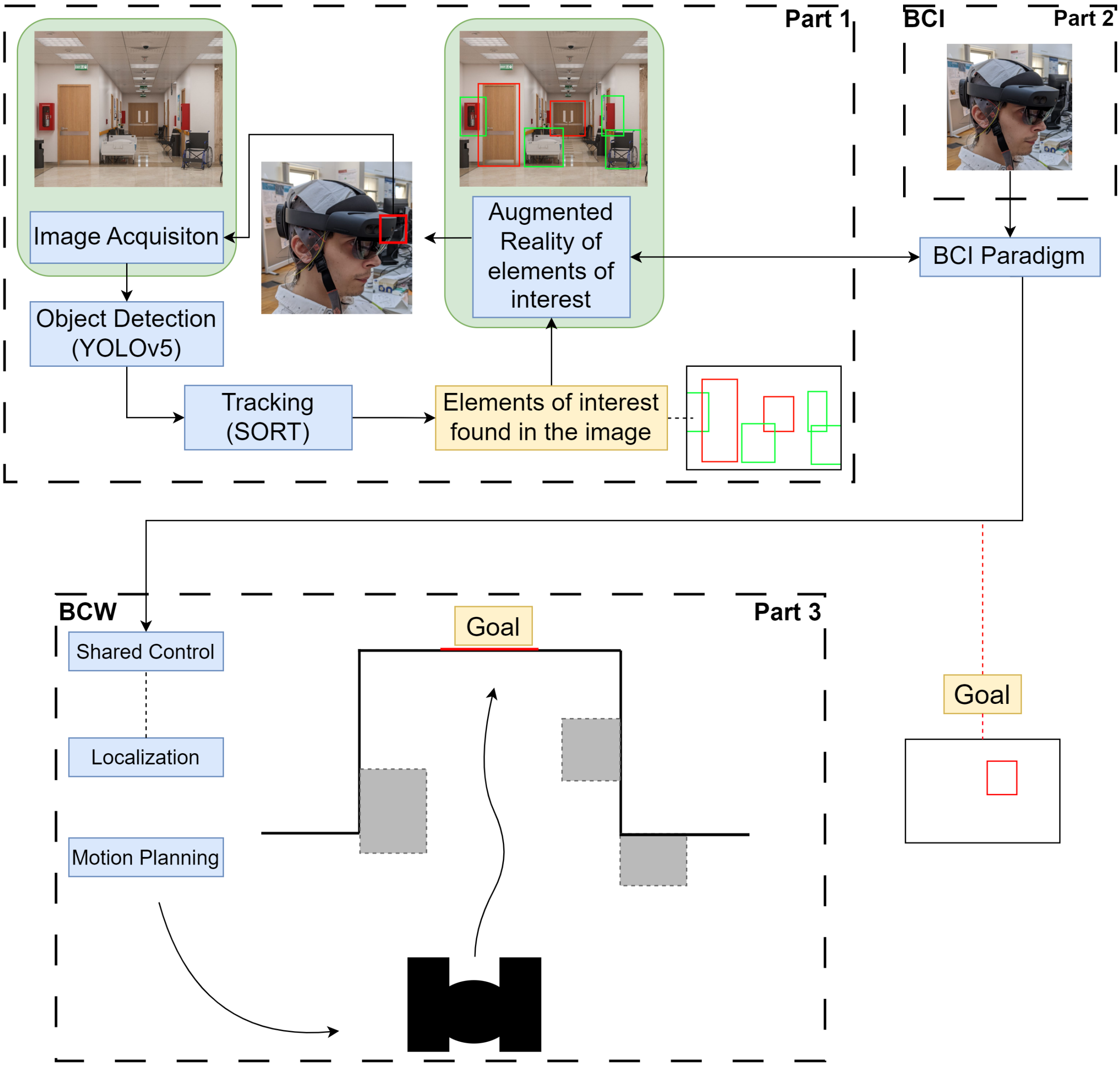


Figure 1.1: Architecture which encompasses the AR interface (Part 1), BCI paradigm (Part 2) and wheelchair navigation approach (Part 3).

variants in the YOLOv5 family. After a thorough analysis, we opted for YOLOv5s as the most appropriate model.

4. **Implementation of SORT considering the detections provided from YOLOv5**
   To improve object tracking accuracy, based on detections from YOLOv5 parameters in the SORT algorithm were finetuned.

5. **Development of the communication architecture to connect to the BCI**
   To integrate our system with the BCI, we implemented a UDP communication architecture. This architecture allowed data exchange between our Computer and the BCI.

6. **Integration of the modules and validation in a real setting**
   Validation of the proposed pipeline/framework was carried out with two volunteers in a

real scenario.

The key implementations and contributions of this study are detailed in the following chapters of this dissertation:

Developed Work (Chapter 4)

This chapter provides an extensive explanation of the methodology used and outlines the various strategies deployed to accomplish the proposed objectives.

Software Tools and Hardware Materials (Chapter 5)

The information about the specific tools is presented in this chapter.

Results and Discussion (Chapter 6)

This chapter presents the results and discussion of the proposed study.

2

Background Material

In this chapter the methods used in the development of this dissertation are described.

2.1 Online Multi-Object Tracking

2.1.1 SORT

The SORT (Simple Online and Realtime Tracking) algorithm is a Multi-Object Tracking algorithm capable of tracking multiple objects in a video [7]. This algorithm consists of three main steps: First, an object detector is used to detect elements of interest in an image. The next step involves predicting the object's position by using a Kalman Filter [8], estimating the new position based on the knowledge of the previous position. Once the new position is observed, the Kalman filter dynamically adjusts its belief.

Finally, the third step involves associating the detected objects across frames to maintain their identity and ensure consistent tracking. This data association problem is solved by using the Hungarian algorithm [9], to link objects from one frame to the next based on their predicted and observed positions. This association step allows SORT to track multiple objects simultaneously while maintaining their identities [10].

2.2 Object Detector

Object Detection is an essential task in the computer vision field, which involves identifying and locating objects in digital images or video frames. This process requires training a computer program or algorithm to recognize specific objects or classes of objects within an image and then outlining them with bounding boxes to show their position in the image. [11]

2.2.1 YOLOv5

YOLOv5 (You Only Look Once, version 5) [6] is a famous deep-learning model, released in 2020 by Glenn Jocher, and is used to detect objects in an image or video. YOLOv5 consists of three parts: The Backbone: CSPDarknet53, the Neck: PANet, and the Head: Yolo Layer. First, the CSPDarknet53 takes the input, an image, to extract features through a series of convolutional layers. YOLOv5 uses a feature pyramid network (FPN) to extract features, helping the model to detect objects of different sizes in the image. The model has multiple detection heads, each responsible for predicting bounding boxes, object classes and object confidences. These detection heads are attached to different levels of the feature pyramid. For each anchor box, the model predicts six values, x_{min}, y_{min}, x_{max} and y_{max}, that represent the bounding box coordinates, object confidence scores and class scores for different object classes. Then, YOLOv5 divides the input image into a grid of cells and each one of these cells is responsible for predicting bounding boxes for objects that are present in that cell. After making predictions at multiple scales, YOLOv5 uses Non-Maximum Suppression (NMS) that discards duplicate detections and maintains the most confident predictions. The output of YOLOv5 consists of a list of bounding boxes, their corresponding class labels and their confidence scores for each detected object in the input image.

There are five versions of the YOLOv5 model, each with variations in size and capabilities. These versions include YOLOv5s (Small), YOLOv5m (Medium), YOLOv5l (Large), YOLOv5x (Extra-Large) and YOLOv5n (Nano). For our research, we decided to use YOLOv5s, the smaller version, due to the requirement of real-time object detection on the HoloLens, known for its limited computational resources, providing an optimal balance between speed and accuracy, providing minimal latency between object detection and augmented reality interactions in the HoloLens environment. During our investigation, we encountered a few challenges while using the YOLOv5s. The main issue we faced was its stability. Due to its smaller model size, YOLOv5s sometimes struggled to detect objects in our BCI and HoloLens setup consistently. This inconsistent detection created problems for our research where accuracy and consistency were important. A representation of the YOLOv5 architecture is presented in Fig. 2.1.

2.3 Brain-Computer Interface

BCI is a growing field that combines neuroscience, computer science and engineering. These systems serve as a direct communication path between the human brain and external devices, such as computers, robotic systems, and in our case, AR interfaces. BCIs allow users to control these devices using their thoughts, translating neural signals into commands. These systems offer a big promise for improving the quality of life for individuals with severe motor impairments or disabilities, as they avoid conventional input methods like keyboards, joysticks or touchscreens [12].

Among the various BCI approaches, P300-based BCI has gained attention due to being non-invasive and having a relatively high accuracy in detecting a user's intentions. They rely on the P300 event-related potential (ERP), a distinct neural response that occurs in the brain when a person recognizes a specific stimulus among a group of stimuli. By detecting the presence or absence of this response, P300 BCI can infer a user's intent, such as selecting an item from a menu or controlling the movement of a wheelchair. These systems hold great promise in scenarios where traditional control interfaces are impractical or impossible to use [13].

In our research, we explored the capabilities of the P300 BCI, integrated with AR, to understand the user's intention. We aimed to provide a more intuitive and accessible for users with severe motor disabilities to interact with the physical and digital worlds.

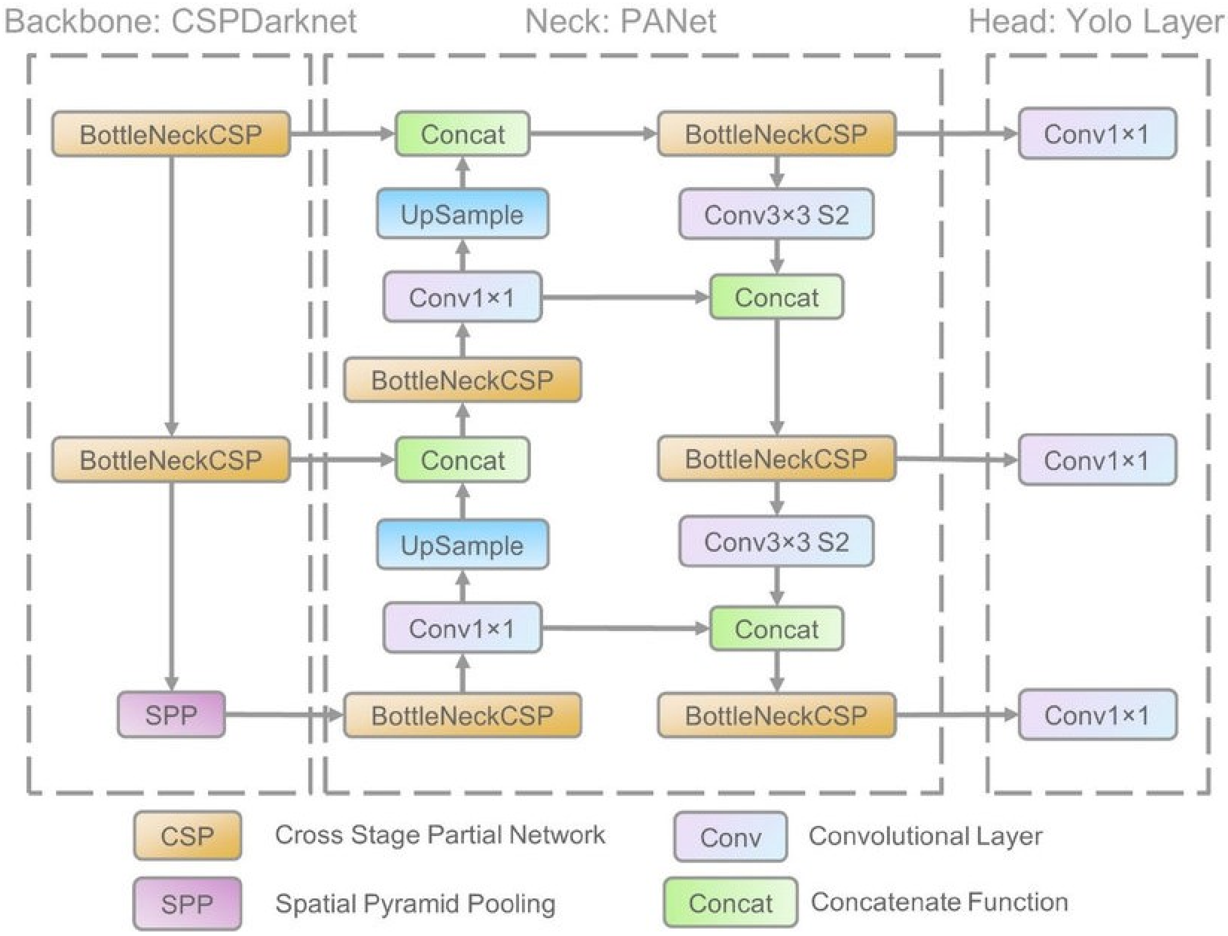


Figure 2.1: Architecture of the YOLOv5 algorithm. This architecture contains three components: the Backbone, CSPDarknet, which initially handles feature extraction from the input data. Next, these features are passed to the Neck, PANet, where feature fusion takes place. Finally, the Head of the network, which is YOLO Layer, serves as the output stage, providing detection results (class, confidence, location, size) [1].

