POLITECNICO DI TORINO

Corso di Laurea Magistrale
Ingegneria del Cinema e dei Mezzi di Comunicazione

Tesi di Laurea

Brain-Computer Interface a Supporto dell'Interazione Uomo-Robot

Relatore:
Andrea Sanna
Federico Manuri

Candidato:
Jacopo Fiorenza

Anno accademico 2021/2022

Abstract

Con lo svilupparsi delle nuove tecnologie e con l'avvento dell'industria 4.0, vi è sempre più la necessità di implementare nuovi supporti che facilitino il lavoro degli operatori e che possano aprire la strada a nuove soluzioni e punti di vista.

Sicuramente non si tratta di un percorso lineare da affrontare, né tanto meno semplice da vedere nel suo insieme, data la gran quantità, sempre in crescita, di nuovi sistemi e supporti in questo ambito.
Se da un lato, in questa nuova rivoluzione industriale, si vede sempre di più il nascere di queste nuove tecnologie e il loro utilizzo in ambienti lavorativi, dall'altro ci si chiede se veramente, allo stato attuale, possano veramente alzare il livello e offrire un servizio sicuro ed efficiente.

L’obiettivo della tesi è stato dimostrare che il potenziale dei nuovi supporti, nello specifico quelli riguardanti le BCI (Brain-Computer Interface), può essere espresso già a partire da ora e che può essere utilizzato per sostenere il processo produttivo sia in termini di usabilità che in termini di sicurezza. L’argomento principale preso in esame in questa trattazione riguarda un contesto di assemblaggio, che include le dinamiche di inter-relazione tra l'uomo e il robot (HRI), andando a sfruttare come ponte tra i due l'utilizzo congiunto di realtà aumentata (AR) e BCI.

Per raggiungere tali obiettivi è stata progettata e realizzata un'applicazione a supporto dell'assemblaggio di un oggetto tramite le sue componenti, che utilizza un visore di Realtà aumentata di alto livello (Hololens) in congiunzione con un dispositivo BCI di tipo EEG non invasivo (NextMind). Il tutto è stato collegato, a titolo di esempio, a un braccio robotico non collaborativo, con il compito di assistere l’operatore porgendogli le componenti al momento opportuno. Inoltre, l’applicazione è stata delineata al fine di permettere il confronto tra due modalità di utilizzo: modalità gesture, per cui si interagisce con l’applicazione tramite i gesti delle mani, e modalità neuro, per la quale l’utente interagisce tramite la BCI e l’utilizzo delle mani non è necessario.

La tesi si conclude riportando i dati e le analisi ottenuti dai test effettuati e specificando le possibili implementazioni future da integrare nell’applicazione sviluppata.

Indice

Introduzione

1. Stato dell’arte

1.1 BCI/Robot
1.2 BCI/MR
1.3 BCI/AR/Robot

2. Design del sistema e realizzazione

2.1 Introduzione: fra gesture e controllo con lo sguardo
2.2 Complicanze iniziali
2.3 Tecnologie Hardware/Software utilizzate
    2.3.1 Tecnologie Software
    2.3.2 Tecnologie Hardware
2.4 Architettura del sistema
    2.4.1 Entità Hardware/Software
    2.4.1 Architettura hardware
    2.4.2 Architettura software

3. Design dell’applicazione

3.1 Introduzione
3.2 Architettura
    3.2.1 Introduzione
    3.2.2 Fase Principale
    3.2.3 Fase di Calibrazione
    3.2.4 Fase di Selezione del Progetto
    3.2.5 Fase di Assemblaggio
3.3 Interfaccia Utente (UI)
    3.3.1 Introduzione
    3.3.2 UI Generale
    3.3.3 UI Tavolo
    3.3.4 UI Oggetto assemblato
    3.3.5 UI Componenti – Modalità “Gesture”
    3.3.6 UI Sezioni – Modalità “Neuro”
    3.3.7 UI Componenti – Modalità “Neuro”
3.4 Architettura del codice
    3.4.1 Introduzione
    3.4.2 Entità di gestione
    3.4.3 Entità oggetto
    3.4.4 Entità UI
    3.4.5 Interfaces

4. Test e risultati

4.1 Introduzione

4.1.1 SUS

4.1.2 NASA - TLX

4.2 Descrizione caso d'uso

4.3 Set-up sperimentale

4.4 Svolgimento dei test

4.5 Risultati

4.5.1 Risultati SUS – System Usability Scale

4.5.2 Risultati NASA – TLX

4.5.3 Tempistiche ed errori

4.6 Analisi

4.6.1 Usabilità

4.6.2 Carico di lavoro (NASA-TLX)

4.6.3 Gesture e Neuro a confronto

5. Conclusioni

Bibliografia

Introduzione

Questa introduzione si propone di illustrare il contesto oggetto della presente tesi di laurea, andando a definire i punti fondamentali e a enunciare i propositi della seguente trattazione.

In primo luogo, è bene comprendere che cosa si intende con il termine “Industry 4.0” e quali sono le tecnologie che utilizza. Il progresso di scienza e tecnologia ha sicuramente portato notevoli cambiamenti nelle dinamiche lavorative e nella produzione di beni e servizi. Per questo motivo i nuovi approcci hanno guidato l’utilizzo di nuovi dispositivi, in particolare quelli che utilizzano le brain-computer interface (BCI) e la mixed reality (MR), i quali vengono progressivamente sempre più utilizzati in molti settori, anche in congiunzione con i robot, e che sono l’oggetto fondamentale degli argomenti trattati.

Di conseguenza, secondariamente, si andrà a introdurre il concetto di BCI e a definire gli obiettivi preposti al fine di dimostrare la loro efficienza sotto diversi punti di vista.
Il discorso riguarda l’utilizzo di tali dispositivi, in congiunzione a quelli di realtà aumentata (AR), a supporto della robotica in campo manifatturiero. In particolare, si vuole presentare un sistema che presti assistenza agli operatori durante il processo di assemblaggio di alcune componenti materiali.
Il modo in cui gli obiettivi sono stati raggiunti sarà approfondito in seguito.

Infine, al fondo dell’introduzione, viene riportata l’organizzazione dei capitoli e i punti fondamentali affrontati in ognuno di essi.

Industria 4.0

Sin dall’inizio dell’era dell’industrializzazione, i passi in avanti compiuti dalla tecnologia hanno portato progressivamente al progredire dei modelli produttivi, ormai passati, chiamati oggi “rivoluzioni industriali”: dalla meccanizzazione (1° rivoluzione), all’utilizzo intensivo dell’energia elettrica (2° rivoluzione), fino alla diffusione della digitalizzazione (3° rivoluzione). [1]

Il termine scientifico di “Industry 4.0” è stato utilizzato per la prima volta in Germania nel 2011 alla Fiera di Hannover, ispirato da un progetto del governo tedesco, dove era stato usato per denotare il processo di trasformazione globale nella creazione di valore.
Oggi, ai tempi in cui questa rivoluzione è in atto, il termine indica soprattutto i cambiamenti in corso nel contesto industriale, guidati dal progresso in ambito IT e dallo sviluppo dell’ambiente in cui le aziende operano. I tradizionali processi di monitoraggio e controllo stanno venendo sostituiti dal controllo decentralizzato, in quanto l’essenza del 4.0 è l’introduzione di sistemi intelligenti collegati per la regolazione automatica della produzione: persone, macchine, equipaggiamenti e prodotti comunicheranno sempre di più fra loro. [2]

In sostanza, le risorse tradizionali sono state convertite in oggetti “smart”, ai quali sono state aggiunte funzionalità identificative, sensoriali e di rete, che cooperano fra loro e con gli operatori umani. Quest’ultimo aspetto è importante da sottolineare, poiché rappresenta una delle caratteristiche fondamentali di questa nuova rivoluzione. Di fatto, la differenza tra *Computer*

Integrated Manifacturing (CIM) e *Industry 4.0* è la concezione del ruolo umano nell'ambiente di produzione. Se nel primo caso si considera una linea produttiva priva del lavoratore, nel secondo caso il ruolo umano è fondamentale per essa. [3]
Le dinamiche di connessione tra uomo-robot e il supporto dei nuovi dispositivi entrano, in questo contesto, a far parte del processo produttivo.
Di conseguenza, il concetto di *Smart Manifacturing*, che indica un sistema che si adatta a vari tipi di condizioni e prodotti, risulta essere un fattore decisamente rilevante. Questo aumenta in generale la qualità, l'efficienza, la flessibilità e la sostenibilità dei prodotti, grazie a un consumo ridotto delle risorse.[4]

L'Industria 4.0, dunque, si definisce come una nuova rivoluzione industriale nata dalla convergenza di spazio fisico e digitale, guidata dalla nascita di nuove e dirompenti tecnologie: Internet of Things (IoT), Cyber-Physical Systems (CPS), Cloud Computing, Big Data, Digital Twin e Intelligenza Artificiale (AI), etc. [5]

Fig. 0.1. *rappresentazione schematica delle nuove tecnologie nell'Industria 4.0. [6]*

In particolare, i dispositivi che utilizzano le Brain-Computer Interface (BCI) e quelli di Mixed Reality (AR, VR e MR) sono tra le tecnologie per cui il loro alto potenziale può portare un contributo significativo nei settori del 4.0.

BCI

Le *Brain-computer interface* (BCI) sono un sistema di comunicazione per cui viene garantita una connessione diretta tra il cervello e una macchina.
Ciò significa che l'attività cerebrale viene misurata da uno strumento e può essere utilizzata per controllare dispositivi esterni, come una sedia a rotelle, un computer o un robot. I dispositivi di BCI sono utilizzati in svariati settori del 4.0, come la medicina, l'industria manifatturiera o l'agricoltura, e rappresentano una tecnologia in continua evoluzione, con un grosso potenziale per il futuro.

Le tecnologie che utilizzano le BCI possono essere divise in due categorie, in base a come viene acquisito il segnale dal cervello: invasive e non invasive.

Quelle invasive installano i dispositivi di BCI direttamente nel cervello o sulla sua superficie, attraverso operazioni di neurochirurgia. Se da un lato l'SNR (signal-to-noise ratio) del segnale cerebrale è molto più alto, garantendo maggiore precisione e controllo, dall'altro c'è il rischio di complicazioni post-operatorie e infezioni.

Le BCI non invasive, invece, acquisiscono il segnale dalle onde cerebrali attraverso dispositivi esterni che possono utilizzare varie tecniche, quali elettroencefalogramma (EEG), magnetoencefalografia (MEG) e la risonanza magnetica funzionale (fMRI). [7]
La risoluzione temporale per misurare i cambi nell'attività neuronale risulta essere molto buona, anche se quella spaziale, per determinare la sorgente attiva nel cervello, è molto più bassa. Inoltre, il segnale è molto più soggetto a presentare artefatti, a causa del movimento dei muscoli e degli occhi.
Ad ogni modo, in base alla percentuale di rischio decisamente più bassa e ai costi ridotti, le tecnologie non invasive sono quelle più comuni nell'industria 4.0.

Un'ulteriore suddivisione può essere identificata nel modo in cui viene esercitato il controllo sulla macchina, computer o dispositivo: BCI attive, reattive e passive.
Quelle attive e reattive permettono all'utente di operare secondo il suo controllo volontario.
Le BCI attive (aBCI) permettono all'utente di potersi immaginare un'azione specifica mentalmente e di tradurla in comandi specifici tramite la risposta del segnale EEG misurato (es. *Motion-Imagery*). In generale, questa tipologia di BCI è integrata sempre in dei sistemi *sincroni*, dove le tempistiche di controllo sono dettate direttamente dall'utente e i ritardi sono molto minori.
Quelle reattive (rBCI), invece, sono tipicamente *asincrone* e traducono in comandi il segnale proveniente dal cervello in risposta a un determinato stimolo visivo; ognuno di questi provoca una differente risposta neuronale.
Per ultime abbiamo quelle passive (pBCI) che misurano l'attività cerebrale dell'utente durante un determinato compito per stimare il suo stato mentale (stress, fatica, zona di interesse...). [8] [9]

Conseguentemente, è importante sottolineare in che modo viene provocato il segnale proveniente dal cervello, al fine di comprendere come vanno ad agire le BCI per la sua ricezione e successiva elaborazione. Si ritiene necessario specificare un'ulteriore categorizzazione per quanto riguarda i segnali EEG di controllo rilevati: *esogeni* (evocati) ed *endogeni* (spontanei).
Quelli esogeni rappresentano una risposta fisiologica a uno stimolo esterno, come un segnale visivo o sonoro, e non richiedono una lunga preparazione dell'utente per il loro utilizzo.

Dall’altro lato, vi sono quelli endogeni, i quali richiedono sessioni di training più lunghe ma sono indipendenti da stimoli esterni, essendo totalmente operativi in base al controllo volontario dell’utente.
Nella maggior parte dei casi, i paradigmi più usati si basano sulle seguenti tipologie di segnali del cervello: *Steady-State Visual Evoked Potential* (SSVEP), *event-related potential* (ERP), *P-300* e “ritmi sensorimotori”(SMR).
Gli SSVEP e ERP sono quelli più comuni e sono provocati da un evento esterno, come il *flickering* di un oggetto a una determinata frequenza; tipicamente non presuppongono un grosso sforzo mentale da parte dell’utente, anche se resta l’eventualità di un affaticamento visivo dovuto alla stimolazione continua. [10] [11]

Il P-300 fa parte della categoria ERP e si basa su una risposta positiva nel segnale EEG che accade circa 300ms dopo un determinato stimolo esterno. A differenza delle sperimentazioni che utilizzano gli SSVEP, che si basano su variazioni di frequenza, lo schema BCI - P300 sfrutta l’effetto dell’evento esterno provocato nell’EEG ed è applicato quando la mole di comandi da proporre all’utente risulta essere considerevolmente alta. Tuttavia, nonostante sia una tipologia di stimolo in reazione a un evento esterno, in neuroscienze il P-300 è considerato essere un potenziale endogeno.

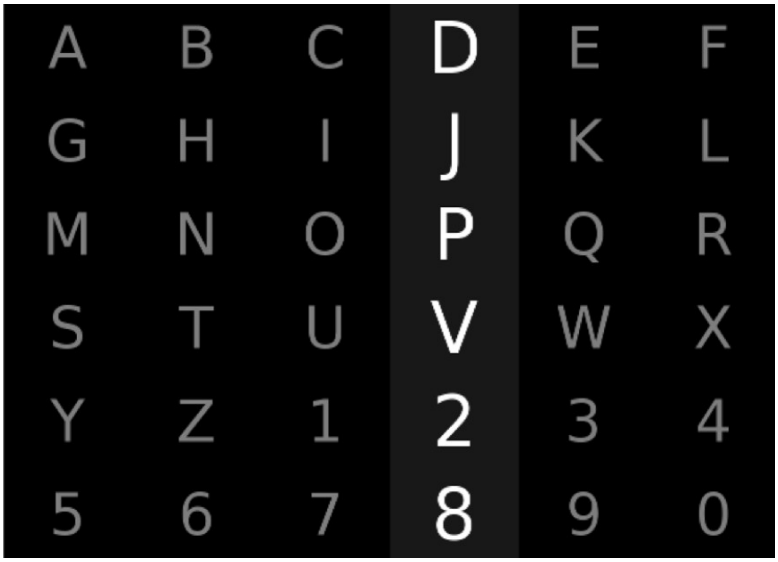


Fig. 0.2. *Esempio di griglia, basata sul P-300, utilizzata in un BCI-speller; le righe e le colonne sono tipicamente illuminate alternatamente per la stimolazione cerebrale dell’utente.*

Ad ogni modo, l’approccio basato sugli SSVEP risulta essere quello più versatile e intuitivo da utilizzare; il progetto che si vuole andare a proporre in questa trattazione utilizza, in contemporanea con un dispositivo di realtà aumentata (Microsoft Hololens), uno strumento BCI non invasivo (NextMind), basato sugli SSVEP, per il controllo da remoto di un robot.
L’obiettivo è di fornire supporto a un operatore umano nelle fasi di assemblaggio di un oggetto e delle sue componenti, in modo tale che non abbia bisogno di utilizzare le mani nel processo e che la *user experience* non ne risenta.

Motivazioni e obiettivi

Le BCI possono portare diversi vantaggi in molte aree applicative.

In primo luogo, esse sono versatili in svariati ambiti e possono essere utilizzate per raggiungere scopi differenti a seconda dei processi coinvolti. Nel campo della medicina sono utilizzate per la riabilitazione di pazienti malati [12], mentre in quello manifatturiero, che rappresenta l'oggetto di tutta questa trattazione, sono utilizzate a scopo di assistenza degli operatori umani [24]. Inoltre, negli ultimi anni l'utilizzo delle BCI si è espanso anche ai settori del gaming [25], dell'intrattenimento[26], dell'educazione [27] e della robotica[15].

Secondariamente, riallacciando la trattazione sempre all'argomento della manifattura, le BCI rappresentano un mezzo comodo per svolgere i compiti in maniera sicura ed efficiente.
Il vantaggio di non utilizzare le mani per controllare un dispositivo esterno (es. un robot) fa in modo che l'operatore possa impiegarle per altre mansioni, ottimizzando i tempi di lavoro e, di conseguenza, aumentando la produttività.

Il progetto proposto, che verrà approfondito più in là nella trattazione, è stato sviluppato seguendo i seguenti obiettivi: introdurre un'applicazione AR che utilizzi le BCI, dimostrarne la comodità di utilizzo, la sicurezza e la corretta usabilità. Inoltre, si vuole dimostrare il vantaggio di utilizzare delle BCI, che non necessitano di alcun movimento delle mani da parte dell'operatore, rispetto all'utilizzo delle *gesture* per il controllo di un robot di supporto.

Organizzazione capitoli

Nel capitolo 1, si andranno a riportare alcuni esempi in letteratura che hanno utilizzato le BCI in vari settori del 4.0. In particolare, si riporteranno i casi in cui sono state utilizzate per il controllo di robot e/o con l'utilizzo congiunto di dispositivi di *augmented reality* (AR).

Nel capitolo 2, si introdurrà il progetto proposto che utilizza le BCI, andando a descriverne l'architettura sia software che hardware.

Nel capitolo 3, verranno riportati i software e i dispositivi hardware utilizzati per lo sviluppo del progetto, riportando le feature e i servizi offerti da ognuno. A titolo informativo, verranno anche riportate alcune complicanze iniziali e in itinere che sono state riscontrate durante le fasi di lavoro.

Nel capitolo 4, verrà analizzata nel dettaglio l'applicazione AR sviluppata, descrivendone il funzionamento, l'architettura del codice e illustrandone l'interfaccia utente.

Nel capitolo 5, verranno riportati i casi d'uso del sistema durante le fasi di testing, i risultati ottenuti e le conseguenti analisi su di essi.

Infine, l'ultimo capitolo è dedicato alle conclusioni desunte da tutto il lavoro svolto e alle possibili integrazioni future.

1. Stato dell’arte

Allo stato attuale la sperimentazione delle BCI nel 4.0 è ancora in corso. Di seguito, vengono riportati e analizzati alcuni esempi trovati in letteratura che utilizzano questa tecnologia, al fine di dimostrare l’adattabilità di queste interfacce per settori diversi e per l’utilizzo congiunto con altri dispositivi (AR, VR, Computer, Robot etc.).

1.1 BCI/Robot

Alcuni utilizzi lampanti, in combinazione con dispositivi robotici, si possono trovare in medicina, molte volte per la riabilitazione dei pazienti sofferenti di malattie neurologiche o in seguito a traumi cerebrali.
Una delle prime BCI di supporto clinicamente rilevanti è stata usata su pazienti affetti dalla sindrome “locked in”, un disagio neurologico per cui il paziente è sveglio e cosciente, ma non può muoversi o parlare. In questo caso, è stata usata una BCI di tipo EEG per tradurre il segnale rilevato dal cervello in lettere e parole su uno schermo.[12]

Un sistema simile è stato utilizzato con successo anche in pazienti tetraplegici in seguito a un ictus del tronco encefalico. Talvolta, questa tipologia di realizzazioni per la riabilitazione sono chiamati Brain-machine interface (BMI) e comprendono la registrazione, la decodifica e la traduzione dei segnali cerebrali in azioni specifiche, senza che venga coinvolto il sistema motorio. In generale, durante la terapia BMI viene chiesto al paziente di cercare/immaginare di muovere l’arto paralizzato. L’attività del cervello viene rilevata attraverso elettrodi invasivi o non invasivi, per poi essere decodificata attraverso un computer e tradotta in comandi per un dispositivo robotico o prostetico.[13]

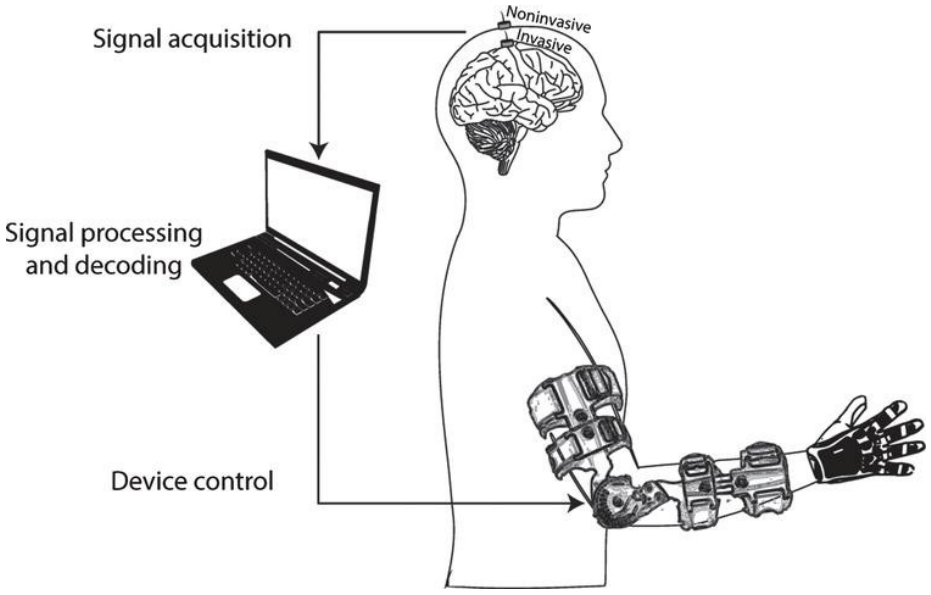


Fig. 1.1. *Architettura generale di una BMI per la riabilitazione degli arti superiori*

Un esempio è stato dato da Lee et al. con lo sviluppo di una BMI tramite *Motion Imagery* (MI) per il controllo di un esoscheletro degli arti inferiori, nel quale venivano utilizzati degli elettrodi EEG e un sistema di decodifica che traducesse il segnale in tre comandi: “cammina di fronte”, “girati a destra” e “girati a sinistra”.[14]

In questo caso, il controllo dell'esoscheletro presentava problematiche aggiuntive rispetto all'utilizzo di altri dispositivi mobili esterni (es. sedia a rotelle elettrica): in primo luogo, è stato considerato il livello di fatica, non trascurabile, dell'utente a causa della postura verticale e all'esercizio fisico; secondariamente, è stato necessario affrontare le problematiche di stabilità del sistema in generale, dovute alla possibile asincronia tra i movimenti del robot e dell'utente.
Per ovviare a queste problematiche, Lee et al. hanno ideato un approccio di decodifica, basato su classificatori *event-related*, che garantisse accuratezza nella classificazione, tempi di calibrazione minori e elaborazioni mentali ridotte.
La tipologia dell'interfaccia è stata sviluppata seguendo l'approccio di una BCI orientata al controllo volontario (endogeno) del dispositivo robotico, preferita all'utilizzo di una stimolazione esterna legata a un evento specifico.

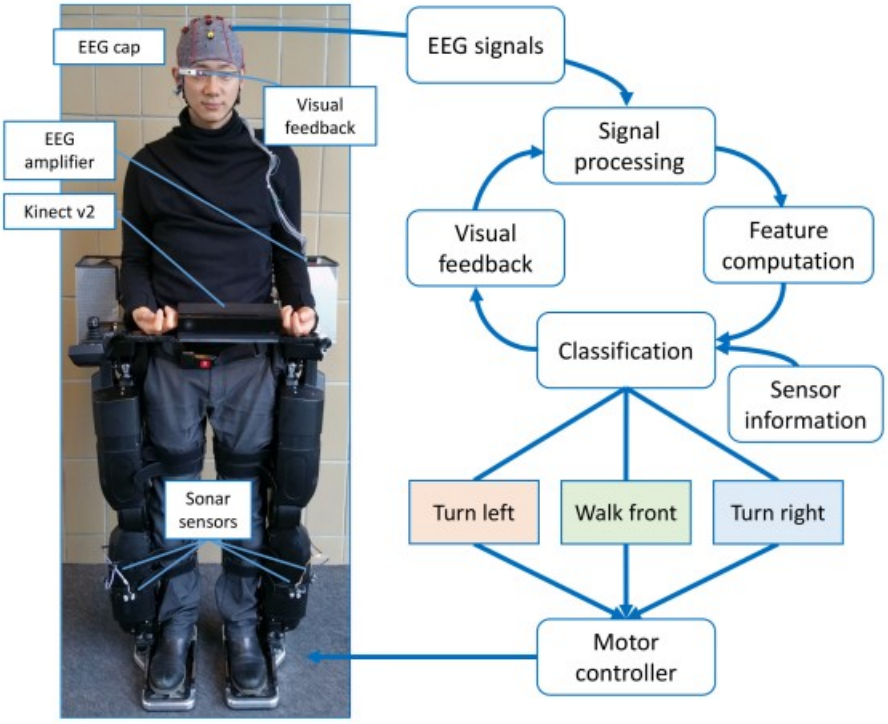


Fig. 1.2. *Illustrazione schematica dell'esoscheletro utilizzato da Lee et al. e del suo funzionamento*

Nonostante questo, alcuni studi per il controllo di robot umanoidi, anche per settori diversi da quello medico/riabilitativo, hanno dimostrato che delle BCI di tipo reattivo sono comunque efficaci. In molti casi, il sistema prevedeva un'interfaccia basata sugli *Steady State Visually Evoked Potential* (SSVEP)[1], per i quali l'utente veniva sottoposto a degli stimoli visivi oscillatori o di *flickering* di frequenza superiore a 6Hz, al fine di generare una risposta cognitiva. Per la selezione dei comandi bastava essere focalizzati su un determinato stimolo, misurando l'attività elettrica corrispondente a quella frequenza di oscillazione dalla regione occipitale del cervello.

Nel loro studio, M. Bryan et al. hanno sperimentato il controllo di un robot umanoide tramite una HBCI (*Hierarchical-BCI)* sviluppata da loro stessi, investigando nel particolare le condizioni per cui potesse essere utilizzabile in spazi di corta portata. Questa proposta di ottimizzazione comprendeva uno schermo LCD con i comandi da dare al robot, con 5 LED a frequenze differenti

1 Visual Evoked Potential (VEP): variazione del potenziale elettrico generata nella corteccia visiva in risposta a uno stimolo visivo. Tipicamente, la frequenza dello stimolo deve essere superiore ai 6Hz.

per la stimolazione SSVEP di fianco a ognuno di essi, un *headset* BCI e un robot-semiumanoide. [16]

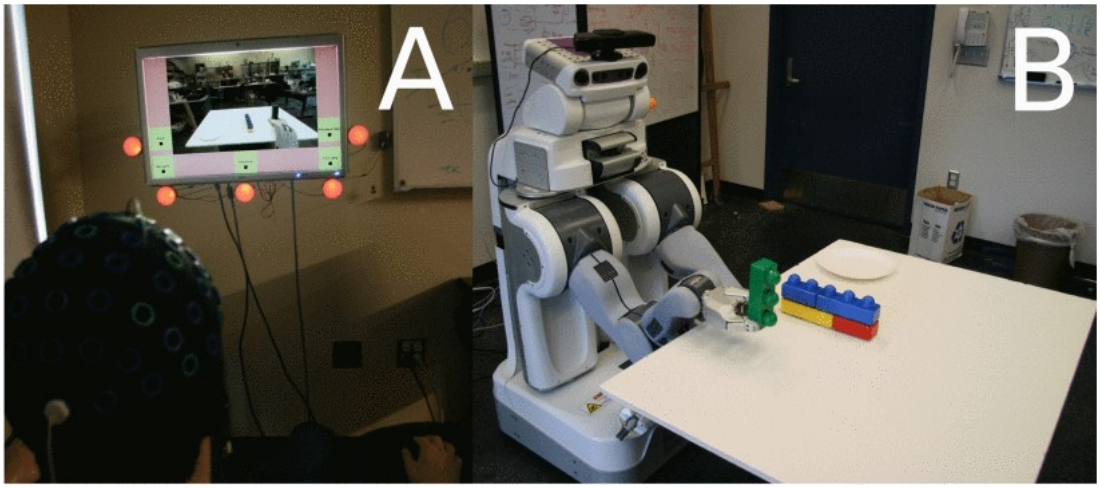


Fig.1.3. *Set-up dell'esperimento condotto da M. Bryan et al. (A) Schermo di selezione dei comandi, con i LED di stimolazione di fianco; (B) Robot semi-umanoide (PR2) per l'esecuzione delle istruzioni.*

Per prendere in considerazione altre ottimizzazioni di architetture simili, Kapeller et. al hanno dimostrato, utilizzando sempre l'approccio del *Visual Evoked Potential* (VEP), come un sistema *code-modulated* (c-VEP) sia in media più efficace di uno *frequency-coded* (f-VEP) per la continuità nell'utilizzo di un'interfaccia BCI-Robot. In questo caso, la stimolazione avviene tramite una sequenza binaria variabile di codice e non tramite una frequenza specifica e fissa.[17]

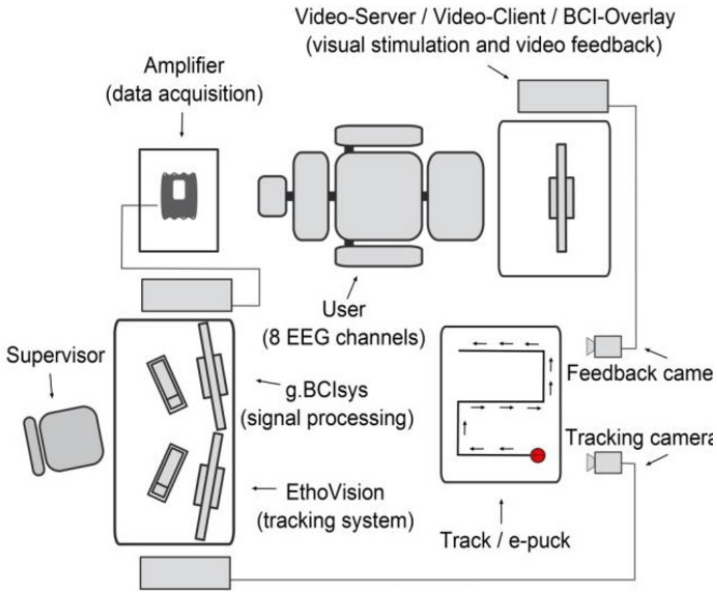


Fig. 1.4. *Rappresentazione schematica del set-up sperimentale realizzato da Kapeller et al.*

Una tipologia analoga di controllo è stata realizzata da B. Saduanov et al. [15], i quali tuttavia hanno utilizzato un approccio basato sul P300. Una matrice 4x4, presentata da uno schermo LCD,

visualizzava i comandi possibili per il robot e la stimolazione era provocata dall'illuminazione alternata delle righe e delle colonne.

L'utilizzo delle BCI legata alla stimolazione esterna è stato preso in considerazione anche per il supporto alle persone diversamente abili, come i sistemi BCI/sedia a rotelle, oppure nel settore della logistica, come l'utilizzo congiunto delle BCI con gli *AGV* (Automated Guided Vehicles).

Per il primo caso, uno studio recente di R. Na et al. ha sviluppato una sedia a rotelle elettrica *embedded* BCI-SSVEP con uno stimolatore ibrido, al fine di ottenere un'alta integrazione dei componenti, un consumo di energia ridotto e una grandezza del sistema contenuta e non ingombrante; l'hardware interno generava gli stimoli, mentre il framework di controllo proposto era responsabile per processare i dati e tradurli nei comandi associati.[27]

In particolare, il sistema era composto da uno stimolatore visuale ibrido *hardware-driven* (HHDVS), da un sensore EEG indossabile con funzionalità Bluetooth, da un'unità di controllo ed elaborazione (EMCM) e dalla sedia a rotelle elettrica fisica.

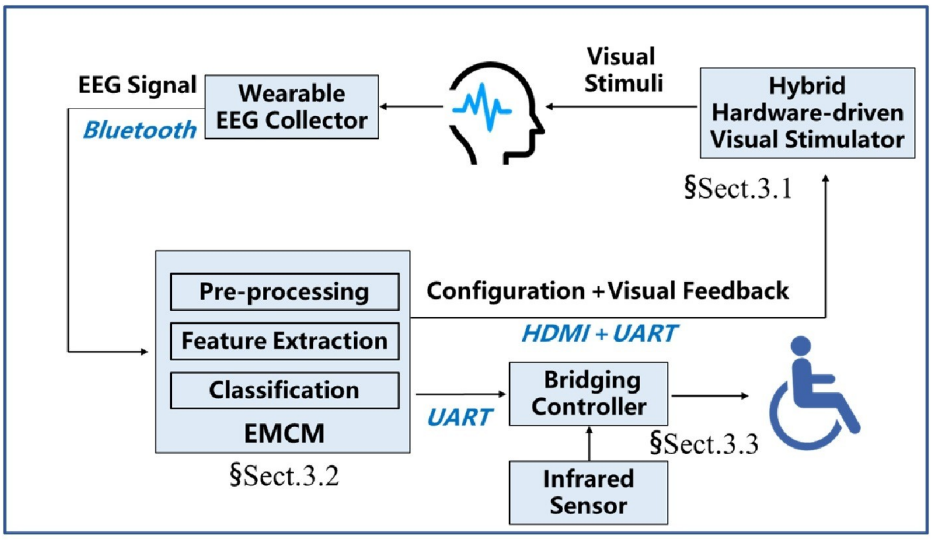


Fig. 1.5. *Architettura della sedia a rotelle elettrica embedded BCI-SSVEP proposta da R. Na et al.*

Per quanto riguarda il settore della logistica e degli AGV, M. Fiolka et al. hanno testato l'utilizzo di un *headset* EEG per il controllo tramite la mente di un veicolo autonomo, al fine di valutare la *user experience* dell'utilizzare una BCI in questo contesto. I dispositivi utilizzati erano un *headset* EEG (EMOTIV EPOC), un AGV (EMILY) e un'applicazione smartphone per i comandi.[28]
Un *headset* EEG simile (EMOTIV Insight) è stato sfruttato da Mamani e Yanyachi per il controllo di un drone in volo (Drone Bepop Parrot), coinvolgendo anche in questo caso un'applicazione mobile sviluppata con Android Studio.[29]
La tecnologia EMOTIV è stata molto sfruttata in questi ambiti, soprattutto per l'accuratezza del segnale EEG rilevato e per la versatilità di utilizzo con diversi paradigmi (SSVEP, P300, Motion-Imagery etc.).

