POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Biomedica

Tesi di Laurea Magistrale

SVILUPPO DI UN SISTEMA PER LA VISUALIZZAZIONE DI INFORMAZIONI RELATIVE ALL' ATTIVAZIONE MUSCOLARE IN REALTA' AUMENTATA.

Relatori:

Prof. Marco Gazzoni
Ing. Giacinto Luigi Cerone
Ing. Alberto Botter

Candidato:

Nicolò Gentili

A.A. 2017/2018

Sommario

L'obiettivo della tesi è stato quello di sviluppare un sistema per la visualizzazione delle informazioni relative all'attivazione muscolare rilevata tramite elettromiografia di superficie in realtà aumentata.

L'elettromiografia di superfice è una tecnica che consiste nell'acquisire e studiare il segnale elettrico generato dai muscoli scheletrici e prelevato sulla superficie cutanea durante la loro contrazione. Nelle procedure riabilitative, questo segnale può essere fornito al paziente come feedback visivo o acustico, noto come biofeedback, in modo da poterlo aiutare ad acquisire il controllo delle attivazioni muscolari. L'utilizzo della realtà aumentata nella tecnica del biofeedback applicata al segnale elettromiografico di superficie, permette di fornire feedback visivi direttamente sopra i distretti muscolare d'interesse, aumentando il coinvolgimento del paziente e rendendo la tecnica più intuitiva e immediata. La realtà aumentata di fatto è una tecnica di *computer vision* che permette di inserire nella scena reale oggetti virtuali, ampliando la normale percezione umana. Il sistema pertanto è stato pensato per unire le potenzialità della realtà aumentata alla tecnica del biofeedback in modo da coinvolgere maggiormente il paziente nelle procedure riabilitative e fornire le informazioni relative alle attivazioni muscolari in forma visiva anche al fisioterapista.

Il sistema di realtà aumentata sviluppato è stato integrato all'interno di un software in grado di registrare e mostrare in tempo reale il segnale elettromiografico acquisito Il sistema può essere diviso in quattro blocchi che sono: 1) acquisizione segnale EMG, 2) acquisizione segnale video, 3) elaborazione ed infine 4) biofeedback.

Il primo blocco si occupa dell'acquisizione del segnale EMG, che dopo essere stato prelevato viene reso disponibile al blocco di processing. Il segnale EMG può essere acquisito con la tecnica bipolare classica o con tecniche di EMG ad alta densità (HD-EMG) nel caso in cui si sia interessati alla distribuzione spaziale delle attivazioni muscolari.

Il secondo blocco si occupa dell'acquisizione dei frame video. Il terzo blocco riceve il segnale EMG e il segnale video rispettivamente dagli altri due e procede all'elaborazione. Il segnale EMG viene processato al fine di estrarre un'informazione di ampiezza, come valore rettificato medio o valore efficace (average rectified value-ARV o root mean square RMS), che viene tradotta in un codice colore indicativo dell'entità della contrazione muscolare registrata. Ogni frame video viene processato per individuare nella scena la posizione di particolari marker, detti "fiducial marker", che vengono posti sopra agli elettrodi di prelievo. I fiducial marker sono individuati all'interno della scena mediante un algoritmo di riconoscimento marker e rappresentano il riferimento su cui poi verrà applicato il feedback visivo. Per il riconoscimento dei marker si è utilizzata ArUco, una libreria OpenSouce estremamente veloce, scritta completamente in C++, che permette di riconoscere i marker presenti nell'ambiente e di stimare la loro posizione e orientamento. L'ultimo blocco combina i risultati dell'elaborazione per generare il biofeedback. Il frame video acquisito viene modificato andando ad aggiungere l'oggetto virtuale all'interno della scena. Sovrapposto ai marker associati alla sonda di tipo bipolare, viene mostrato un poligono colorato, mentre per un sistema HD-sEMG viene mostrata una mappa colore. Una volta generato il nuovo frame si procede al repainting, ottenendo così la visualizzazione dell'attivazione muscolare.

Al fine di verificare la funzionalità del sistema sono state eseguite alcune prove sperimentali. In particolare l'esercizio di *leg extension* e il *curl concentrato manubrio singolo*.