3

State of the Art

This chapter summarizes relevant literature containing the Object Detection, Microsoft HoloLens 2, and the Brain-Computer Interface technologies related to the investigation in this dissertation.

3.1 Object detection

Object detection is one of the key tasks of the computer vision area and has been one of the most popular topics to research in recent years [14]. Object detection must detect objects of interest in an image or video and provide the respective bounding box around the detected object.

In a general way, object detection starts by obtaining an image or video. Then, significant features are extracted using Convolutional Neural Networks (CNNs) - CNNs are deep learning models designed to process images. The extracted features provide a complete representation of the input data, allowing the model to recognize the image's patterns, textures and context. After the feature extraction, the next step is to use an object detection algorithm to locate and classify the objects within the image.

Object detection methods can be divided into One-stage and Two-stage methods [15]. One-stage methods can predict the bounding boxes and probabilities in a single step. They predict the locations and the classes of the objects using default boxes with different sizes. Popular One-stage methods are YOLO and SSD (Single Shot Multibox Detector). On the other hand, Two-stage methods break down the object detection process into two main stages: region proposal and object classification. In the first stage, the algorithm proposes regions that are likely to contain objects (region proposals). These proposed regions are refined and classified into specific object classes in the second stage. Two popular two-stage methods are Faster R-CNN (Region-based Convolutional Neural Networks) and Mask R-CNN.

YOLO was first introduced by Joseph Redmon, et al. in 2015. The main purpose of YOLO is to perform real-time object detection predicting both object bounding boxes and class probabilities in just one stage through the neural network.

In the past, object detection methods with two-stage methods were costly in terms of computation as they depended on region proposals and object classification, however, YOLO introduced a one-stage approach that was faster and more accurate, causing a significant impact in the area of object detection.

YOLO uses a deep convolutional neural network to process the complete image at once. The network divides the image into a grid and predicts bounding boxes and class probabilities for each grid cell. Each bounding box is associated with a specific confidence score that indicates the confidence that the model has in the presence of an object within that box, moreover, YOLO performs non-maximum suppression to select the most appropriate bounding box.

In [16], Alexey Bochkovskiy et al. introduced YOLOv4, which, by that time, represented a significant improvement over previous versions of the YOLO model. YOLOv4 achieved an optimal balance between speed and accuracy with improvements in the backbone architecture, using the CSPdarknet53. Instead of using FPN, like in the YOLOv3, the authors used PANet for parameter aggregation across various backbone levels for distinct detector levels. Data augmentation was utilized to increase the input data's variability so that the capacity of the object detection model, to variations in images obtained from various environments, is improved. The study concluded that YOLOv4 was the fastest and most accurate detector in terms of accuracy and speed when compared with the YOLOv3, EfficientDet, ATSS, ASFF, and CenterMask.

YOLOv5 was released in 2020 by Glenn Jocher. Compared with the YOLOv4, the YOLOv5 focused on reducing the model size, increasing speed, and improving ease of use. For bounding box prediction, this algorithm adopted an anchor-free approach, meaning that the network directly predicts the bounding box coordinates without relying on predefined anchor boxes. In what concerns the backbone, YOLOv5 uses a Focus structure with CSPdarknet53 as a backbone, and like the YOLOv4, it also adopts the CSP structure, and the neck segment acquires the FPN + PAN structure [17]. The Focus layer was first introduced in YOLOv5, this layer replaces the first three layers in the YOLOv3 algorithm. When using a Focus layer, there are several benefits that can be gained, including a reduction in the amount of the required CUDA memory, a decrease in the layer size, and an increase in both forward and backpropagation [18].

Siddhi Chourasia et al. conducted a study [19] on three object detectors, YOLOv4, YOLOv5, and YOLOv7, which were compared to determine their performance in detecting safety helmets within an image to identify which of these detectors revealed the best performance in this task. The authors concluded that in terms of overall performance, YOLOv7 achieved the highest average accuracy with 96.4%, beating YOLOv5, which achieved 95.1%, and YOLOv4, with 93.6%.

SSD (Single Shot Multibox Detector) was introduced in 2016 by Wei Liu et al. [20]. Similar to the YOLO, SSD has the ability to perform object detection in a single step, making this detector efficient and ideal for applications that demand real-time processing. SSD takes a different path when compared to traditional methods that use region proposal algorithms, instead, it uses a convolutional method and a single neural network that predicts the object bounding boxes and their class confidence directly from the image. Then, the network generates a set of default bounding boxes, anchors, which will cover different object sizes and shapes by varying in scales and aspect ratios, resulting in an improved performance. The authors demonstrated that when given the VGG-16 base architecture, SSD was revealed to be superior in terms of accuracy and speed. The SSD512 model had a beer accuracy compared to the Faster R-CNN on PASCAL VOC and COCO datasets, being three times faster. The SSD300 model ran faster than the YOLOv1, at 59 FPS, and with superior detection accuracy.

Jeong-ah Kim et al. [21] compared three object detectors, Faster-RCNN, YOLOv4, and SSD. The purpose of the study was to identify different types of vehicles. The authors trained these algorithms using a dataset of automobiles and analyzed their performance to determine the best model for recognizing vehicles. After analyzing those object detectors, it was determined that YOLOv4 had the best performance in terms of both speed and accuracy for the given task. Although Faster R-CNN was found to be the fastest among RCNN models, the use of a CNN resulted in a low FPS. The SSD was considered fast but compared to the YOLOv4 had a lower accuracy because of its use of a lighter architecture (MobileNet-v1). The authors also mentioned that introducing FPN (Feature Pyramid Network) to the YOLOv4, improved the model to detect small objects.

Faster R-CNN is an object detector model, introduced in 2016 by Shaoqing Ren, et al. [22], that combines region proposal network (RPN) and Fast R-CNN into a single network by sharing their convolutional features [23]. The RPN is a method that is used to generate region proposals by sliding a small network over the convolutional feature map, making the proposal process efficient. Then, these proposals are refined and classified by the Fast R-CNN algorithm, sharing convolutional features with the RPN reducing the computation, and improving the performance. Faster R-CNN is known for its state-of-the-art performance on benchmark datasets like COCO and Pascal VOC, making this an effective solution for different real-world applications. The combination of RPN and Fast R-CNN simplifies the object detection pipeline and improves accuracy and speed.

In [24] the authors studied two object detectors, YOLOv5, a well-known one-stage method, and Faster R-CNN, a multi-stage method, for applications in autonomous machine vision. In this study, spacecraft images are used as training data, with labels for solar panels and satellite bodies, and the testing process includes capturing videos of a target satellite under different conditions and labeling them for similar characteristics. After training both algorithms, Faster R-CNN outperformed YOLOv5, although YOLOv5 offered a faster inference rate. This study suggests YOLOv5 as the preferred real-time object detector due to its speed.

Mask R-CNN (Mask Region-based Convolutional Neural Network), introduced in 2017 by Kaiming He et al., is an object detector that extends Faster R-CNN by adding a pixel-wise segmentation component to its capabilities, this means that, in addition to predicting the bounding box and class labels for an object in an image or video, this detector can also predict object masks, which accurately delineates the boundaries of objects at a pixel level, making this detector a powerful instance segmentation model [25].

In [26], Wei Li et al. created two COCO datasets based on the Baidu AI insect detection Dataset and IP102 Dataset (a Large-Scale Benchmark Dataset for Insect Pest Recognition) and compared YOLOv5, Faster-RCNN and Mask-RCNN detectors, on the two coco datasets, to find which one is more efficient for insect detection. The authors conclude that Yolov5 for insect pest detection on Baidu AI dataset with simple backgrounds is a better option due to its accuracy of over 99%, having a computational speed faster than Faster-RCNN and Mask-RCNN, but when applied to the IP102 dataset, its accuracy drops to around 97%, while Faster-RCNN and Mask-RCNN maintained accuracy of 99%.

3.1.1 Object Detection and Tracking

Object detection and tracking are essential tasks in computer vision, allowing the detection, recognition, and subsequent tracking of objects in images or frames of a video [27]. Because of security measures, object detection and tracking have been important by providing surveillance, threat identification, and response capabilities, indispensable tools for providing the safety of people and properties [28].

Gioele Ciaparrone et al. [29] conducted a study on the use of Deep Learning for Multiple Object Tracking (MOT) in single-camera videos. This study examines the four essential stages that make up a standard MOT pipeline: detection, feature extraction, affinity, and association. They explore how Deep Learning enhances detection and feature extraction at each stage. This study highlights the importance of detecting high-quality features and utilizing Convolutional Neural Networks (CNNs) for extracting appearance features. It also highlights the need to adapt Single Object Tracking (SOT) trackers and global graph optimization techniques to address Multiple Object Tracking (MOT) challenges, which is made possible through the integration of Deep Learning.

Ricardo Pereira et al. [30] proposed a study focused on Multi-Object Tracking (MOT), more specifically, SORT [7] and Deep-SORT [31], for applications in assistive mobile robot navigation. The authors present new data association cost matrices for the SORT algorithm based on intersection over union, Euclidean distances, and bounding box measurements to improve object detection accuracy across frames. YOLOv3 [32] is utilized for the object detection and classification of the objects in the frames. After that, the detections are then used as inputs for the Multi-Object Tracking (MOT) evaluation study, to track and associate the objects across frames. The SORT method achieved higher results of accuracy and precision when compared to the Deep-SORT method. The proposed A data association metric achieved the best performance on both evaluated object tracking methods showing a significant improvement on the MT evaluation metric, which could be crucial to successful navigation tasks on robotic platforms. As expected, results showed that the object tracking overall performance has a high dependency on the object detector performance. The SORT is faster than the Deep-SORT, reaching 50 FPS on the overall pipeline (YOLOv3 + SORT).

3.2 Augmented Reality

3.2.1 Augmented Reality and Object Detection

Augmented Reality (AR) allows users to preview virtual items in the real world [33]. AR has applications from mobile apps to smart glasses, in many industries like medicine, entertainment, military and industrial manufacturing [34]. Augmented reality and object detection have many applications when combined, such as recognizing and detecting objects in the real world as shown by Haythem Bahri et al. [35], which proposes an implementation of an object detector, YOLOv3, in the Microsoft Hololens 1 glasses for detect and recognize objects. The authors established a TCP/IP connection between the Hololens, as a client, and the desktop as the server, to process the object detection on a desktop. They concluded that the YOLOv3 presented an improved result, with 5 fps, when compared to the previous version of YOLO.

Another example of integrating augmented reality with object detection is the utilization of the YOLOv5 algorithm for the identification and interpretation of American Sign Language, with the purpose of including people with speech or hearing difficulties, in verbal communication. [36] The proposed solution utilizes YOLOv5 to detect the numbers and alphabets, corresponding to each gesture, captured through a camera. To train and evaluate the model, the MU HandImages ASL dataset [37], which achieved 95% precision, was used.

3.3 Brain-Computer Interface

Brain-Computer Interface (BCI) can establish communication and control between the brain and external devices [38]. The field of BCI has been growing, with multiple research studies being conducted, for example, a BCI that enables communication and control in the metaverse by skin touch [39] and a BCI that can control a robotic arm [40].

Some research, concerning the BCI has been carried out at the Institute of Systems and Robotics of the University of Coimbra, are explained next. In order to improve the user experience of a brain-actuated wheelchair, Ricardo Pereira et al. [4] developed a DEVIS. The main objective of this system is to create an intuitive approach for selecting navigation targets, particularly for individuals with severe motor disabilities. This system consists of three main components: a Dynamic Visual Interface (DVI), that presents a possible navigation goal in multiple visual cues, an RGB image-based perception module for scene classification, object detection and object tracking, and a P300-based BCI used to select the navigation target. The authors concluded that the proposed DEVIS successfully achieved real-time functionality, allowing the user to interact without any delays or latency problems. Users were also able to select the navigation target with high accuracy using the BCI.