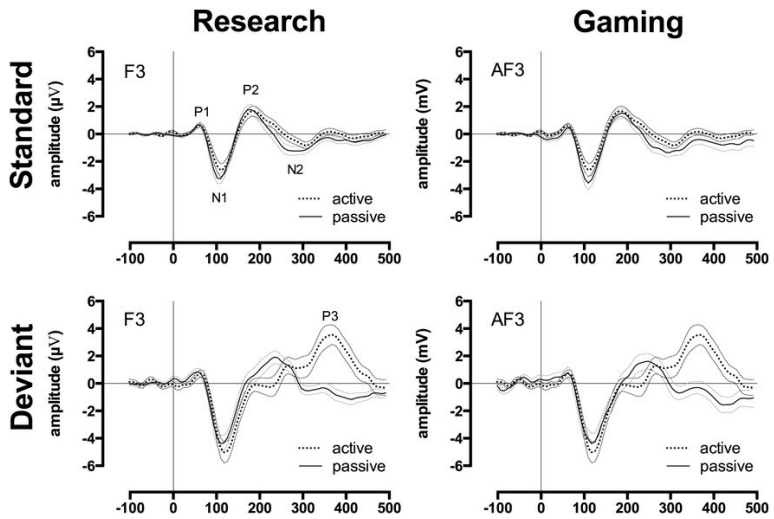


Fig. 1.6. *Comparazione dei segnali EEG rilevati da dispositivi clinici (Research) e dalla tecnologia EMOTIV (Gaming); [30]*

Come si è visto per i veicoli a guida autonoma (AGV), le BCI sono utilizzate spesso nelle sperimentazioni insieme a dispositivi intelligenti o sistemi neurali.[31]
Gli approcci tipici coinvolgono interfacce tattili (bottoni), sonore o visuali (schermi LCD, VR, AR), ma, in molti casi, l'intuitività di tali interfacce e l'esecuzione dei comandi non risultano essere sempre banali e intuitive per gli utenti. (es. *gesture*).

Ecco perché si ritiene necessario specificare che l'approccio basato sulle BCI rappresenta non solo un mezzo alternativo per la selezione e l'esecuzione dei comandi, ma una vera e propria metodologia d'azione da considerare come intuitiva, semplice e utile agli scopi dell'utente.

1.2 BCI/MR

Negli ultimi anni, i sistemi che coinvolgono le *brain-computer interface* comprendono talvolta dei dispositivi di *mixed-reality* (MR).

Nell'ambito delle simulazioni di volo, F. Dehais et al. hanno sperimentato l'utilizzo di r-BCI (*reactive-BCI*) con le p-BCI (*passive-BCI)* per misurare anche il livello di attenzione dell'utente; il sistema comprendeva un simulatore di volo, un dispositivo EEG indossabile basato sugli SSVEP (NextMind) e uno schermo su cui apparivano i comandi.[8]

Fig. 1.7. *Rappresentazione del caso d'uso della sperimentazione attuata da F. Dehais et al.; (in alto a sinistra) BCI reattiva, con i bottoni selezionabili in modalità switch on/off; (in basso a sinistra) BCI passiva, per monitorare il livello di attenzione sul radar per evitare le collisioni; (a destra) pilota nel simulatore di volo che indossa l'headset BCI (Nextmind);*

Sempre in merito a questo argomento, sono stati sperimentati diversi sistemi per l'utilizzo delle BCI con l'ausilio della *Virtual Reality* (VR). Si ritiene importante sottolineare che, soprattutto in questi casi, i problemi che si possono presentare vanno a provocare un deciso impatto sulla *user experience* (UX) e sull'efficienza effettiva del sistema.
Uno dei problemi che coinvolge i sistemi basati sul VEP è dato dall'eccessivo sforzo dell'occhio nel focalizzarsi su un determinato stimolo visivo. Per aggirare questo ostacolo, Wang et al. hanno progettato un sistema BCI indossabile in combinazione con un HMD (*Head-mounted display*) per la simulazione VR del volo di un drone. L'architettura progettata permetteva di ottimizzare al meglio gli stimoli visivi in relazione al movimento della testa dell'utente, utilizzando comandi *switch* per diminuire l'affaticamento visivo dell'utente. [18]

Nonostante questo, al fine di prevenire fenomeni di *motion-sickness* o scombussolamento, queste simulazioni coinvolgono alcune volte un'applicazione desktop-VR (*es. zSpace)*, invece di un HMD che copra tutta la visuale dell'utente.[32]

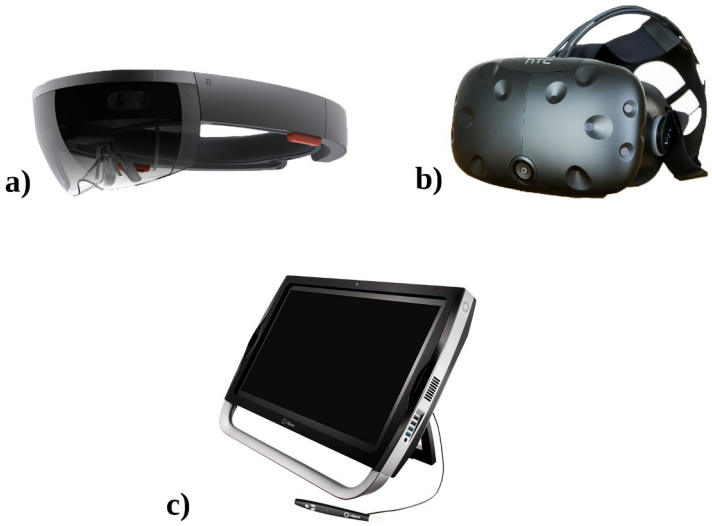


Fig. 1.8. *Alcuni esempi di dispositivi MR: a) Microsoft Hololens (AR), b) HTC Vive (immersive VR), c) zSpace (desktop VR).*

Realizzazione simili di strutture che utilizzano le BCI si ritrovano anche nel settore ludico e video-ludico, come sistemi di misurazione del livello di stress durante le fasi di gioco.[34]
H. Arora et al. hanno realizzato un prototipo che non solo mirasse a migliorare il livello di concentrazione del videogiocatore, ma anche a coinvolgerlo maggiormente e a offrirgli una *user experience* decisamente più appagante e soddisfacente.[35]

In riferimento alla realizzazione progettuale proposta in questa trattazione, si vuole porre particolare attenzione alle sperimentazioni di sistemi BCI che coinvolgono la realtà aumentata (AR). Ci sono diverse motivazioni per cui combinare queste due tecnologie può essere vantaggioso: in primo luogo, dal punto di vista delle BCI, la realtà aumentata offre nuovi spunti per collocare i feedback visuali/sonori nell'ambiente reale, oltre a offrire nuove interazioni e a migliorare la *user experience* (UX); secondariamente, dal punto di vista della AR, i paradigmi che coinvolgono le BCI offrono un'interazione *hands-free* con gli oggetti virtuali e forniscono informazioni sullo stato mentale dell'utente durante l'utilizzo.[33]

In campo medico, per esempio, lo schema AR-BCI viene sfruttato per il monitoraggio di determinati valori, come i parametri vitali di un paziente, o per la riabilitazione di persone che abbiano avuto un qualche trauma neurologico.
In merito al primo punto, Arpaia et al. hanno proposto un sistema integrato AR-BCI per il monitoraggio del paziente basato sugli SSVEP; in questo caso, il sistema trasferisce i parametri vitali del soggetto in tempo reale sugli occhiali AR, l'utente li indossa e, tramite la stimolazione visiva, seleziona i valori che gli interessa visualizzare nello specifico.[11]

Per quanto riguarda la riabilitazione, invece, Y. Yao et al. hanno ideato un sistema BCI/AR a supporto di pazienti che siano stati colpiti da ictus; le fasi del progetto prevedevano una parte di AR/SSVEP, dove l'utente si abituava all'ambiente e i segnali EEG venivano catturati in tempo reale, e una parte AR/MI (Motion Imaginery), dove il paziente immaginava il movimento dell'arto

leso con un seguente feedback visuale sul dispositivo AR; la sperimentazione è stata possibile tramite lo sviluppo di un'applicazione sul software Unity.[19]

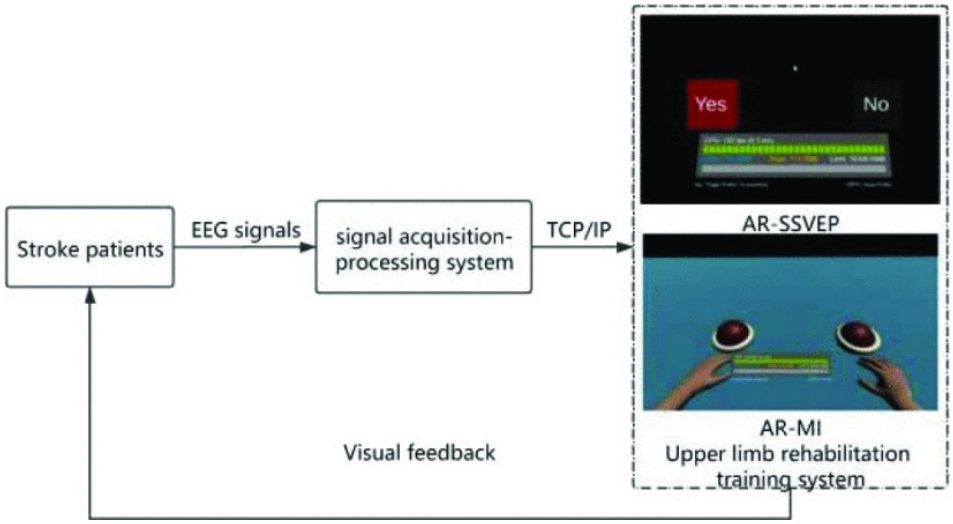


Fig. 1.9. *Rappresentazione schematica dell'idea proposta da Y. Yao et al.*

Altre ricerche hanno dimostrato come tali architetture possano essere utili per il supporto alla chirurgia durante le operazioni, come la "visione a raggi X" del corpo del paziente tramite visori AR proposta da Blum et al. L'obiettivo dell'applicazione proposta era quello di combinare una BCI con un rilevatore di movimento dello sguardo (dispositivo AR HMD), in modo tale che quest'ultimo definisse l'area di visione mentre il primo definisse il livello di zoom desiderato dall'utente. Il vantaggio principale per il chirurgo risiede nella comodità di non dover utilizzare le mani per interagire con gli ologrammi, soprattutto in contesti di operazioni chirurgiche. [20]

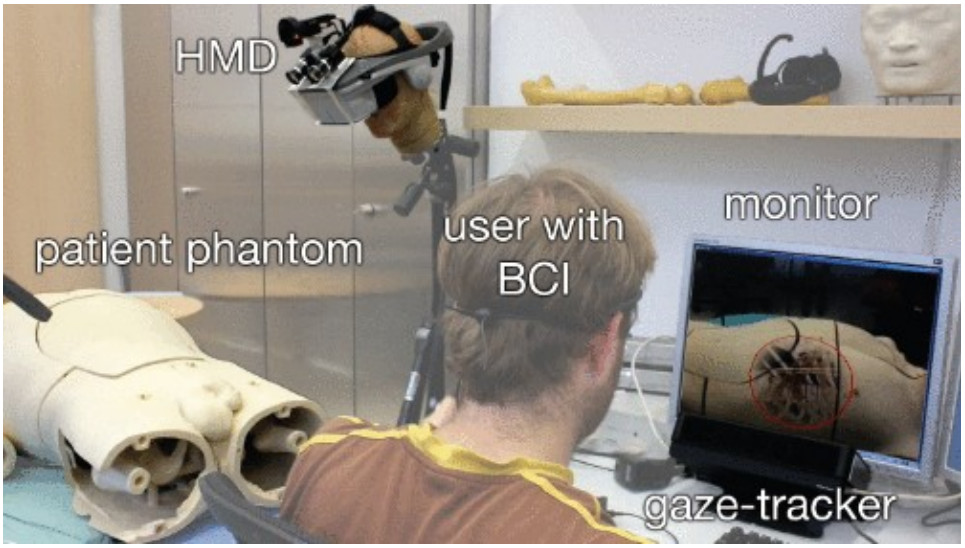


Fig. 1.10 *Set-up sperimentale del sistema realizzato da Blum et al.*

Un ulteriore terapia in cui questa combinazione è stata sperimentata è la cosidetta "terapia dell'esposizione". Per la cura di fobie e ansie, Acar et al. hanno sviluppato un sistema EEG che comprendeva uno smartphone che visualizzava l'entità soggetto della paura del paziente, aiutandoli in un ambiente totalmente sicuro; il segnale EEG misurato andava a verificare l'efficacia effettiva della terapia.

1.3 BCI/AR/Robot

Dopo aver illustrato alcuni casi d'uso riguardanti la combinazione di BCI/Robot e di BCI/AR, in questo ultimo sottocapitolo si andranno a citare alcuni esempi di sperimentazioni che hanno utilizzato il paradigma BCI/AR/Robot; l'unione di queste tre tecnologie rappresenta proprio il progetto che si vuole andare a proporre in questa trattazione.

Le tecnologie BCI e AR sono utilizzate nella robotica moderna principalmente per il controllo di robot mobili (umanoidi e non) e per il controllo di bracci robotici. Tramite la realtà aumentata è possibile avere una visione chiara dei comandi su cui si vuole porre l'attenzione, senza comunque perdere la cognizione dell'ambiente reale circostante.

Questo è stato dimostrato da realizzazioni come quelle di Escolano et al. [36], i quali hanno sviluppato un sistema BCI basato sul P-300 per il controllo di un robot mobile, e di Gergondet et al. [37], che hanno invece proposto un'architettura che utilizzava gli SSVEP.

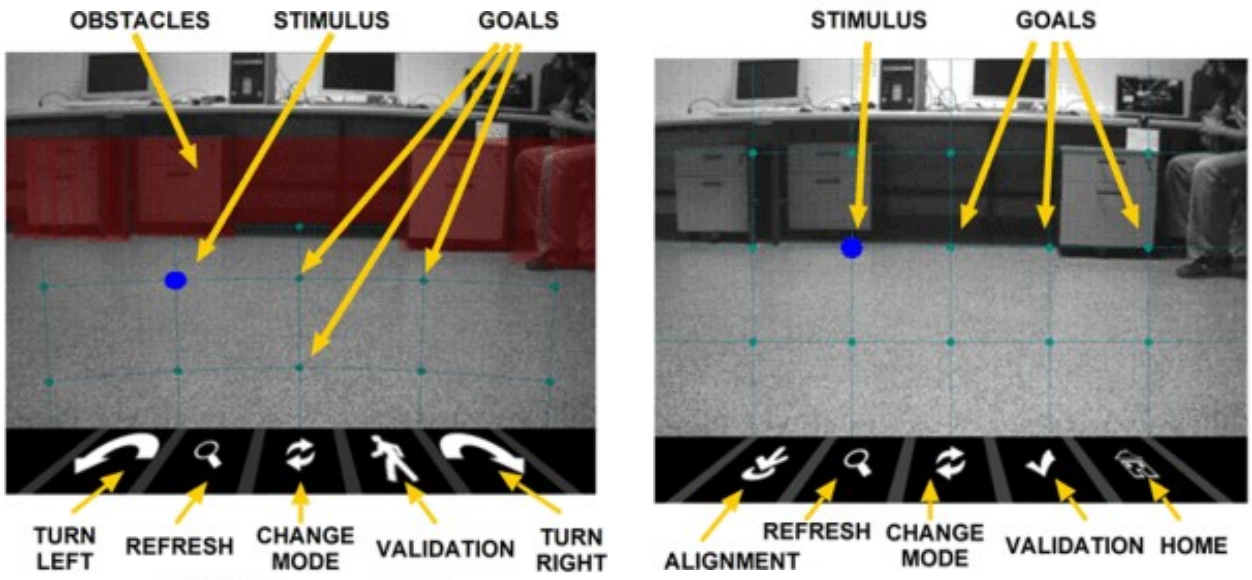


Fig. 1.11. *Visuale di controllo, tramite una telecamera installata sul dispositivo, per il controllo del robot tramite BCI nella proposta di Escolano et al.; (sinistra) visuale in modalità "robot navigation"; (destra) visuale in modalità "camera exploration".*

Il paradigma BCI/AR/Robot è utilizzato in svariati settori.

In merito al campo della chirurgia, per cui la sperimentazione sta prendendo piede e continua a evolversi nel tempo, questa tipologia di sistemi è molto utilizzata per il training virtuale; Barresi et al. hanno descritto nella loro proposta un'architettura BCI/AR/Robot che alleni l'utente nel compimento delle task e che lo mantenga concentrato. In questo contesto, attraverso il paradigma chiamato *brain controlled augmented-reality* (BcAR), l'operatore controlla un robot-chirurgico dall'interfaccia AR restando ben consapevole dello spazio che lo circonda; inoltre, i segnali EEG permettono di monitorare il livello di attenzione dell'utente sui compiti specifici da svolgere. [21]

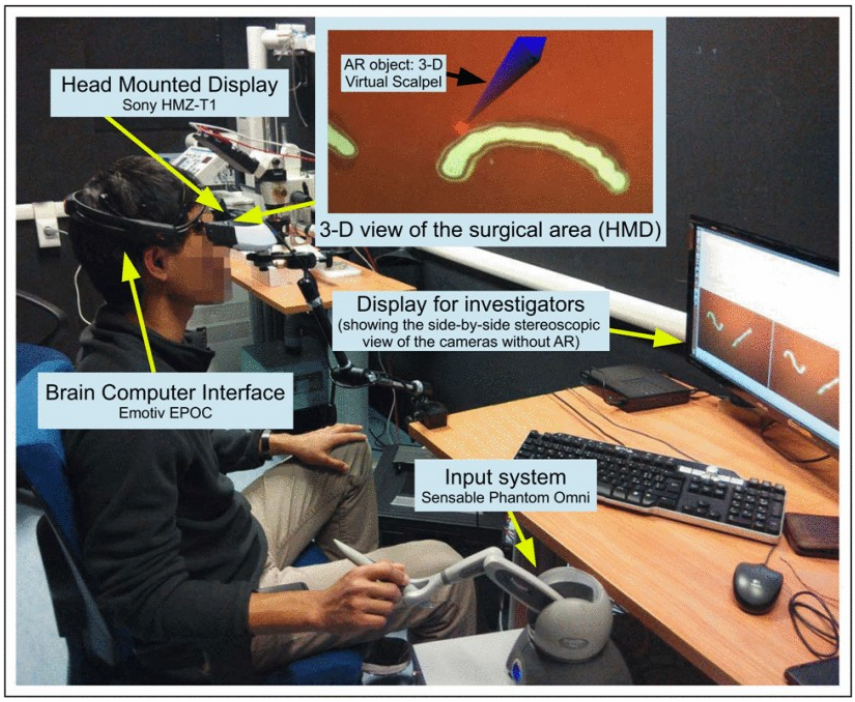


Fig. 1.12. *Set-up del modello BCI/AR/Robot elaborato da Barresi et al.*

Questo tipo di approccio viene utilizzato anche in ambito IoT in generale.
Alcuni risultati sperimentali ottenuti da Si-Mohammed et al. hanno dimostrato che una BCI e un dispositivo OST-HMD (*Optical See-Through Head-Mounted Display*) sono compatibili nel loro utilizzo congiunto, specificando anche una certa tolleranza nel funzionamento della BCI per piccoli movimenti della testa; il tutto è stato sviluppato per la creazione di un sistema-prototipo che controllasse un robot mobile. Anche in questo caso i comandi visualizzati sullo schermo AR dell'utente si basavano sugli SSVEP, provocati da determinati layout di stimolazione.[22]

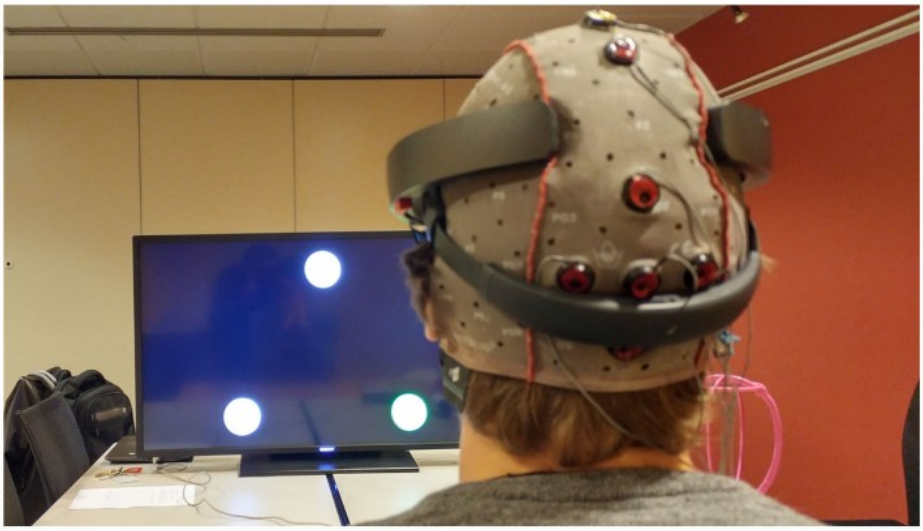


Fig. 1.13. *Set-up dello studio condotto da H. Si-Mohammed et al. ; l'utente indossa il dispositivo AR (Hololens) insieme ai sensori EEG per innescare i comandi tramite i layout di stimolazione (rappresentati sia su uno schermo LCD, sia sul dispositivo Hololens).*

In altri casi, invece degli SSVEP, si è scelto di operare secondo una modalità basata sul *P-300*. Attraverso un sistema asincrono, per cui si vuole rilevare costantemente se l'utente sta cercando di comunicare con la BCI, Lenhardt e Ritter hanno dimostrato, attraverso un prototipo, l'efficacia nel

controllare un comune robot tramite la mente e la realtà aumentata; le dinamiche dei compiti prevedevano principalmente lo spostamento di oggetti e il loro *pick and place* nell'ambiente.[23]

Una realizzazione simile, anche se nuovamente basata sugli SSVEP, si trova nel progetto proposto e realizzato da X. Chen et al.; con la combinazione di AR e BCI, utilizzando Microsoft Hololens e un sensore EEG, e di algoritmi di Computer Vision, lo studio mirava a controllare un braccio robotico per dinamiche di *pick and place*.

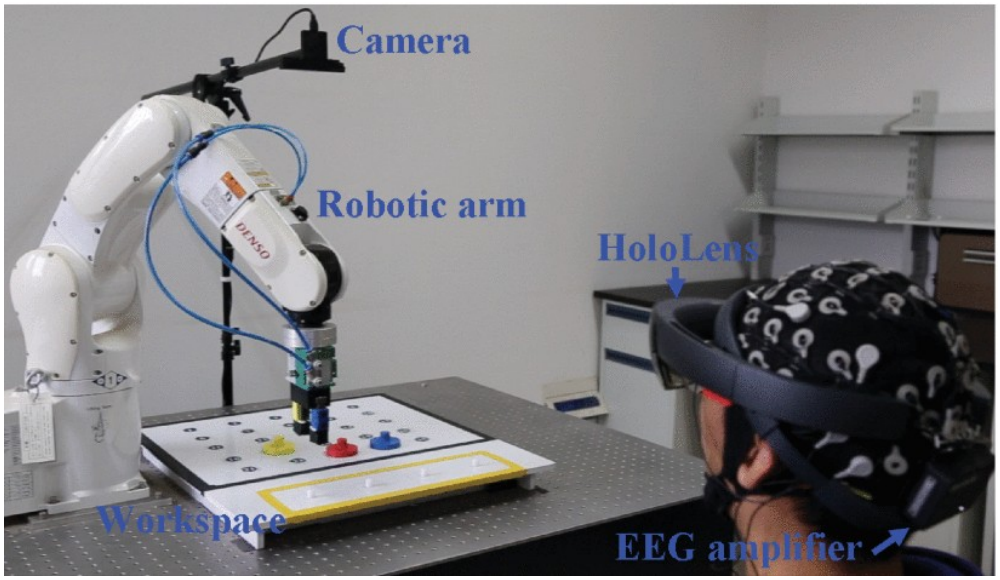


Fig. 16. *Set-up sperimentale del sistema proposto da X. Chen et al.*

Come negli esempi precedenti, il progetto che si vuole andare a proporre in questa trattazione mira a dimostrare l'efficacia della tecnologia BCI basata sugli SSVEP, nel contesto specifico dell'assemblaggio di componenti, per il controllo di un braccio robotico. In particolare, è stata sviluppata un'applicazione per Hololens 2 con la quale si vuole andare a evidenziare la comodità di un contesto operativo *hands-free* e a confrontare quest'ultima modalità con l'utilizzo delle *gesture*.

Nei capitoli successivi si andrà a illustrare nel dettaglio il funzionamento dell'architettura hardware e software del sistema, descrivendo le varie entità che vanno a comporlo e specificandone i punti di forza nell'utilizzare ognuno di essi.

2. Design del sistema e realizzazione

2.1 Introduzione: fra gesture e controllo con lo sguardo

Il progetto proposto in questa tesi riguarda la realizzazione del prototipo di un'applicazione, per il controllo di un robot non collaborativo, che utilizzi le BCI e la realtà aumentata (AR) in un tipico contesto di assemblaggio.

L'idea di base era quella di fornire un supporto concreto all'operatore-utente nelle fasi di costruzione di un determinato oggetto, in modo tale da non dover impiegare le mani per prendere le componenti, ma che gli fossero portate direttamente sul tavolo qualora ne avesse bisogno; l'applicativo è stato realizzato sul software Unity.

Per dimostrare l'efficacia di questo approccio, sono state messe a confronto le due modalità operative presenti nell'applicazione: modalità *gesture* e modalità *neuro*.
Nel primo caso, l'utente gestisce l'applicazione esclusivamente utilizzando le mani; sono presenti tutte le dinamiche di manipolazione degli ologrammi, di selezione dei bottoni per le istruzioni al robot e le ottimizzazioni relative a questa tipologia operativa; tutte le componenti presenti negli oggetti e nell'interfaccia utente che utilizzano la BCI sono disattivate.
Nel secondo caso, l'utente ha la possibilità di fornire istruzioni al robot semplicemente guardando gli elementi dell'interfaccia utente preposti a tale scopo; inoltre, può adoperare lo sguardo anche per ottenere informazioni riguardo al progetto specifico su cui sta lavorando e per manipolare, entro un certo limite, gli ologrammi visualizzati.

Il paradigma seguito per il controllo del robot è stato quello basato sugli SSVEP estratti tramite un dispostivo EEG non invasivo (NextMind), realizzando delle interfacce utente di stimolazione sugli oggetti virtuali visualizzati su un HMD-AR (Hololens 2); come si è potuto evincere dai casi studio trattati precedentemente, si tratta di un approccio efficace in molti ambiti ed è anche il più intuitivo da realizzare in termini di usabilità e comodità.
Inoltre, l'utilizzo di un *headset* AR permette all'utente di continuare ad avere una chiara percezione dell'ambiente circostante; questo aspetto è da sottolineare per quanto riguarda la sicurezza, che sugli ambiente lavorativi del settore manifatturiero risulta essere molto importante. A questo proposito, sono state integrate nell'applicazione delle interfacce utente per avvisare l'operatore nelle fasi di movimento del robot.

2.2 Complicanze iniziali

Tipicamente, sistemi di questo tipo presentano a priori delle problematiche da risolvere, sia in termini di *user experience* e di usabilità del sistema, sia in termini di difficoltà tecniche dovute alla compatibilità dei dispositivi e dei software utilizzati.

Nelle fasi iniziali del progetto, il dispositivo HMD-AR scelto per la realizzazione dell'applicazione era Microsoft Hololens (1° Generazione) in base alla grande varietà di feature offerte dal toolkit a

disposizione (MRTK) e dalla confermata compatibilità con l'*headset* BCI utilizzato, NextMind.

Tuttavia, durante le fasi di lavoro, sono emerse alcune difficoltà che hanno avuto come conseguenza la migrazione dell'applicativo sulla versione successiva e più recente del dispositivo (Hololens 2).

Di seguito, sono elencate alcune delle motivazioni per cui è stata presa la scelta in questione:

- Si è verificata la mancata compatibilità di Hololens 1 e il dispositivo NextMind utilizzando il plugin "Windows XR" nella versione di Unity 2019.4.32f; il problema riguardava la mancata capacità della BCI di rilevare il corretto *refresh rate* dal display del dispositivo Hololens, causando il mancato feedback da parte dell'interfaccia utente di stimolazione; il bug è stato risolto andando a utilizzare il plugin Windows Mixed Reality, nonostante fosse deprecato e non fosse più presente nella versione 2020 di Unity.

- Le modalità per indossare il dispositivo Hololens e *l'headset* NextMind risultavano difficoltose; questo causava un allungamento delle fasi di calibrazione della BCI, come conseguenza della non sufficiente stabilità dei dispositivi e dei movimenti compiuti dalla testa; inoltre, si è ritenuto doveroso evitare questa problematica per non aumentare il livello di stress dell'utente in fase di test e per fare in modo di non ricavarne dei dati non attinenti allo studio in questione

- Il *Mixed Reality Toolkit* (MRTK), offerto da Microsoft per gli sviluppatori Unity, contiene molti più elementi e strumenti utili per la realizzazione su Hololens 2 di quelli per Hololens 1; attualmente, la maggior parte delle sperimentazioni che hanno scelto Hololens utilizzano la seconda versione per la comodità di utilizzo, per il miglioramento degli algoritmi di *tracking* e *Spatial Awareness* e per l'aggiunta di nuove e innovative funzionalità (es. *Hand Tracking*).

Fig. 2.1. *Microsoft Hololens (1° Generazione) rilasciato nel 2016;*[38]

Nel sottocapitolo successivo, verranno illustrate le architetture Hardware e Software del prototipo del sistema definitive che si vuole andare a proporre.

Nello specifico, saranno introdotti i dispositivi hardware utilizzati nella sperimentazione e riportate le dinamiche di interconnessione tra di loro durante l'esecuzione del sistema; verranno schematizzati i processi in esecuzione dal lato software, riassumendo le SDK e le componenti di Unity sfruttate, e sarà reso noto il design progettuale messo a punto per la realizzazione dell'applicazione.

2.3 Tecnologie Hardware/Software utilizzate

Di seguito sono elencati i vari elementi per cui è stata possibile la realizzazione del sistema, andando a riportare per ognuno di essi le feature offerte e quelle utilizzate, nonché le motivazioni di tali scelte.

2.3.1 Tecnologie Software

UNITY

Unity è un *game-engine* di tipo *cross-platform* sviluppato da Unity Technologies e rilasciato nel 2005 con la sua versione 1.0.

Fig. 2.2. *Tipica interfaccia utente del software Unity* [39]

Il sistema è multi-piattaforma, ciò significa che il codice per la programmazione di un applicativo, che in questo caso avviene tramite il linguaggio C#, è possibile scriverlo una volta sola ed è adattabile ad altri possibili sistemi (Pc, Playstation, Xbox, UWP[2] etc.).

2 *UWP - Universal Windows Platform*

Con l'obiettivo iniziale di rendere lo sviluppo videoludico accessibile al pubblico e con il progredire delle versioni di aggiornamento, il software è diventato uno standard molto utilizzato sia dagli sviluppatori indipendenti, sia dalle case produttrici di videogiochi. In particolare, è molto popolare nei contesti di *mobile gaming* per iOS e Android; alcuni esempi si ritrovano in realizzazioni come *Angry Birds 2*, *Call of Duty: Mobile* e *Pokemon GO*.

Nonostante all'inizio sia nato come un software principalmente orientato ai videogames, attualmente la piattaforma offre servizi per le più svariate aree applicative e per diversi settori.
Alcuni di questi sono riportati di seguito:

- *Game consulting*: viene fornito supporto agli sviluppatori di videogiochi per l'ottimizzazione delle performance, per il design e la pianificazione, per la velocità di realizzazione e il miglioramento della *game experience*;

- *Digital Twin:* attraverso la piattaforma è possibile replicare digitalmente gli ambiente reali, realizzati nell'ambito della ricerca, dell'architettura e dei più svariati settori;

- *Simulation and Robotics*: i modelli che utilizzano l'intelligenza artificiale e il machine learning possono fare affidamento sui data set proposti da Unity, per cui è possibile migliorare e rendere valide le sperimentazioni in questo campo

- *Mixed Reality*: sono messi a disposizione strumenti e architetture per lo sviluppo di applicazioni VR/AR per l'intrattenimento, per la ricerca e in campo *educational.*

Fig. 2.3*. Esempio di simulazione nel settore del training virtuale immersivo*
[40]

Inoltre, le realizzazioni che utilizzano Unity sono agevolate dall'*Asset Store* messo a disposizione dalla piattaforma, per cui è possibile caricare gruppi di oggetti e script specifici oppure importarli direttamente all'interno del progetto su cui si sta lavorando; gli asset possono comprendere anche interi ambienti 3D e rappresentano uno strumento molto utile soprattutto per lo sviluppo indipendente.

Nel caso specifico progetto proposto in questa trattazione, la scelta di adoperare Unity come *game engine* è stata presa per diversi motivi.
In primo luogo, la natura multi-piattaforma del software permette una facile adattabilità del sistema a più dispositivi; il fatto che sia presente una gran quantità di plugin (es. OpenXR, Windows Mixed reality, Unity AR etc.) è un fattore decisamente rilevante, che ha permesso anche il cambio di HMD "in itinere" passando da Hololens (1° generazione) a Hololens 2.
Secondariamente, il NextMind Engine, responsabile della decodifica/traduzione dei segnali e integrato direttamente nell'*head-set* BCI/EEG, utilizza una SDK compatibile allo stato attuale solamente con Unity.

In base alle precedenti considerazioni, l'applicazione che verrà presentata successivamente è stata sviluppata integralmente sulla versione di Unity 2020.3.26f1 , attraverso il quale è stato possibile il funzionamento della *brain-computer interface* e il successivo invio dei dati per il controllo del braccio robotico.

VISUAL STUDIO

Visual Studio è un ambiente di sviluppo multi-linguaggio realizzato da Microsoft, la cui prima versione risale al 1997.