INDICE

1. Introduzione

L'obiettivo della tesi è stato quello di sviluppare un sistema di realtà aumentata che permettesse di visualizzare informazioni relative all'attivazione muscolare rilevata tramite elettromiografia di superficie (*surface electromyography*-sEMG). In particolare ci siamo proposti di sviluppare un sistema multipiattaforma, con particolare attenzione alla portabilità su device quali *smartglasses* e dispositivi mobili in generale.

L'elettromiografia di superficie è una tecnica che permette di acquisire e studiare il segnale elettrico generato dai muscoli scheletrici durante la loro contrazione. Il segnale che viene acquisito e studiato prende il nome di segnale elettromiografico (sEMG) ed è una rappresentazione del potenziale elettrico generato dalla depolarizzazione della membrana esterna della fibra muscolare [20].

Le informazioni relative all'attività elettrica dei muscoli durante la loro contrazione è di particolare importanza in ambito clinico, in quanto permette di studiare e valutare diverse condizioni che coinvolgono il sistema neuromuscolare. Queste informazioni possono essere utilizzate: per effettuare analisi del movimento e del cammino, studi in neurologia, valutazione clinica dei tempi di attivazione muscolare e nella tecnica del biofeedback [22].

In particolare, la tecnica del biofeedback consiste nel misurare segnali fisiologici e mostrarli al paziente in tempo reale in una forma comprensibile (audio o video) al fine di aiutarlo ad acquisire il controllo dei processi fisiologici stessi [18]. La tecnica del biofeedback applicata al segnale EMG, consiste nel misurare l'attività muscolare di uno o più distretti muscolari e fornirla al paziente sullo schermo di un PC in molteplici forme (segnale nel tempo, barre proporzionali all'ampiezza del segnale).

Il sistema che viene proposto in questo lavoro intende fornire al medico, fisioterapista e paziente, le informazioni, in tempo reale, sull'attività muscolare sotto forma di realtà

aumentata, permettendo di fornire il feedback visivo direttamente sopra la zona anatomica d'interesse, aumentando la coerenza fra il segnale fisiologico misurato e il biofeedback.

La realtà aumentata è una tecnologia di *computer vision*, che permette di inserire nella scena reale oggetti virtuali, in tempo reale, permettendo di ampliare la percezione sensoriale umana. La realtà aumentata in ambito medico ha avuto successo in diverse applicazioni quali: formazione del personale sanitario [12], applicazioni chirurgiche [13] e riabilitazione [15] [16] [17].

A nostra conoscenza, non esistono sistemi per la visualizzazione di informazioni relative all'attivazione muscolare in realtà aumentata. Per questo motivo, abbiamo iniziato a studiare la possibilità di utilizzare questa tecnologia di *computer vision* al fine di:

1. Aumentare l'efficacia del biofeedback EMG;
2. Fornire al fisioterapista un nuovo strumento che mostri le informazioni relative all'attività muscolare direttamente sull'arto del paziente.

Il sistema di realtà aumentata (AR) sviluppato è stato integrato come *plugin* (estensione delle funzionalità originarie di un software) in bluPlot, un software in grado di registrare e mostrare in tempo reale il segnale EMG acquisito. Il sistema bluePlot + AR può essere diviso in quattro blocchi:

1. blocco di acquisizione del segnale EMG;
2. blocco di acquisizione del segnale video;
3. blocco di elaborazione dei segnali;
4. biofeedback.

Il primo blocco si occupa dell'acquisizione del segnale EMG, che dopo essere stato prelevato viene reso disponibile al blocco di elaborazione dei segnali. Il prelievo del segnale EMG viene effettuato grazie ad un sistema di acquisizione sviluppato all'interno del LISiN (*Laboratorio di Ingegneria del Sistema Neuromuscolare e della riabilitazione motoria*), il DUEPro. Tale sistema permette di acquisire segnali elettromiografici bipolari che vengono inviati a bluePlot. Oltre alla possibilità di

acquisire segnali tramite il Due, è anche possibile acquisire segnali da matrici di elettrodi tramite il dispositivo MEACS, che conta 32 canali. In questo modo è possibile fornire una mappa colore opportunamente calcolata, la quale fornisce indicazioni sulla distribuzione spaziale delle attivazioni muscolari.