Another research, also conducted at the ISR of the University of Coimbra, proposes a new approach that improves the usability and reliability of Brain-Controlled Wheelchairs (BCWs) for users with severe motor disabilities. The authors' main issues were the low reliability of decoding electroencephalographic (EEG) signals and the high cognitive workload associated with constant wheelchair control. To overcome these challenges, Aniana Cruz et al. [41] introduced a multi-component approach, including a self-paced P300-based BCI, that would allow users to change between control and noncontrol states without needing extra tasks, simplifying the user experience, while a dynamic time-window command allowed balancing the reliability and speed of the BCI system. Seven healthy participants and six participants with motor disabilities validated the system. Every participant could successfully control the BCI with an accuracy greater than 93%. The authors conclude that the proposed approach improves the usability and reliability of BCW.

3.4 Assistive Robotics

There are some studies to explore the application of assistive r

obotics with HoloLens glasses, including its application in a robotic wheelchair. In [42], the authors propose an augmented reality system making use of Microsoft HoloLens glasses for robotic wheelchair navigation addressing the challenge faced by disabled people who use robotic wheelchairs with built-in assistive features, such as shared control. Usually, users struggle to form a mental model of their wheelchair's behavior in diverse environmental conditions.

The system would display visual feedback to the patient as a way of explaining the underlying dynamics of the wheelchair's shared controller and its predicted future states. This work shows experimental evidence that additional effort should be taken to limit the amount of alerts presented to the users so as to not overwhelm them. This research, at that time, represented pioneering research in integrating AR headsets with robotic wheelchairs, showing valuable insights for design considerations. It also highlighted the importance of positioning virtual objects in visible locations, away from the mobile base. The results highlight how AR cues, such as a virtual rear-view mirror, can be useful elements in designing robotic wheelchairs. Implementing these cues can improve the process of retrieving information and decrease the mental effort required to operate a robotic wheelchair, leading to a better experience for the user.

3.5 HoloLens and BCI integration

Table 3.1 presents a comparison of findings from three research articles that explore the integration of AR with BCI technologies. Each article focuses on different aspects of this integration, including the choice of AR head-mounted displays (HMDs) and BCI technologies.

Table 3.1: Comparison of Articles Using HoloLens and BCI.

Article	**HoloLens Version**	**BCI Technology**	**Research Objective**	**Key Findings**
Yasmine Mustafa et al. [43]	HoloLens 1	SSVEP-based BCI	BCI-AR Integration	Proposed an adaptive ensemble classification system to handle inter-subject variability, achieving a mean accuracy of 80% on PC and 77% on HoloLens. Demonstrated robustness to head movements during SSVEP commands.
Dennis Dietz et al. [44]	HoloLens 1	P300-based BCI	Mixed Reality Interaction	Proposed a mixed reality interface using P300 brain signals. Demonstrated controlling real-world devices like a TV remote, suggesting potential applications for people with physical disabilities.
Pasquale Arpaia et al. [45]	Epson Moverio BT-350, Oculus Rift S and HoloLens 1	SSVEP-based BCI	Performance comparison of AR head-mounted displays in SSVEP-based BCI	Choice of AR HMD significantly impacts SSVEP detection. HoloLens and Oculus Rift S showed better classification accuracy than Epson Moverio BT-350.

4

Developed Work

In this chapter, we provide a comprehensive presentation of the work undertaken to achieve the objectives of the proposed dissertation.

4.1 Methodology

In this dissertation, our main objective was to integrate object detection, augmented reality and brain-computer interface technologies so that users with severe mobility issues could identify and interact with objects of interest within an environment. For the task of object detection, the YOLOv5 model was chosen due to its capacity for real-time processing and accuracy. The integration of augmented reality, through Microsoft Hololens 2, implicated the development of a program to establish a TCP/IP (Transmission Control Protocol/Internet Protocol) connection between the Hololens 2 device and our computer, to project the results of YOLOv5 into the Microsoft Hololens.

4.2 Hardware and Software Configurations

Configuring both hardware and software was essential to successfully connect the computer and the Hololens 2. Those configurations are explained next.

4.2.1 Hardware Configurations

In what concerns the hardware, we had to configure the computer and the Hololens 2. On the computer, it was necessary to disable the private network firewall as it could interfere with the Hololens connection. We also had to create a hotspot, with the computer, so that Hololens could successfully connect to the internet.

For the Microsoft Hololens 2 part, it was necessary to turn *”Developer Mode”* On, on settings, to enable the deployment of the Visual Studio code.

4.2.2 Software Configurations

To be able to deploy the client code into the Hololens we had to configure both Visual Studio and Unity. Next, the steps to deploy are presented:

- Create a project in Unity. Within the project create a game object and a script;
- Place the client's code in the script;
- After creating the project, go to File > Build Settings;
- On the Platform type select 'Universal Windows Platform';
- Click on 'Add Open Scenes.' and check 'Development build' and 'Script Debugging';
- On Player Settings check: 'InternetClient', 'InternetClientServer', 'PicturesLibrary', 'PrivateNetworkClientServer', 'SharedUserCertificates', 'Webcam', 'Microphone', 'Location', 'HumanInterfaceDevice', 'Objects3D', 'SpatialPerception', 'RemoteSystem' and 'GazeInput';
- Create a 'Builds' folder within the project and build it inside that folder;
- In the 'Builds' folder, open the Visual Studio solution;
- Build configuration: 'Release'; Processor: 'ARM64'; Add the IP of the glasses in Debug Properties.

4.3 Hololens - Server Communication

To exchange information between the Microsoft Hololens 2 and the Computer, first, it was necessary to connect both devices to the same network. After that, we established a TCP/IP connection between the server (computer) and the client (Hololens 2). This task involved specifying the port in the server-side code, while the IP address of the computer and the port information were configured in the client-side code. TCP/IP works by breaking the information into packets, sending it to a known IP address and using port numbers to identify specific services. We used this protocol for data communication because this protocol ensures that data is received when compared to UDP [46]. Then, the server received the image from the client and processed it using YOLOv5 and SORT to detect and track relevant objects. After detecting an object, YOLOv5 sends the coordinates to Unity to create bounding boxes using a specific function.

The diagram in Fig. 4.1 represents the data exchange between the server and the client over TCP/IP.

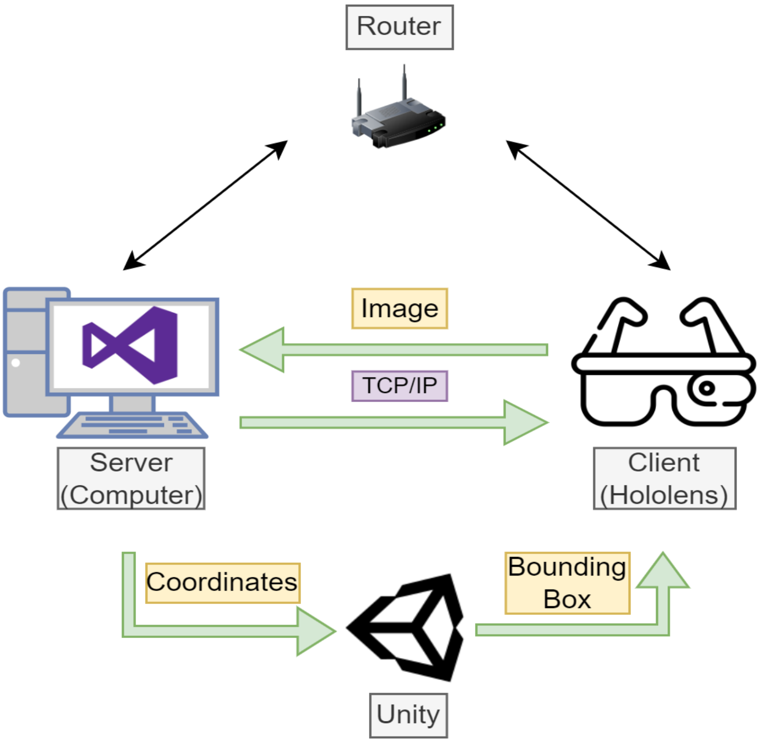


Figure 4.1: Data exchange between Microsoft Hololens and computer over TCP/IP.

4.4 Image Acquisiton

For image acquisition, we first tried using the UnityEngine.Windows.WebCam.PhotoCapture API for capturing photos with the Hololens. Our initial concept involved sending images from the client to the server, using this API, to be processed, however, we eventually opted not to use this approach because we faced significant delays in acquisition. Instead, on the server side, we would directly access the images through OpenCV's VideoCapture function using a link. The link contained the username and password of the Windows Device Portal along with the IP address of the Hololens. This approach improved real-time processing and provided faster access to the image data. The real-time camera is accessible through the following link:

```
https://username:password@IP_Adress/api/holographic/stream/live_high.mp4?holo=
false&pv=true&mic=false&loopback=false&RenderFromCamera=false
```

4.5 Object Detection + Object Tracking

For both Object Detection and Object Tracking were used YOLOv5 [6] and SORT [7] algorithms, respectively, on the server side, as mentioned above.

In our thesis, we were inspired by the work of Guilherme Carvalho et al. [47], who integrated YOLOv3 with SORT (a tracking algorithm) to achieve real-time object tracking. After conducting tests, we discovered that YOLOv5 produced better results in terms of both accuracy and speed. We experimented with both YOLOv5m and YOLOv5s but chose YOLOv5s for our research project involving the Hololens. This decision was based on the fact that YOLOv5s is more lightweight and can operate effectively on the Hololens's limited hardware resources, despite its slightly lower accuracy when compared to YOLOv5m.

We also utilized the SORT algorithm that Guilherme implemented, but we changed one of the parameters, *min_hits*, which is the number of minimum consecutive track matches to consider as a track as valid. Guilherme used $min\_hits = 3$, but we used $min\_hits = 5$. A higher *min_hits* value will make the tracker more robust to noise, but it will also make it less likely to detect new objects. In our case, we chose to use $min\_hits = 5$ because we wanted to make the tracker more robust to noise, since Hololens, has a relatively small field of view and can be affected by reflections and other kinds of noise. By increasing the *min_hits* value, we were able to reduce the number of false positives caused by noise. However, increasing this value also made it less likely to detect new objects. This is because the tracker would need to see an object for at least 5 consecutive frames before it would consider it a valid track.

The server starts by specifying a TCP port for communication, which is set to port 40001. It initializes an empty result list that will store the bounding box coordinates among other parameters. Afterward, it starts a pre-trained YOLOv5 Small model object detection on a GPU if available. The server then enters an infinite loop that receives continuous video frames from the Hololens and performs object detection using the YOLOv5. To simplify the processing by the YOLOv5 algorithm, we resized the input image to dimensions of 416×416 pixels. The SORT algorithm is then applied to track objects across frames. After this, we have two equations that modify the bounding box coordinates. These calculations transform the bounding box coordinates from the format $[x1, y1, x2, y2]$ (where $x1$ and $y1$ represent the top-left vertex, and $x2$ and $y2$ represent the bottom-right vertex) to the format $[x, y, width, height]$ since they are more convenient for processing.

$$width = x_2 - x_1 \tag{4.1}$$

$$height = y_2 - y_1 \tag{4.2}$$

Equation 4.1 subtracts the x coordinate of the top-left vertex of each bounding box from the x coordinate of the bottom-right vertex. It calculates the width of each bounding box. Similarly, Equation 4.2 subtracts the y coordinate of the top-left vertex of each bounding box from the y coordinate of the bottom right corner. It calculates the height of each bounding box.

The tracked objects' positions and IDs are collected into a results list. The code also listens for a UDP signal, coming from the BCI, that indicates if the user selected any target, this value is also added to the list. Depending on this signal, the bounding boxes of objects are consequently colored. Additionally, it constantly updates the detected and tracked objects' information and sends this information to a client over TCP/IP, to be displayed on the Hololens.

Algorithm 1 outlines the key steps involved in the server code, from initialization to continuously processing video frames and performing object detection while utilizing the SORT algorithm for object tracking.

Algorithm 1: Server side Pseudocode

```
Load YOLOv5 model
Initialize SORT algorithm
Connect to client through TCP/IP
Receive Video from the camera with OpenCV
while True do
    Get frames from the video
    Resize frames to $416 \times 416$ pixels
    Perform object detection using YOLOv5
    Obtain bounding box coordinates $[x, y, w, h]$
    Calculate confidence score, class, and class name for detected objects
    Run the SORT algorithm for object tracking
end while
```

4.6 Bounding Box Display

To project the bounding boxes onto the Hololens, it was necessary to make use of the Unity platform. As explained above, Unity is a game development platform that besides being used to create video games and mobile applications, can also be useful for applications in Microsoft Hololens 2. For that matter, we made use of Unity to deploy those bounding boxes, through a *GameObject*. GameObjects are considered the most important concept in Unity Editor because they store the fundamental elements of a game or application scene [48].