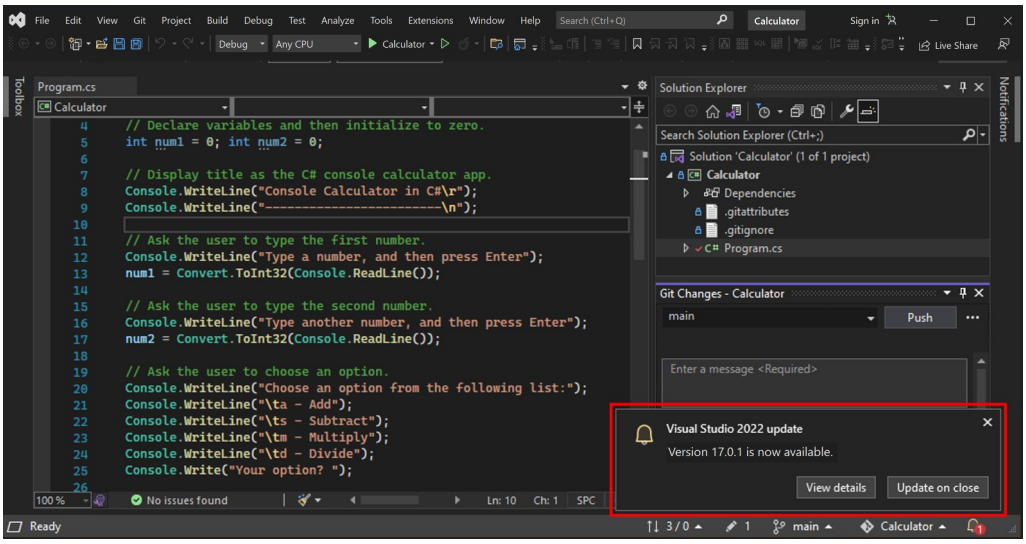


Fig. 2.4. *Esempio di interfaccia utente di Visual Studio*

Il software supporta diversi linguaggi di programmazione, quali C#, Visual Basic .NET, Java e Javascript e si integra perfettamente con Unity per la compilazione degli script. Inoltre, la tecnologia IntelliSense permette la correzione di eventuali errori sintattici durante la scrittura del codice, nonché possiede un potente debugger per la correzione in runtime e diversi strumenti per il monitoraggio delle prestazioni.
Nel progetto proposto, si è utilizzato Visual Studio per un duplice scopo:

- Scrivere gli script per il funzionamento dell'applicazione, tramite il collegamento con Unity e l'integrazione delle diverse librerie necessarie (es. MRTK, NextMind SDK)

- Attuare il *deploy* tramite USB dell'applicazione sviluppata attraverso Unity sul dispositivo Hololens 2.

PYCHARM

Pycharm è un ambiente di sviluppo integrato (IDE) realizzato da JetBrains e utilizzato nell'ambito della programmazione; nello specifico, l'utilizzo è esclusivo per il linguaggio Python e la prima versione risale al 2010.

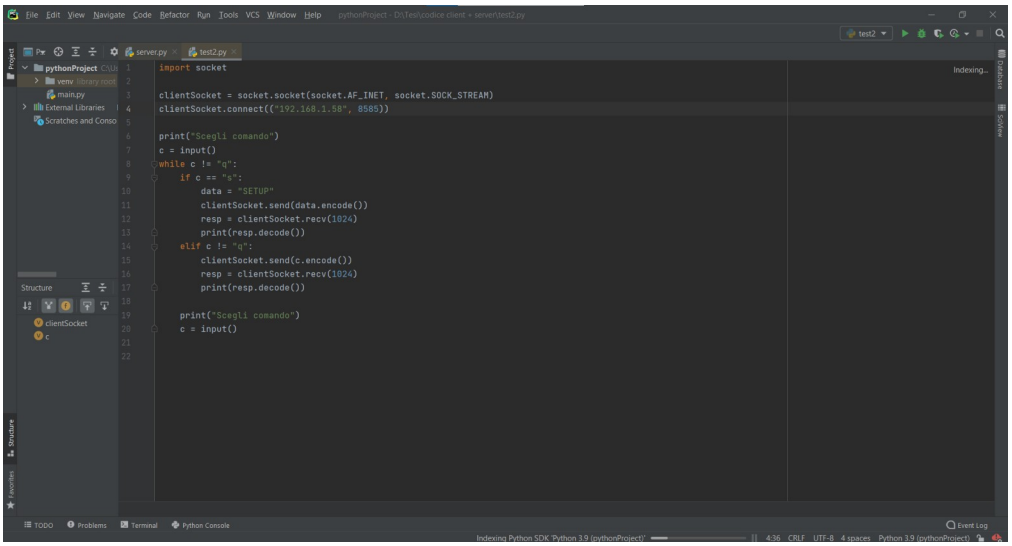


Fig. 2.5 *Tipica interfaccia utente del software Pycharm*

Anch'esso, come Visual Studio, comprende diverse feature, quali il rilevamento degli errori nella scrittura del codice e strumenti per l'analisi delle prestazioni; inoltre, il software è provvisto di un'API per cui gli sviluppatori possono realizzare i loro plugin personali, al fine di estendere gli strumenti messi a disposizione dall'applicativo.

Nel progetto proposto, PyCharm è stato utilizzato su un terminale di controllo per la scrittura degli script di movimento del braccio robotico, con l'apertura di una connessione in rete locale per il passaggio di dati reciproco del sistema Hololens/Terminale/Robot.

2.3.2 Tecnologie Hardware

NEXTMIND

La BCI utilizzata nel progetto ha coinvolto l'utilizzo del dispositivo NextMind, sviluppato dall'omonima azienda fondata da Sid Kouider, specialista in neuroscienza cognitiva.

Fig. 2.6 *Il sensore BCI-EEG indossabile di Nextmind.*

La tecnologia si basa sull'approccio degli *Steady State Visually Evoked Potential* (SSVEP) e permette di estrarre i segnali dal cervello tramite EEG non invasiva; in sostanza, permette di controllare un'interfaccia utente, in tempo reale, direttamente tramite l'attenzione visuale verso un determinato oggetto.

Allo stato attuale il "NextMind Dev. Kit", la versione messa in commercio, è specifica per gli sviluppatori e comprende tre elementi fondamentali:

- Il **NextMind Sensor,** che rileva in maniera precisa i segnali EEG dalla corteccia visiva; si pone direttamente sulla parte occipitale della testa, in modo tale da garantire libertà di movimento all'utente, tramite il supporto di una banda elastica; il sensore è possibile anche utilizzarlo insieme a *headset* VR/AR;

- Il **NextMind Engine**, responsabile della decodifica dei segnali rilevati dal sensore, trasmessi via Bluetooth; il funzionamento si affida ad algoritmi che combinano il machine learning e le neuroscienze, al fine di convertire l'attenzione dell'utente nei relativi feedback di output;

- La **NextMind SDK**, utilizzata specificatamente su Unity e contenente tutti gli strumenti per lo sviluppo di un'applicazione che utilizzi questa tecnologia.

Nel progetto proposto, il Nextmind è stato utilizzato come BCI per far si che l'utente non debba utilizzare le mani durante le fasi di assemblaggio di un oggetto. In particolare, sono state realizzate delle interfacce di stimolazione, essendo una BCI basata sugli SSVEP, per controllare la *user interface* (UI) dell'applicazione e per azionare le dinamiche di *pick and place* del braccio robotico.

MICROSOFT HOLOLENS 2

Hololens 2 è un HMD-AR sviluppato e costruito da Microsoft, la cui data di rilascio ufficiale risale al 2019.

Fig. 2.7 *Microsoft Hololens 2*

Il dispositivo è stato utilizzato in diverse applicazioni AR sviluppate non solo nel campo della ricerca scientifica, ma anche in esecuzioni pratiche vere e proprie per i più svariati settori. Alcuni di questi riguardano: la produzione industriale, l'ingegneria e l'edilizia, l'assistenza sanitaria e l'istruzione.

Rispetto al suo predecessore, Hololens 2 offre una vasta quantità di feature aggiuntive che contribuiscono a creare un ambiente di lavoro comodo (es. *hand tracking, eye tracking*) e stimolano la collaborazione e l'interconnessione di più utenti (es. Microsoft Mesh). Inoltre, le modalità per indossare il dispositivo sono decisamente migliorate in confronto alla prima versione; la possibilità di alzare la visiera frontale per far riposare gli occhi, pur mantenendo l'*headset* indossato sul capo, ha rappresentato una comodità in più durante le fasi di testing della sperimentazione trattata.

Per quanto riguarda lo sviluppo dell'applicazione trattata in questa tesi, è stato utilizzato il ***Mixed Reality Toolkit*** **(MRTK)** offerto da Microsoft; si tratta di un progetto specificatamente pensato per Unity, il quale offre una serie di funzionalità e componenti per lo sviluppo di app di realtà mista.
Alcuni di questi servizi comprendono:

- La fornitura del sistema di input multipiattaforma e gli elementi predefiniti per le interazioni spaziali e l'interfaccia utente;

- La possibilità di creare in maniera rapida i prototipi tramite simulazione nell'editor, il quale permette di visualizzare le modifiche in modo immediato;

- La garanzia di un'operatività basata su un framework estendibile, in modo tale che gli sviluppatori possano sostituire le componenti di base

- Il supporto di una vasta gamma di piattaforme: OpenXR, OpenVR, Oculus, dispositivi mobili etc.

Inoltre, l’utilizzo di questo toolkit ha permesso una migrazione dell’applicazione da Microsoft Hololens (1° Generazione) a Microsoft Hololens 2 senza alcun tipo di problematica aggiuntiva.

L’Hololens 2 è stato utilizzato nel sistema proposto, tramite il supporto del MRTK, come display AR per la visualizzazione dell’applicazione sviluppata con Unity e per la ricezione delle interfacce di stimolazione preposte al funzionamento della BCI.

E.DO ROBOT

E.DO è un robot modulare e multi-asse, prodotto da Comau, sviluppato principalmente per il mondo della formazione.

Fig. 2.8 *e.Do Robot.*

Il braccio robotico è costituito da 6 assi, che si interfacciano e comunicano tra loro in modo modulare e indipendente; conseguentemente, possiede 6 gradi di libertà e le sue principali applicazioni sono legate allo spostamento e allo smistamento automatico di determinati oggetti (*pick and place*).

Inoltre, gli sviluppatori hanno la possibilità di programmarlo in maniera semplice utilizzano l’apposita applicazione (e.DO App), oppure le SDK realizzate per Matlab o Python.

Come caso analogo ai robot che vengono utilizzati tipicamente nell’industria, e.DO Robot rappresenta un buon caso d’uso sperimentale ai fini del progetto proposto. In particolare, il braccio robotico ha avuto la responsabilità del *pick and place* delle componenti sul tavolo dell’utente, rispettando i comandi in arrivo dal terminale di controllo.

2.4 Architettura del sistema

2.4.1 Entità Hardware/Software

La sperimentazione del sistema ha coinvolto diverse entità hardware e software interconnesse fra di loro:

- Un'**applicazione** sviluppata tramite **Unity 2020.3.23f1**; l'utilizzo congiunto con Hololens 2 è stato possibile grazie all'API OpenXR open-source, adattabile a svariati dispositivi e specificamente raccomandata dalla documentazione Microsoft; le interazioni con l'headset BCI sono state realizzate grazie all'SDK di Nextmind, che attualmente è disponibile esclusivamente per Unity;

- Il **Microsoft Hololens 2**, indossato dall'utente per la visualizzazione degli ologrammi, delle interfacce utente e delle componenti dell'oggetto che deve assemblare, nonché per la ricezione degli stimoli al fine di far funzionare correttamente l'interfaccia cervello-computer;

- Il **NextMind Sensing Device**, posto sulla parte occipitale della testa dell'utente con il supporto di una banda elastica opportunamente regolata; il sensore EEG è responsabile della ricezione dei potenziali SSVEP provocati dalle interfacce di stimolazione visualizzate, mentre il NextMind Engine integrato nel dispositivo si occupa della parte di decodifica e traduzione dei segnali sull'applicativo; la connessione con Hololens 2 avviene via Bluetooth;

- Un **pc fisso**, preposto ad agire come terminale di controllo per l'esecuzione dei comandi da mandare al braccio robotico; per raggiungere tale scopo, è stato realizzato uno script Python con le seguenti funzionalità:
  - Aprire una socket per la connessione con il robot tramite il suo indirizzo IP;
  - Ricevere i dati in arrivo dall'applicazione, in particolare delle stringhe a indicare il setup del sistema pc/robot o a rappresentare un componente specifico dell'oggetto da assemblare in esame;
  - Elaborare il dato e conseguentemente mettere in azione il robot in base al comando specifico ricevuto; in particolare, vengono fornite al braccio robotico le coordinate di un punto specifico nello spazio reale, andando a specificare i movimenti da compiere e la loro velocità.
- Il **Robot e.DO**, responsabile del *pick and place* delle componenti di un determinato oggetto da assemblare sul tavolo dell'utente;
- Un **modem Wi-Fi**, per la connessione in rete locale del dispositivo Hololens, del terminale di controllo e del braccio robotico.

2.4.1 Architettura hardware

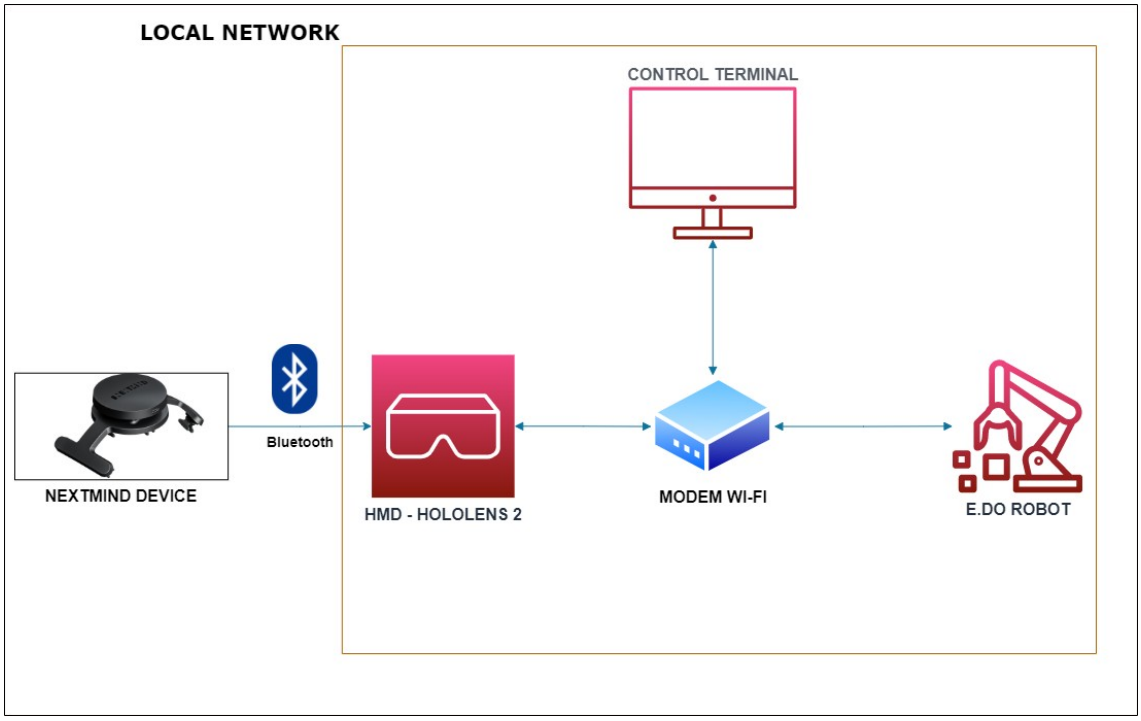


Fig. 2.9. *Il diagramma raffigura i collegamenti hardware in atto durante il funzionamento del sistema;*

Le diverse interconnessioni che coinvolgono le entità hardware durante l'esecuzione sono le seguenti:

- Il dispositivo NextMind è connesso tramite tecnologia Bluetooth a Hololens 2, mentre quest'ultimo instaura una connessione in rete locale con il modem Wi-FI, che riceve i pacchetti e li manda al terminale di controllo;
- Dopo l'elaborazione, i pacchetti contenenti i dati per le esecuzioni dei movimenti del robot vengono inviati al modem, che li inoltra al braccio robotico e.DO;
- Il robot manda indietro un feedback del confermato successo o fallimento di una determinata istruzione, che viene ricevuto dal terminale dopo il passaggio dal modem;
- Infine, il terminale restituisce un pacchetto analogo al dispositivo Hololens, sempre passando per il modem Wi-Fi, in modo che vengano ricevuti i dati per la conferma dell'esecuzione di un comando o per il suo fallimento.

2.4.2 Architettura software

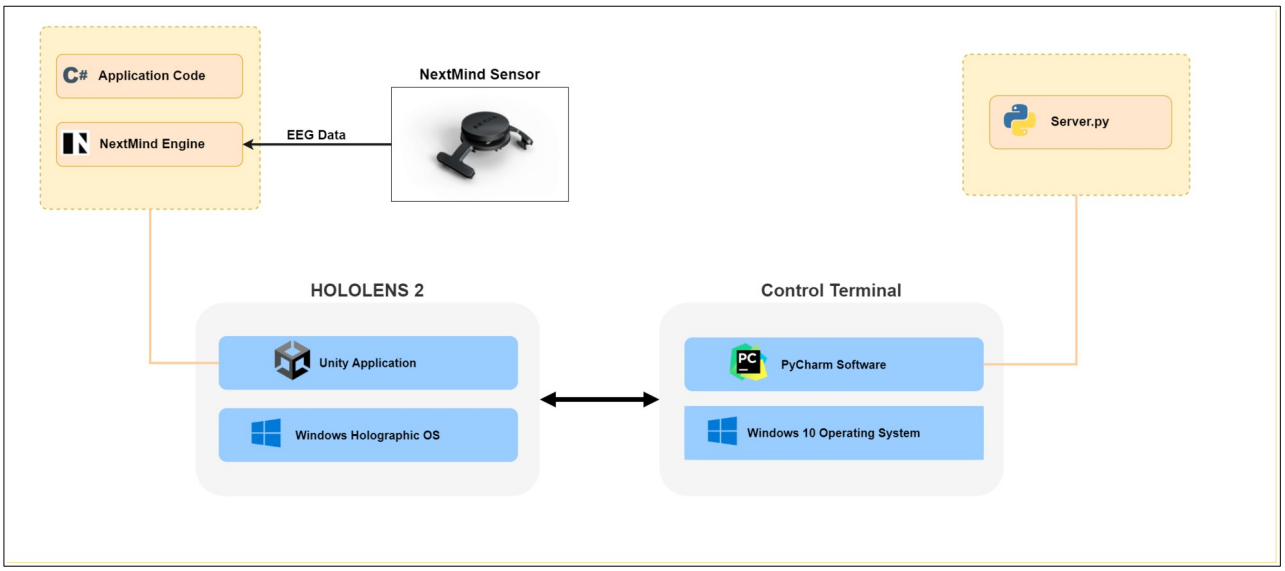


Fig. 2.10. *Rappresentazione schematica dell'architettura software;*

Le seguenti fasi vanno a descrivere i processi in esecuzione, come si può vedere dalla figura 19, durante il funzionamento del sistema:

- Sul dispositivo Hololens sono attivi i processi del sistema operativo integrato, Windows Holographic, e quelli dell'applicazione Unity installata;

- In base all'interazione dell'utente con le interfacce di stimolazione nell'applicativo, il NextMind Engine, integrato grazie alla SDK di NextMind per Unity, processa il segnale EEG rilevato dal sensore e lo traduce in un feedback;

- Il feedback viene successivamente interpretato da Unity in base al codice scritto per le esigenze dell'applicazione; in particolare, quando si interagisce con l'interfaccia preposta per il *pick and place* delle componenti da parte del robot, i dati vengono mandati al terminale di controllo ed elaborati tramite uno script Python sul software PhyCharm;

- Le coordinate delle componenti selezionate dall'utente sono già state definite tramite lo script Python, dunque il robot è stato programmato in maniera tale da prelevare gli oggetti in punti fissi e prestabiliti;

- Infine, i processi dello script Python ricevono il feedback dal robot e lo rimandano all'applicazione Unity, la cui architettura di codice provvederà all'aggiornamento dell'interfaccia utente.

3. Design dell’applicazione

3.1 Introduzione

In questo capitolo verrà illustrato nel dettaglio il funzionamento dell’applicazione di supporto all’assemblaggio sviluppata tramite Unity.

Il concetto di tale applicativo si basa su un duplice scopo: permettere all’utente di ricavare informazioni rispetto all’oggetto da assemblare, visualizzate sullo schermo tramite l’interfaccia utente progettata, e garantirgli una selezione dei pezzi comoda ed efficace tramite la BCI.
Dopo aver valutato diverse possibilità in fase di ideazione, si è stabilito che l’opzione migliore per l’assemblatore dovesse essere quella di avere l’oggetto composto come ologramma AR direttamente sul suo tavolo da lavoro. Questa dinamica permette sia l’adattabilità del sistema ad ambiti diversi, sia la comodità per l’assemblatore di non utilizzare le mani e selezionare velocemente le componenti.

Il proposito riguardante la selezione si articola nella fase di assemblaggio, dove l’utente può accedere alle varie componenti dell’oggetto e inviare al robot il comando per il *pick and place* di una componente specifica. Al fine di effettuare il confronto tra *gesture* e BCI, sono state integrate due modalità operative attraverso cui l’utente può selezionare le varie parti dell’oggetto che necessita: modalità ***gesture*** e modalità ***neuro*.**
In modalità *gesture*, l’utente interagisce con i pezzi esclusivamente utilizzando le mani; questo comprende sia le opzioni che forniscono informazioni all’operatore, sia la selezione specifica.
In modalità *neuro*, invece, queste dinamiche interattive sono esclusivamente accessibili tramite la focalizzazione dell’utente riguardo a un determinato elemento; le opzioni disponibili sono pressoché uguali alla modalità *gesture*, mentre l’interfaccia utente è diversa in relazione all’integrazione di elementi di stimolazione per il funzionamento della BCI utilizzata (NextMind).

Nei prossimi sottocapitoli, in primo luogo, verranno illustrate le diverse fasi di funzionamento dell’applicazione, andando a specificare nel dettaglio quando sono disponibili e le azioni che l’utente può compiere per ognuna di esse. Si porrà particolarmente attenzione alla fase di assemblaggio, in quanto rappresenta lo scopo principale di tutta l’applicazione.

Secondariamente, verranno riportati tutti le tipologie degli elementi utilizzati facenti parte dell’interfaccia utente; il discorso comprende i bottoni, i pannelli di informazione e di *warning*, i menù e le varie interfacce di stimolazione BCI (*NeuroTag*).

Infine, sarà schematizzata l’architettura del codice scritto per il funzionamento dell’applicazione, descrivendo le entità di gestione, quelle degli oggetti specifici, le entità UI e le *interfaces* integrate.

3.2 Architettura

3.2.1 Introduzione

Il percorso dell'utente attraverso l'applicazione avviene tramite le tre scene Unity che sono state create: *Main Menu Scene*, *Calibration Scene* e *Project Scene.*

Nella *Main Menu Scene,* l'utente entra a contatto per la prima volta con l'interfaccia dell'applicativo; questa ha la principale funzione di collegare le altre due scene. Si sottolinea che il caricamento delle scene avviene prevalentemente in modalità additiva, in modo tale da avere la possibilità di accedervi contemporaneamente, qualora ce ne fosse bisogno, durante l'esecuzione del sistema. In merito a questa scena, l'accessibilità è sempre garantita tramite l'interfaccia utente (con bottoni specifici).

Nella *Calibration Scene,* l'utente procede a collegare a Hololens 2 il dispositivo NextMind e a calibrarlo correttamente; alla fine della procedura, se il punteggio di calibrazione è buono, l'utente può passare nuovamente al main menu. Questa scena è accessibile sia nella *Project Scene,* sia nel *Main Menu*.

Nella *Project Scene,* avviene la fase di assemblaggio in senso stretto: la selezione delle componenti di un determinato oggetto, tramite *gesture* o tramite BCI, per innescare le dinamiche di *pick and place* del robot; questa scena è esclusivamente accessibile solo dalla *Main Menu Scene*.

In funzione delle scene di cui fanno parte, è possibile individuare 4 fasi di interazione dell'utente con l'applicazione:

- **Fase Principale** (Main Menu Scene);
- **Fase di Calibrazione** (Calibration Scene);
- **Fase di Selezione del Progetto** (Project Scene);
- **Fase di Assemblaggio** (Project Scene).

I passaggi fra le diverse scene sono identificati nella figura 3.1, così come si possono notare le fasi di interazione dell'utente interne ad esse.

Nei sottocapitoli successivi verranno illustrate le funzionalità di ognuna di queste fasi, ponendo particolarmente attenzione alla fase di assemblaggio. Quest'ultima rappresenta il fulcro di tutta la trattazione, in quanto proprio in questa fase vengono attivate le dinamiche di *pick and place* del robot tramite BCI o *gesture.*

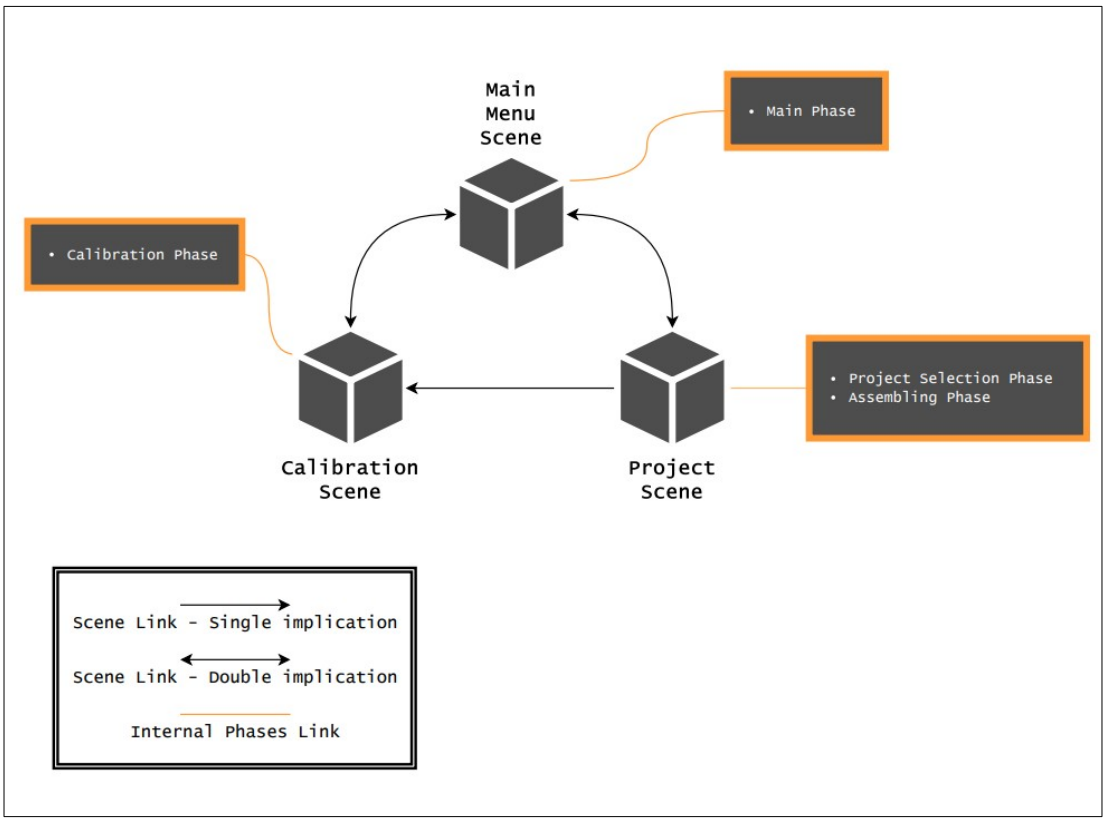


Fig. 3.1. *Rappresentazione schematica dell'architettura delle scene su Unity, con le relative fasi incluse in esse;*

3.2.2 Fase Principale

La fase principale avviene all'interno della scena "Main Menu" ed è la prima fase che l'utente incontra all'apertura dell'applicazione.

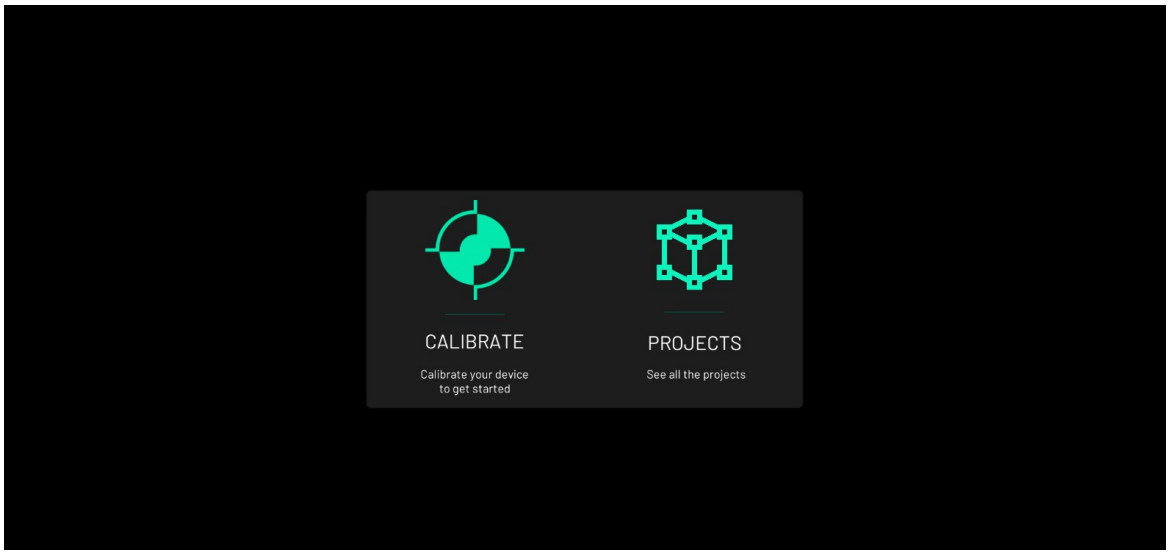


Fig. 3.2 *Interfaccia utente del main menu che definisce pienamente la fase principale*[3]*;*

3 Le immagini con sfondo totalmente nero sono state catturate da un simulazione tramite emulatore Desktop.

Siccome rappresenta il menù principale, l'unica funzionalità individuabile è quella di permettere all'utente di passare da una scena all'altra. Per questo motivo, l'unica e semplice interfaccia presente comprende due bottoni: "Calibrate" e "Projects".

Rispettivamente, l'utente verrà reindirizzato alla "Calibration Scene" o alla "Project Scene", aggiungendo la scena caricata a quelle attive e disattivando la visibilità dell'interfaccia di menu principale.

Inoltre, l'interfaccia include un constraint di *gaze tracking*, per cui le sue coordinate sono legate al movimento della testa dell'utente.

3.2.3 Fase di Calibrazione

Durante questa fase l'utente procede alla calibrazione del dispositivo NextMind, il supporto BCI scelto per lo sviluppo del sistema.

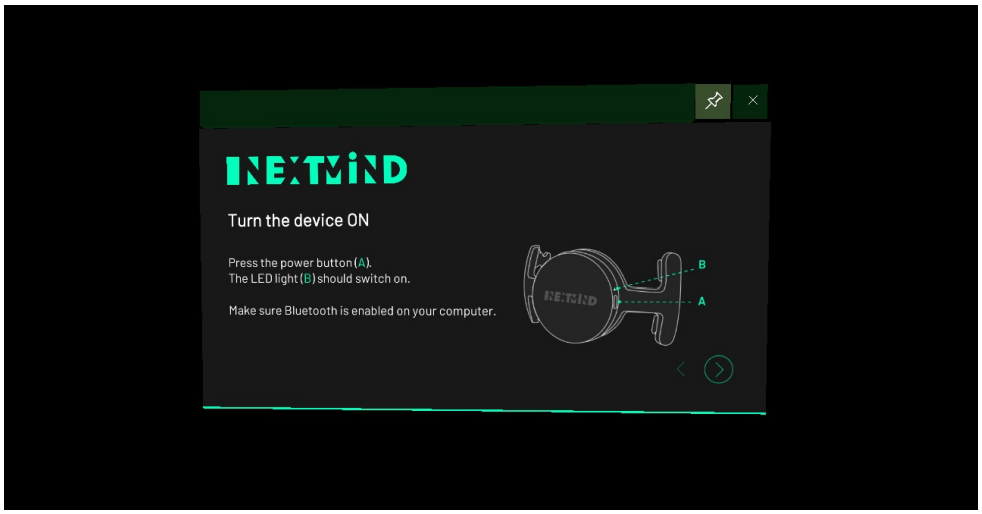


Fig. 3.3 *Pannello iniziale presentato all'utente durante la fase di calibrazione;*

L'interfaccia comprende 9 differenti pannelli, la cui visibilità è resa attiva in funzione dell'interazione dell'utente con essi, e una barra superiore, per cui è possibile spostare l'interfaccia tramite *gesture* nella posizione desiderata. Come per il main menu, anche in questo caso è attivo il *gaze tracking* e i pannelli saranno sempre orientati in base all'orientamento della testa dell'utente.

I pannelli principali, insieme alle loro funzionalità, sono i seguenti:

- *Device Connection Panel*: in questa sezione preliminare viene verificata la connessione Bluetooth del dispositivo NextMind con Hololens 2; nel caso non sia stato effettuato il *pairing*, è possibile procedere all'associazione dei due dispositivi; in caso contrario, la connessione Bluetooth avviene in automatico;

- *Adjust Device Panel:* in questo pannello viene effettuata una verifica del contatto degli elettrodi con lo scalpo della testa dell'utente; l'interfaccia raffigura un'immagine degli elettrodi del dispositivo, con un colore tendente al rosso o al verde a seconda della qualità più o meno alta del contatto; in questa sotto-fase, all'utente è consigliato di sistemare in maniera ottimale il dispositivo sul retro della testa; tipicamente, per una calibrazione efficace, gli elettrodi devono essere tutti verdi;

- *MonoTarget Calibration Panel:* a questo punto avviene la calibrazione vera e propria; all'utente viene consigliato di restare fermo e di concentrarsi su un'interfaccia circolare di stimolazione, comprendente un materiale di *flickering* e delle linee verdi al centro; nel caso in cui la calibrazione stia andando bene, le linee verdi formeranno un triangolo per fornire un feedback all'utente;

- *Result Panel*: in questa parte, infine, viene fornito all'utente il punteggio ottenuto durante la calibrazione e calcolato dal NextMind Engine; l'intervallo di valori che rappresentano questa valutazione va da 1 (calibrazione scarsa) a 5 (calibrazione ottima); se il punteggio è basso, all'utente viene data la possibilità di ripetere la calibrazione del dispositivo;

Nel caso in cui la calibrazione sia soddisfacente, è possibile ritornare al Main Menu, per il quale l'utente può passare alla fase di selezione del progetto interagendo con il bottone "Projects".