Il secondo blocco si occupa della gestione del flusso video: aperta la connessione con la camera presente nel dispositivo in uso, questo riceve i video frame, ne controlla il formato e la risoluzione e li invia al terzo blocco.

Il terzo blocco riceve il segnale EMG e il segnale video dai due blocchi precedenti e procede all'elaborazione. Il segnale EMG viene processato al fine di estrarre un'informazione d'ampiezza, come valore rettificato medio o valore efficace (average rectified value-ARV o root mean square RMS) e trasformato in un codice colore indicativo dell'entità della contrazione muscolare registrata. Il codice colore va dal blu, indicativo del muscolo a riposo, al rosso, indicativo invece della contrazione massimale del muscolo. Per indicare contrazioni intermedie non massimali vengono utilizzate le sfumature dal rosso al blu. Il segnale video viene processato per individuare nella scena la posizione di particolari marker, detti *fiducial marker*, che vengono posti sopra agli elettrodi per il prelievo del segnale EMG. Questi *fiducial marker* sono individuati agevolmente all'interno della scena mediante un algoritmo di riconoscimento dei marker e rappresentano il riferimento su cui poi verrà applicato il feedback visivo.

L'ultimo blocco combina i risultati dell'elaborazione dei segnali per generare il biofeedback: il video frame acquisito viene modificato andando ad aggiungere l'oggetto virtuale all'interno della scena: se sovrapposto ai marker associati alla sonda di tipo Due, viene mostrato un poligono colorato, mentre per il Meacs viene mostrata una mappa colore. Una volta generato il nuovo frame si procede al *repainting*, ottenendo così la visualizzazione dell'attivazione muscolare.

Per il riconoscimento dei marker si è utilizzata ArUco, una libreria OpenSouce estremamente veloce, scritta completamente in C++, che permette di riconoscere i marker presenti nell'ambiente e di calcolare la loro posizione in modo quanto più possibile affidabile. Tale libreria inoltre si appoggia ad OpenCV (*Open Source Computer Vision Library*), una libreria open source per computer grafica ampiamente usata, con

una comunità da più di 47 mila utenti ed un numero di download che supera i 14 milioni.

La libreria ArUco fa parte di OpenCV, ma non viene fornita nella release ufficiale di quest'ultima. Infatti cercando la documentazione e consultando i vari moduli, è possibile trovarla sotto la voce *Extra modules*. Pertanto per poter importare ed utilizzare ArUco all'interno dell'ambiente di sviluppo usato (Qt), è stato necessario ricompilare il tutto. La compilazione della libreria è stata fatta usando CMake, una famiglia di strumenti open source e multipiattaforma progettati per creare, testare e "confezionare" software.

Nelle seguenti sezioni, verranno fornite alcune nozioni necessarie per contestualizzare il discorso e per comprendere al meglio le motivazioni che hanno portato alla realizzazione di tale sistema.

1.1 Stato dell'Arte: la realtà aumentata in campo medico

La realtà aumentata (*Augmented Reality* - AR) è una tecnologia emergente che permette di aggiungere nell'ambiente reale oggetti virtuali e di interagire con essi. AR differisce dalla realtà virtuale (VR) in quanto, in quest'ultima, l'utente viene immerso completamente in un mondo virtuale e l'interazione con gli elementi avviene nel mondo virtuale. Di fatto, per realizzare applicazioni AR si utilizza sempre la scena che rappresenta il mondo reale, mentre per quelle VR la scena viene creata completamente in maniera artificiale.

AR può essere pensata come un ampliamento della percezione sensoriale umana, dato che normalmente, le informazioni aggiunte nell'ambiente reale non sarebbero percepite con i cinque sensi. La realtà aumentata quindi, può essere utilizzata per una visione diretta o indiretta di un ambiente fisico reale, i cui elementi vengono amplificati da input sensoriali, generati artificialmente, come suoni o immagini.

L’aggiunta di questi elementi virtuali direttamente all’interno del campo di percezione dell’utente, lo aiutano ad eseguire determinate operazioni.

Figura 1.1: Esempi di applicazioni AR comuni.

L’interesse per la realtà aumentata in ambiente medico nasce dalla necessità di visualizzare i parametri clinici del paziente ed il paziente stesso, all’interno dello stesso spazio fisico [11].