After the processing stage, which involves YOLO and SORT, the server sends, to the client, a list containing the coordinates of the detected objects, x, y, w (width) and h (height). Since these coordinates are relative to the image with dimensions 416 pixels in width and 416 pixels in height, we must transform these coordinates to match the scale and dimensions of the Hololens. Equations 4.3 and 4.4 represent the transformation of x and y, respectively. fx and fy correspond to an offset determined by iterative adjustments to align precisely with the Hololens' field of view because we could not find the transformation/calibration matrix between the physical camera and the in-game camera. The values chosen were $fx = -480$ and $fy = -290$.

$$x = fx + sx \times \frac{x \times aspectW}{1280.0f} \times Camera.main.pixelWidth; \tag{4.3}$$

$$y = fy + sy \times \frac{y \times aspectH}{720.0f} \times Camera.main.pixelHeight; \tag{4.4}$$

sx and sy, which also appear in these equations, represent a scaling factor applied to x and y, these scaling factors also were determined through iterative adjustments. The values chosen were

$sx = 1.80$ and $sy = 1.40$. Following this transformation, we determined the aspect ratios for both width and height, $aspectH = \frac{720.0f}{416.0f}$ and $aspectW = \frac{1280.0f}{416.0f}$. Both aspect ratios are calculated as the ratio of the width and height of the desired screen resolution, 1280 and 720 pixels, to the reference width, 416 pixels. The purpose of these transformations is to map or scale coordinates from one coordinate system to another. Then we can use the API *Camera.main.pixelWidth* and *Camera.main.pixelHeight*, to get the width and height of the Camera viewport in pixels.

Similar to x and y, w and h are computed using the same approach, with the exception of the offset. Because Unity uses a left-hand coordinate system where Y is pointing downwards, equation 4.6 has a negative sy to reverse the direction of the Y-axis because, without this negative sign, increasing values of h would move coordinates downward on the screen.

$$w = sx \times \frac{w \times aspectW}{1280.0f} \times Camera.main.pixelWidth; \tag{4.5}$$

$$h = -sy \times \frac{h \times aspectH}{720.0f} \times Camera.main.pixelHeight; \tag{4.6}$$

To draw the bounding box, we had to create four vertices. Unity takes the screen coordinates, x and y, and the depth, p, and returns a Vector3 with the equivalent world coordinates. Depth represents the distance to the camera.

Bottom left 4.7, top left 4.8, bottom right 4.9 and top right 4.10 are the 4 vertices of the bounding box. After testing, we found that a depth value of 0.35 provided the best visual representation for our bounding box.

$$bottomleft = (x, y) \tag{4.7}$$

$$topleft = (x, y + h) \tag{4.8}$$

$$bottomright = (x + w, y) \tag{4.9}$$

$$topright = (x + w, y + h) \tag{4.10}$$

After creating the vectors, we developed a function that accepted the start vector, end vector, and a bounding box color as parameters. Inside the function, a GameObject is created with these parameters, and the color of the bounding box is determined based on the value received from the BCI.

The P300-based BCI allows the user to select, by flashing the DVI (Dynamic Visual Interface System), visual cues associated with each potential target with the user's intent. These potential

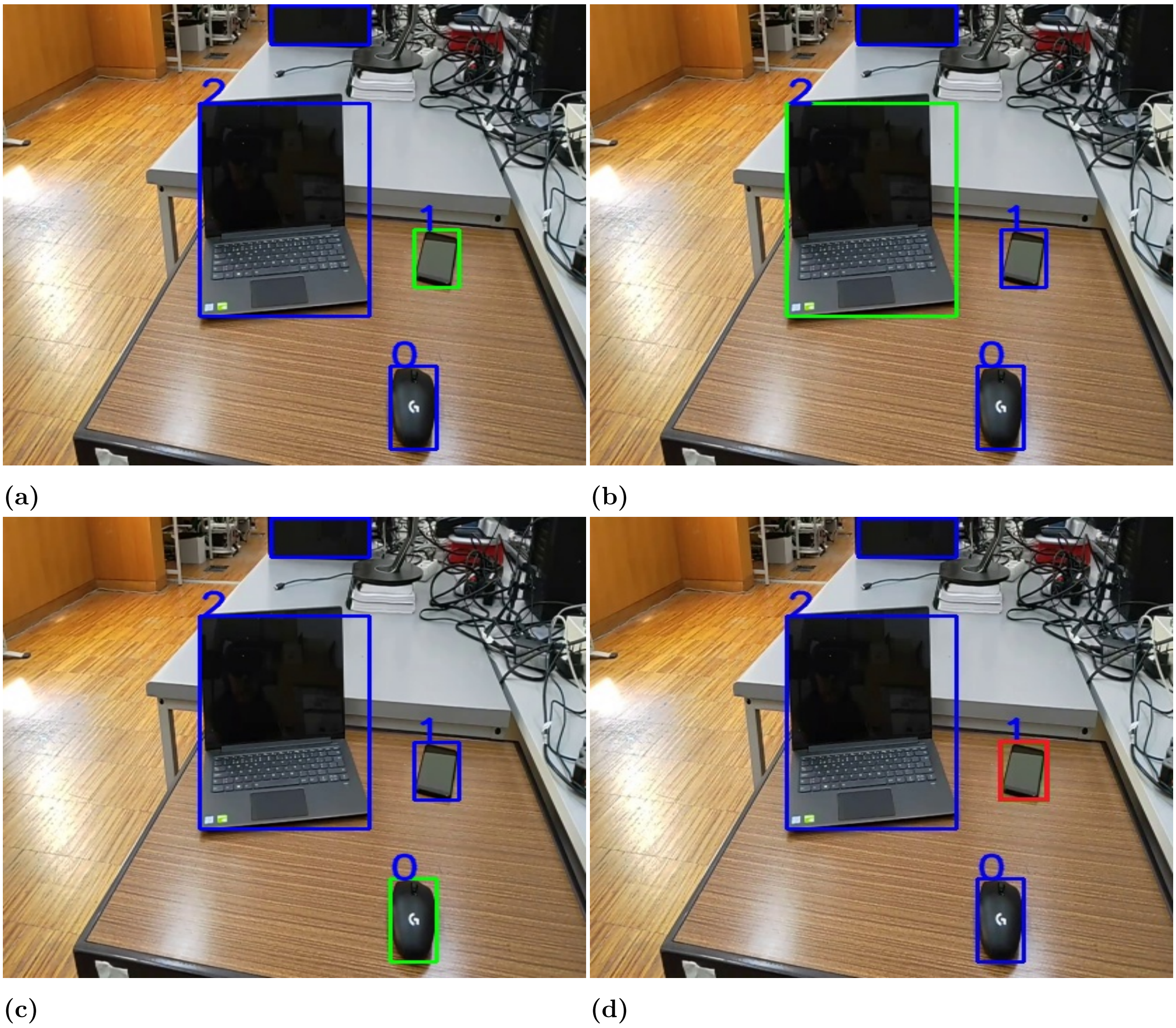


Figure 4.2: Visual stimulus 4.2a, 4.2b, 4.2c, and selection of desired target 4.2d

targets randomly flash by smoothly transitioning the color of the bounding box from blue to green, as seen in Fig. 4.2a, Fig. 4.2b, and Fig. 4.2c. However, once a user's intention is detected, the bounding box of the selected target turns red, Fig. 4.2d. In this case, it's important to note that the number of potential targets can't exceed 9. If the object detector detects more than 9 objects in an image, any additional detections beyond the initial 9 are eliminated.

When using the self-paced mode, a constant stream of images is displayed with overlaying bounding boxes. If the BCI control state is detected, the image will freeze and the visual cues will begin to flash in accordance with the oddball paradigm. This paradigm involves presenting a sequence of stimuli that includes a rarely appearing ”target stimulus” and a more frequently appearing ”non-target” stimulus. Once the BCI detects the intended target, it will be highlighted in red and the image stream will resume [4].

Algorithm 2 presents the pseudocode for color selection based on BCI commands and object tracking.

4.7 Brain-Computer Interface Communication

In this section, we explore BCI technology and its role in developing our framework. The P300 BCI is a system that allows users to establish communication and control over devices using their brain activity, Fig. 4.3a. This technology integrates with AR and will be explored in more detail next.

4.7.1 BCI Setup

In our experiment, we placed a computer, a smartphone, and a computer mouse on a table in front of the user, as seen in Fig. 4.3a. We used this setup as the main focus of our research, monitoring the user's interactions with the objects. To achieve this, we combined the use of the HoloLens with the P300 BCI. The HoloLens displayed bounding boxes around these objects, highlighting them, as seen in Fig. 4.2a, Fig. 4.2b and Fig. 4.2c. By monitoring the user's brain activity through the P300 cap BCI, we could determine which object the user pretended to select. Figure 4.3 illustrates a participant prepared for the experiment, wearing both the BCI and HoloLens.

4.7.2 Integration with Augmented Reality

To integrate the BCI with the HoloLens, we needed to connect the BCI system to the same network as the HoloLens and computer. This network connectivity was important in establishing communication that allowed data to flow between the BCI, HoloLens and the computer. The server would send to the BCI the number of boxes that were detected within the environment, this number was limited to a maximum of nine boxes. Subsequently, the BCI processed the received data and performed the task of allowing the user to select a specific box from among the detected boxes. The user's intent was directly reflected in this choice. In order to maintain the accuracy of communication and avoid the risk of losing data, the BCI would send back to the server the command it received. This confirmation step was implemented because data can sometimes be lost in the communication process.

Figure 4.4 represents the data exchange between the Hololens, computer and the BCI over both TCP/IP and UDP.

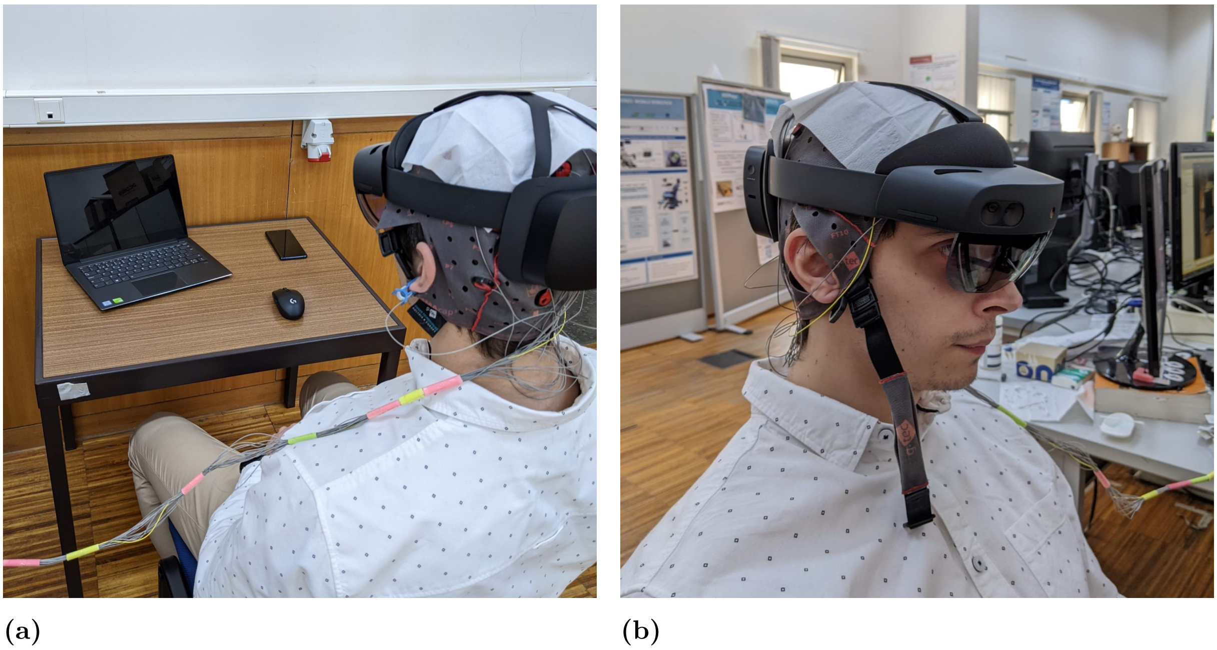


Figure 4.3: Experimental Setup: A comprehensive view of the experimental setup showcasing the BCI integrated with AR. The scene includes a selection of real-world objects ready for detection.