3.2.4 Fase di Selezione del Progetto

In questa fase, che avviene nella *Project Scene*, l'utente entra in contatto per la prima volta con l'oggetto da assemblare e con le modalità operative disponibili.

In primo piano, l'interfaccia presenta il nome del progetto corrente, il modello tridimensionale dell'oggetto corrispondente e un menu di fianco a esso. Quest'ultimo comprende un bottone di **Info**, che fornisce un pannello con le informazioni riguardo al progetto, e un bottone di **Start Assembling**, che permette all'utente di entrare nella fase di assemblaggio.

Le interazioni dell'utente con il resto degli elementi dell'interfaccia permettono le seguenti funzionalità:

- *Project Selection*: il bottone "Next" posto sotto l'oggetto corrente permette all'utente di passare da un progetto all'altro; le sue coordinate sono strettamente legate al gruppo che comprende mesh, titolo e menu laterale;

- ***Table Definition***: è possibile definire la posizione di un tavolo da lavoro nello spazio reale, trascinando una mesh piana nel punto desiderato, alla quale verrà legato l'oggetto tramite le sue coordinate spaziali; la definizione del tavolo da lavoro è accessibile sia da un menu in alto, che segue l'utente tramite *gaze tracking*, sia da un menù "a mano", visualizzabile tenendo il palmo della mano rivolto verso l'alto; la definizione del tavolo è possibile anche durante la fase di assemblaggio;

- ***Operational Mode Selection***: l'utente può già scegliere quale modalità operativa (*gesture/neuro)* vuole andare ad utilizzare per l'assemblaggio; anche in questo caso, si può accedere alla selezione della modalità sia dal menu in alto, sia dal menu "a mano"; questa definizione è disponibile anche durante l'assemblaggio; in questa fase, l'interfaccia non presenta nessun cambiamento al variare della modalità selezionata;

Si specifica che non è possibile passare alla fase di assemblaggio se non è stato definito alcun tavolo da lavoro, in quanto l'oggetto non sarebbe legato a nessuna coordinata nello spazio.
Questo non significa che, con il tavolo definito, l'utente non possa muovere l'oggetto a suo piacimento, ma che possa comodamente riportarlo sul tavolo qualora ne avesse bisogno.

Una volta definito il tavolo e selezionato il progetto, l'utente ha la possibilità di passare alla fase di assemblaggio.

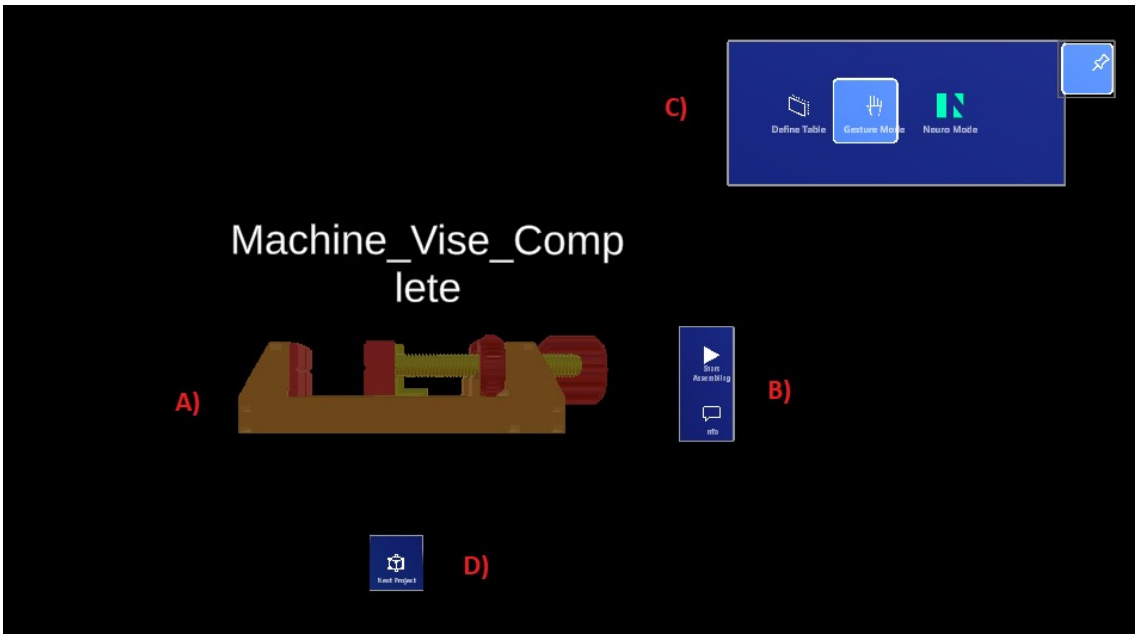


Fig. 3.4 *Interfaccia utente della Project Scene; A) Mesh del progetto corrente; B) Menu laterale con i bottoni di Start Assembling e Info; C) Menu superiore, per il cambiamento della modalità operativa e per la definizione del tavolo; D) Bottone di Next Project, per passare da un progetto all'altro;*

3.2.5 Fase di Assemblaggio

Una volta iniziata questa fase, la mesh del progetto selezionato si trasferirà automaticamente sul tavolo definito dall'utente.

In un primo momento, l'oggetto è ancora completamente assemblato e presenta un'interfaccia utente leggermente diversa dalla fase di progetto, come si può evincere dalla figura .

Tale interfaccia utente permette le seguenti funzionalità:

- *Manipolazione dell'oggetto assemblato*, con un vincolo sulla scala perché non sia né troppo grande, né troppo piccolo;
- *Ancoraggio/Disancoraggio* dell'oggetto in un punto dello spazio;
- *Return* dell'oggetto nella posizione del tavolo
- ***Explode***, per cui viene visualizzato il modello esploso dell'oggetto tramite un'animazione e possono essere individuate le sue componenti;

Lo stato dell'oggetto quando si interagisce con quest'ultima funzionalità, quella di esplosione, rappresenta la fase di assemblaggio in senso stretto; la selezione delle componenti, per il *pick and place* del robot sul tavolo, e le altre opzioni sono disponibili proprio in questa fase.

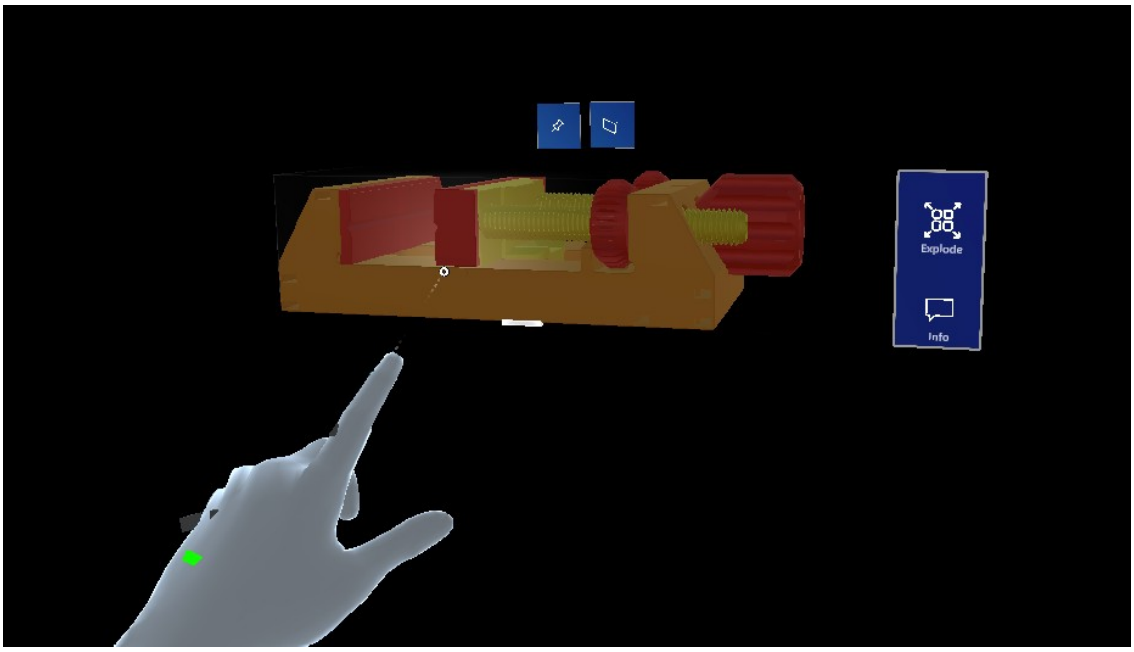


Fig. 3.5 *Modello completamente assemblato e UI associata all'oggetto;*

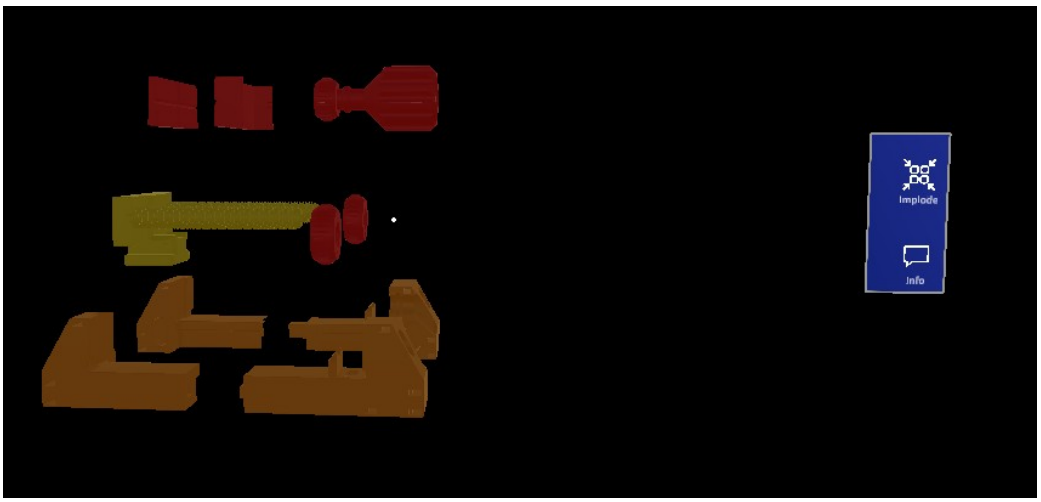


Fig. 3.6 *Modello esploso dello stesso oggetto, ottenuto tramite l'interazione con il bottone "Explode"; Quest'ultimo, dopo l'esplosione, subisce una variazione nel testo, diventando "Implode", e nell'icona;*

Il modo tramite cui avvengono queste interazioni dipende dalla modalità operativa scelta: *gesture mode* o *neuro mode*, che vengono definite dall'utente all'inizio della sessione di assemblaggio (rimane comunque la possibilità di cambiare modalità in ogni momento).

Modalità Gesture

In questa modalità, l'utente a piena libertà di manipolazione, tramite *gesture*, delle mesh che rappresentano i componenti dell'oggetto. In particolare, le interazioni possibili tramite manipolazione sono quelle di traslazione, rotazione e scalamento.

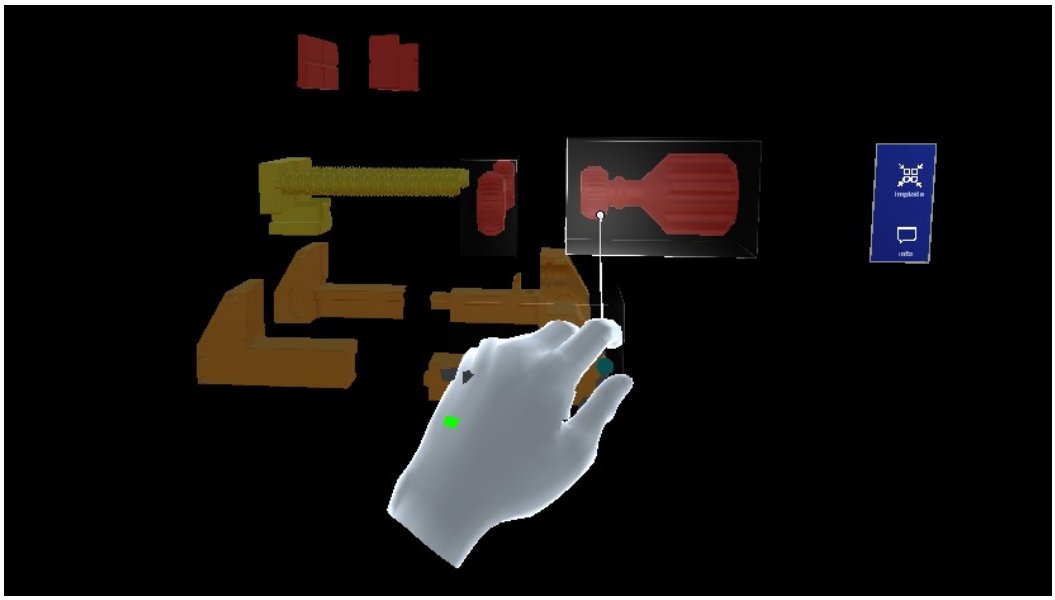


Fig. 3.7 *Interfaccia visualizzata durante la gesture mode;*
sullo sfondo è possibile osservare il modello esploso dell'oggetto;

L’interfaccia utente (UI) di gesture mode per ogni componente è visualizzabile nel momento in cui la distanza tra questi e l’oggetto assemblato sia maggiore di un certo valore.

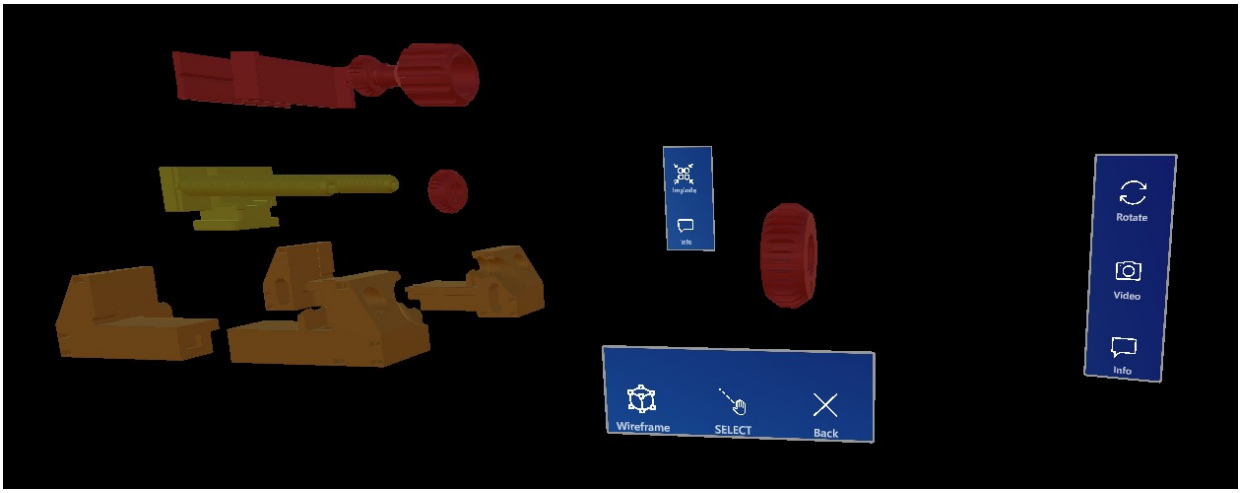


Fig. 3.8 *UI specifica di uno dei componenti dell’oggetto;*

Le interazioni specifiche tramite tale interfaccia sono le seguenti:

- *Rotate*, per la quale l’oggetto ruota automaticamente su sé stesso di 90° intorno al suo asse verticale locale;

- *Video*, per cui viene visualizzato un video informativo per cui viene mostrato come dovrebbe essere montato il componente;

- *Info*, che genera un pannello contente informazioni riguardo al componente (nome, dimensioni, peso etc.);

- *Wireframe*, che scambia il materiale di default del componente con uno di wireframe (e viceversa);

- ***SELECT,*** la meccanica principale che innesca la dinamica di *pick and place* e manda al terminale di controllo l’indice del componente selezionato; in questo modo, il robot è in grado di capire quale componente portare sul tavolo;

- *Back*, per cui l’oggetto ritorna nella posizione originaria del modello esploso;

Modalità Neuro

In questa modalità, l'oggetto è interagibile solamente tramite la focalizzazione verso le interfacce di stimolazione e non è possibile manipolarlo liberamente.

Per rendere più chiara la trattazione seguente, è bene specificare la terminologia che verrà utilizzata per indicare le interfacce di stimolazione di NextMind, la tecnologia BCI utilizzata.

Con il termine **NeuroTag** s'intende il componente associato a una determinata interfaccia utente, la quale comprende un materiale di *flickering* per la stimolazione; tale componente permette la focalizzazione dell'utilizzatore verso l'interfaccia associata e l'innesco di una certa azione programmata precedentemente.

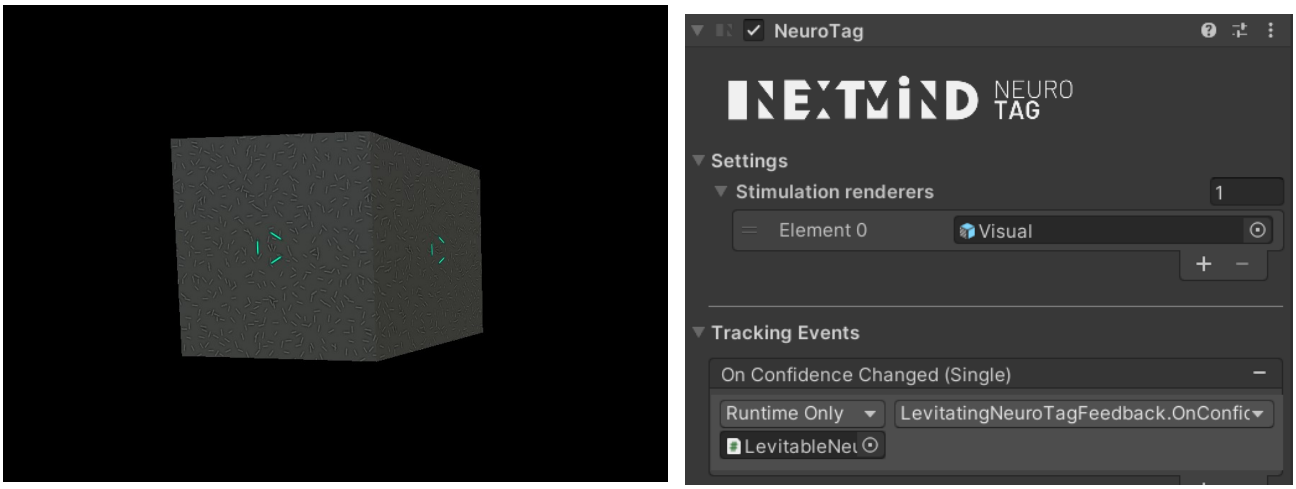


Fig. 3.9 Esempio di c*oppia componente NeuroTag (destra) / mesh associata (sinistra), alla quale è stato assegnato il materiale di stimolazione (di flickering);*

La maggior parte dei NeuroTag sono concentrati nell'interfaccia di neuro mode, che per ogni componente è accessibile solo dopo averlo portato la sezione associata in primo piano.
Con il termine **Sezione** si intende un gruppo di componenti del modello esploso; la sezione è definita sistematicamente a seconda dell'animazione di esplosione predefinita per l'oggetto.

Le fasi attraverso cui passa l'utente durante le interazioni sono le seguenti:

- *Selezione di una specifica **Sezione** dell'oggetto;* una volta che il NeuroTag associato alla sezione è innescato, tutta la sezione viene portata in primo piano e la UI delle componenti è resa visibile;

- Possibilità di interagire con la UI in tre modalità diverse:

    - ***SELECT rapido*** di un componente, per cui viene azionata la dinamica di *pick and place* da parte del robot sul tavolo dell'utente; nel momento in cui il *SELECT* è confermato, il componente ritorna nella posizione originaria del modello esploso;

- ***Explore,*** ossia visualizzare le *feature*, tramite l'innesco del NeuroTag, per cui è possibile accedere al menù completo del componente; quest'ultimo comprende tutte le funzioni già presenti nell'interfaccia in *gesture mode* (info, wireframe etc.), compreso il ***SELECT****;* la differenza sostanziale è che gli inneschi sono prodotti dalla focalizzazione dell'utente sui NeuroTag e non tramite le *gesture*;

- ***Back*** della sezione, riportando tutte le componenti nella loro posizione iniziale nel modello esploso e disattivando la UI;

Fig. 3.10 *Interfaccia presentata durante la neuro mode; si possono individuare le tre sezioni (azzurra, verde e arancione) contenenti le componenti e l'interfaccia di stimolazione associata;*

Gestione dei SELECT

Trattandosi dell'operazione più importante, vale a dire quella di innesco del *pick and place* di un componente da parte del robot, è bene specificare alcune precisazioni per quanto riguarda la meccanica di SELECT:

- Per innescare il NeuroTag di SELECT o di SELECT rapido, l'utente deve rimanere concentrato sull'interfaccia di stimolazione per 2 secondi; allo scadere dei due secondi, la UI del componente si disattiva e questi ritorna nella sua posizione iniziale nel modello esploso;

- L'utente NON può attuare il SELECT due volte sullo stesso componente;

- L'utente NON può attuare un SELECT mentre il robot è in movimento; questo per evitare il verificarsi di errori nella ricezione delle richieste e per mantenere l'attenzione dell'utente sul braccio robotico;

- L'utente NON può attuare un SELECT se il client non è connesso al router della rete locale;

In tutti gli ultimi tre casi, l'interfaccia utente provvede a notificare l'impossibilità dell'azione tramite la visualizzazione di determinati pannelli di avviso.

Nel prossimo sottocapitolo, verranno specificati tutti gli elementi di *user interface* che vanno a comporre l'intera applicazione e tutti le fasi di interazione dell'utente.

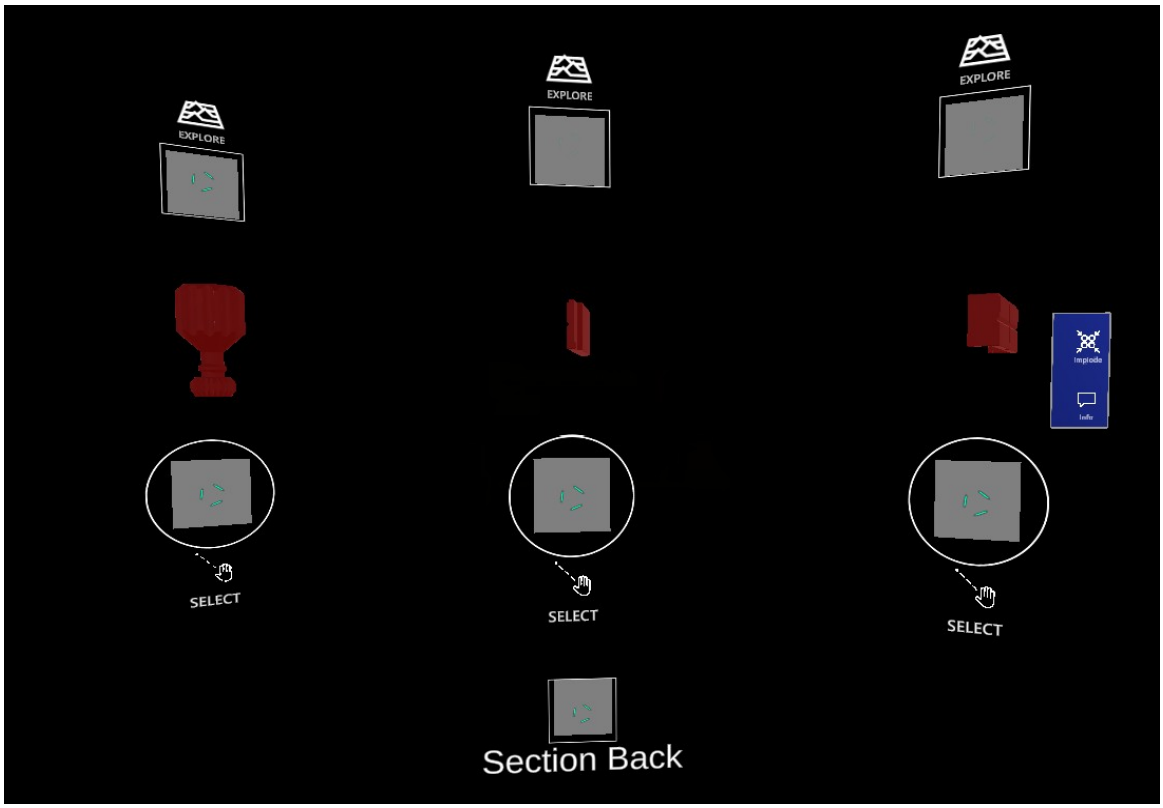


Fig. 3.11 Rappresentazione di una Sezione in primo piano; nella parte alta di ogni componente è presente il NeuroTag "Explore", mentre in quella bassa vi è quello di SELECT rapido; il NeuroTag di "Section Back" è chiaramente visibile in basso al centro;

3.3 Interfaccia Utente (UI)

3.3.1 Introduzione

In questo sotto-capitolo si vuole andare a illustrare tutti gli elementi di UI (*user interface*) che sono stati utilizzati per lo sviluppo dell'applicazione, includendo i modelli tridimensionali dell'oggetto usato nella sperimentazione.

Gli asset specifici da cui sono stati presi tali elementi sono elencati di seguito:

- **Mixed Reality Toolkit (MRTK)**; l'asset di Microsoft per lo sviluppo di applicazioni di realtà mista offre un notevole assortimento di bottoni (di tipo *button* e di tipo *toggle*), menu, pannelli e checkbox; in tutti questi elementi sono già integrate le dinamiche caratteristiche utilizzate con Hololens (manipolazione, visuale radiale, etc.) e sono totalmente personalizzabili per i più svariati utilizzi; la maggior parte dei bottoni e dei menu dell'applicazione sono stati costruiti a partire da quelli offerti da questo asset;

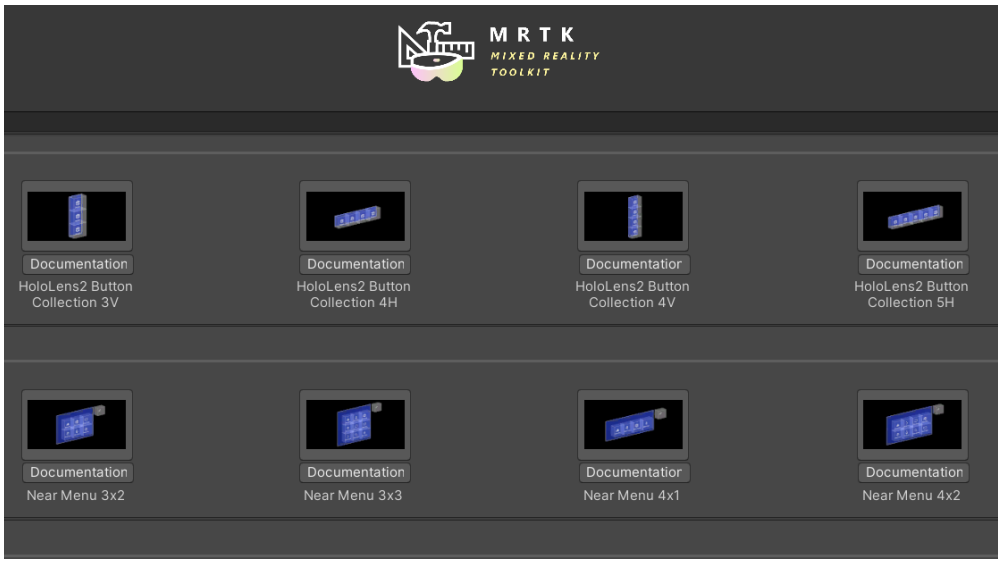


Fig. 3.12. *Toolbox all'interno di Unity con gli elementi offerti da MRTK*

- **NextMind SDK**; comprende varie scene di esempio o predefinite, come quella della calibrazione, e una buona quantità di prefab che utilizzano i NeuroTag; Alcuni degli elementi integrati nell'applicazione sono riportati di seguito:
    - Materiali di stimolazione, con texture di *flickering* per il funzionamento della BCI;
    - Pannelli per la calibrazione, che sono stati personalizzati successivamente;
    - Feedback triangolari proposti dall'asset, innescati quando viene azionato il NeuroTag associato;

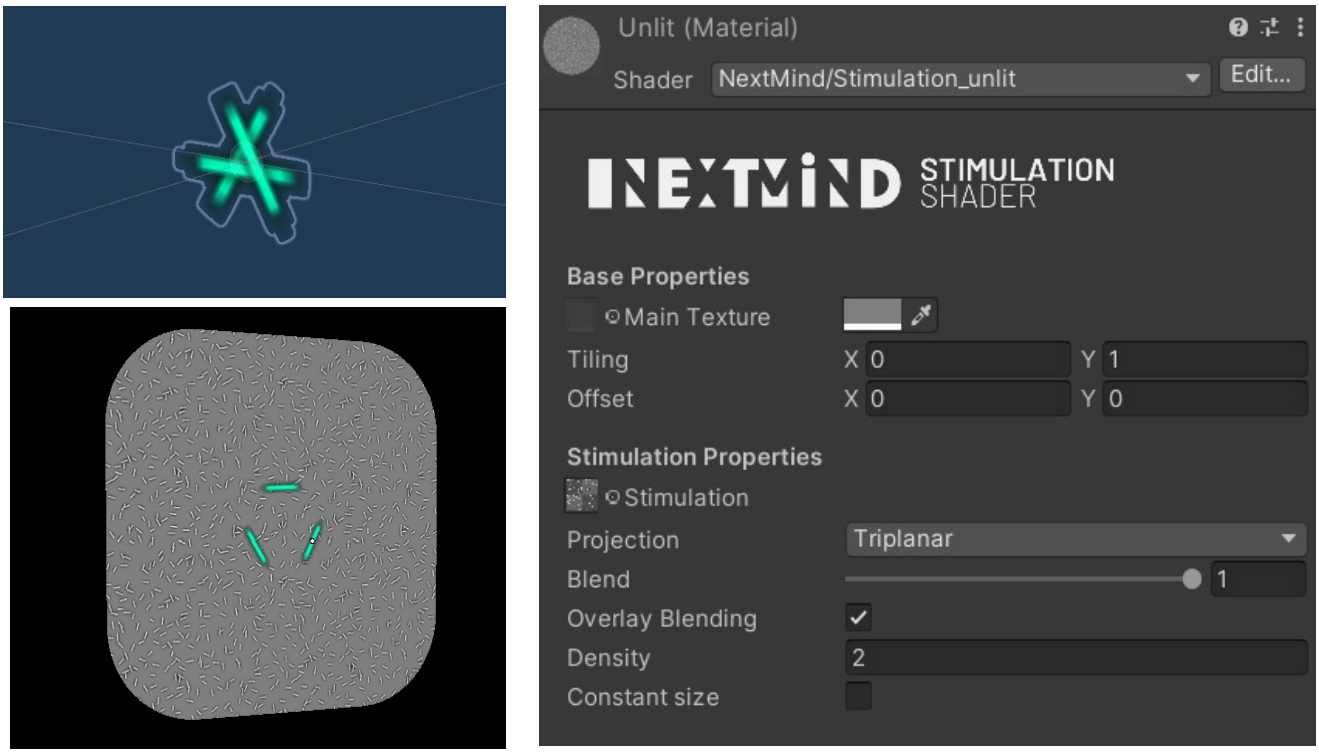


Fig. 3.13. *(Destra) Shader di stimolazione di NextMind; (in alto a sinistra) feedback triangolare, azionato quando viene attivato il neurotag associato; (in basso a sinistra) esempio di combinazione del materiale e del triangolo su una mesh;*

- **Yet ANOTHER Machine Vise**; sono stati scaricati online i CAD dell'oggetto che è stato utilizzato durante la progettazione e nelle fasi di testing [41]; i modelli tridimensionali sono stati prima importati sul software di modellazione Blender, per la conversione del formato, e successivamente importati su Unity; l'assemblaggio delle componenti e l'animazione di esplosione sono stati completamente realizzati sul software;

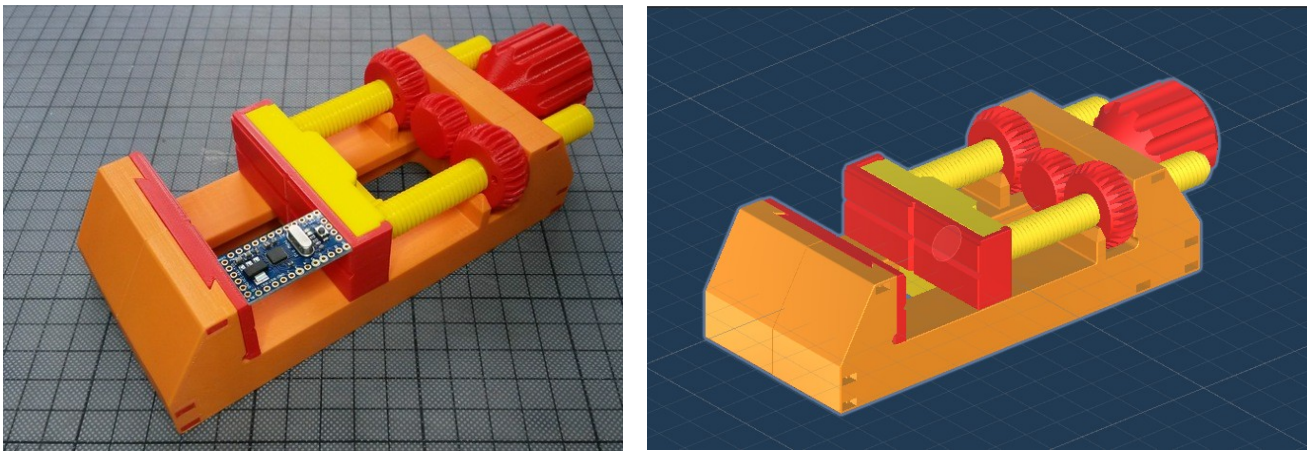


Fig. 3.14 – 3.15. *Modello completo reale (a sinistra); modello 3D su Unity (a destra);*

Nei paragrafi successivi verranno illustrati tutto gli elementi UI utilizzati durante la fase di assemblaggio, in quanto rappresenta la sezione fondamentale del sistema e comprende la maggior parte delle interfacce utente di tutta l'applicazione.

3.3.2 UI Generale

La UI generale comprende tutti gli elementi generici che l'utente incontra durante la sessione di assemblaggio (pannelli, menu, bottoni etc.); in particolare, si tratta di quelle entità che sono sempre visualizzabili dall'utente e disponibili, oppure sono molto comuni o appaiono in situazione specifiche.

Tali entità sono elencate di seguito:

- **Hand Menu**; si tratta di un menù offerto da MRTK e personalizzato per i propositi dell'applicazione; è visualizzabile di fianco alla mano dell'utente quando questi alza il palmo verso l'alto; questo menu contiene i seguenti elementi:
    - Due bottoni MRTK di tipo *toggle*, "Gesture mode" e "Neuro Mode" per passare da una modalità operativa all'altra; se uno dei due bottoni viene selezionato, l'altro si disattiva;
    - Un bottone MRTK di tipo *button*, "Define Table" per la definizione del tavolo;
    - Un bottone MRTK di tipo *button*, "Calibrate" per aggiungere alla scena attiva la schermata di calibrazione;
    - Due bottoni MRTK di tipo *button*, "Exit Assembling" e "Main Menu", che reindirizzano rispettivamente l'utente alla fase di selezione del progetto e alla fase principale.

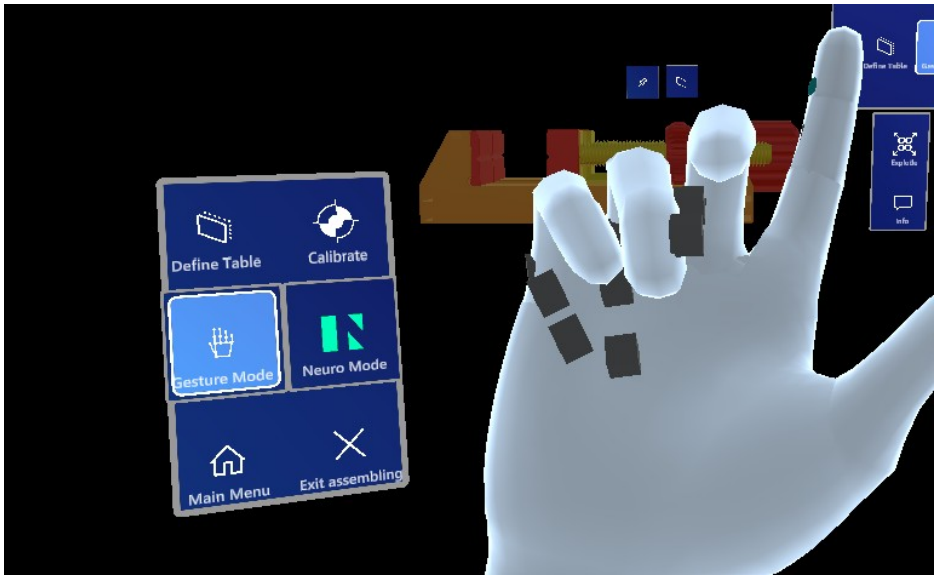


Fig. 3.16. *Interfaccia dell'Hand Menu vista dall'emulatore per desktop;*

- **Menu superiore**; anche in questo caso, si tratta di un menu MRTK personalizzato successivamente; uno script di visuale radiale permette al menu di seguire il movimento della testa dell'utente tramite *gaze tracking*; gli elementi contenuti in questo menù sono pressochè gli stessi dell'hand menu, a meno dei bottoni "Calibrate" e "Main Menu".

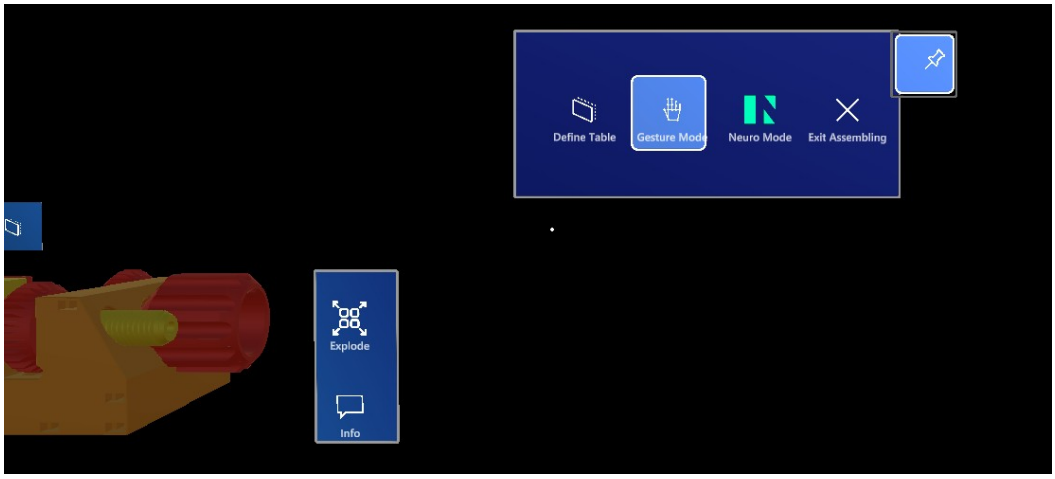


***Fig*. 3.17.** *Interfaccia del menu superiore visualizzata dall'emulatore per desktop;*

- **Connected Device Status**; è un prefab offerto dalla SDK di NextMind, al quale è stato aggiunto un elemento di visuale radiale perché seguisse il movimento della camera attiva; la sua funzione è quella di notificare all'utente lo stato di connessione, la percentuale di batteria rimasta e il nome del dispositivo NextMind utilizzato; al fine di non recare fastidio all'utente, è stato posto nella parte bassa dell'interfaccia;

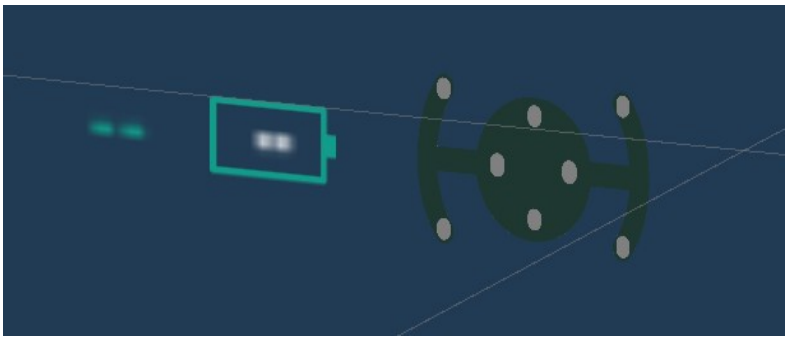


Fig. 3.18 *Pannello dello stato corrente del device;*

- **Warning panel;** i pannelli di warning sono costruiti a partire da un basico pannello con sfondo blu e utilizzano anch'essi la visuale radiale; i casi in cui vengono visualizzati i pannelli di warning sono i seguenti:

    - L'utente cerca di selezionare un componente quando il robot è in movimento (*Selection Unavailable)*;

- L’utente cerca di selezionare un componente due volte (*Component Already Selected);*
- L’utente cerca di selezionare un componente, ma il client non è connesso o la connessione con la rete locale è stata interrotta (*Client not Connected);*
- L’utente cerca di entrare nella fase di assemblaggio, cliccando sul bottone di *Start Assembling*, ma non ha ancora definito un tavolo da lavoro (*Assembling not ready).*

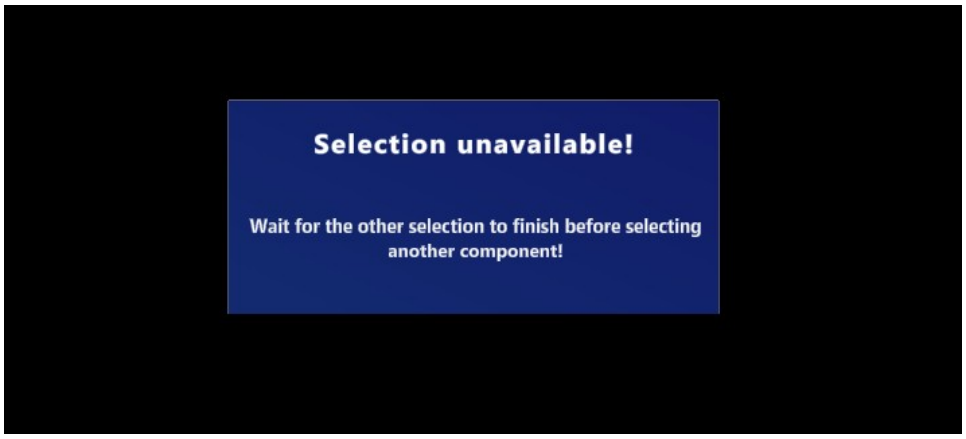


Fig. 3.19 *Esempio di Warning Panel; si tratta del caso in cui il robot si sta muovendo e l’utente ha cercato di selezionare un componente;*

- **“Robot Is Moving” Panel** (Fig. 3.20)**;** questo pannello si attiva tutte le volte che il braccio robotico sta compiendo un movimento e si disattiva quando questi è di nuovo in stato di riposo; incorpora una componente di visuale radiale, in modo tale che sia subito visibile all’utente;

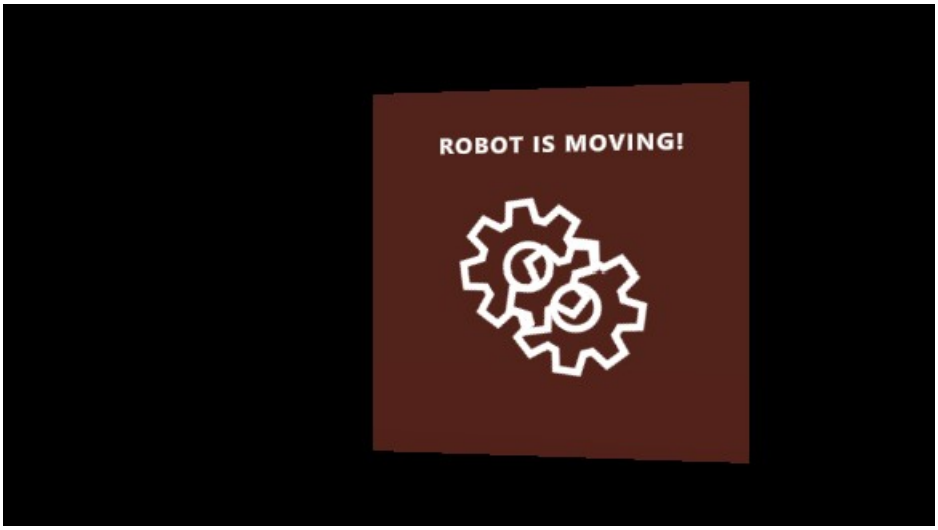


Fig. 3.20

3.3.3 UI Tavolo

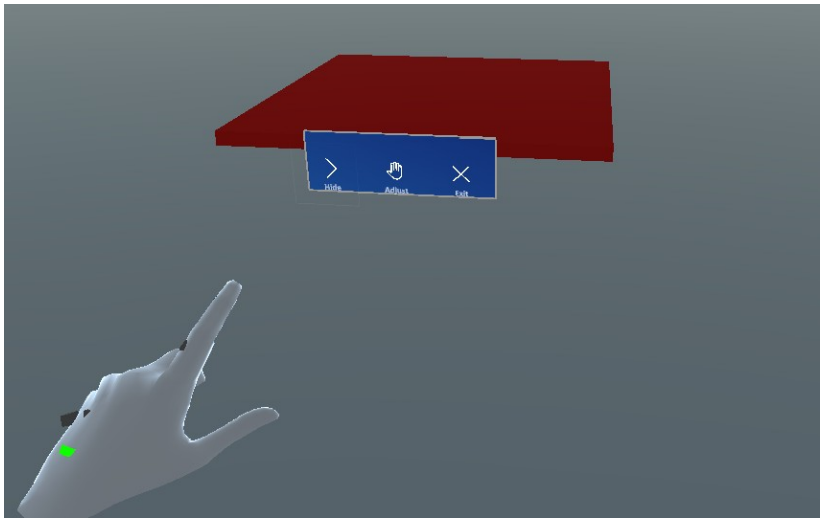
La UI per la definizione del tavolo da lavoro comprende un set di bottoni, che seguono la mesh e si adattano sui bordi a seconda della posizione dell'utente, e un materiale che cambia colore a seconda della situazione.

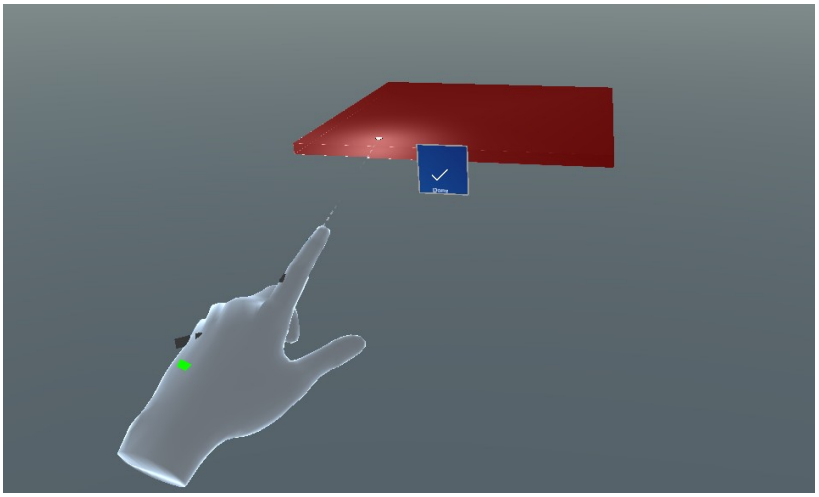
In particolare, sono identificati di seguito i cambiamenti della UI in base alla fase di definizione del tavolo corrispondente:

- L'utente, tramite il bottone *Adjust*, ha la possibilità di traslare la mesh tramite gesture nello spazio reale; a questo punto vi è solo più il bottone *Done* di conferma attivo;

- Durante la traslazione, tramite un *groundcheck* posto alla base della mesh, il materiale diventa verde se viene rilevata una superficie (il layer della superficie deve essere quello di "Spatial Awareness"); se non vi è nessun contatto, la mesh rimane rossa;

- A questo punto, l'utente può confermare la definizione del tavolo e selezionare il pulsante di exit; la definizione del tavolo è terminata.





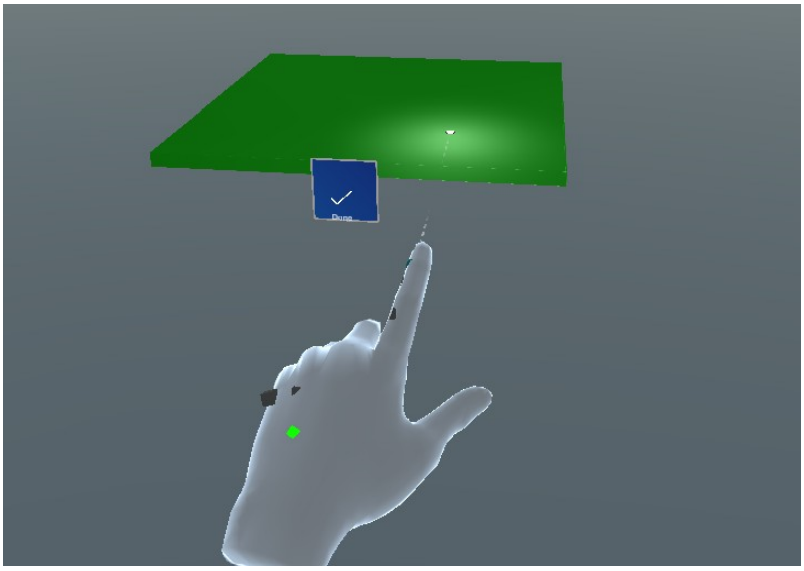


Fig. 3.20-21-22 *Immagini della UI della mesh durante la sequenza di definizione del tavolo*[4]*;*

Ad esclusione della definizione del tavolo, l'unico momento in cui la mesh è visualizzabile è quando viene fatto ritornare l'oggetto assemblato su di essa (tramite un bottone della UI propria dell'oggetto assemblato). In questo caso, il materiale diventa visibile per un momento, con un'animazione di *Fade In*, e poi scompare di nuovo, con un'animazione di *Fade Out.*

3.3.4 UI Oggetto assemblato

La UI dell'oggetto assemblato comprende due gruppi distinti, che si attivano rispettivamente quando l'utente è in fase di selezione del progetto e di assemblaggio:

- ***Gruppo di Selezione***; si tratta della UI che comprende i bottoni di *Start Assembling* e I*nfo*, oltre al titolo che rappresenta il nome dell'oggetto;

- ***Gruppo di Assemblaggio***; comprende il menu laterale, con i bottoni di *Explode/Implode* e di *Info*, e due bottoni sopra l'oggetto, *Anchor* e *Return*; questi ultimi sono utili rispettivamente per ancorare allo spazio l'oggetto (Bottone di tipo *Toggle* per l'attivazione/disattivazione della manipolazione) e per farlo ritornare sul tavolo (Bottone di tipo *Button*).

In entrambi i casi, la UI comprende anche il pannello informativo dell'oggetto assemblato, reso visibile in funzione dell'interazione dell'utente con il bottone di *Info*.

Inoltre, l'oggetto incorpora un Box Collider che viene utilizzato principalmente per il Bounds Control, ossia il controllo dei limiti dell'oggetto tramite l'omonimo componente; in questo modo, sono anche visualizzabili i bordi della Bounding Box in cui l'oggetto è rinchiuso.

4 Immagini tratte dall'emulatore, con la simulazione del contatto con una superficie di "Spatial Awareness"

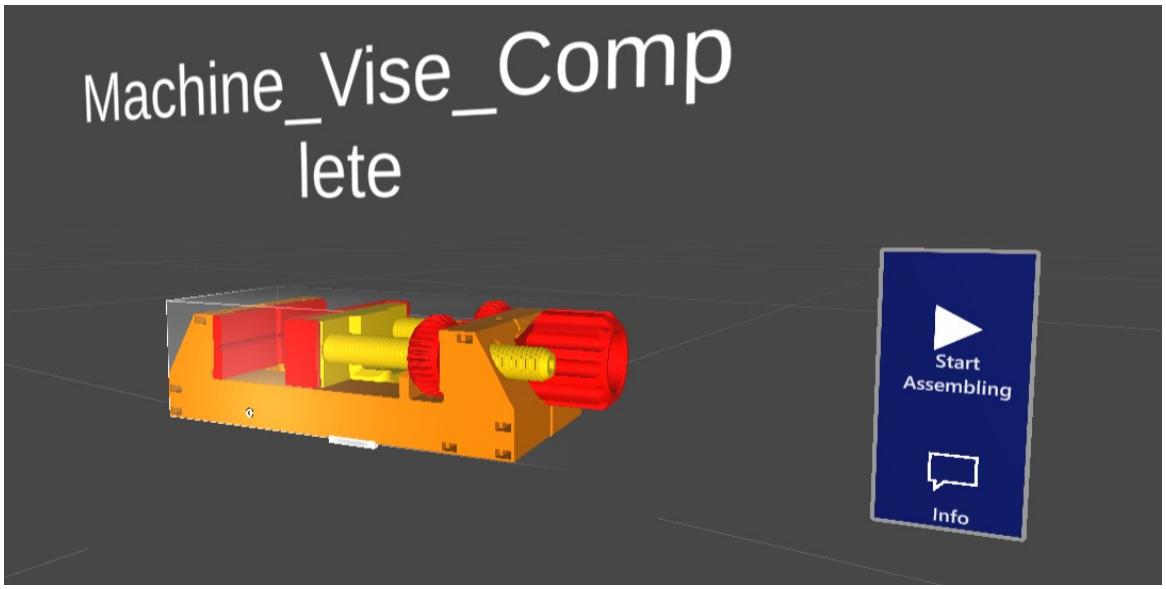


Fig. 3.23 *Gruppo di Selezione;*

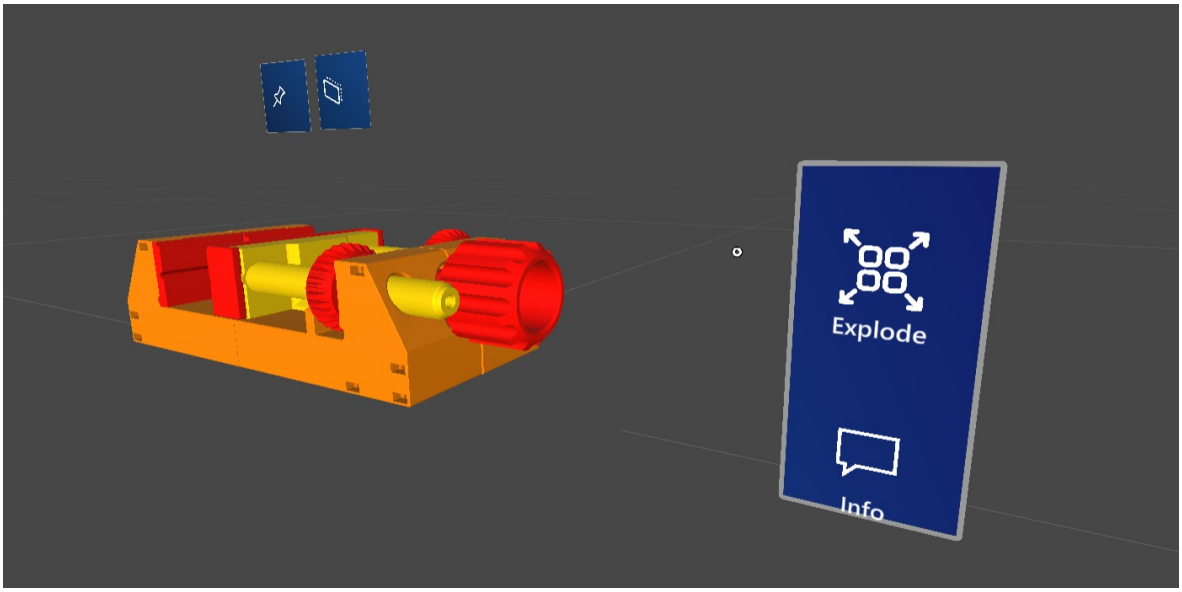


Fig. 3.24. *Gruppo di assemblaggio;*

3.3.5 UI Componenti – Modalità “Gesture”

I componenti in modalità *gesture* presentano una UI composta da due barre di bottoni, dove entrambe si adattano in posizioni specifiche della *Bounding Box* del componente:

- ***Vertical App Bar***; si posiziona alla destra del componente, con un *offset* di 0.2 rispetto al limite destro della *Bounding Box*; include i bottoni di *Rotate, Video* e *Info* che incorporano le rispettive funzionalità; in particolare, le ultime due rendono possibile la visualizzazione del pannello e del video informativi;

- ***Horizontal App Bar***; si posiziona davanti al componente, con un *offset* di 0.1 rispetto al limite frontale della *Bounding Box;* include i bottoni di *SELECT, Wireframe* e *Back* che incorporano le rispettive funzionalità;

Si specifica sempre che entrambe si adattano in base alla posizione della testa, dunque si spostano lungo i bordi della *Bounding Box* del componente in maniera tale da essere sempre rivolti verso l’utente.

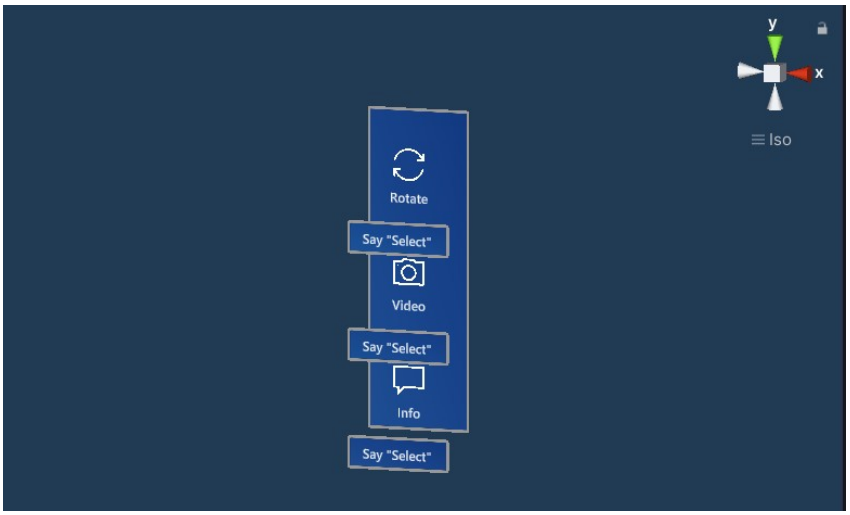


Fig. 3.25. *VerticalAppBar;*

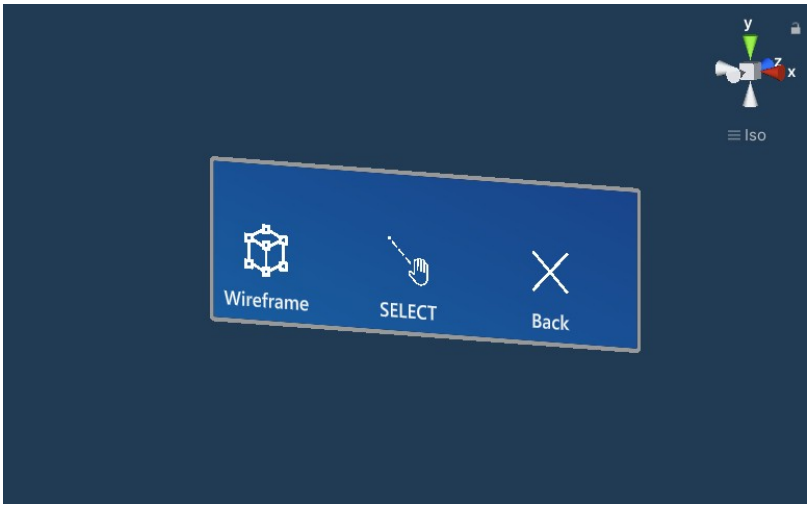


Fig. 3.26 *HorizontalAppBar;*

3.3.6 UI *Sezioni* – Modalità “Neuro”

Una S*ezione* del modello esploso è identificata totalmente da una *Bounding Box* invisibile e da un NeuroTag associato. In particolare, quest’ultimo è definito come **NeuroPad**, una mesh a sezione rettangolare, con i bordi arrotondati, che incorpora un materiale di stimolazione e viene utilizzato per la selezione delle *Sezioni* del modello esploso; per ottimizzare la focalizzazione, si è deciso di includere nello script un adattamento della scala, rendendo l’interfaccia più grossa o più piccola in base alla distanza dell’utente da tale oggetto.
Quando la focalizzazione dell’utente è rilevata, oltre al feedback triangolare, viene visualizzato anche un bagliore verde intorno all’oggetto; dopo un secondo, l’evento associato al NeuroTag viene invocato e la *Sezione* viene portata in primo piano, mentre tutti i NeuroPad scompaiono.

Si specifica che questo elemento non richiede una tempo di focalizzazione dell’utente: l’interfaccia fornisce un feedback con tempismo automatico, ma l’innesco è comunque istantaneo. Nel resto delle interfacce di tipo *neuro*, invece, è richiesto un certo tempo di focalizzazione da parte dell’utente.

Fig. 3.27. *Rappresentazione del NeuroPad e della sua interfaccia di stimolazione; si nota al centro anche il feedback triangolare proprio di NextMind SDK;*

L’altro elemento UI della *Sezione* è il bottone di *Section Back*, che viene visualizzato soltanto nel momento in cui questa viene portata in primo piano; tale interfaccia è sempre interagibile tramite la focalizzazione, in quanto include un NeuroTag e una materiale di stimolazione.

Il tempo di focalizzazione richiesto all’utente per questa interfaccia è di due secondi, durante i quali un quadrato verde, con scala iniziale a zero, aumenta di grandezza fino raggiungere il quadrato bianco più esterno. Se l’utente perde la focalizzazione, il processo si interrompe e il contorno quadrato verde torna ad avere la scala uguale a 0; anche in questo caso, viene fornito un ulteriore feedback tramite un bagliore progressivamente sempre più visibile.

3.3.7 UI Componenti – Modalità “Neuro”

In modalità *Neuro*, l’interfaccia dei componenti si attiva sistematicamente quando la *Sezione* di cui fanno parte viene portata in primo piano; tali componenti rimangono visibili all’utente, mentre tutti gli altri vengono nascosti per evitare confusione.

Gli elementi della UI di tipo *Neuro* hanno tutti configurazioni simili, con qualche variazione per quanto riguarda la forma e la grandezza per quelli responsabili del SELECT. Ad ogni modo, ogni entità presenta le seguenti caratteristiche:

- Un quadrato riempito da un materiale di stimolazione (di *flickering*) e dal feedback triangolare di NextMind SDK;

- Un bagliore verde la cui componente alpha aumenta in modo direttamente proporzionale alla focalizzazione dell’utente sul NeuroTag;

- Un ulteriore feedback composto da due contorni, uno interno (verde) e uno esterno (bianco), di forma circolare (per il SELECT) o quadrata (per i restanti elementi); il contorno interno aumenta di grandezza in maniera direttamente proporzionale alla focalizzazione dell’utente, partendo da scala 0, fino a raggiungere la scala del contorno più esterno.

I gruppi UI di cui fanno parte gli elementi dell’interfaccia sono elencati di seguito:

- **Quick Menu**; si tratta dell’interfaccia preliminare presentata all’utente quando la *Sezione* è portata in primo piano e presenta due pannelli di stimolazione. In particolare, questi ultimi prendono il nome di ***MiniPad*** e rappresentano una variante dei ***NeuroPad*** illustrati nella sotto-sezione 3.3.6:

    - ***Quick SELECT***; si tratta del *MiniPad* che aziona le dinamiche di *pick and place* del componente e si posiziona sotto al componente. Rappresenta un metodo veloce per selezionare le componenti con il minor numero di passaggi;

    - ***Explore;*** si tratta del *MiniPad* che porta in primo piano il componente corrispondente, facendo tornare al modello esploso tutti gli altri della sezione e rendendoli non più visibili. Inoltre, attiva l’interfaccia di tutte le feature disponibili, che rappresenta l’altro gruppo UI;

- **Explore Menu;** il gruppo di elementi che rappresenta tutte le opzioni disponibili per il componente, che corrispondono alle stesse presentate per la UI in modalità *gesture.* Si tratta di un’interfaccia composta da 6 *Minipad* organizzati su due piani orizzontali, uno superiore e l’altro inferiore al componente; tra questi, c’è ovviamente anche il ***SELECT;***

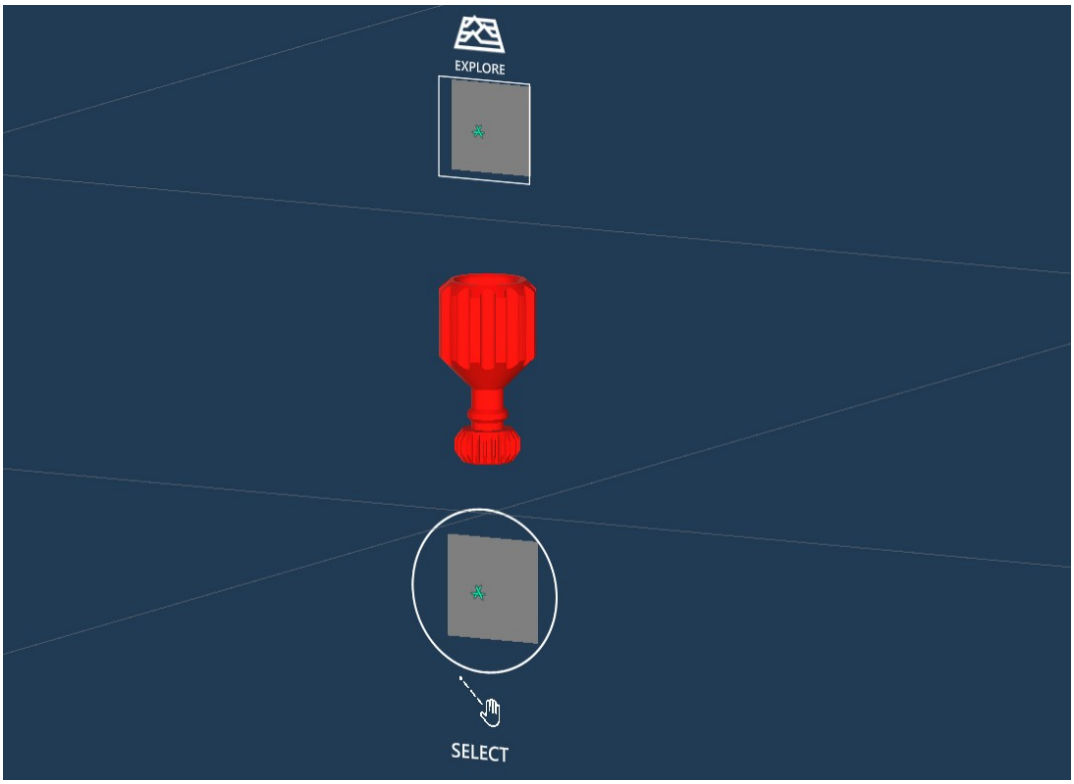


Fig. 3.28. *Quick Menu;*

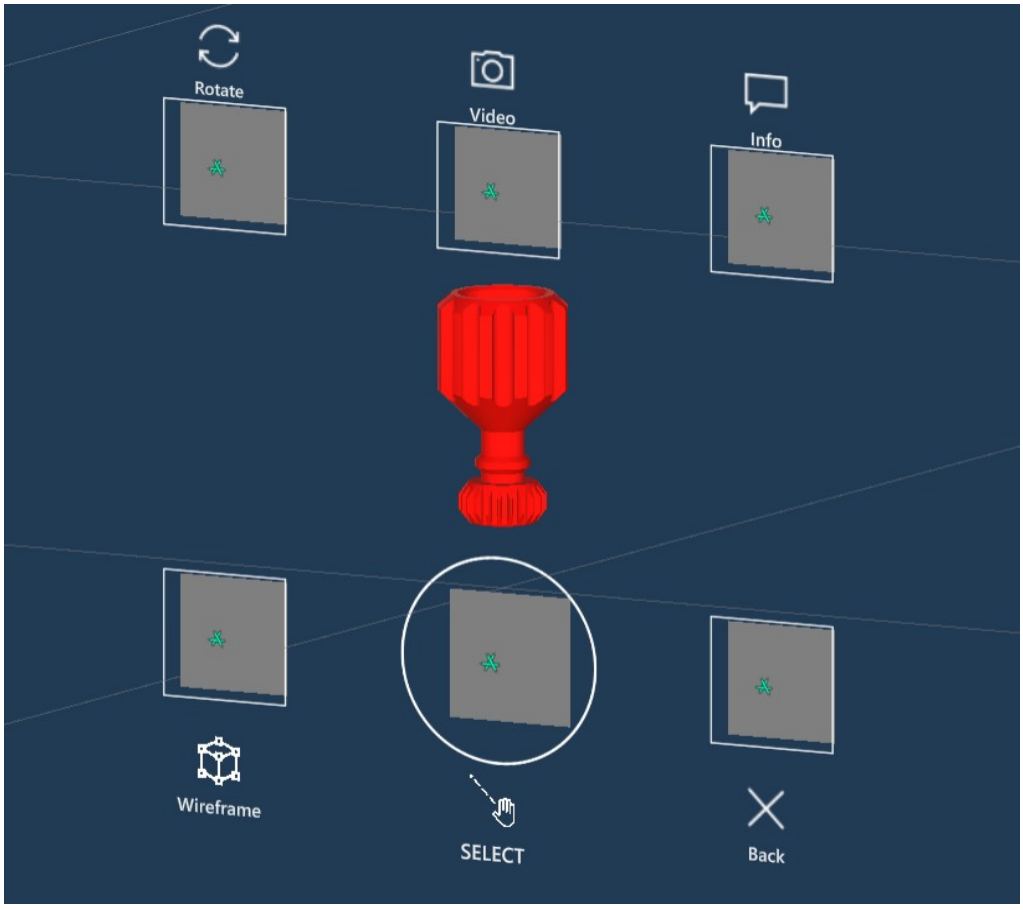


Fig. 3.29 *Explore menu;*

Fig. 3.30. *Sequenza di esempio dei momenti di allargamento del cerchio verde per il NeuroTag di SELECT;*

3.4 Architettura del codice

3.4.1 Introduzione

In questo sotto-capitolo verrà illustrata l'architettura di codice realizzata per il funzionamento dell'applicazione Unity.