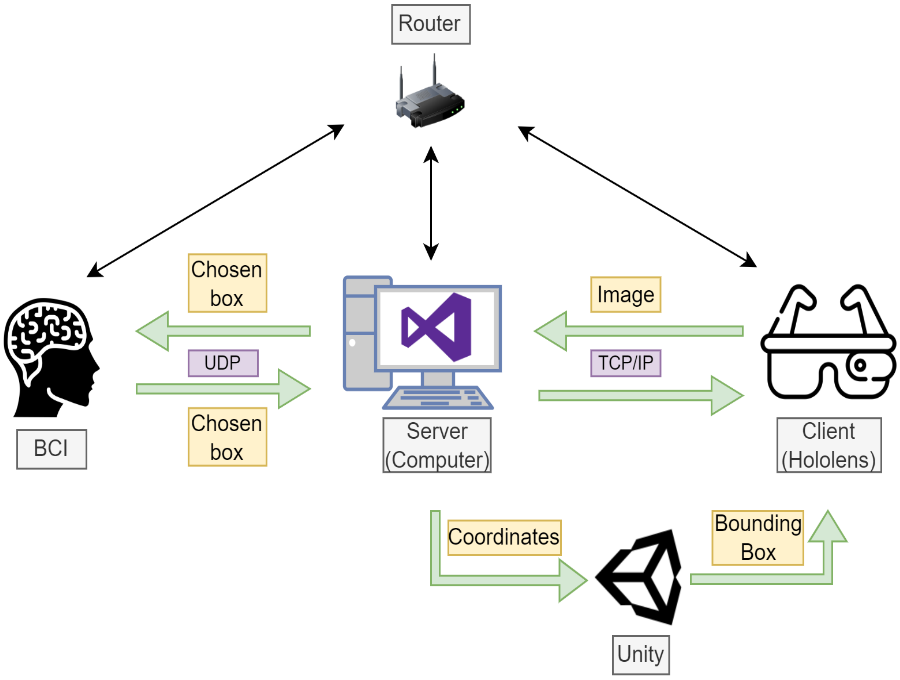


Figure 4.4: Data Exchange Between Microsoft Hololens, Computer and BCI over TCP/IP and UDP.

Algorithm 2: Pseudocode for color selection based on BCI commands and tracked objects.

```
$flash \leftarrow$ false
$lastBCICommandTrackID \leftarrow -1$
$ct \leftarrow 0$
$modified \leftarrow$ false
foreach $boundingBox$ $in$ $bbs$ do
    if $BCICommand = -2$ then
        $modified \leftarrow$ true
        if $flash = true$ then
            if $tracking\_id = lastBCICommandTrackID$ then
                $color \leftarrow$ green
            else
                $color \leftarrow$ blue
            end if
        else
            if $tracking\_id = lastBCICommandTrackID$ then
                $color \leftarrow$ red
            else
                $color \leftarrow$ blue
            end if
        end if
    else
        if $BCICommand \mod 10 = 0$ then
            if $ct = BCICommand$ then
                $flash \leftarrow$ true
                $lastBCICommandTrackID \leftarrow$ tracking_id
                $color \leftarrow$ green
                $modified \leftarrow$ true
            else
                $color \leftarrow$ blue
            end if
        else
            $selectBCI \leftarrow BCICommand \mod 10$
            if $ct = selectBCI$ then
                $modified \leftarrow$ true
                $flash \leftarrow$ false
                $lastBCICommandTrackID \leftarrow$ tracking_id
                $color \leftarrow$ red
            else
                $color \leftarrow$ blue
            end if
        end if
    end if
    $ct \leftarrow ct + 1$
    if $modified = false$ then
        $lastBCICommandTrackID \leftarrow -1$
    end if
end foreach
```

5

Software Tools and Hardware Materials

This chapter comprehends the software tools and hardware materials used for the development of this dissertation.

5.1 Software Tools

5.1.1 Visual Studio

Visual Studio is an integrated development environment (IDE) created by Microsoft in 2002 and is used for software development, with many applications including, desktop applications, web applications, mobile apps, and cloud-based applications, among others. Visual Studio provides the developer a comprehensive set of tools, services, and features making it easier for developers to build high-quality software across different platforms and technologies [49].

Since this dissertation used a TCP/IP connection, Visual Studio 2022 was utilized to establish the connection between the server, in Python, and the client in C#. Unity was integrated with Visual Studio for the deployment of the C# code (client) into Microsoft HoloLens.

5.1.2 Unity

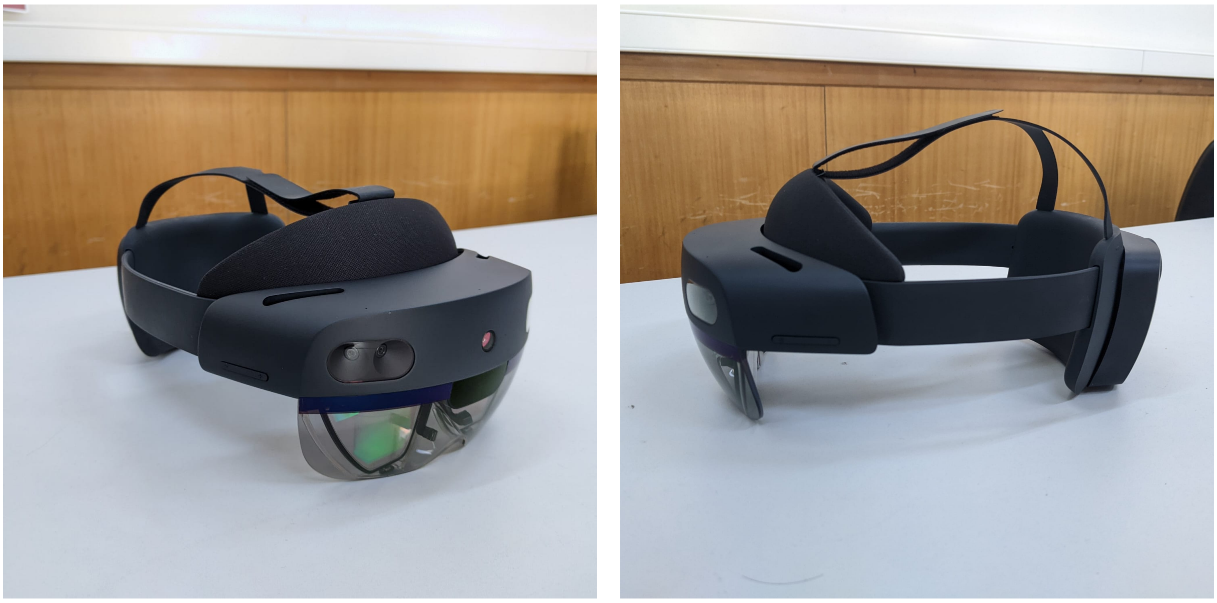
Unity is a cross-platform game engine that was developed by Unity Technologies in 2005. Unity allows developers to create video games, interactive experiences, 2D, 3D, virtual reality (VR) and augmented reality (AR) applications [50].

The Unity engine provides a user-friendly interface and an extensive collection of tools and features, supporting various platforms, including Windows, macOS, Linux, Android, iOS and many others.

We used version 2021.3.16f1 of Unity and as explained above, Visual Studio and Unity were integrated to deploy the client code, in C#, to the Microsoft HoloLens.

5.1.3 Windows Device Portal

The Windows Device Portal (WDP) is a web server tool that allows developers to remotely manage and configure Windows devices, including PCs, tablets, phones and the Mi-



(a) Front view

(b) Side view

Figure 5.1: Microsoft Hololens 2.

Table 5.1: Microsoft HoloLens 2 specifications.

CPU	Qualcomm Snapdragon 850 Compute Platform
Memory	4 GB RAM
Storage	64 GB
Display resolution	2048 × 1080
Field of view	52°
Camera	8 MP, 1080p30 video
Microphones	Five channel array
Eye tracking	Yes
Biometric security	Yes (Iris Scan)
Hand tracking	Yes
Connectivity	IEEE 802.11 2x2 WiFI, Bluetooth LE 5.0, USB Type-C

crosoft HoloLens [51]. To access WDP the user must type the IP of the device, in this case, the Microsoft HoloLens, in a web browser, allowing the developer to access many information about the device.

5.2 Hardware Materials

5.2.1 Microsoft HoloLens 2

Microsoft HoloLens 2, see Fig. 5.1, is a mixed reality (MR) head-mounted display that was developed by Microsoft in 2019. It is designed to overlap the digital elements with the physical world, allowing users to interact with holographic content in the real world, unlike virtual reality (VR), which creates a fully digital environment.

In Table 5.1 Microsoft HoloLens 2 specifications are presented.

5.2.2 Brain-Computer Interface

The BCI material consisted of three essential components: the BCI cap, the amplifier, and the electrode driver box.

The BCI cap, shown in Fig. 5.2 a), is a specially designed headwear that captures and records neural activity with great accuracy. The cap is equipped with an array of electrodes that detect and measure the electrical signals produced by the brain. The electrodes are carefully arranged, according to our study's needs, to guarantee complete spatial coverage, allowing monitoring of neural activity from different regions of the brain at the same time.

The Electrode Driver Box, shown in Fig. 5.2 b), connects electrodes and manages signal transmission, optimizing signal quality before sending neural data to the computer for analysis. The device plays a crucial role in ensuring the high quality of data collected, which contributes to the success of our research.

The Amplifier, shown in Fig. 5.2 c), is a crucial component in our BCI material. Its main function is to improve the faint electrical signals that are detected by the BCI cap. Typically, neural signals are weak and need to be amplified to be accurately analyzed and interpreted. The Amplifier serves as a tool to magnify the electrical signals while preserving their integrity.

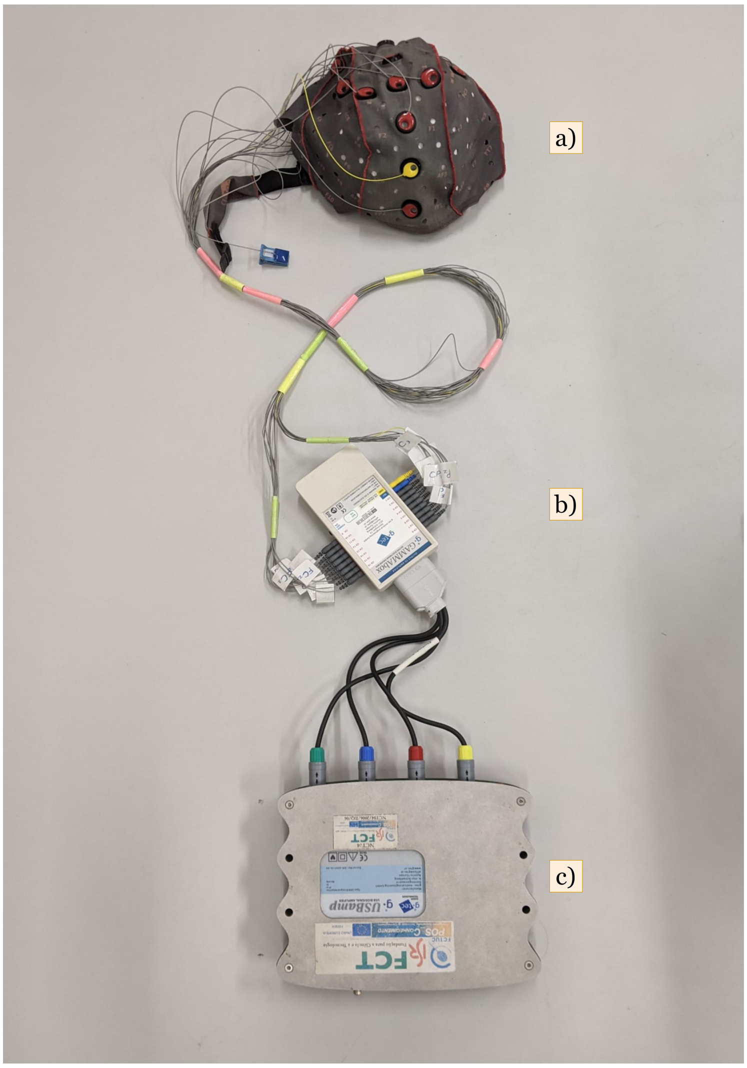


Figure 5.2: BCI materials. **a)**Cap, **b)**Electrode Driver Box and **c)**Amplifier.