Si specifica che i componenti della maggior parte dei bottoni e dei menu utilizzano script propri del MixedRealityToolkit (MRTK) di Microsoft, mentre i NeuroTag utilizzano script originali del NextMind Engine per il funzionamento della BCI.

In particolare, si identificano diverse entità di gestione appartenenti alle due SDK che permettono il funzionamento del sistema:

- ***MixedRealityToolkit (Script)***; l'oggetto associato a questo script è create nel momento in cui viene il setup della scena; permette di gestire le impostazioni dell'esperienza AR, gli input, le dinamiche di manipolazione, la consapevolezza spaziale e le impostazioni relative alla camera; questo componente è personalizzabile andando a creare nuovi profili per ogni sezione di impostazioni;

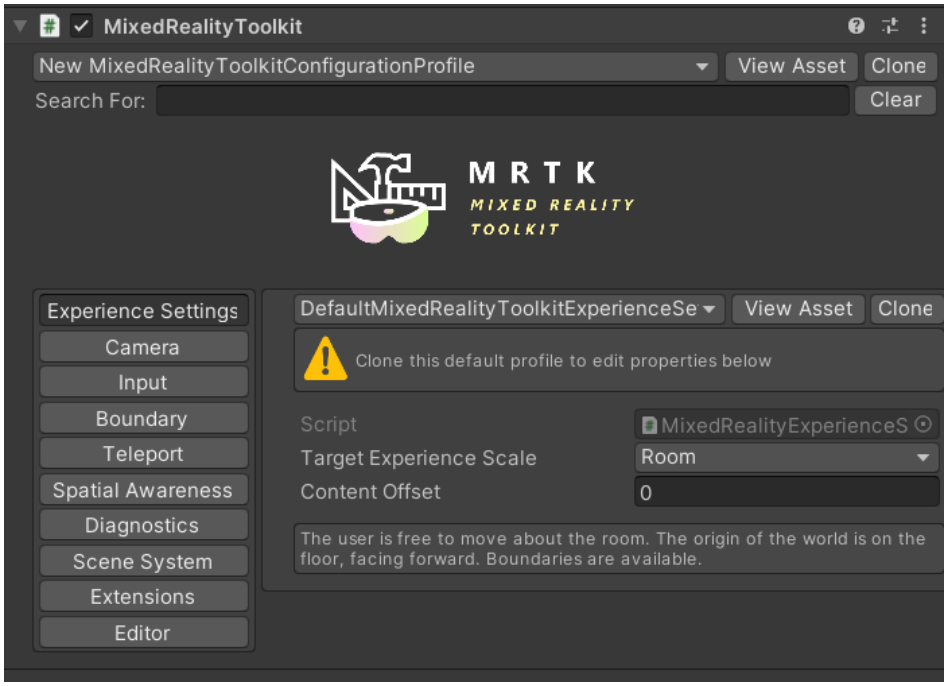


Fig. 3.31

- ***NeuroManager;*** gestisce la comunicazione tra ogni NeuroTag nella scena e il *NextMind Engine*; inoltre, consente allo sviluppatore di effettuare debug anche senza il dispositivo fisico, simulando la focalizzazione sui NeuroTag e azionando il *flickering* delle interfacce di stimolazione;

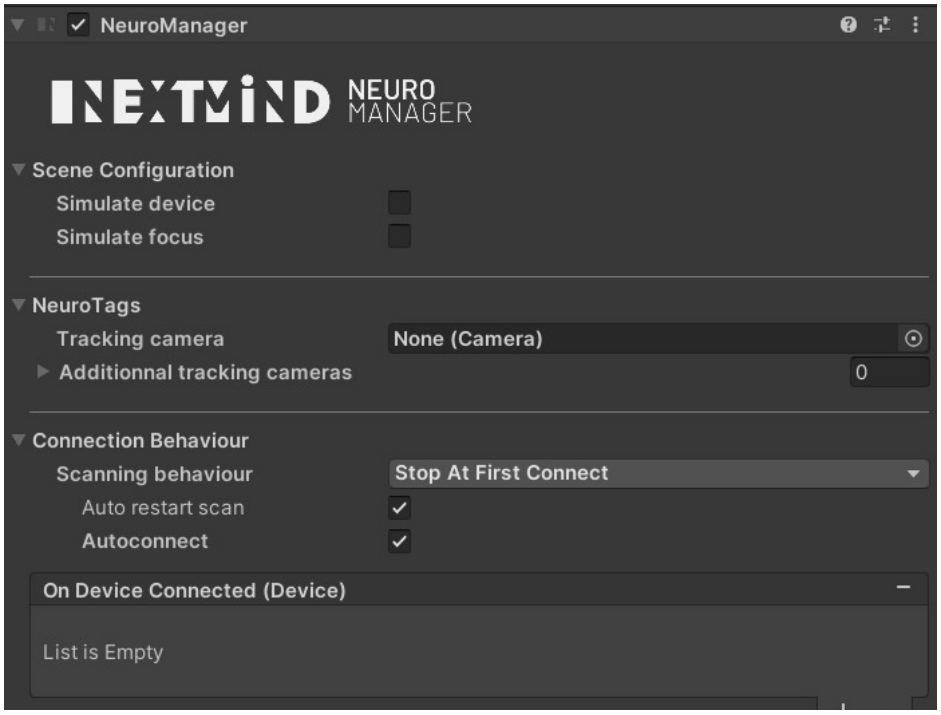


Fig. 3.32

- ***Calibration Manager***; fa parte del NextMind Engine per la gestione della calibrazione del dipositivo e viene utilizzato nella *Calibration Scene*; l'intero processo può essere descritto in maniera semplificata nel seguente modo:

  - Viene invocato il metodo *StartCalibration*;

  - Vengono inizializzati i NeuroTag preposti alla calibrazione con *OnInitialize(NeuroTag)*;

  - Avviene l'iterazione del seguente processo per 12 volte, ognuna con un NeuroTag diverso:

    - *OnStartCalibrating(NeuroTag)*;
    - 3 secondi di attesa;
    - *OnEndCalibratin(NeuroTag)*;

  - La calibrazione termina e i risultati vengono inviati a un'altra entità per la loro gestione.

- ***Steps Manager***; script della NextMind SDK utilizzato durante la fase di calibrazione per la gestione dei diversi pannelli, i quali incorporano tutti uno script del tipo astratto *Step*;

Inoltre, vi sono degli script ricorrenti dell'asset MRTK che vengono integrati all'interno dell'oggetto assemblato e delle componenti singole; principalmente, si tratta di elementi preposti alle dinamiche di manipolazione:

- ***BoundsControl***; riceve un collider di override, o ne crea uno nel caso non sia presente, e definisce dei bordi illuminati sui contorni di tale *Collider*; la visualizzazione può assumere stili diversi, a seconda delle impostazioni dello sviluppatore, e può essere azionata tramite prossimità, tramite puntatore o essere sempre attiva;

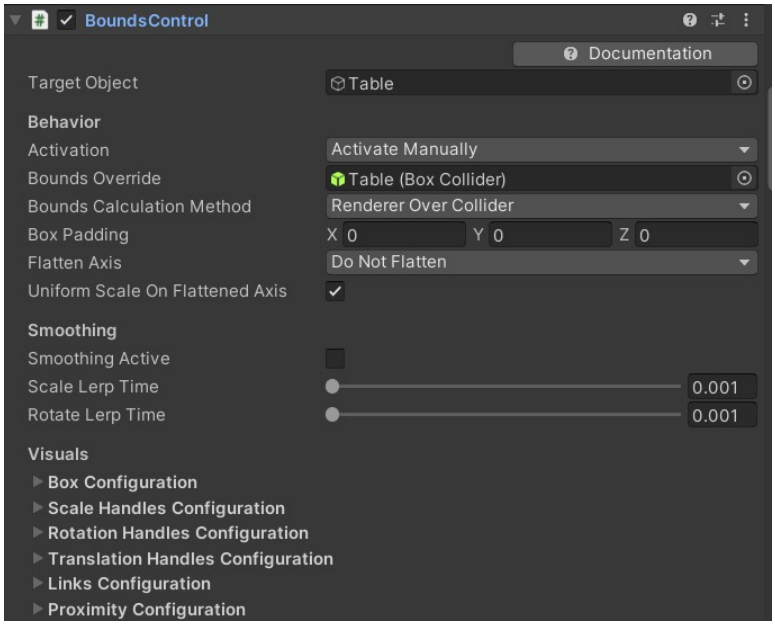


Fig. 3.33

- ***ObjectManipulator***; riceve un oggetto di tipo *Transform*, il quale rappresenta l'oggetto specifico da manipolare; le varie impostazioni permettono di determinare la tipologia di manipolazione (una mano o due mani) e quali azioni sono disponibili (rotazione, traslazione e scalamento).

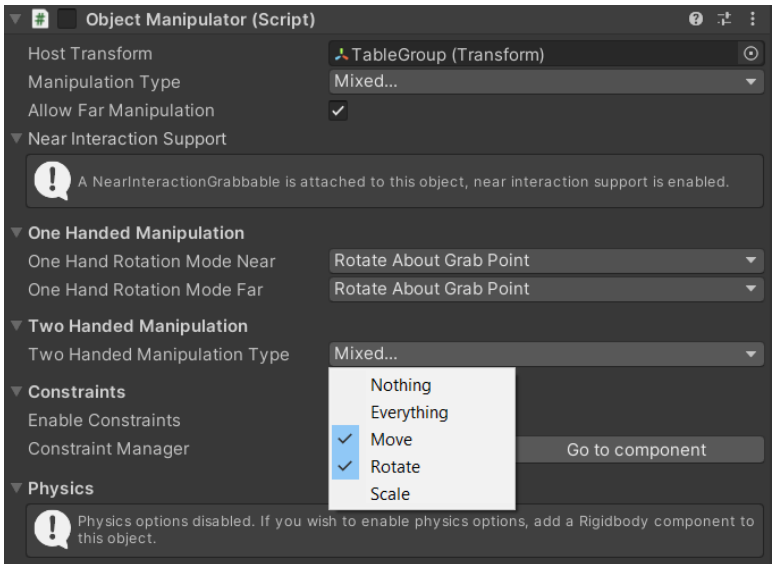


Fig. 3.34

Infine vi sono i **NeuroTag**, appartenenti alla SDK di NextMind, che permette a un qualunque *Unity GameObject* di essere interagibile tramite la BCI per innescare determinate azioni.
Ognuno di questi elementi possiede un'interfaccia visuale di stimolazione che genera una risposta nel cervello dell'utente (tramite il paradigma SSVEP), la quale viene rilevata e decodificata dal NextMind Engine per misurare l'attenzione dell'utilizzatore.

Lo script contiene diversi eventi, i quali diventano attivi nel momento in cui il NeuroTag diventa attivo a sua volta nella scena; tra questi, vi sono:

- *OnTriggered*; evento invocato una volta sola quando viene rilevata l'attenzione dell'utente;
- *OnMaintained;* evento invocato ogni volta che l'*engine* riceve conferma dell'attenzione dell'utente (ogni 250 ms circa);
- *OnReleased*; evento invocato quando l'attenzione dell'utente viene rilasciata e non c'è più la conferma da parte dell'*engine*;
- *OnConfidenceChanged*; evento invocato quando il valore di *confidence* dell'attenzione sul NeuroTag cambia di valore;

Fig. 3.35 *Script NeuroTag visualizzato dall'editor di Unity;*

Con questa premessa è possibile andare ad analizzare il resto dell'architettura del codice originale sviluppato per l'applicazione.

Con questo approccio si è voluto creare una struttura semi-adattiva a diverse tipologie di oggetto da assemblare, nonostante ne sia stato utilizzato solo uno per gli scopi preposti alla ricerca.
In tale paradigma, gli script sono assegnati a un determinato oggetto sistematicamente da specifiche **Entità di gestione**, ognuna preposta all'amministrazione di una determinata categoria (oggetti, UI etc.). Gli script assegnati in questo modo, che definiscono delle **Entità oggetto** o **Entità UI**, provvederanno eventualmente a loro volta ad integrare altre componenti, a creare altre entità o ad aggiungere delle funzionalità specifiche. Inoltre, sono state incluse delle ***interface*** nel codice che fungono da *observer* di un determinata entità, al fine di garantire una corrispondenza uno a molti per alcuni eventi che la richiedano.

La decisione di prendere questa direzione segue due motivazioni principali: comodità nell'adattamento del codice a cambiamenti drastici e proseguimento della sperimentazione per integrazioni future.

Nei paragrafi successivi verranno definite le entità citate, andando a illustrare la struttura generale e specifica del codice C# sviluppato su Visual Studio e integrato su Unity.

3.4.2 Entità di gestione

Le entità di gestione sono fondamentalmente quattro, di cui le prime tre seguono il pattern del *Singleton*. Questo assicura che della loro classe venga creata una e una sola istanza, fornendo alle altre entità un punto di accesso globale a tale oggetto; in questo modo l'accessibilità è garantita in ogni parte del codice.

Project Scene Manager

Si tratta dell'entità di gestione fondamentale per il funzionamento della *Project Scene* e include diversi *observer* di tipo *IProjectSceneObserver* per l'innesco di eventi specifici in oggetti di altre classi.

Si occupa di varie mansioni:

- Istanziamento degli oggetti assemblati nella scena e gestione della loro visualizzazione in base all'interazione dell'utente; il manager riceve direttamente dall'editor di Unity una lista di *prefab* che rappresentano gli oggetti assemblati da creare, attraverso l'impostazione di appositi campi tramite il parametro *SerializeField*;

- Gestione della meccanica di definizione del tavolo tramite i metodi *OnTableDefinitionStarted* e *OnTableDefinitionEnded*;

- Cambiamento della modalità operativa (*gesture* o *neuro*) in base all'interazione dell'utente; quando ciò avviene, viene innescato l'evento *OnOperationalModeChanged* in tutti gli *observer* presenti;

- Inizializzazione del robot quando inizia la fase di assemblaggio tramite un metodo asincrono;

- Ritorno al menù principale o aggiunta della *Calibration Scene* qualora l'interazione dell'utente lo richieda.

Inoltre, include l'*interface* denominata *UIObserver*, che permette l'osservazione dell'entità dell'oggetto assemblato corrente.

UI Manager

Si tratta dell'entità preposta alla gestione della UI generale e include anch'essa l'*interface* dal nome *UIObserver* per l'osservazione dell'oggetto corrente.

I compiti fondamentali sono i seguenti:

- Instanziamento della UI specifica dell'oggetto assemblato e innesco della dinamica di definizione dei *Listeners* di tale *user interface;*

- Fornitura dei *prefab* di UI alle entità che rappresentano i pezzi dell'oggetto assemblato (*es. HorizontalAppBar, NeuroUI* etc.) e alle *Sezioni*; i *prefab* sono assegnati allo script direttamente dall'editor di Unity, ognuno associato al campo corrispondente impostato tramite il parametro *SerializeField*;

- Fornitura dei giusti materiali ai pezzi dell'oggetto assemblato e alle *Sezioni*;

- Gestione della visibilità dei pannelli di *Warning* e del pannello *RobotIsMoving*; nel primo caso, a seconda dell'evento, si occupa anche di assegnare il giusto testo al pannello (*es. Selection Unavailable*);

- Gestione degli elementi *Hand Menu* e *Menu Superiore*;

Robot Handler

Si tratta dell'entità di gestione preposta all'invio dei messaggi al robot *e.DO*.

Integra fondamentalmente tre metodi, che rappresentano le 3 funzionalità specifiche che caratterizzano questa entità:

- ***StartClient***; inizializza la connessione in rete locale aprendo una nuova socket IP e invia i dati per il *Setup* del robot; una volta terminata l'inizializzazione, il client resta in ascolto fino

alla chiusura dell'applicazione e riceve i messaggi che gli vengono inviati; in base a ciò che riceve, viene posta a *true* oppure a *false* una variabile booleana denominata *RobotIsMoving* per la notifica alle altre entità di tale evento;

- ***SelectComponent*** (*string index)*; viene chiamato quando l'evento SELECT è innescato; riceve l'indice del componente come parametro e invia il dato al terminale di controllo;

- ***CloseClient****;* viene chiamato alla chiusura dell'applicazione, nel metodo di override *OnDestroy*; ha il compito di chiudere il client, se connesso.

Components Manager

Si tratta dell'unica entità di gestione che non è un *Singleton*, siccome viene istanziata in relazione all'oggetto assemblato specifico; contiene un importante numero di metodi e controlla sostanzialmente tutti gli elementi che coinvolgono un componente/pezzo dell'oggetto.

Include *l'interface* dal nome *IProjectSceneObserver* per l'osservazione del cambiamento della modalità operativa scelta dall'utente (*gesture o neuro*)

In particolare, le funzionalità fondamentali sono le seguenti:

- Inizializzazione dei componenti, assegnando a ognuno di loro uno script di tipo *ObjectComponent* e impostando l'indice per ognuno di essi (assegnato progressivamente in base all'ordine);

- Gestione generale della visibilità della UI, dell'attivazione dei *Collider* e dei NeuroTag dei componenti; questo include anche la visualizzazione e l'aggiornamento della corretta UI sulle componenti in base alla modalità operativa corrente;

- Impostazione generale del parametro booleano *IsSelectable* all'interno dei componenti, al fine di gestire quando è o non è possibile un SELECT;

- Creazione delle *Sezioni* dell'oggetto quando il modello è esploso, con l'assegnazione delle componenti per ognuna tramite un algoritmo basato sull'altezza locale a cui tali componenti si trovano; in altre parole, se le componenti si trovano in un determinato *range* di altezza, allora fanno parte della stessa sezione; il codice assicura anche che un componente non possa far parte di più sezioni;

3.4.3 Entità oggetto

Le entità oggetto rappresentano elementi fisicamente presenti nella scena, quali l'oggetto assemblato, le componenti, le *Sezioni* e il tavolo da lavoro.

Assembled Object

Si tratta dello script preposto a gestire l'oggetto assemblato in generale e controlla la maggior parte degli script assegnati al *GameObject* di cui fa parte; ciò include il controllo dei limiti (*BoundsControl*) e della manipolazione tramite *gesture* (*ObjectManipulator*).

L'oggetto include tra i parametri la posizione attuale del tavolo da lavoro, così come è stato definito dall'utente; nel momento in cui il tavolo viene ridefinito (traslato in un altra posizione) viene invocato il metodo *OnTablePositionUpdate,* che riceve come parametro la nuova posizione del tavolo e aggiorna conseguentemente la variabile propria della classe.
Nel caso l'utente interagisca con la UI per far ritornare l'oggetto sul tavolo, viene invocato il metodo asicrono *GoBackToTablePosition*; quest'ultimo trasla l'oggetto progressivamente, attraverso un processo di iterazione a tempo e di interpolazione lineare della posizione, fino a quando non raggiunge le stesse coordinate del tavolo da lavoro.

Il metodo più importante è rappresentato da ***OnExplode***, che viene invocato quando l'utente interagisce con il bottone *Explode/Implode*; a questo punto, il metodo può agire in due modi in base al valore della variabile booleana di classe *IsExploded*:

- ***False***; significa che l'oggetto non è esploso, dunque viene innescata l'animazione di esplosione dell'oggetto tramite l'impostazione di un determinato parametro booleano sull'*Animator*, che gestisce le animazioni; a questo punto entrano in gioco diverse dinamiche:

    - Il *Collider* esterno, quello assegnato all'oggetto assemblato, viene disattivato, così come l'*ObjectManipulator* e il *BoundsControl;* vengono invece attivati i *Collider* e i *BoundsControl* assegnati ai componenti singoli (i pezzi che compongono l'oggetto);

    - Se si tratta della prima esplosione, vengono create le *Sezioni* dell'oggetto una volta che l'animazione di esplosione si è conclusa; questo risultato è stato raggiunto attraverso una *Coroutine* che resta in attesa per il tempo dell'animazione e poi esegue le istruzioni per la creazione delle *Sezioni*;

    - Nel caso si operi in *Neuro Mode*, vengono resi attivi in NeuroTag e le *Sezioni* di componenti; in caso contrario, vengono resi attivi i manipolatori dei componenti;

    - Si attua un check tramite *Raycasting* che permette di individuare l'altezza a cui si trova l'oggetto assemblato da una certa superficie; nel caso sia troppo bassa, l'oggetto viene traslato in alto di un valore necessario a evitare il *clipping* delle componenti su tale superficie;

- Il parametro booleano *IsExploded* viene posto a *true*.

- ***True;*** significa che l'oggetto è esploso, perciò viene innescata l'animazione inversa sempre tramite il componente *Animator*; anche in questo caso, vengono eseguite varie istruzioni:

  - Il *Collider* esterno, quello dell'oggetto assemblato, viene riattivato insieme agli script *BoundsControl* e *ObjectManipulator*;

  - Il metodo *AssembleComponents*, proprio di *ComponentsManager*, viene invocato per riportate nella loro posizione originale tutte le componenti dell'oggetto (nel caso l'utente le abbia spostate);

  - Viene disattivata completamente la UI di tutti i componenti dell'oggetto, così come i NeuroTag, gli script *BoundsControl* e *ObjectManipulator* e le *Sezioni*.

Object Component

Si tratta dell'entità rappresentante del componente specifico, inteso come uno dei pezzi che va a comporre l'oggetto assemblato.

Le caratteristiche fondamentali sono elencate di seguito:

- Contiene 4 variabili booleane:
  - ***ManipulationStatus****;* definisce se il componente sta venendo manipolato oppure no;
  - ***IsInSection****;* definisce l'appartenenza o meno a una *Sezione*;
  - ***IsAlreadyonTable****;* definisce se il *SELECT* è già stato attuato per il componente;
  - ***IsSelectable****;* definisce se il *SELECT* è possibile in un determinato momento per il componente (es. *false* quando il robot è in movimento);

- Eredita dall'interfaccia *INeuroTagObserver* per l'osservazione degli eventi che coinvolgono i NeuroTag; in particolare *OnNeuroTagTriggered* e *OnNeuroSectionTriggered*, rispettivamente per l'innesco delle azioni che l'oggetto deve compiere in relazione a un NeuroTag o alla *Sezione* associata;

- Include 3 *Coroutine*, che definiscono azioni diverse in un certo tempo:

  - ***SmoothTranslateToLocalPosition***; riceve come parametro una posizione e attua una traslazione locale, tramite interpolazione lineare, fino a quando l'oggetto non raggiunge quella posizione; viene usata per far tornare la mesh del componente al suo posto nel modello esploso;

  - ***MoveInTheForeground***; riceve due parametri di tipo *float*, denominati *offsetX* e *offsetZ*; anche in questo caso, la mesh del componente viene traslata secondo la direzione frontale verso cui l'utente è rivolto (*camera forward*) e successivamente ruotata su sé stessa per essere rivolta verso la camera attiva; in entrambi i casi, viene seguito il

metodo dell'interpolazione lineare;

- ***RotateComponentCoroutine;*** ruota la mesh del componente di 90° secondo l'asse verticale; viene utilizzata quando l'utente interagisce con il bottone *Rotate* o innesca l'omonimo NeuroTag.

L'elemento più importante è definito dal metodo asincrono ***OnSelect***, il quale va ad azionare le dinamiche di *pick and place* del componente secondo i seguenti passaggi:

- Viene verificata l'esistenza di un'istanza di *RobotHandler* nella scena e l'avvenuta connessione del client con la rete locale;
- Viene effettuato l'aggiornamento della UI, chiamando il metodo *OnSelect* presente nell'oggetto di tipo *ComponentUI* legato al componente;
- Viene verificato che la variabile *IsSelectable* sia uguale a *true* e che la variabile *IsAlreadyOnTable* sia uguale a *false*;
- Se tutte le condizioni precedenti sono soddisfatte, viene invocato il metodo *SelectComponent* di *RobotHandler* per inviare l'indice corretto al terminale di controllo;
- Infine, il componente ritorna nella sua posizione originaria del modello esploso tramite il metodo *GoBackToAssembleObject*; nel caso si tratti di un *QuickSelect*, tutta le componenti della *Sezione* tornano al loro posto tramite il metodo *SectionBackToAssembledObject* di *ObjectSection*;

Object Section

La suddetta classe rappresenta una *Sezione* specifica dell'oggetto assemblato.

In particolare, gli elementi che costituiscono la classe *Object Section* sono i seguenti:

- Una lista di componenti assegnate a tale S*ezione* in base all'altezza a cui si trovano nel momento in cui l'oggetto è esploso; ogni componente può far parte di una e una sola *Sezione*.
- Un oggetto *Container* che rappresenta i limiti della *Sezione*; tale oggetto è caratterizzato da un *Collider* di tipo trigger, non interattivo tramite le *gesture*, e un materiale invisibile;
- Un *NeuroPad* posizionato di fianco al limite destro del *Container*, che innesca la dinamica di posizionamento in primo piano della *Sezione*;
- Un oggetto derivato dal prefab "*backNeuroButton*", che include un NeuroTag interno e un materiale di stimolazione; l'innesco di tale NeuroTag fa in modo che gli oggetti in primo

piano della *Sezione* ritornino al loro posto;

- Una lista di I*NeuroTagObserver*, che contiene tutti gli osservatori dei NeuroTag presenti nella classe (il *NeuroPad* e il *backNeuroButton*).

La classe include anche una serie di metodi per la gestione della visibilità di tutti gli elementi elencati, oltre a un metodo di calcolo del punto medio delle posizioni delle componenti; quest'ultimo è utile per il posizionamento corretto della *Sezione*.

I metodi fondamentali, invece, sono sostanzialmente due:

- ***MoveSectionInTheForeground***;
    - Per ogni I*NeuroTagObserver* nella lista interna alla classe, viene innescato il metodo *OnNeuroSectionTriggered;* quest'ultimo riceve un parametro float che rappresenta l'offset sull'asse orizzontale di posizionamento in primo piano delle componenti;
    - L'offset viene calcolato progressivamente tramite interpolazione lineare, in modo tale che le componenti siano equispaziate davanti all'utente;
    - Dopodiché viene attivato il *backNeuroButton* e vengono disabilitati i *Container* e i *NeuroPad* di tutte le *Sezioni*;
- ***SectionBackToAssembledObject***;
    - Riceve una stringa come parametro che rappresenta l'indice di un componente, chiamata *exceptionIndex*;
    - Per ogni componente, viene invocato il metodo *GoBackToAssembledObject* al fine di farlo ritornare nella posizione originale sul modello esploso, con l'eccezione del componente il cui indice è uguale a *exceptionIndex;*
    - Nel caso in cui *exceptionIndex* abbia il valore "NULL", tutte le componenti della *Sezione* ritornano nella posizione originale;

Working Table

La classe *WorkingTable* rappresenta il tavolo da lavoro definito dall'utente; contiene un materiale di default e due materiali per verificare che il tavolo resti su una superficie (blu, verde, rosso)

Tale verifica viene effettuata nell'*override* del metodo *Update*, chiamato a ogni frame ed ereditato dalla classe *MonoBehaviour* interna a Unity.

I metodi fondamentali sono due:

- ***OnStart;*** invocato quando ha inizio la fase di definizione del tavolo; la posizione della mesh rappresentante il tavolo viene spostata di fronte all'utente e viene attivato il suo *Collider* al fine di permettere la manipolazione tramite *gesture*;

- ***OnExit;*** invocato quando l'utente esce dalla definizione del tavolo; il materiale della mesh ritorna a essere quello di default e viene disattivato il suo *Collider*;

Sono presente anche due *Coroutine* che integrano un'animazione di *Fade* sul materiale della mesh associata al tavolo:

- *MaterialFading*; riceve un parametro booleano denominato *visible*, che permette il *FadeIn* o il *FadeOut* a seconda del suo valore (rispettivamente, *true* e *false*);

- *MaterialGlow*; compie un'animazione di *FadeIn* e una di *FadeOut* perché l'utente identifichi il tavolo in determinate occasioni (quando l'oggetto torna sul tavolo, per esempio).

3.4.4 Entità UI

Le entità UI rappresentano script preposti alla gestione degli elementi di interfaccia utente presenti nei vari oggetti, quali collezioni di bottoni, menù e NeuroTag.

ObjectUI

La classe *ObjectUI* è legata all'omonimo prefab e gestisce i parametri di tutti gli elementi interni ad esso, come la visibilità o l'impostazione del testo (es. titolo, descrizione del pannello di info etc.).
Si specifica che questa classe rappresenta l'interfaccia generale dell'oggetto assemblato e non delle sue singole componenti; conseguentemente, non essendo presenti NeuroTag, la gestione riguarda esclusivamente bottoni interagibili tramite *gesture*.

Al suo interno, è presente una lista di *IUIObserver* per l'osservazione delle dinamiche di interazione delle utente con la *user interface*.

Il metodo *SetOnClickListeners* si occupa di impostare, per ogni bottone dell'interfaccia, gli eventi da invocare quando vengono cliccati dall'utente; questi ultimi sono cinque:

- ***OnStartAssembling;*** se il tavolo è stato definito, invoca il metodo *OnAssemblingSessionStarted* per ogni osservatore e attiva i bottoni utili per la fase di assemblaggio (*Explode, Anchor* e *Back*); in caso contrario, viene invocato un metodo di *UIManager* per notificare all'utente che il tavolo non è stato definito;

- ***OnExplode;*** invoca il metodo omonimo dello script *AssembledObject,* il quale cambia le

icone e il testo del bottone *Explode* a seconda che si tratti di un'esplosione o di un'implosione (modello esploso/modello assemblato);

- ***OnBackToTable;*** invoca il metodo *GoBackToTablePosition* dello script di tipo *AssembledObject* sull'oggetto assemblato corrente;

- ***OnAnchor;*** invoca il metodo *OnToggleAnchor* dello script di tipo *AssembledObject* sullo stesso oggetto assemblato;

- ***OnInfoPanelToggled;*** disattiva lo script *RadialView* sul pannello di info dell'oggetto;

ComponentUI

La classe *ComponentUI* rappresenta l'interfaccia utente legata a un componente specifico e gestisce tutti i suoi elementi, includendo sia quelli della modalità *neuro* che quelli della modalità *gesture*.
Integra al suo interno l'*interface* denominata *INeuroTagObserver* per l'osservazione dei NeuroTag legati a tutte le entità UI interne.

Gli elementi fondamentali inclusi sono i seguenti:

- Due oggetti di tipo ***ComponentAppBar,*** che rappresentano la barra verticale e quella orizzontale della UI di tipo *gesture*; tali elementi vengono instanziati in fase di inizializzazione e vengono loro impostati i vari parametri di adattamento al componente (scala, target da seguire etc.);

- Un oggetto denominato ***NeuroUI*** che rappresenta gli elementi di interfaccia utente di tipo *neuro*; anch'esso contiene tre *ComponentAppBar* che vengono instanziate e impostate a dovere in fase di inizializzazione;

- Un elemento di tipo ***BoundsControl*** per il controllo dei limiti del componente; questi viene aggiunto in fase di inizializzazione e legato al *Collider* della mesh rappresentante tale componente;

A parte i metodi di gestione della UI, come quelli preposti alla visibilità di determinati elementi, sono presenti quelli ereditati dall'*interface*: *OnNeuroTagTriggered* e *OnNeuroSectionTriggered*.
Come nel caso di *ObjectComponent*, il primo agisce in base al parametro che gli viene passato, mentre il secondo si limità ad attivare l'elemento *NeuroUI*.

L'unico metodo che non viene invocato tramite l'*interface* è ***OnSelect***, che viene invece richiamato da *ObjectComponent* nell'omonimo metodo per motivazioni di sincronia tra le azioni.
In particolare, il metodo si limita ad attivare dei *WarningPanel* e a impostarne il contenuto in base al successo o al fallimento del *SELECT*.

ComponentAppBar e NeuroPad

Si tratta di due entità associate rispettivamente alle componenti e alle *Sezioni:*

- ***ComponentAppBar;*** entità legata a una specifica collezione di bottoni e può essere di tipo *neuro* oppure di tipo *gesture:*
    - *Neuro;* tutti i bottoni vengono inizializzati secondo il *NeuroTag* che contengono, aggiungendo i giusti *listener* a ognuno quando viene rilevata la focalizzazione dell'utente;
    - *Gesture;* tutti i bottoni vengono inizializzati secondo il loro componente *Interactable*, per cui è possibile aggiungere eventi da invocare nel momento in cui avviene il click dell'utente;
- ***NeuroPad;*** entità associata all'omonimo prefab e legata a una *Sezione* particolare;

In entrambi i casi, l'interfaccia utente segue uno specifico *target*, rispettivamente un componente dell'oggetto assemblato e una *Sezione* specifica.

Per la gestione degli eventi legati ai NeuroTag, le entità *ComponentAppBar* di tipo *neuro* e le entità *NeuroPad* includono una lista di osservatori di tipo *INeuroTagObserver*.