6

Results and Discussion

6.1 BCI-Hololens Setup and Validation Protocol

In this chapter, we provide a detailed description of the setup and procedures that we followed to prepare and validate the Hololens-BCI system.

6.1.1 Cap Placement and Electrode Selection

The user begins by wearing an EEG cap and applying a conductive gel to improve conductivity between the electrodes and scalp. This gel is placed between the electrode sensors and scalp to remove any substances that could block the flow of electrical signals. The conductive gel has a low impedance, which means it has a small resistance to electrical signals, resulting in more precise readings of the brain activity.

The location of the electrodes on the scalp is crucial for capturing specific brain activity and depends on the research being conducted. The electrodes were placed at Fz, Cz, C3, C4, CPz, Pz, P3, P4, Po7, PO8, POz and Oz channels, according to the international 10-20 standard system. The right earlobe was utilized as a reference and the ground electrode was placed at AFz. The EEG signals were sampled at 256 Hz and pre-processed using a notch filter at 50 Hz and a bandpass filter between 0.5 and 30 Hz.

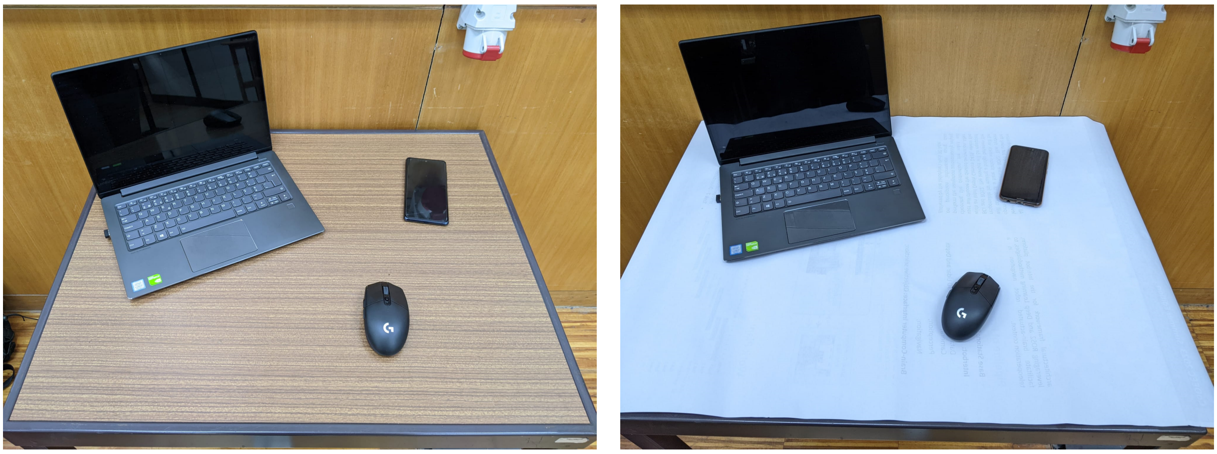
Figure 6.2 shows the placement of the electrodes used in our research.

6.1.2 Experimental Setup

We positioned on a table, three distinct objects as seen in Fig. 6.1a: a computer, a smartphone, and a computer mouse. These items were carefully selected to represent common objects users may encounter in real-life situations. The Hololens and BCI systems used these objects as targets for user interaction.

The user, wearing both the Hololens and the EEG cap, was seated at a fixed distance of 50 centimeters from the table. This distance was chosen to enable comfortable interaction and alignment between the user's line of sight through the Hololens and the objects on the table.

During the course of our experiments, we made a significant adjustment to the setup to improve the performance of the object detection component, which utilized the YOLOv5 model. We modified the color of the table's top surface to white, Fig. 6.1b. This adjustment was



(a) Table configuration with original surface (b) Table configuration with white surface

Figure 6.1: Table Configurations with Original and White Surfaces.

made to increase the contrast between the objects and the table surface, as the YOLOv5 model occasionally encountered challenges in consistently detecting all three objects simultaneously when the table had a neutral color. Transitioning to a white table surface improved object detection.

6.2 Signal Processing

To analyze biosignals, we send the signals from the bioamplifier to MATLAB and Simulink. Next, we use a Butterworth bandpass filter to filter the biosignals. This type of filter is designed to pass signals of a certain frequency while attenuating signals outside of that range. This is crucial for analyzing biosignals as they typically have a narrow frequency range. By filtering the signals with a Butterworth bandpass filter, we could ensure that we were analyzing the relevant frequencies for the biosignal.

6.3 BCI - Hololens Train

6.3.1 User Training/Calibration

Before using the P300 BCI, a calibration phase was carried out. During this phase, the user was presented with a series of stimuli in the form of green-flashing bounding boxes displayed on the HoloLens according to the oddball paradigm. The user received instructions to concentrate on a particular item (target) placed in front of them and mentally count the number of times the bounding box flashed green. Each object flashed nine times before moving to the next item of the sequence. The choice of nine counts was motivated by the need to collect a large number of samples, which would prove instrumental in training the classifier. The distinction between target and non-target stimuli during this phase was crucial in improving the classifier's overall performance. This calibration phase took about 45 minutes, to gather the EEG data to train the P300 classifier.

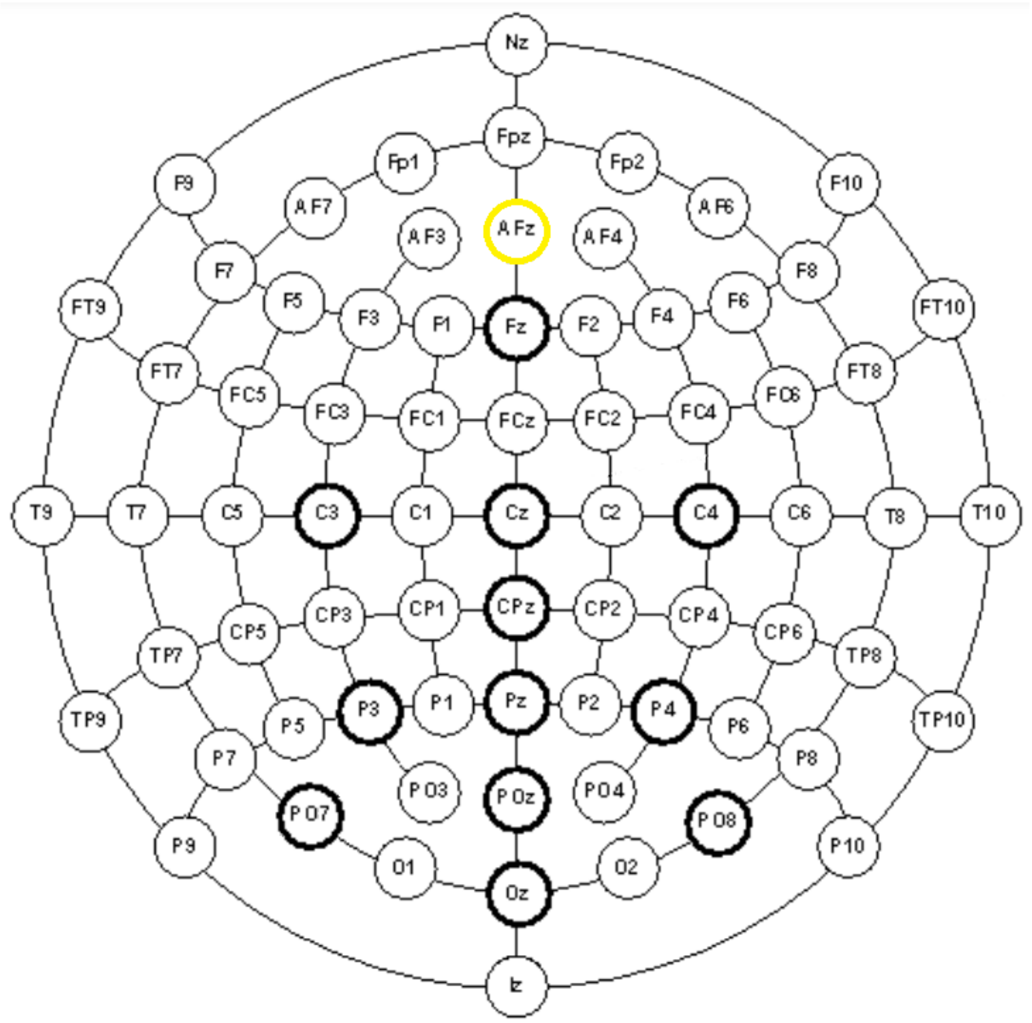


Figure 6.2: Electrode locations, according to the international 10-20 extended system. Bold circles indicate the channels used in our research. Yellow circle indicates the ground.

6.3.2 Data Acquisition

The EEG signals are acquired with a g.USBamp bioamplifier, equipped with 12 positioned electrodes. The right earlobe was utilized as a reference and the ground was placed at AFz allowing us to capture the user's cognitive responses. As the user focused their attention on specific objects displayed through the HoloLens, the P300 BCI recorded P300 event-related potentials from the user's brain. This data is then transmitted to a computer for processing.

The BCI system operates in a sequential mode, where it captures an image from HoloLens and flashes each of the three target objects for 150 *ms*, with a stimuli interval of 250 *ms*. To ensure a reliable object selection, each one of the three target objects is flashed a total of nine times within the same image. This resulted in a total of 27 flashes occurring in a single captured image. The reason behind this approach is that the HoloLens environment is highly dynamic, with objects and their positions changing quickly. Given the current pipeline of our BCI system, we found it difficult to adapt to this dynamic environment. Therefore, by flashing each target object multiple times within a single image, we expected to improve the accuracy. The process of capturing an image, flashing the objects and moving on to the next image is repeated in the

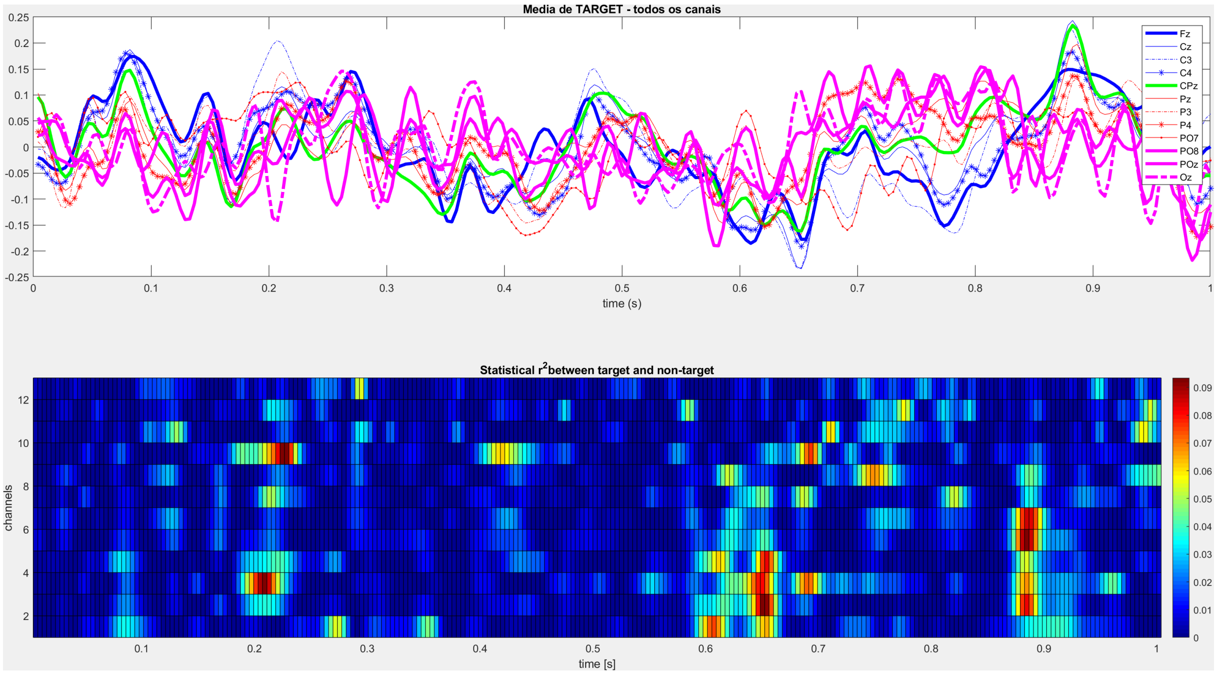


Figure 6.3: Grand average and R-squared graph for Participant 1.

validation phase as well.