ProjectMenu

Entità associata ai menu presenti nell'applicazione, in particolare *HandMenu* e *Menu Superiore*.

Eredita metodi da due *interface: IProjectSceneObserver* e *IUIObserver*.

Nel primo caso, il metodo ereditato è *OnOperationalModeChanged*; questi notifica al menù che è stata cambiata la modalità operativa e sono necessari dei cambiamenti dell'interfaccia utente.
La necessità di utilizzare questo approccio deriva dal fatto che, nonostante la modalità operativa venga cambiata proprio in un oggetto di tipo *ProjectMenu*, è necessaria la sincronia tra *HandMenu* e *MenuSuperiore*; in altre parole, la UI deve essere aggiornata su entrambi i menu nello stesso momento.

Secondariamente, il metodo ereditato da *IUIObserver* è *OnAssemblingSessionStarted*; questi si limita a rendere visibile il bottone *ExitAssembling* quando l'utente entra nella fase di assemblaggio.

3.4.5 Interfaces

Sono state incluse nell'architettura del codice tre *interface* che fungono da osservatori, per garantire una corrispondenza uno a molti in relazione a determinati eventi.

- **INeuroTagObserver;** contiene due metodi:
    - *OnNeuroTagTriggered(string ActionType)*; include come parametro una stringa per definire l'azione da eseguire; viene invocato dalle entità *ComponentAppBar* e *NeuroPad* per l'impostazione dei *listener* in relazione ai vari eventi su *ObjectComponent* e su *ComponentUI*;
    - *OnNeuroSectionTriggered*; viene invocato principalmente dai *NeuroPad* associati alle *Sezioni*;
- **IUIObserver;** include il metodo *OnAssemblingSessionStarted*, invocato da varie entità per eseguire le azioni corrispondenti a tale evento.
- **IProjectSceneObserver;** include il metodo *OnOperationalModeChanged*, a cui viene passato un parametro del tipo enumerativo *OperationalMode* definito in *ProjectSceneManager*; viene utilizzato per notificare il cambiamento della modalità operativa alle entità poste come osservatrici.

Entità di gestione	• ProjectSceneManager • UIManager • ComponentsManager • RobotHandler
Entità oggetto	• AssembledObject • ObjectComponent • WorkingTable • ObjectSection
Entità UI	• ObjectUI • ComponentUI • NeuroPad • ComponentAppBar • ProjectMenu
Interface	• INeuroTagObserver • IUIObserver • IProjectSceneObserver

Tabella 3.1. *Entità presenti nell'architettura di codice divise secondo le loro tipologie*.

4.Test e risultati

4.1 Introduzione

Dopo aver effettuato le verifiche necessarie al corretto funzionamento del sistema in generale, è iniziata la fase di testing sugli utenti.

Oltre ai dati anagrafici dei tester prescelti, sono stati definiti i parametri fondamentali da misurare in merito agli obiettivi di questa trattazione:

- **Usabilità** del sistema in generale, valutando i risultati ottenuti in associazione al contesto dell'assemblaggio nell'industria 4.0; in tale analisi sono state incluse le considerazioni dei dati ottenuti in termini di tipologia di utenza scelta per i test;

- **Carico di lavoro** per l'utente durante l'utilizzo del sistema, ponendo particolare attenzione all'aspetto di interfaccia uomo-BCI e uomo-macchina, al fine di comprenderne l'efficacia e di valutarlo in termini di frustrazione, stress mentale e stress fisico dell'utilizzatore.

Per raggiungere tali scopi, sono stati utilizzati due paradigmi di misurazione: **SUS** e **NASA-TLX.**

4.1.1 SUS

Il **SUS (System Usability Scale)** è una scala di valutazione dell'usabilità di un sistema. L'asserzione da cui si parte è indicata dal fatto che non esiste una misurazione di tale parametro assoluta; di fatto, l'usabilità dipende fortemente dal contesto in cui definita. [42]

Secondo una valutazione in ambito industriale, che desta l'interesse di questa trattazione, il processo completo di analisi e di selezione delle metriche specifiche può essere spesso non adatto né pratico; nella maggior parte dei casi, tutto quello che viene richiesto è un indice generale dell'usabilità di un sistema, al fine di confrontarlo con i suoi predecessori o con sistemi attuali analoghi. Questo sistema di scala è stato definito partendo proprio da questi presupposti, cercando di soddisfare le misure fondamentali per valutare l'usabilità di un sistema secondo lo standard ISO 924-11: efficacia, efficienza e soddisfazione.
La valutazione di questi parametri avviene tramite un questionario di dieci affermazioni proposte, dopo l'utilizzo del sistema in questione, a un utente specifico; quest'ultimo ha il compito di assegnare un certo grado di accordo o disaccordo per ogni affermazione secondo una scala da 1 a 5.

Il punteggio finale va da 0 a 100 e viene calcolato sommando i contributi singoli per ogni affermazione; si sottolinea che i contributi singoli non sono i punteggi puri assegnati dall'utente, ma sono calcolati da questi ultimi in base all'affermazione specifica.
Inoltre, non si tratta di una percentuale ma di un valore indicativo dell'usabilità generale del sistema; il punteggio medio, in riferimento agli standard dell'industria, si aggira intorno al valore di 68 in scala SUS.

4.1.2 NASA - TLX

Il **NASA-TLX** è uno strumento di valutazione proprio della NASA che permette la valutazione soggettiva del **carico di lavoro (*Workload)*** di un utente nell'interazione con vari sistemi uomo-macchina.

Tramite una procedura di misurazione multidimensionale, il NASA-TLX produce un punteggio totale basato sulla media dei risultati riguardo a 6 diversi parametri:

- **MENTAL DEMAND**, che indica lo sforzo mentale dell'utente nel compiere una determinata task (decisioni, calcoli, ricerca, memoria...);

- **PHYSICAL DEMAND**, che misura lo sforzo fisico dell'utente per compiere una task (tirare, torcere, sollevare, spingere…);

- **TEMPORAL DEMAND**, che valuta quanta pressione dal punto di vista temporale ha percepito l'utente nel compiere una task, nonché il tempo per portarla a termine (ritmo, velocità, rapidità…);

- **PERFORMANCE**, che misura quanto l'utente ha avuto successo nel finire una task e il suo grado di soddisfazione nel farlo;

- **EFFORT,** che misura lo sforzo complessivo compiuto dall'utente per raggiungere il proprio livello di perfomance;

- **FRUSTRATION**, che valuta lo stress accumulato, la frustrazione e l'irritazione dell'utente nel compiere una task;

La procedura completa consiste nel sottoporre all'utente, dopo l'utilizzo dell'applicazione presa in esame, due valutazioni distinte che definiscono altrettante fasi:

- Valutazione singola dei parametri precedenti per ogni task compiuta, chiedendo all'utente di valutare ognuno di essi tramite una scala da 1 a 20;

- Determinazione dei pesi da assegnare ad ogni parametro, presentando all'utente 15 coppie di valori che si formano dalla combinazione dei 6 parametri di valutazione; per ogni coppia, l'utente dovrà decidere quale dei due parametri ha influito di più durante l'utilizzo del sistema.

- I pesi sono calcolati in base a quante volte l'utente ha selezionato un determinato parametro durante l'assegnazione, mentre il punteggio totale pesato per ogni parametro deriva dalla moltiplicazione di pesi e punteggi singoli non pesati.

- Infine, il valore di carico di lavoro globale viene calcolato sommando i contributi pesati di ogni parametro e dividendo per 15.

Sia il SUS che il NASA-TLX sono stati molto utili per misurare l'efficienza del sistema durante le fasi di testing, le quali sono state preparate in merito a un generico caso d'uso per l'assemblaggio di un oggetto.

Durante i test sono state fatte provare agli utenti le due modalità presenti nell'applicazione: gesture e neuro. Entrambe sono state valutate successivamente tramite il SUS, per l'usabilità dell'esperienza generale, e il NASA-TLX, per le task specifiche richieste all'utente.

Nei paragrafi successivi verranno esposte le modalità con cui queste sperimentazioni sono state svolte, la descrizione del caso d'uso in esame, l'utenza scelta per i test e i risultati ottenuti; riguardo a questi ultimi, verrà svolta nella parte finale una breve analisi statistica per confrontare le due modalità operative dell'applicazione e per l'usabilità generale del sistema.

4.2 Descrizione caso d'uso

Attraverso i CAD reperiti online [41], le varie componenti del oggetto da assemblare sono state stampate in 3D e poste su uno strato di spugna, opportunamente ritagliato in corrispondenza di ogni componente per mantenere la loro posizione inalterata durante le fasi di testing.

Il modello originale dell'oggetto prevedeva l'inserimento di alcune clip sottili per fissare alcune componenti e garantire una giusta stabilità della struttura. Tuttavia, tali elementi sono stati eliminati nel caso d'uso proposto al fine di permettere la rapida ripetizione dei test; l'aggiunta delle clip è stata omessa principalmente per la loro difficoltà di rimozione dopo l'inserimento.

In base alle posizioni delle componenti, sono state fornite al robot, tramite uno script Python eseguito dal terminale di controllo, le coordinate spaziali e gli indici specifici di ognuna; questa fase è stata fondamentale per garantire un *pick and place* delle componenti efficace da parte del robot, senza che questi urtasse per sbaglio altri elementi circostanti.
In questa caso sono state anche definite le posizioni di "Home" (braccio robotico verticale) e di "Riposo" (braccio in attesa di ricevere istruzioni). La posizione di rilascio delle componenti, invece, è stata identificata tramite le coordinate spaziali in corrispondenza di un altro strato spugnoso posto vicino alla postazione dell'utente, facilmente raggiungibile anch'essa dal robot tramite un movimento di rotazione.

Per quanto riguarda le operazioni da far svolgere agli utenti, la fase di testing vera e propria dell'applicazione ha coinvolto due **Task** diverse:

- **Task 1 - Ricerca informazioni**; la task consisteva nell'andare a cercare una determinata componente all'interno del modello esploso dell'oggetto assemblato e trovare le informazioni relative ad essa nel pannello di *Info* specifico. In particolare, le informazioni richieste da cercare erano le dimensioni e il peso della componente*;*

- **Task 2 – SELECT delle componenti**; la task consisteva nel selezionare sequenzialmente le

componenti, in base all'ordine di assemblaggio fornito all'utente, per innescare la dinamica di *pick and place* del robot. Nell'effettivo, l'assemblaggio vero e proprio dell'oggetto reale non è stato richiesto all'utente specifico, ma solamente il SELECT delle sue componenti;

Le modalità con cui svolgere queste operazioni, *gesture* o *neuro*, sono state definite progressivamente in base alla sessione di test specifica; la priorità è stata data in ogni caso alla modalità neuro, in quanto rappresenta l'oggetto principale della trattazione.

4.3 Set-up sperimentale

I test hanno avuto luogo in una stanza d'ufficio, posizionando opportunamente gli elementi del caso d'uso proposto in questa trattazione.
La postazione dell'utente, con indosso i dispositivi, è stata delineata sullo spazio libero di una lunga scrivania. Davanti a questa, sempre sulla scrivania, è stato posto uno schermo LCD per mostrare delle istruzioni specifiche per ogni fase della sessione di testing; tale elemento è stato utile per fornire informazioni aggiuntive all'utente, oltre a quelle verbali, nel caso in cui questi necessitasse una corrispondenza visuale tra il suddetto schermo e ciò che vedeva sull'applicazione.

Lo strato spugnoso di rilascio è stato posizionato sulla scrivania alla destra della postazione dell'utente, così come il robot e.DO e l'altro strato di spugna contenente le componenti; tale posizionamento è stato progettato in modo tale da permettere al tester di vedere chiaramente il braccio robotico in movimento e di prendere le componenti dal punto di rilascio.
Lo schermo del terminale di controllo è stato invece posto sul pavimento, dietro la scrivania e non visibile dalla postazione; le motivazioni principali sono state quelle di non distrarre l'utente durante l'esecuzione dell'applicazione e di liberare spazio sulla scrivania.

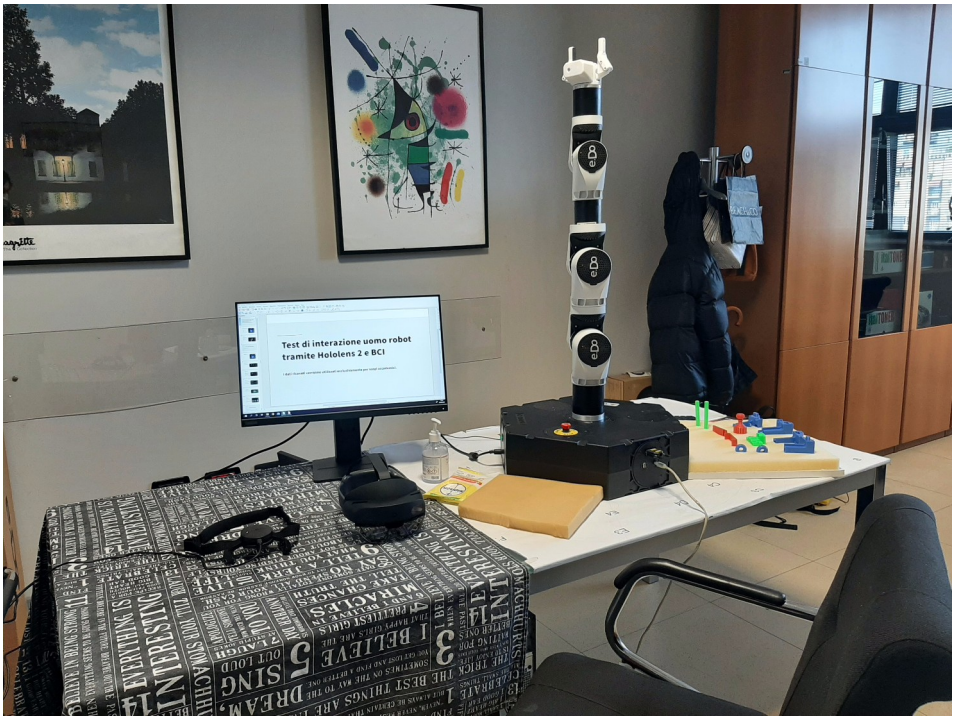


Fig. 4.1 *Set-up sperimentale arredato per le fasi di testing;*

4.4 Svolgimento dei test

L'applicazione è stata testata su 16 utenti di generi misti ed età compresa tra i 22 e i 29 anni, tutti studenti o laureati presso il Politecnico di Torino in vari corsi di laurea.

Di questi 16 utenti, 11 di loro avevano già usato un dispositivo di realtà mista (VR/AR) prima di allora, mentre nessuno di loro aveva mai provato una BCI. Di contro, la maggior parte di loro non utilizzava spesso dispositivi di realtà mista, né applicazioni AR.

Le modalità operative dell'applicazione, *neuro* e *gesture*, allo stato attuale sono state sperimentate rispettivamente su 10 e 6 utenti; è stato deciso di dare la priorità alla parte *neuro* per avere dei dati confrontabili al fine di validare l'utilizzo della BCI nell'ambito dell'assemblaggio.

I test sono stati svolti secondo le seguenti fasi:

- *Accoglienza e sanificazione;* durante l'inizio della sessione, all'utente sono state fatte igienizzare le mani, in accordo alle norme sanitarie per l'emergenza COVID; per le stesse motivazioni, sono stati altrettanto sanificati i dispositivi Hololens 2 e NextMind;

- *Set-up dispositivi;* dopo una breve spiegazione della procedura sperimentale e dei propositi dei test, l'utente è stato fatto accomodare alla postazione di lavoro e gli sono stati fatti indossare i dispositivi (prima NextMind, dopo Hololens 2);

- *Accesso a Hololens; s*uccessivamente, dopo aver illustrato all'utente le modalità di utilizzo delle *gesture*, è stato fatto effettuare al tester l'accesso al dispositivo Hololens 2 e gli è stata fatta aprire l'applicazione sviluppata;

- *Calibrazione;* Tramite istruzioni verbali e visive, presenti sull'apposito schermo LCD, il tester è stato guidato attraverso le fasi di calibrazione del NextMind,

- *Tutorial;* l'utente è stato poi reindirizzato verso una scena di tutorial aggiunta appositamente per i test; in quest'ultima, è stata fatta prendere confidenza all'utente con le meccaniche di manipolazione e con la focalizzazione propria di NextMind;

- *Selezione* progetto *e definizione tavolo*; una volta terminata la fase di tutorial, è stata fatta iniziare la fase di selezione del progetto, guidando l'utente passo passo nella definizione del tavolo e della modalità operativa scelta per la sessione di testing;

- *Fase di assemblaggio;* all'utente è stato richiesto di svolgere le due task secondo la modalità operativa precedentemente definita; per la seconda task, quella di SELECT, i primi due passi sono stati guidati, mentre i restanti sono stati lasciati svolgere autonomamente al tester;

- *Compilazione questionari*; al termine delle due task, sono stati fatti togliere i dispositivi all'utente e gli si sono stati fatti compilare i questionari per i dati anagrafici, il SUS e il NASA-TLX.

Fig. 4.1. *Immagine scattata durante uno dei test, in cui l'utente è impegnato nella fase di assemblaggio e il robot sta rilasciando un componente sul tavolo;*

4.5 Risultati

Tutti gli utenti, con più o meno difficoltà, sono riusciti a completare le task assegnategli rispetto a una o all'altra modalità operativa (gesture/neuro).

In alcuni casi rari, la task di assemblaggio è stata interrotta per malfunzionamenti dovuti ai limiti del robot e.DO; tuttavia, dopo averlo calibrato nuovamente, il sistema è sempre tornato completamente funzionante e l'utente ha potuto terminare la task in corso.

Di seguito, si riportano tutti i risultati ricavati dalle sessioni di testing sugli utenti; in particolare, vengono riportati i dati legati al SUS, al NASA-TLX e alle tempistiche e agli errori dei tester.

4.5.1 Risultati SUS – System Usability Scale

Di seguito sono riportati i dati ottenuti dai questionari compilati dagli utenti per la valutazione dell'usabilità tramite il SUS. Le tabelle sono divise in due colonne, dove il primo valore indica il numero identificativo del tester mentre il secondo indica il punteggio totale ottenuto da tale utente; il punteggio è stato calcolato tramite il contributo effettivo per ogni punteggio singolo di ogni domanda, andando a sommare tutti i risultati e moltiplicando per 2.5:

$$C_i = S_i - 1 \ ,$$

dove C_i è il contributo effettivo e S_i è il punteggio (*Score*) ottenuto dall'utente nelle domande di indice i = 1, 3, 5, 7, 9.

$$C_j = 5 - S_j \ ,$$

dove C_i è il contributo effettivo e S_j è il punteggio (*Score*) ottenuto dall'utente nelle domande di indice j = 2, 4, 6, 8, 10.

$$TS = [(\sum C_i) + (\sum C_j)] \cdot 2.5 \ ,$$

dove TS è il punteggio totale (*Total Score*) calcolato tramite la sommatoria di tutti i contributi e una moltiplicazione, misurato su una scala da 0 a 100.

NEURO MODE

User Id	**Total Score**
2	62,5
3	67,5
6	92,5
7	77,5
8	72,5
9	75
10	82,5
11	70
12	77,5
13	80

TOTAL	757,5

***Tabella 4.2**. Risultati ottenuti in merito all'esperienza generale in neuro mode;*

GESTURE MODE

User Id	Total Score
1	37,5
4	77,5
5	67,5
14	90
15	75
16	87.5

Tabella 4.1 *Risultati ottenuti in merito all'esperienza generale in gesture mode;*

4.5.2 Risultati NASA – TLX

Di seguito sono riportati i dati ottenuti dai questionari compilati dagli utenti per la valutazione del carico di lavoro tramite il NASA-TLX. Le tabelle presentate di seguito si riferiscono alle due Task richieste all'utente durante l'esperienza: *Ricerca Informazioni* e *SELECT delle componenti.*

Per ogni tabella, la prima colonna si riferisce al numero identificativo dell'utente; dalla seconda alla settima sono rappresentati i **pesi** (prima tabella del paragrafo) e i punteggi singoli **pesati** (seconda Tabella del paragrafo) per ogni parametro di valutazione ottenuti dall'utente. L'ultima colonna della seconda tabella di ogni sezione si riferisce al *carico di lavoro globale*, calcolato secondo lo standard del NASA-TLX; i calcoli di punteggi singoli pesati (S_i) e di *Global Workload* (GWL) vengono riportati di seguito:

$$S_i = RS_i \cdot W_i \ ,$$

dove S_{i}, RS_{i} e W_{i} sono rispettivamente il punteggio pesato (*Score*) , il punteggio non pesato (*Raw Score*) e il peso del parametro con indice i (*Weight*).

$$GWL = \frac{(\sum S_i)}{15} \ ,$$

dove GWL è il carico di lavoro globale (*Global Workload*), misurato su una scala da 0 a 100.

NEURO MODE – TASK 1

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration
2	2	2	3	4	3	1
3	3	3	5	1	3	0
6	3	1	4	4	3	0
7	0	2	4	3	3	3
8	3	2	3	3	2	2
9	2	0	4	5	2	2
10	5	0	3	3	3	1
11	5	0	2	4	3	1
12	4	3	1	5	2	0
13	2	2	3	5	2	1

***Tabella 4.3**. Pesi assegnati a ogni parametro da ogni utente per la Task 1;*

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration	Global Workload
2	70	40	150	120	210	60	43,3
3	135	135	300	25	150	0	49,6
6	105	10	180	20	45	0	24
7	0	60	60	45	45	45	17
8	45	70	60	135	30	50	26
9	50	0	260	200	120	60	46
10	350	0	240	45	180	5	54,6
11	325	0	90	80	225	35	50,3
12	260	150	45	125	130	0	47,3
13	110	10	105	175	150	20	38

***Tabella 4.4**. Score singoli pesati per ogni parametro e Global Workload per ogni utente per la Task 1;*

NEURO MODE – TASK 2

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration
2	2	4	2	3	2	2
3	2	5	4	1	3	0
6	3	1	4	5	2	0
7	1	4	3	5	2	0
8	3	0	4	4	3	1
9	4	0	2	5	2	2
10	3	0	4	1	5	2
11	5	0	2	4	3	1
12	5	0	1	3	4	2
13	0	1	3	4	3	4

Tabella 4.5. *Pesi assegnati a ogni parametro da ogni utente per la Task 2;*

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration	Global Workload
2	130	20	120	120	130	130	43,3
3	90	300	320	50	225	0	65,6
6	105	5	140	25	40	0	21
7	40	140	120	100	100	0	33,3
8	195	0	340	320	135	30	68
9	80	0	40	250	60	30	30,6
10	285	0	380	15	500	70	83,3
11	375	0	120	100	195	40	55,3
12	350	0	40	120	260	110	58,6
13	0	5	180	120	255	240	53,3

Tabella 4.6. *Score pesati per ogni parametro e Global Workload per ogni utente per la Task 2;*

GESTURE MODE – TASK 1

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration
1	1	1	3	5	2	3
4	0	3	4	5	1	2
5	2	1	3	5	4	0
14	3	0	5	4	2	1
15	3	1	2	0	4	5
16	5	0	2	4	2	2

Tabella 4.7. *Pesi assegnati a ogni parametro da ogni utente per la Task 1;*

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration	Global Workload
1	45	50	40	50	40	65	49.3
4	25	70	45	60	35	60	56,3
5	25	60	35	80	40	25	51,6
14	75	0	25	80	10	5	13
15	15	25	10	0	60	350	30,67
16	150	0	50	20	30	40	19,33

Tabella 4.8. *Score pesati per ogni parametro e Global Workload per ogni utente per la Task 1;*

GESTURE MODE – TASK 2

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration
1	1	1	3	5	2	3
4	0	3	4	5	1	2
5	2	1	3	5	4	0
14	3	0	5	4	2	1
15	3	1	2	0	4	5
16	5	0	2	4	2	2

Tabella 4.9. *Pesi assegnati a ogni parametro da ogni utente per la Task 2;*

User Id	Mental Demand	Physical Demand	Temporal Demand	Performance	Effort	Frustration	Global Workload
1	65	40	55	30	20	70	42.3
4	45	70	65	70	50	50	62,6
5	25	70	55	75	45	30	61,3
14	0	375	300	180	60	30	63
15	165	25	110	0	220	450	64,67
16	225	0	50	40	45	20	25,33

Tabella 4.10. *Score pesati per ogni parametro e Global Workload per ogni utente per la Task 2;*

4.5.3 Tempistiche ed errori

Di seguito sono riportati i risultati riguardanti il tempo impiegato da ogni utente per ogni Task nelle due modalità; per le task di tipo *Neuro*, sono riportati anche gli **errori di selezione** dei NeuroTag commessi da ogni tester; nelle tabelle si sono anche evidenziati i tempi più brevi e più lunghi per l'esecuzione della task.

NEURO MODE

User Id	**Tempo (min)**	**Errori**
2	4,05	0
3	0,95	0
6	0,92	0
7	0,75	0
8	1,65	0
9	0,5	0
10	1,05	0
11	0,72	0
12	0,38	0
13	0,62	0

Tabella 4.11. *Tempi ed errori per la Task 1 (Neuro);*

User Id	Tempo (min)	Errori
2	17,6	2
3	14,03	0
6	12,93	0
7	13,02	1
8	13,06	0
9	12,13	3
10	11,92	3
11	12,35	1
12	11,23	3
13	12,03	0

Tabella 4.12. *Tempi ed errori per la Task 2 (Neuro);*

GESTURE MODE

User Id	Tempo (min)
1	1
4	0,41
5	0,55
14	0,35
15	0,6
16	0,3

Tabella 4.13. *Tempi per la Task 1 (Gesture);*

User Id	Tempo (min)
1	12
4	9,41
5	9,48
14	13,9
15	13,65
16	9,81

Tabella 4.12. *Tempi per la Task 2 (Gesture);*

4.6 Analisi

4.6.1 Usabilità

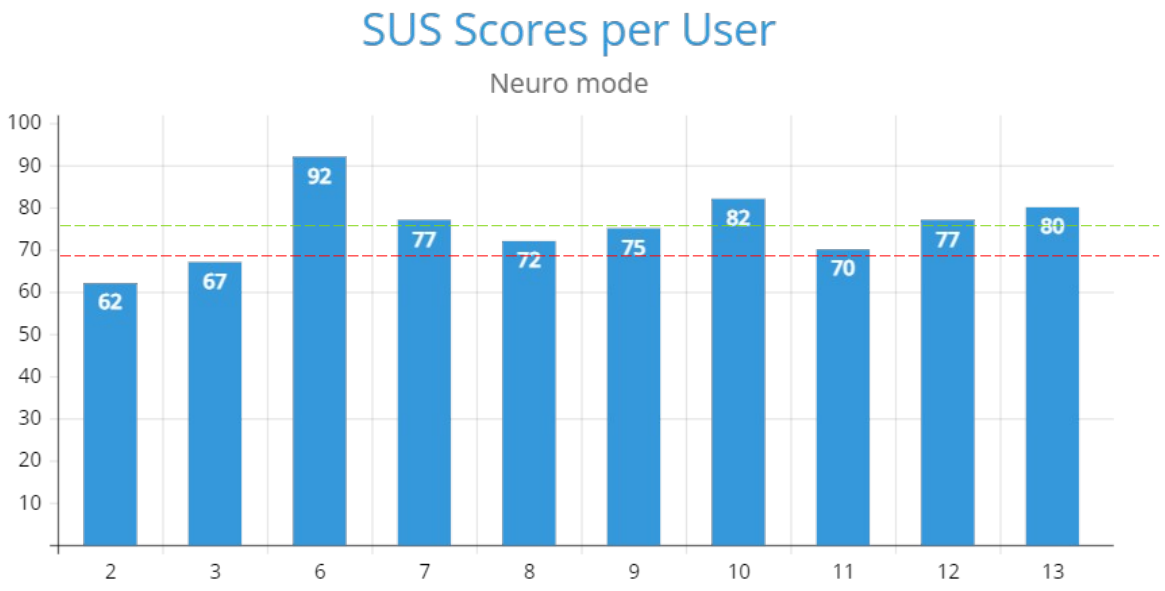


Grafico 4.1.

Il grafico 4.1 mette in mostra i risultati ottenuti in *neuro mode*, per ogni utente, tramite il SUS.
Il punteggio medio è di 75.75 (linea verde) ed è chiaramente sopra la media di 68 per gli standard dell'industria (linea rossa), con una differenza di 7.75. La varianza totale sui risultati di tutti gli utenti è di 63.8, a indicare una distribuzione semi-uniforme. Il massimo è stato assegnato dall'utente 6, con un punteggio pari a 92, mentre il minimo è stato assegnato dall'utente 2, con un punteggio pari a 62.

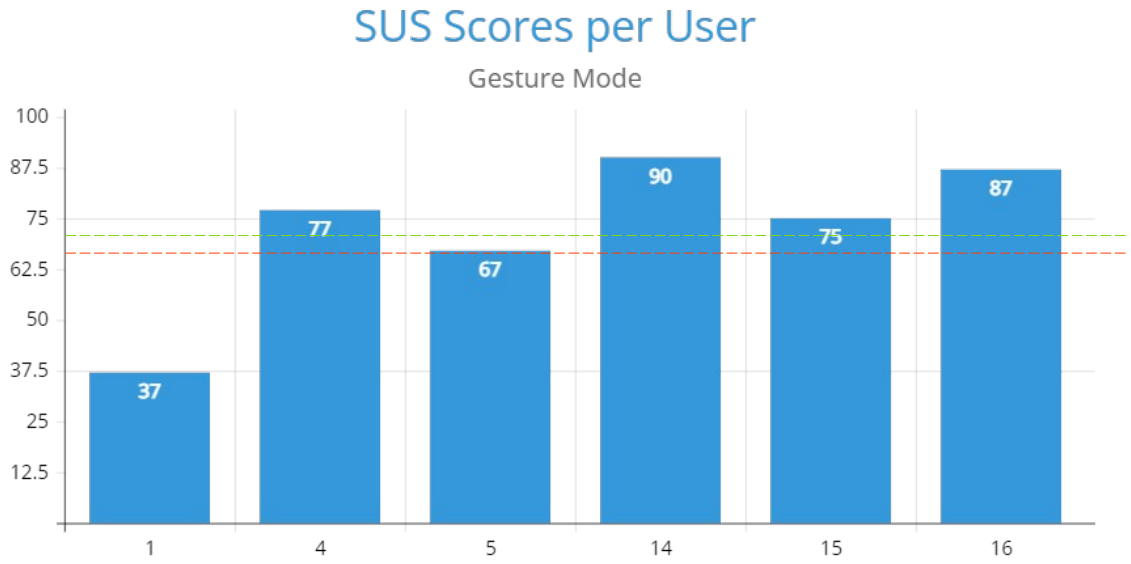


Grafico 4.2.

Il grafico 4.2 mette in mostra i risultati ottenuti in *gesture mode*, per ogni utente, tramite il SUS.
Il punteggio medio è di 72.5 (linea verde) ed è chiaramente sopra la media di 68 per gli standard dell'industria (linea rossa), con una differenza di 4.5. La varianza totale sui risultati di tutti gli utenti è di 302.08, maggiore di quella per *neuro mode*. Il massimo è stato assegnato dall'utente 14, con un punteggio pari a 90, mentre il minimo è stato assegnato dall'utente 1, con un punteggio pari a 37.

4.6.2 Carico di lavoro (NASA-TLX)

I primi due istogrammi istogrammi rappresentati in ognuno dei paragrafi successivi indicano i punteggi medi pesati ottenuti in una delle due modalità, per ogni Task e per ogni parametro del NASA-TLX: *Temporal Demand* (TD), *Mental Demand* (MD), *Frustration* (FR), *Physical Demand* (PD) e *Effort* (EF).

Dopodichè, viene attuato per ogni modalità un confronto tra le due Task, andando a rilevare i punti in comune e a confrontarne il carico di lavoro totale (*Global Workload*).

NEURO MODE

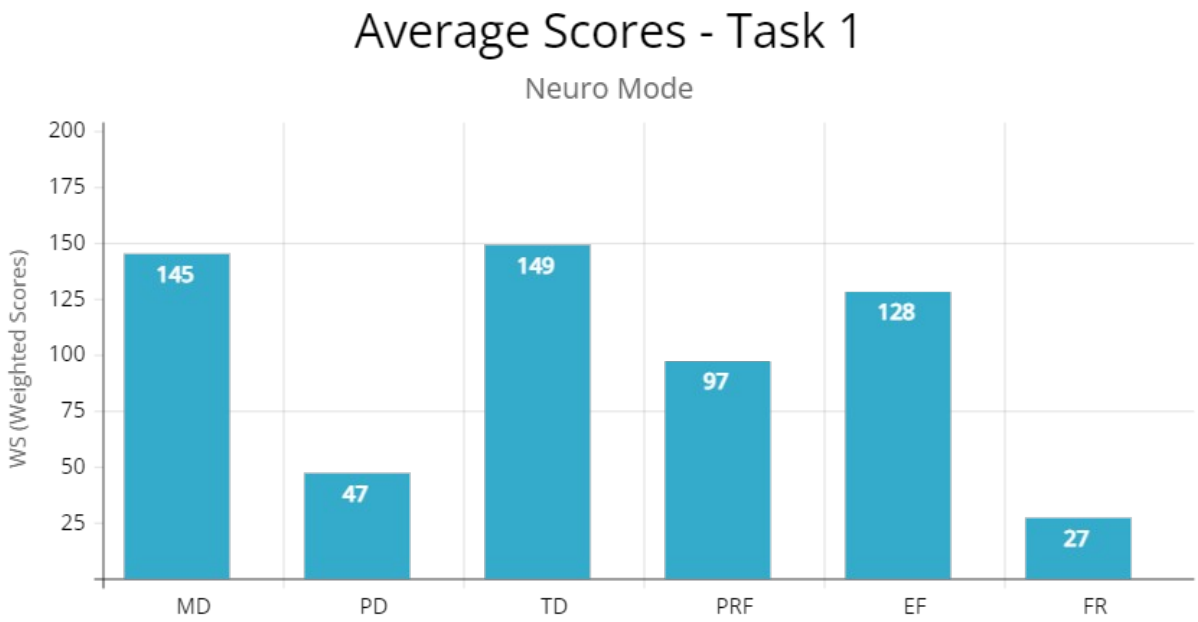


Grafico 4.3

Si nota dal grafico 4.3 che i punteggi medi più alti sono uguali a 149 e 145, rispettivamente per i parametri di *Temporal Demand* (TD) e di *Mental Demand* (MD), a indicare un carico di lavoro maggiore riguardo al fattore temporale e allo sforzo mentale; di contro, i punteggi minimi sono rappresentati dai parametri di *Frustration* (FR) e *Physical Demand* (PD), con punteggi medi rispettivamente di 27 e 47. Il tempo medio su tutti gli utenti per l'esecuzione della Task è di 1.16 min, con una varianza di 1.04 sul totale dei tempi.

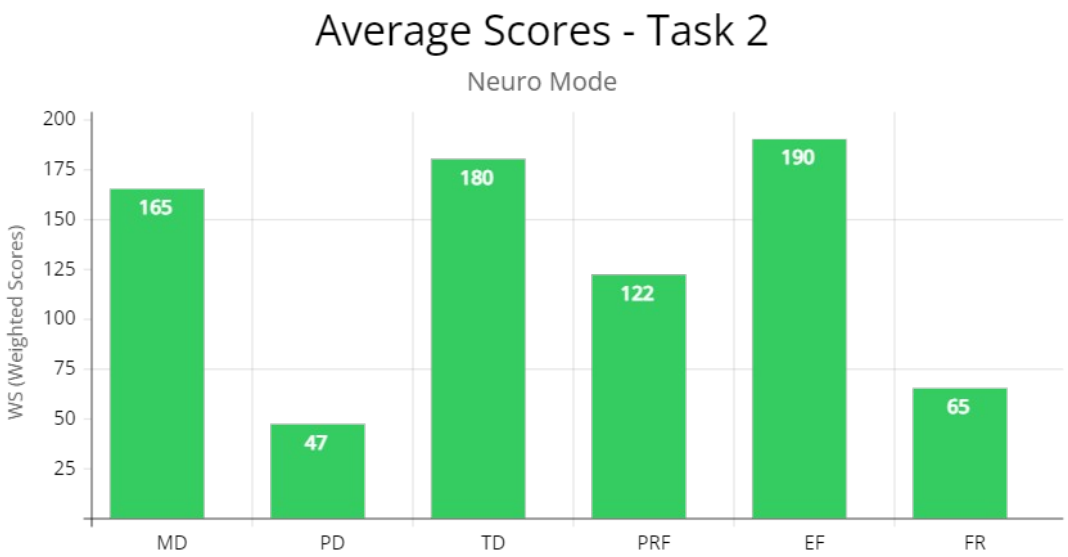


Grafico 4.4.

Dal grafico 4.4 si nota che i punteggi medi più alti sono uguali a 190 e 180, rispettivamente per i parametri di *Effort* (EF) e di *Temporal Demand* (TD), a indicare un carico di lavoro maggiore riguardo al fattore temporale e alla sforzo generale da compiere; il parametro di *Mental Demand* risulta comunque abbastanza alto, con un punteggio medio di 165. Di contro, i punteggi minimi sono rappresentati dai parametri di *Frustration* (FR) e *Physical Demand* (PD), con punteggi medi rispettivamente di 65 e 47. Il tempo medio su tutti gli utenti per l'esecuzione della Task è di 13.03 min, con una varianza di 2.86 sul totale dei tempi.

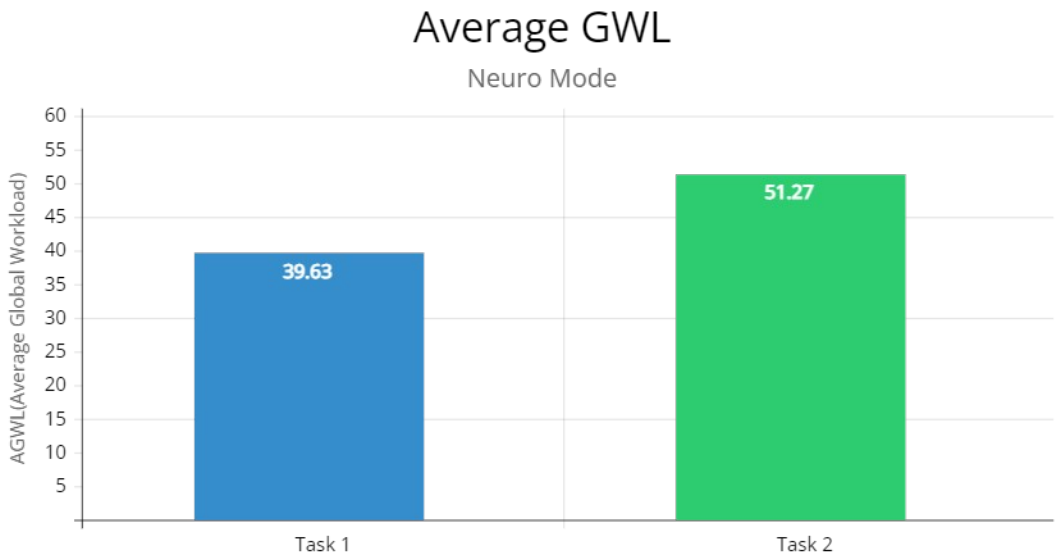


Grafico 4.5.

L'istogramma 4.5 raffigura il carico di lavoro medio (*AGWL – Average Global Workload)* per entrambe le task, ricavato dai *GWL* singoli per ogni utente che sono stati calcolati come illustrato nel paragrafo 4.5.2. La prima raggiunge un carico medio di 39.63, mentre il valore della seconda è di 51.27.

GESTURE MODE

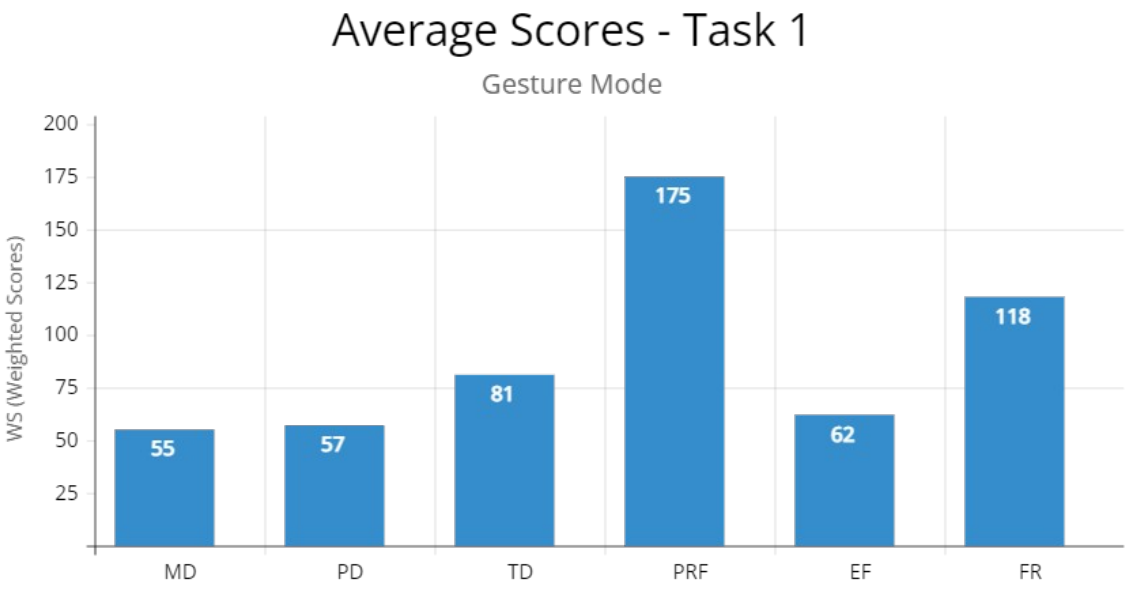


Grafico 4.6.

Si nota dal grafico 4.6 che i punteggi medi più alti sono uguali a 175 e 118, rispettivamente per i parametri di *Performance* (PRF) e di *Frustration* (FR), a indicare un maggior successo nel compimento della task ma anche una maggior frustrazione; di contro, i punteggi minimi sono rappresentati dai parametri di *Mental Demand* (MD), Physical Demand (PD), con punteggi medi rispettivamente di 55 e 57. Il tempo medio su tutti gli utenti per l'esecuzione della Task è di 0.54 min, con una varianza di 0.05 sul totale dei tempi.

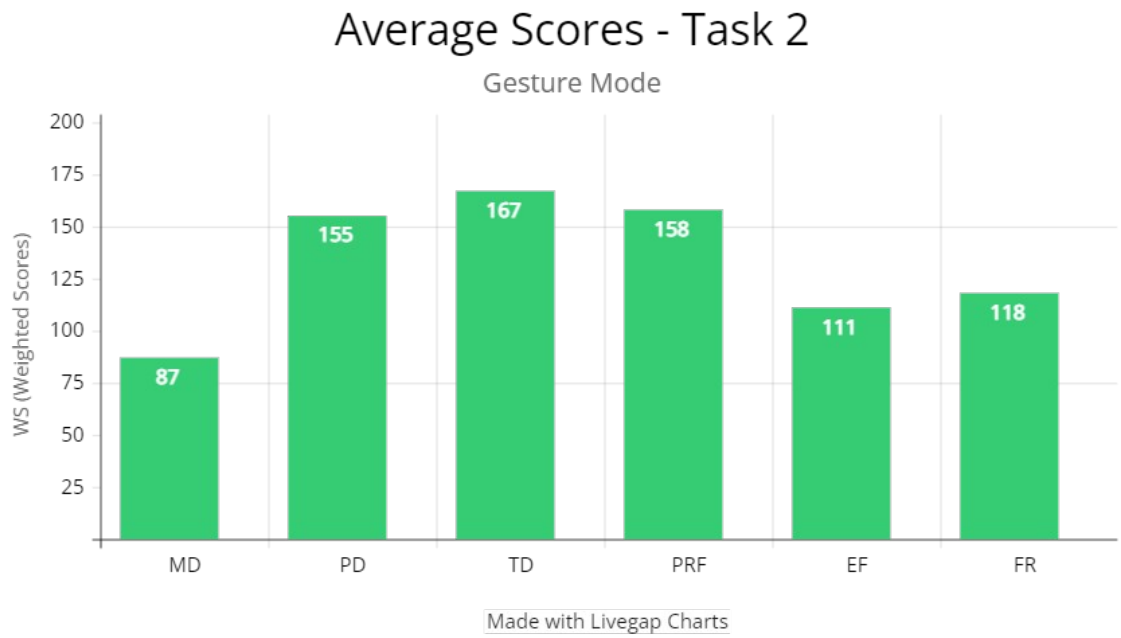


Grafico 4.7.

Dal grafico 4.7 si evince che i punteggi medi più alti sono uguali a 167, 158 e 155 , rispettivamente per i parametri di *Temporal Demand* (TD), di *Performance* (PRF) e di *Physical Demand (PD);* questi indicano un maggior successo nel compimento delle task, ma con un aumento del fattore temporale e di richiesta fisica. Dall'altro lato, il punteggio minimo è rappresentato dal parametro di *Mental Demand* (*MD*) a indicare un basso sforzo mentale nel compimento della task.
Il tempo medio su tutti gli utenti per l'esecuzione è di 11.36 min, con una varianza di 3.64 sul totale dei tempi.

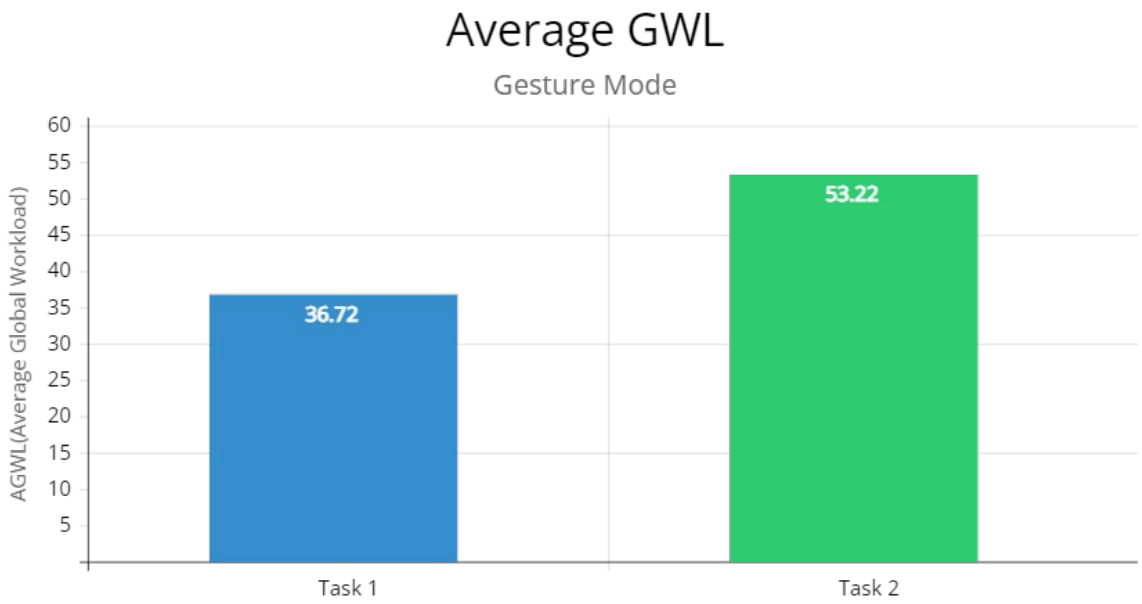


Grafico 4.7.

Il grafico 4.7 raffigura il carico di lavoro medio (*AGWL – Average Global Workload)* per entrambe le task, ricavato dai *GWL* singoli per ogni utente che sono stati calcolati come illustrato nel paragrafo 4.5.2. La prima raggiunge un carico medio di 36.72, mentre il valore della seconda è di 53.22.

4.6.3 Gesture e Neuro a confronto

USABILITÀ

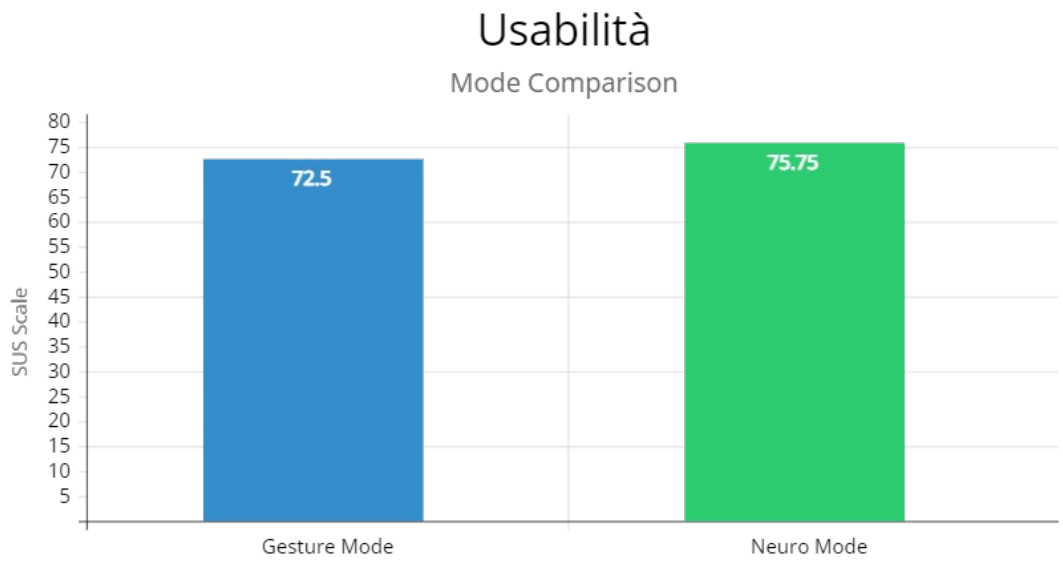


Grafico 4.8.

I dati hanno dimostrato un indice di usabilità, misurato tramite la scala SUS, maggiore per la modalità *neuro*, con un valore di 75.75; per la modalità *gesture* il valore di 72.5 resta comunque totalmente confrontabile. (Grafico 4.8)

CARICO DI LAVORO

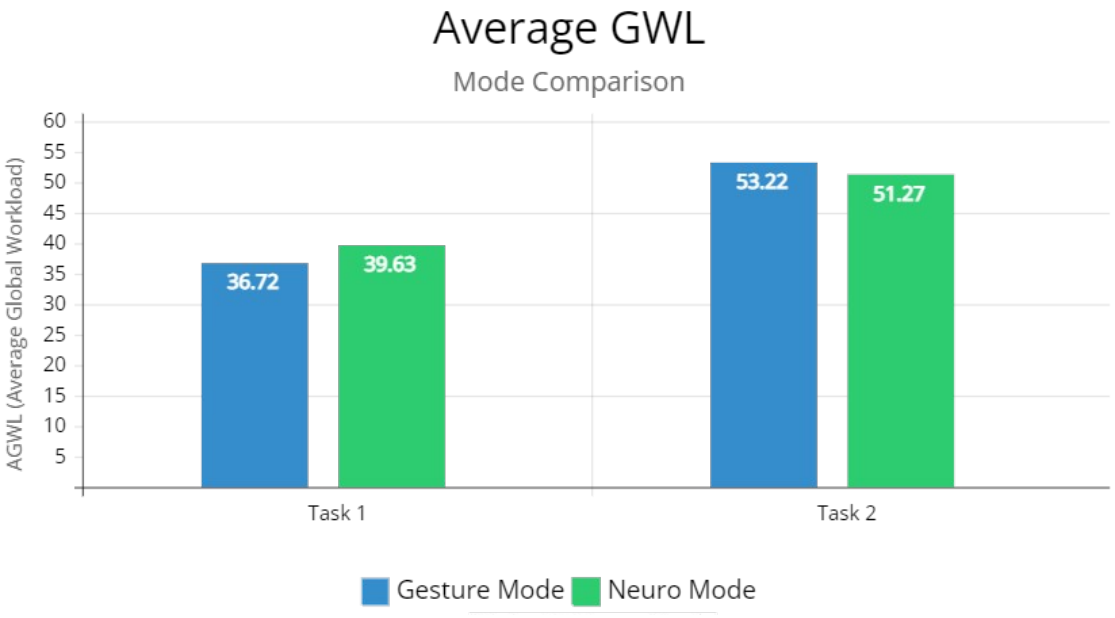


Grafico 4.9.

Il grafico 4.9 mostra come anche nel caso del carico di lavoro medio globale (AGWL), le due modalità restino totalmente confrontabili: per la Task 1, il carico di lavoro è leggermente più alto per la modalità *neuro*, con un valore di 39.63; per la Task 2, invece, è più alto il valore della modalità *gesture*, che arriva a 53.22.

TEMPI MEDI DI ESECUZIONE

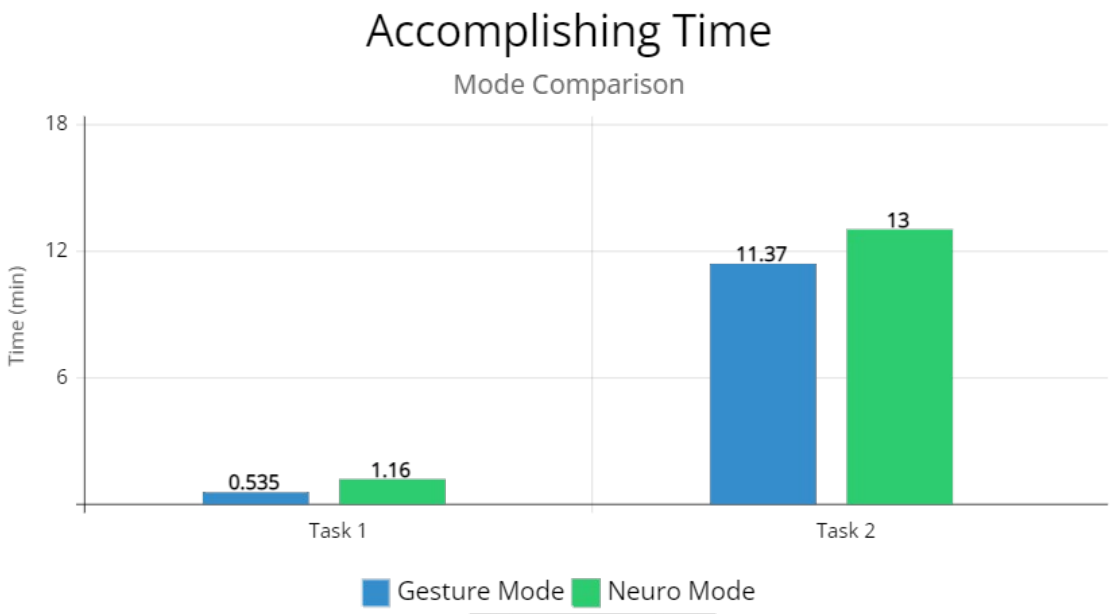


Grafico 4.9.

Come per i risultati precedenti, i tempi medi perché l'utente compia una determinata task sono molto simili. Per entrambe le task, i tempi sono leggermente minori in modalità *gesture*, con 0.535 minuti per la Task 1 e 11.37 minuti per la Task 2; in modalità *neuro*, invece, i tempi si allungano leggermente, con 1.16 minuti per la Task 1 e 13 minuti per la Task 2.

5. Conclusioni

I risultati ottenuti, nonostante i dati siano ancora relativamente pochi e la sperimentazione sia in fase iniziale, hanno dimostrato l'efficacia dell'utilizzo di una BCI in un contesto di assemblaggio.

Tutti gli utenti che hanno utilizzato la *neuro mode* come modalità operativa hanno completato le task senza particolari problemi, con una bassa percentuale di errore nella selezione dei *NeuroTag.*
Tramite i dati misurati con il SUS è stato mostrato come l'usabilità del sistema in questa modalità sia sopra la media per lo standard dell'industria. Inoltre, si è dimostrata la totale comparabilità tra *neuro mode*, che utilizza la BCI, e *gesture mode*, che utilizza la tecnologia Hololens 2; tale comparazione è stata dimostrata sia in termini di usabilità, sia in termini di carico di lavoro globale (GWL) e di tempistiche per l'esecuzione.

Le problematiche prevalenti per la modalità *neuro* dell'applicativo sono state provocate maggiormente da fattori estranei allo studio proposto, quali la sensibilità alta del NextMind in relazione ai movimenti della testa dell'utente e la calibrazione dello stesso; nella maggior parte dei casi, le questioni sono state risolte stringendo maggiormente il dispositivo attorno al capo dell'utente, per una maggior stabilità e accuratezza (anche se dalle fasi di test è stata rilevata una maggior quantità di stress in merito a questo motivo, siccome il dispositivo risultava essere scomodo da indossare dopo circa un'ora di utilizzo).

Per quanto riguarda i parametri singoli, dall'analisi tramite il NASA-TLX si è potuta evincere una maggior quantità di carico di lavoro per il fattore temporale e mentale nella modalità *neuro*; in alcuni casi, l'utente doveva rimanere concentrato per più tempo su un NeuroTag per innescarlo, oppure avvicinarsi ad esso, e ciò provocava un maggiore sforzo generale. Nonostante questo, i tempi sono rimasti comunque confrontabili con la parte *gesture*, che comunque a volte presentava problematiche dovute all'accuratezza nella manipolazione degli ologrammi.

Sicuramente, tramite alcune integrazioni future, l'applicazione potrà essere migliorata sotto diversi aspetti:

- Al momento, sia per la parte *gesture*, che per la parte *neuro* il SELECT delle componenti avviene in due passaggi. Nel primo caso, l'utente avvicina a sé il componente tramite manipolazione e successivamente clicca il pulsante di SELECT; nel secondo, l'utente innesca il NeuroTag di una *Sezione* e poi quello di SELECT del componente. Inoltre, l'utente utilizza del tempo per cercare la componente che necessita nel modello esploso.
  Una riduzione dei passaggi potrebbe sicuramente portare l'applicazione a livello successivo; anche delle dinamiche di ordinamento automatico per le componenti, di modo che l'utente possa trovare subito quella desiderata, potrebbero contribuire alla medesima causa.

- Un *feature* di NextMind permette il rilevamento della percentuale di attenzione su un determinato NeuroTag; dal lato sviluppatore, si potrebbe pensare di utilizzare questa possibilità per innescare i NeuroTag in base a una certa soglia (di *default*, la soglia è del 100% di attenzione). Sicuramente in questo caso sarebbe opportuno gestire il lato codice in maniera tale da non rilevare falsi positivi;

- Al momento, l’utente può attuare il SELECT solamente di una componente alla volta mentre il robot sta compiendo i movimenti. Per accorciare i tempi di questa operazione, potrebbe essere opportuno integrare una coda di selezione per le componenti, di modo che l’assemblatore possa attuare il SELECT su più elementi e poi il robot scorra la coda automaticamente.

Per concludere, con lo svilupparsi delle tecnologie legate alle BCI (NextMind in primis), potrebbero entrare in gioco dinamiche ancora diverse, come l’innesco automatico tramite il pensiero dell’utente di una certa azione, il monitoraggio continuo dell’attenzione generale dell’utente oppure l’integrazione dei dispositivi BCI direttamente su quelli AR/VR.

Bibliografia

[1] Lasi, H., Fettke, P., Kemper, H. G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. Business & information systems engineering, 6(4), 239-242.

[2] Gubán, M., & Kovács, G. (2017). INDUSTRY 4.0 CONCEPTION. Acta Technica Corviniensis-Bulletin of Engineering, 10(1), 111.

[3] Saurabh Vaidya, Prashant Ambad, Santosh Bhosle, Industry 4.0 – A Glimpse, Procedia Manufacturing, Volume 20, 2018, Pages 233-238, ISSN 2351-9789,

[4] G. Saravanan, Shailesh S. Parkhe, Chetan M. Thakar, Vaibhav V. Kulkarni, Hari Govind Mishra, G. Gulothungan, Implementation of IoT in production and manufacturing: An Industry 4.0 approach, Materials Today: Proceedings, 2021, ISSN 2214-7853,)

[5] Daqiang Guo, Mingxing Li, Zhongyuan Lyu, Kai Kang, Wei Wu, Ray Y. Zhong, George Q. Huang, Synchroperation in industry 4.0 manufacturing, International Journal of Production Economics, Volume 238, 2021, 108171, ISSN 0925-5273

[6] https://www.cleanpng.com/png-industry-4-0-fourth-industrial-revolution-automati-4352575

[7] Van Gerven, Marcel, et al. "The brain–computer interface cycle." Journal of neural engineering 6.4 (2009): 041001.

[8] Dual Passive Reactive Brain Computer Interface: a Novel Approach to Human-Machine Symbiosis Frédéric Dehais, Simon Ladouce, Ludovic Darmet, Nong Tran-Vu, Giuseppe Ferraro, Juan Torre Tresols, Sebastien Velut and Patrice Labedan; Artificial and Natural Intelligence Toulouse Institute, Universite de Toulouse, Toulouse, France; Department for Aerospace Vehicles Design and Control, ISAE-SUPAERO, Universite de Toulouse, Toulouse France; School of Biomedical Engineering, Science and Health Systems, Drexel University, PA, USA

[9] Fabien Lotte, Raphaëlle Roy. Brain–Computer Interface Contributions to Neuroergonomics. Neuroergonomics: The Brain at Work and in Everyday Life, Elsevier, pp.43-48, 2019. ffhal-01946095f

[10] Dal Seno, Bernardo, Matteo Matteucci, and Luca Mainardi. "Online detection of P300 and error potentials in a BCI speller." Computational intelligence and neuroscience 2010 (2010).

[11] Pasquale Arpaia, Egidio De Benedetto, Luigi Duraccio, Design, implementation, and metrological characterization of a wearable, integrated AR-BCI hands-free system for health 4.0 monitoring, Measurement, Volume 177, 2021, 109280, ISSN 0263-2241,

[12] Surjo R. Soekadar, Niels Birbaumer, Marc W. Slutzky, Leonardo G. Cohen, Brain–machine interfaces in neurorehabilitation of stroke, Neurobiology of Disease, Volume 83, 2015, Pages 172-179, ISSN 0969-9961,

[13] López-Larraz, E. et al. ‘Brain-machine Interfaces for Rehabilitation in Stroke: A Review’. 1 Jan. 2018 : 77 – 97

[14] Kyuhwa Lee, Dong Liu, Laetitia Perroud, Ricardo Chavarriaga, José del R. Millán, A brain-controlled exoskeleton with cascaded event-related desynchronization classifiers,Robotics and Autonomous Systems,Volume 90, 2017, Pages 15-23, ISSN 0921-8890

[15] B. Saduanov, T. Alizadeh, J. An and B. Abibullaev, "Trained by demonstration humanoid robot controlled via a BCI system for telepresence," 2018 6th International Conference on Brain-Computer Interface (BCI), 2018, pp. 1-4, doi: 10.1109/IWW-BCI.2018.8311508.

[16] M. Bryan et al., "An adaptive brain-computer interface for humanoid robot control," 2011 11th IEEE-RAS International Conference on Humanoid Robots, 2011, pp. 199-204, doi: 10.1109/Humanoids.2011.6100901.

[17] Kapeller Christoph, Hintermüller Christoph, Abu-Alqumsan Mohammad, Prueckl Robert, PeerPeer Angelika, Guger Christoph, 2013/01/01, “A BCI using VEP for continuous control of a mobile robot”

[18] M. Wang, R. Li, R. Zhang, G. Li and D. Zhang, "A Wearable SSVEP-Based BCI System for Quadcopter Control Using Head-Mounted Device," in IEEE Access, vol. 6, pp. 26789-26798, 2018, doi: 10.1109/ACCESS.2018.2825378.

[19] Y. Yao, B. Yang, X. Xia, Z. Peng, S. Gao and X. Meng, "Design of Upper Limb Rehabilitation Training System Combining BCI and AR Technology," 2021 40th Chinese Control Conference (CCC), 2021, pp. 7131-7134, doi: 10.23919/CCC52363.2021.9550315.

[20] Blum, Tobias, et al. "Superman-like X-ray vision: Towards brain-computer interfaces for medical augmented reality." 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). IEEE, 2012.

[21] G. Barresi, E. Olivieri, D. G. Caldwell and L. S. Mattos, "Brain-Controlled AR Feedback Design for User's Training in Surgical HRI," 2015 IEEE International Conference on Systems, Man, and Cybernetics, 2015, pp. 1116-1121, doi: 10.1109/SMC.2015.200.

[22] H. Si-Mohammed et al., "Towards BCI-Based Interfaces for Augmented Reality: Feasibility, Design and Evaluation," in IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 3, pp. 1608-1621, 1 March 2020, doi: 10.1109/TVCG.2018.2873737.

[23] Alexander Lenhardt and Helge Ritter, An Augmented-Reality Based Brain-Computer Interface for Robot Control, Bielefeld University, CITEC, Universitaetsstr. 2123, 33615 Bielefeld, Germany

[24] D. Kuhner, L.D.J. Fiederer, J. Aldinger, F. Burget, M. Völker, R.T. Schirrmeister, C. Do, J. Boedecker, B. Nebel, T. Ball, W. Burgard, A service assistant combining autonomous robotics, flexible goal formulation, and deep-learning-based brain–computer interfacing, Robotics and

Autonomous Systems, Volume 116, 2019, Pages 98-113, ISSN 0921-8890,

[25] Kerous, B., Skola, F. & Liarokapis, F. EEG-based BCI and video games: a progress report. Virtual Reality 22, 119–135 (2018). https://doi-org.ezproxy.biblio.polito.it/10.1007/s10055-017-0328-x

[26] I. Pedersen, P. Mirza-Babaei and N. Gale, "iMind: An alternative dialogue between viewers and artists," 2014 IEEE Games Media Entertainment, 2014, pp. 1-2, doi: 10.1109/GEM.2014.7048085.

[27] Rui Na, Chun Hu, Ying Sun, Shuai Wang, Shuailei Zhang, Mingzhe Han, Wenhan Yin, Jun Zhang, Xinlei Chen, Dezhi Zheng, An embedded lightweight SSVEP-BCI electric wheelchair with hybrid stimulator, Digital Signal Processing, Volume 116, 2021, 103101, ISSN 1051-2004, https://doi.org/10.1016/j.dsp.2021.103101.

[28] M. Fiolka, J. Jost and T. Kirks, "Explorative Study of a Brain-Computer Interface for Order Picking Tasks in Logistics,"2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2020, pp. 253-256, doi: 10.1109/IHMSC49165.2020.10135.

[29] M. A. Mamani and P. R. Yanyachi, "Design of computer brain interface for flight control of unmanned air vehicle using cerebral signals through headset electroencephalograph,"2017 IEEE International Conference on Aerospace and Signals (INCAS), 2017, pp. 1-4, doi: 10.1109/INCAS.2017.8123499.

[30] Badcock, Nicholas A., et al. "Validation of the Emotiv EPOC® EEG gaming system for measuring research quality auditory ERPs."PeerJ1 (2013): e38.

[31]D. Kuhner, L.D.J. Fiederer, J. Aldinger, F. Burget, M. Völker, R.T. Schirrmeister, C. Do, J. Boedecker, B. Nebel, T. Ball, W. Burgard, A service assistant combining autonomous robotics, flexible goal formulation, and deep-learning based brain–computer interfacing, Robotics and Autonomous Systems, Volume 116, 2019, Pages 98-113, ISSN 0921-8890, https://doi.org/10.1016/j.robot.2019.02.015.

[32] Zhengdong Zhou, Lingwei Zhang, Shisong Wei, Xuling Zhang, Ling Mao, Development and evaluation of BCI for operating VR flight simulator based on desktop VR equipment, Advanced Engineering Informatics, Volume 51, 2022, 101499, ISSN 1474-0346,

[33] Hakim Si-Mohammed, Ferran Argelaguet Sanz, Géry Casiez, Nicolas Roussel, Anatole Lécuyer. BrainComputer Interfaces and Augmented Reality: A State of the Art. Graz Brain-Computer Interface
Conference, Sep 2017, Graz, Austria. ff10.3217/978-3-85125-533-1-82ff. ffhal-01625167f

[34] P. Samal and R. Singla, "EEG Based Stress Level Detection During Gameplay,"2021 2nd Global Conference for Advancement in Technology (GCAT), 2021, pp. 1-4, doi: 10.1109/GCAT52182.2021.9587468.

[35] H. Arora, A. P. Agrawal and A. Choudhary, "Conceptualizing BCI and AI in Video Games,"2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2019, pp. 404-408, doi: 10.1109/ICCCIS48478.2019.8974549.

[36] Escolano C, Antelis JM, Minguez J. A telepresence mobile robot controlled with a noninvasive brain– computer interface. IEEE SMC. 42, 3 (2012), 793–804

[37] Gergondet P, Druon S, Kheddar A, Hintermüller C, Guger C, Slater M. Using brain-computer interface to steer a humanoid robot. IEEE ROBIO. (2011), 192–197.

[38] *https://docs.microsoft.com/it-it/hololens/hololens1-hardware*

[39] https://www.beniculturalionline.it/post.php?n=2492

[40] https://unity.com/solutions/digital-twin