6.3.3 Results of BCI Calibration

The calibration/training phase is important because it prepares the model to be used in the validation phase, indicating how well the model will perform on new data. If the model is not well-trained or calibrated, it may not be accurate on new data. The results of this calibration, for the two participants, are explained and shown below.

Figure 6.3 shows the graph for the Grand Average (top) and for the R-squared (bottom) for Participant 1. The grand average graph displays the data collected from these 12 channels, and averaged together, over a one-second interval. When examining the grand average from Participant 1, it is evident that the graph exhibits a substantial amount of oscillatory behavior and does not display the typical event-related potential P300. The presence of these irregularities makes it challenging to discriminate between target and non-target signals, as the graph displays many oscillations making it difficult to identify any significant patterns. However, when examining the R-squared graph, it's important to note that even though there are some temporal windows with a certain level of discrimination (though relatively low), these temporal windows do not align with the expected response windows, considering any potential delays. It is possible that these deviations are associated with ocular artifacts.

The Grand Average graph of Participant 2, as seen in Fig. 6.4 demonstrates a well-defined pattern, a P300 peak at around 500 ms preceded by an N200 100ms earlier, which points to it being the classic pattern but with a delay of 200ms.

Figure 6.5 presents Participant 2's Target and Non-Target Grand Average, revealing an SSVEP effect overlapping with the P300 ERP, indicating a response to all stimuli.

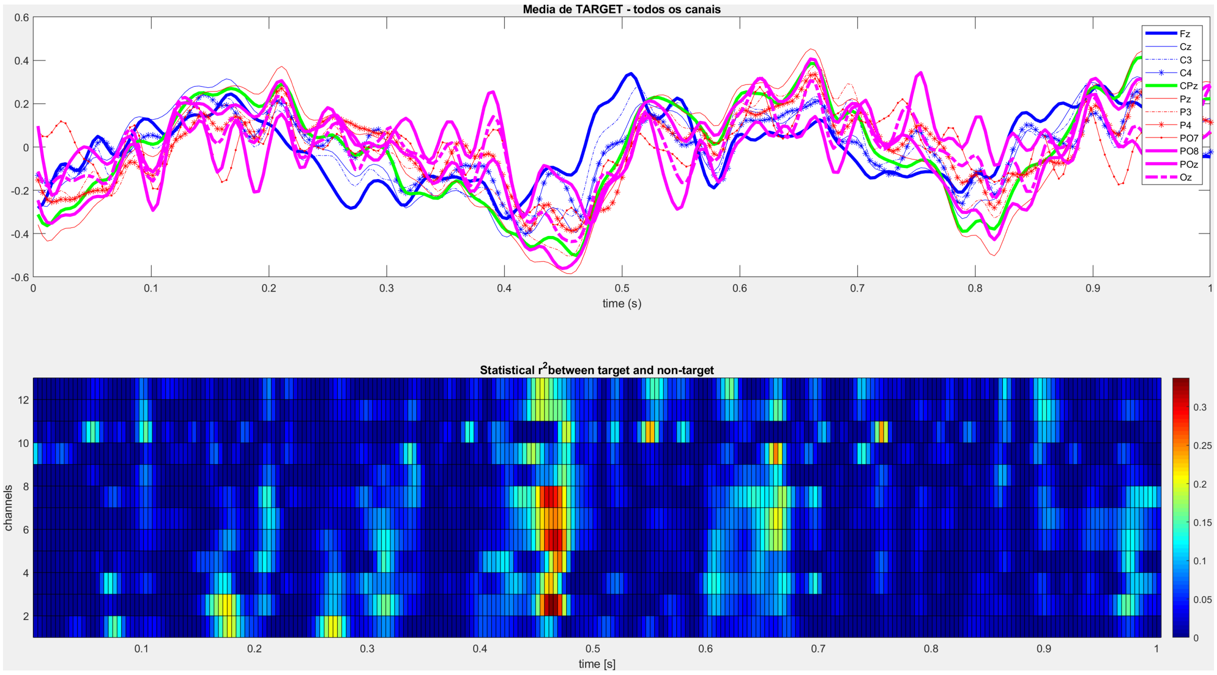


Figure 6.4: Grand average and R-squared graph for Participant 2.

Figures 6.6a and 6.6b, present the relationship between the expected error rate and the number of stimulus repetitions during online sessions for the participants. This graph demonstrates the evolution of participant performance as they are exposed to varying levels of stimulus repetition. The results suggest that the performance improves with a higher number of repetitions, indicating the presence of the P300 signal. This improvement is observed through cross-validation, highlighting the significance of stimulus repetition in the binary classification between target and non-target.

6.4 BCI - Hololens Online Validation

6.4.1 Data Acquisition

Similar to data acquisition during the training phase 6.3, in the validation phase the process remained the same, with one difference: instead of flashing nine times, the stimuli flashed five times. We used a g.USBamp bioamplifier with twelve electrodes to acquire EEG signals. While the users focused their attention on particular objects displayed on the HoloLens, the P300 BCI recorded P300 event-related potentials from their brains. Later, we sent this data to a computer for further analysis.

6.4.2 Results

To calculate the accuracy of the test phase, we conducted experiments with the participation of two individuals. Accuracy was determined by the number of correct predictions relative to the total cases, expressed as a percentage.

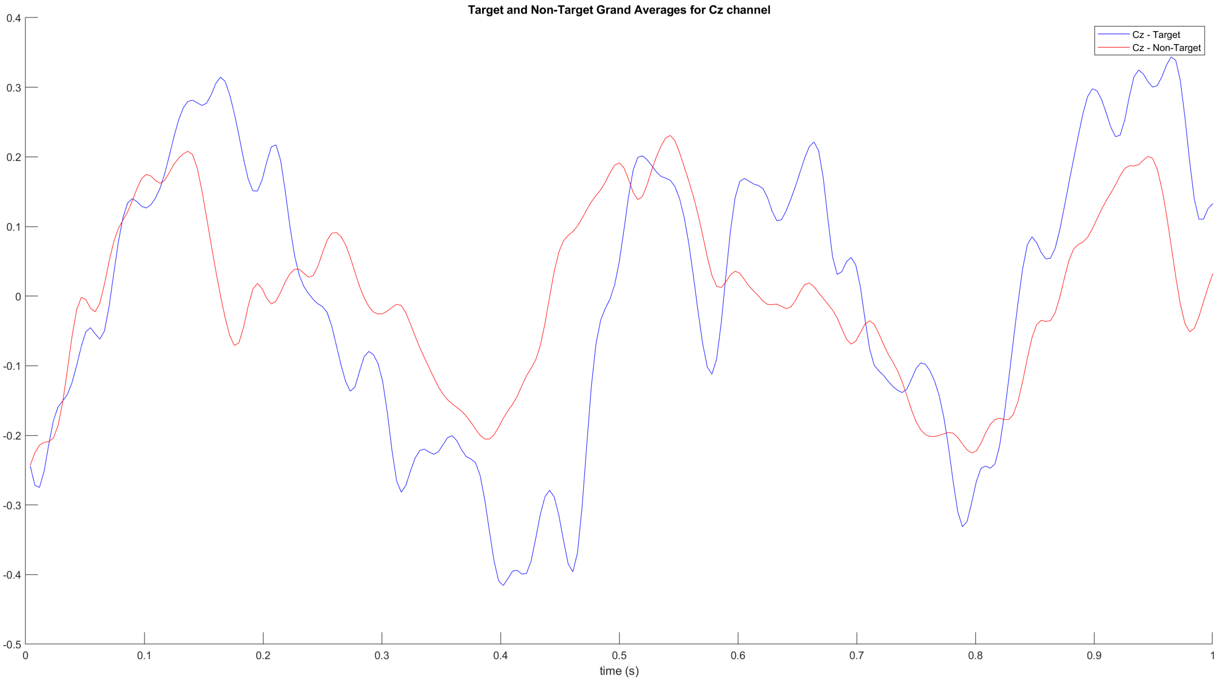


Figure 6.5: Target and Non-Target Grand Average for Participant 2.

$$Accuracy = \frac{Correct\ Predictions}{Total\ Number\ of\ Cases} \times 100\%$$

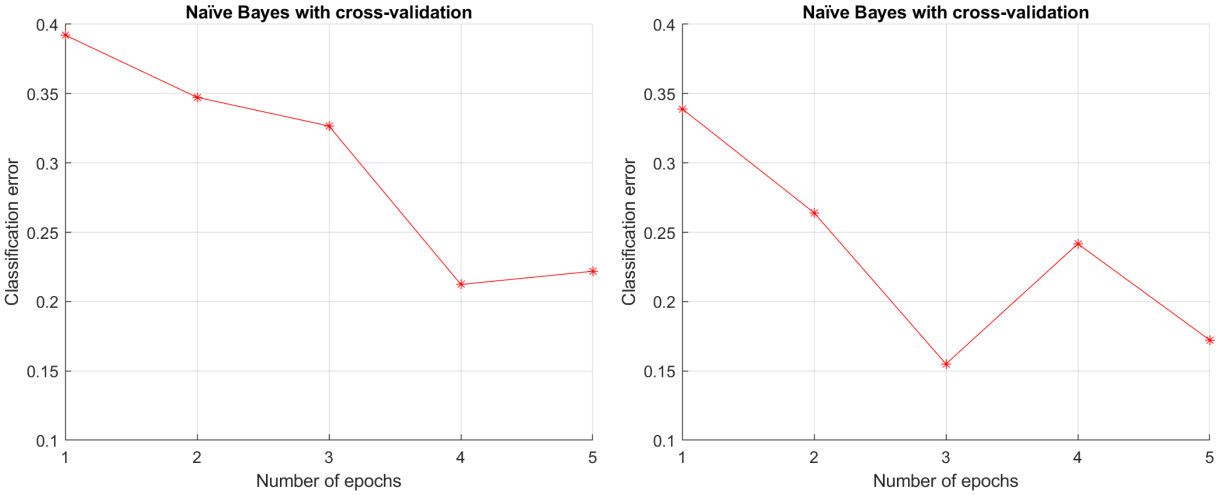
For Participant 1, in a scenario with 3 classes, we observed a total of 50 cases, and our system made 16 correct predictions, resulting in an accuracy rate of 32%, which approximates a chance level.

For Participant 2, in a scenario with 3 classes, we observed a total of 49 cases and among these cases, 16 predictions were accurate, leading to an accuracy rate of 27.11%, which approximates a chance level.

6.5 User Experience

In this section, we provide insights into the experiences of Participants 1 and 2 during their engagement in our experiment. We administered a questionnaire [52] to gather feedback from the participants regarding their perspectives. Taking into account the participants' feedback during the experiments can improve future experiments. Here we display two distinctive experiences, with fatigue being a main contributor to poor performance. More feedback is needed in order to understand if this factor, or other factors, are important enough to significantly impact the users' experience.

Participant 1 perceived the overall workload as low, indicating that the task did not seem to be demanding. They also found task difficulty to be low, indicating that it wasn't particularly complex or difficult to handle. Despite the low workload and task difficulty, Participant 1 reported a high level of effort. Reported Participant 1's fatigue during the experience might have contributed to their perception of the task as requiring significant effort and thus considering



(a) Participant 1.

(b) Participant 2.

Figure 6.6: Expected error for the online sessions with cross-validation for each participant (Y-axis) with the specific number of stimulus repetitions (X-axis).

their performance as poor. Participant 1 also considered the mental demand as high but the physical demand as low, which aligned with the overall workload.

Participant 2 found the workload and task difficulty to be low indicating that the task did not seem to be overwhelming and challenging. Participant 2 expressed that they put in a lot of effort in completing the task. They found the physical demands to be low but considered the mental demands to be high because of the cognitive aspects of the task. In general, this participant considered their performance to be good.

7

Conclusion

This dissertation explored the integration of Microsoft HoloLens 2 and Brain-Computer Interface technologies, aiming for future incorporation with assistive robotics to assist people with motor disabilities in driving robotic wheelchairs.

We have established TCP/IP and UDP/IP connections, that connect the Microsoft HoloLens to the computer and the Brain-Computer Interface respectively. These connections serve as the foundation of our communication framework, enabling the exchange of data and commands between the various components of our system. The trial that combined Microsoft HoloLens with Brain-Computer Interface did not produce any conclusive outcomes.

The acquisition from the HoloLens camera was successful, and notable enhancements were made in optimizing the frame rate for captured images sent to the object detection algorithm. This improvement was achieved by OpenCV's VideoCapture function in conjunction with Windows Device Portal credentials, a more effective approach than relying on the standard Unity API function for this task.

Furthermore, the implementation of object detection using YOLOv5 and object tracking using SORT was successfully executed within the HoloLens environment. In particular, the rendering of bounding boxes generated by the object detection and tracking algorithms, using Unity, was also accomplished. These bounding boxes effectively encapsulated all objects perceived by the HoloLens, aligning them accurately according to the user's field of view.

After conducting experiments with two participants, we concluded that there seems to be a latency issue in the system which may vary unpredictably. Additionally, there appears to be an SSVEP effect that could overlap with the P300 signal, reducing the ability to distinguish between target and non-target. The study's small number of participants also limits the conclusions that can be drawn, as the patterns observed between participants were significantly different.

In the future, the integration of this system with assistive robotics still holds the potential to significantly enhance the quality of life for individuals with motor disabilities. Although we have encountered challenges, including communication delays, object detection instability, and variations in user performance, we remain enthusiastic about the transformative possibilities this technology offers to individuals facing motor disabilities.

7.1 Future Work

As we look ahead for future work, it is evident that extensive research and development are necessary to overcome the challenges we faced in integrating Brain-Computer Interface with Microsoft HoloLens. Although our initial attempts were promising, we encountered significant obstacles in achieving a smooth integration.

Improving Object Detection Accuracy: The accuracy of YOLOv5 may need further improvement for reliable object detection and tracking. Future research should explore techniques such as fine-tuning the model with domain-specific data, investigating alternate object detection algorithms, or leveraging ensemble methods to improve accuracy.

Reducing Hololens-BCI Communication Delay: Addressing communication delays between Hololens and BCI is crucial for a smoother user experience. Research should focus on optimizing data transmission protocols, exploring edge computing solutions to reduce latency, or implementing predictive algorithms to compensate for delays.

Real-world Testing: Expand testing and experimentation to more diverse real-world scenarios, environments, and user populations. Collect data in various contexts to assess the system's robustness, adaptability, and effectiveness across different use cases.

Create a dataset from the user's point of view: Creating a dataset from the user's point of view improves flexibility, control and customization to effectively address our research objectives. This is especially important when existing datasets do not align with our specific research context or requirements.

User-Centered Design Iterations: Refine the user interface and interaction design. Incorporate user feedback to ensure the system aligns with their needs.

Integration with Assistive Robotics: Expand the integration to control and navigate assistive robotic platforms beyond object detection and tracking.

Bibliography

[1] Renjie Xu, Haifeng Lin, Kangjie Lu, Lin Cao, and Yunfei Liu. A Forest Fire Detection System Based on Ensemble Learning. *Forests*, 12:217, 02 2021.

[2] Augmented Reality. `https://www.grandviewresearch.com/industry-analysis/augmented-reality-market/`, 2023.

[3] Ricardo Pereira, Aniana Cruz, Luís Garrote, Gabriel Pires, Ana Lopes, and Urbano J. Nunes. Dynamic Environment-based Visual User Interface System for Intuitive Navigation Target Selection for Brain-actuated Wheelchairs. In *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 198–204, 2022.

[4] Ricardo Pereira, Aniana Cruz, Luís Garrote, Gabriel Pires, Ana Lopes, and Urbano J. Nunes. Dynamic environment-based visual user interface system for intuitive navigation target selection for brain-actuated wheelchairs. In *2022 31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 198–204, 2022.

[5] Wen Chen, Shih-Kang Chen, Yi-Hung Liu, Yu-Jen Chen, and Chin-Sheng Chen. An electric wheelchair manipulating system using ssvep-based bci system. *Biosensors*, 12(10), 2022.

[6] Glenn Jocher. Ultralytics yolov5, 2020.

[7] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.

[8] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960.

[9] Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 52, 1955.

[10] Daniel Pleus. SORT Stages. `https://www.linkedin.com/pulse/object-tracking-sort-deepsort-daniel-pleus/`.

[11] Zhihao Wu, Chengliang Liu, Chao Huang, Jie Wen, and Yong Xu. Deep Object Detection with Example Attribute Based Prediction Modulation. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2020–2024, 2022.

[12] Aniana Cruz, Gabriel Pires, Ana Lopes, Carlos Carona, and Urbano Nunes. A Self-Paced BCI With a Collaborative Controller for Highly Reliable Wheelchair Driving: Experimental Tests With Physically Disabled Individuals. *IEEE Transactions on Human-Machine Systems*, 51:109–119, 12 2020.

[13] Rui Zhong, Mengmeng Li, Qingling Chen, Jing Li, Guangjian Li, and Weihong Lin. The P300 Event-Related Potential Component and Cognitive Impairment in Epilepsy: A Systematic Review and Meta-analysis. *Frontiers in Neurology*, 10, 2019.

[14] Rodrigo Verschae and Javier Ruiz-del Solar. Object Detection: Current and Future Directions. *Frontiers in Robotics and AI*, 2, 2015.

[15] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object Detection in 20 Years: A Survey, 2023.

[16] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

[17] Siddhi Chourasia, Rhugved Bhojane, and Lokesh Heda. Safety Helmet Detection: A Comparative Analysis Using YOLOv4, YOLOv5, and YOLOv7. In *2023 International Conference for Advancement in Technology (ICONAT)*, pages 1–8, 2023.

[18] Upesh Nepal and Hossein Eslamiat. Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs. *Sensors*, 2022.

[19] Siddhi Chourasia, Rhugved Bhojane, and Lokesh Heda. Safety Helmet Detection: A Comparative Analysis Using YOLOv4, YOLOv5, and YOLOv7. In *2023 International Conference for Advancement in Technology (ICONAT)*, pages 1–8, 2023.

[20] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot MultiBox detector. In *Computer Vision – ECCV 2016*, pages 21–37. Springer International Publishing, 2016.

[21] Jeong-ah Kim, Ju-Yeong Sung, and Se-ho Park. Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition. In *2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia)*, pages 1–4, 2020.

[22] Jason Brownlee. Object Recognition with Deep Learning. `https://machinelearningmastery.com/object-recognition-with-deep-learning/`, 2023.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, 2016.

[24] Trupti Mahendrakar, Andrew Ekblad, Nathan Fischer, Ryan White, Markus Wilde, Brian Kish, and Isaac Silver. Performance Study of YOLOv5 and Faster R-CNN for Autonomous Navigation around Non-Cooperative Targets. In *2022 IEEE Aerospace Conference (AERO)*, pages 1–12, 2022.

[25] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN, 2018.

[26] Wei Li, Tengfei Zhu, Xiaoyu Li, Jianzhang Dong, and Jun Liu. Recommending Advanced Deep Learning Models for Efficient Insect Pest Detection. *Agriculture*, 12:1065, 07 2022.

[27] Behzad Mirzaei, Hossein Nezamabadi-pour, Amir Raoof, and Reza Derakhshani. Small Object Detection and Tracking: A Comprehensive Review. *Sensors*, 23(15), 2023.

[28] G.L. Foresti. Object recognition and tracking for remote video surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(7):1045–1062, 1999.

[29] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera. Deep learning in video multi-object tracking: A survey. *Neurocomputing*, 381:61–88, mar 2020.

[30] Ricardo Pereira, Guilherme Carvalho, Luís Garrote, and Urbano J. Nunes. Sort and DeepSORT Based Multi-Object Tracking for Mobile Robotics: Evaluation with New Data Association Metrics. *Applied Sciences*, 12(3), 2022.

[31] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple Online and Realtime Tracking with a Deep Association Metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649, 2017.

[32] Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement, 2018.

[33] Ronald T. Azuma. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 08 1997.

[34] Ming Chen, Chen Ling, and Wenjun Zhang. Analysis of Augmented Reality application based on cloud computing. In *2011 4th International Congress on Image and Signal Processing*, volume 2, pages 569–572, 2011.

[35] Haythem Bahri, David Krčmařík, and Jan Kočí. Accurate Object Detection System on HoloLens Using YOLO Algorithm. In *2019 International Conference on Control, Artificial Intelligence, Robotics Optimization (ICCAIRO)*, pages 219–224, 2019.

[36] Tasnim Ferdous Dima and Md. Eleas Ahmed. Using yolov5 algorithm to detect and recognize american sign language. In *2021 International Conference on Information Technology (ICIT)*, pages 603–607, 2021.

[37] Andre Barczak, Napoleon Reyes, M Abastillas, A Piccio, and Teo Susnjak. A New 2D Static Hand Gesture Colour Image Dataset for ASL Gestures. *Res Lett Inf Math Sci*, 15, 01 2011.

[38] Lusheng Liu, Bo Wen, Minjie Wang, Aiping Wang, Jing Zhang, Yuan Zhang, Song Le, Lihua Zhang, and Xiaoyang Kang. Implantable Brain-Computer Interface Based On Printing Technology. In *2023 11th International Winter Conference on Brain-Computer Interface (BCI)*, pages 1–5, 2023.

[39] Myoung-Ki Kim, Jeong-Hyun Cho, Hye-Bin Shin, and Seong-Whan Lee. Towards Brain-based Interface for Communication and Control by Skin Touch. In *2023 11th International Winter Conference on Brain-Computer Interface (BCI)*, pages 1–5, 2023.

[40] Byeong-Hoo Lee, Jeong-Hyun Cho, and Byoung-Hee Kwon. Hybrid Paradigm-based Brain-Computer Interface for Robotic Arm Control. In *2023 11th International Winter Conference on Brain-Computer Interface (BCI)*, pages 1–4, 2023.

[41] Aniana Cruz, Gabriel Pires, Ana Lopes, Carlos Carona, and Urbano J. Nunes. A Self-Paced BCI With a Collaborative Controller for Highly Reliable Wheelchair Driving: Experimental Tests With Physically Disabled Individuals. *IEEE Transactions on Human-Machine Systems*, 51(2):109–119, 2021.

[42] Mark Zolotas, Joshua Elsdon, and Yiannis Demiris. Head-Mounted Augmented Reality for Explainable Robotic Wheelchair Assistance. pages 1823–1829, 10 2018.

[43] Yasmine Mustafa, Mohamed Elmahallawy, Tie Luo, and Seif Eldawlatly. A Brain-Computer Interface Augmented Reality Framework with Auto-Adaptive SSVEP Recognition, 2023.

[44] Dennis Dietz. MR-Braintap: Increasing Freedom through Mixed Reality-Brain Computer Interface. In *CHI '18 workshop: Designing Interactions for the Ageing Populations*, CHI '18, New York, NY, USA, 2018. ACM.

[45] Pasquale Arpaia, Egidio De Benedetto, Lucio De Paolis, Giovanni D'Errico, Nicola Donato, and Luigi Duraccio. Highly wearable SSVEP-based BCI: Performance comparison of augmented reality solutions for the flickering stimuli rendering. *Measurement: Sensors*, 18:100305, 2021.

[46] Faizan Badshah, Syed Tauhid Ullah Shah, Syed Roohullah Jan, and Izaz Ur Rahman. Communication Between Multiple Processes on Same Device Using TCP/IP Suite. In *2017 International Conference on Communication, Computing and Digital Systems (C-CODE)*, pages 148–151, 2017.

[47] Guilherme de Sousa Carvalho. Kalman Filter-Based Object Tracking Techniques for Indoor Robotic Applications. Master's thesis, 2021.

[48] GameObject. `https://docs.unity3d.com/Manual/GameObjects.html`.

[49] Visual Studio. `https://en.wikipedia.org/wiki/Visual_Studio`.

[50] Unity. `https://en.wikipedia.org/wiki/Unity_(game_engine)`.

[51] Windows Device Portal. `https://learn.microsoft.com/en-us/windows/uwp/debug-test-perf/device-portal`.

[52] Sofien Gannouni, Kais Belwafi, Nourah Alangari, Hatim AboAlsamh, and Abdelfettah Belghith. Classification Strategies for P300-Based BCI-Spellers Adopting the Row Column Paradigm. *Sensors*, 22(23), 2022.

[53] Microsoft HoloLens 2. `https://en.wikipedia.org/wiki/HoloLens_2`.

[54] Augmented Reality Applications. `https://www.liainfraservices.com/blog/top-5-applications-of-augmented-reality-ar//`, 2023.

[55] Reza Fazel-Rezai, Brendan Allison, Christoph Guger, Eric Sellers, Sonja Kleih, and Andrea Kübler. P300 Brain Computer Interface: Current challenges and emerging trends. *Frontiers in neuroengineering*, 5:14, 07 2012.

[56] Sandra G. Hart and Lowell E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In Peter A. Hancock and Najmedin Meshkati, editors, *Human Mental Workload*, volume 52 of *Advances in Psychology*, pages 139–183. North-Holland, 1988.