Grazie al recente sviluppo tecnologico in ambito AR, sono stati proposti sistemi per applicazioni mediche utilizzabili in ambiti diversi del campo sanitario, come:

- Formazione del personale sanitario;
- Applicazioni chirurgiche;
- Riabilitazione.

Formazione del personale sanitario

La realtà aumentata offre importanti opportunità nel campo dell’educazione del personale sanitario. Grazie alla capacità di aggiungere elementi virtuali nella scena reale e di interagire con essi, l’AR è in grado di simulare situazioni reali più rilevanti, dando la possibilità, a chi sta facendo pratica, di analizzare il problema e risolverlo in un ambiente sicuro e protetto [12].

Barsom et al. [12] hanno effettuato una *review* sull’utilizzo di sistemi AR per l’educazione del personale sanitario, individuato tre categorie differenti quali: chirurgia laparoscopica, neurochirurgia e ecocardiografia. Hanno inoltre valutato

l'efficacia di sette applicazioni, selezionate fra le varie categorie, che sono state individuate come le più rilevanti.

Tutti i sistemi indagati, si sono rivelati utili ed efficienti, andando a migliorare le abilità degli operatori. In particolare, per la valutazione delle procedure chirurgiche, sono stati considerati il tempo di esecuzione dell'intervento, la lunghezza del percorso effettuato per raggiungere il sito interessato, la fluidità dei movimenti e la quantità di tessuto danneggiato. Questi parametri possono essere valutati in maniera oggettiva e quantitativa e quindi utilizzati per effettuare la valutazione dei sistemi. Per quanto riguarda invece le applicazioni in ecocardiografia, l'alto grado di realismo fornito dai sistemi ha portato ad un miglioramento delle capacità diagnostiche degli operatori.

Nella figura sottostante vengono riportati tre dei sistemi analizzati da Barsom et al. uno per ogni categoria.

Figura 1.2: Esempi di applicazioni di realtà aumentata presentati nel lavoro di Barsom et al. [12]

Applicazioni chirurgiche

Recentemente sono stati introdotti sistemi AR anche in applicazioni chirurgiche. In particolare l'utilizzo sempre maggiore di tecniche mini-invasive come quelle in laparoscopia, pone il problema della visualizzazione degli organi coinvolti nell'intervento. Di fatto, con queste tecniche, il chirurgo perde la possibilità di vedere

direttamente i tessuti interessati, i quali invece vengono mostrati grazie all'utilizzo di una camera e di uno schermo. L'integrazione della realtà aumentata in queste procedure è stata al centro di molte ricerche negli ultimi anni, in quanto, l'utilizzo di sistemi AR, può migliorare l'esperienza visiva del chirurgo aggiungendo nella scena elementi 3D estratti da immagini intraoperatorie o preoperatorie [13].

Uno dei problemi principali riguarda la localizzazione accurata dell'endoscopio, grazie alla quale è possibile sovrapporre coerentemente l'immagine reale e quella virtuale. Bernhardt et al. [13] hanno proposto un metodo innovativo per ovviare al problema, che permette di localizzare automaticamente l'endoscopio senza dispositivi esterni di *tracking* o senza analizzare immagini endoscopiche. Questa tecnica utilizza un sistema CT per *imaging*. Dato che l'endoscopio è composto da titanio, la sua superficie introduce un artefatto nelle acquisizioni, che può essere individuato valutando l'intensità dei voxel (unità volumetrica dell'immagine). Inoltre, sapendo che la forma è tubulare è possibile recuperare la posizione dell'endoscopio sovrapponendo un modello 3D cilindrico sulla sua superficie. Gli autori inoltre suppongono, che l'asse ottico e quello del cilindro corrispondano. Questa assunzione introduce un errore medio più piccolo di 15 pixel (circa 1mm) che riportano come accettabile per gli interventi laparoscopici. L'angolo di rollio viene determinato mediante l'utilizzo di accelerometri solidali con il sensore CCD.

È importante notare che la relazione trovata vale solamente fintanto che la camera rimane immobile. Applicazioni AR di questo tipo possono essere usate per individuare vasi nascosti, evidenziare parti importanti come vene principali o fungere da guida durante l'ablazione termica.

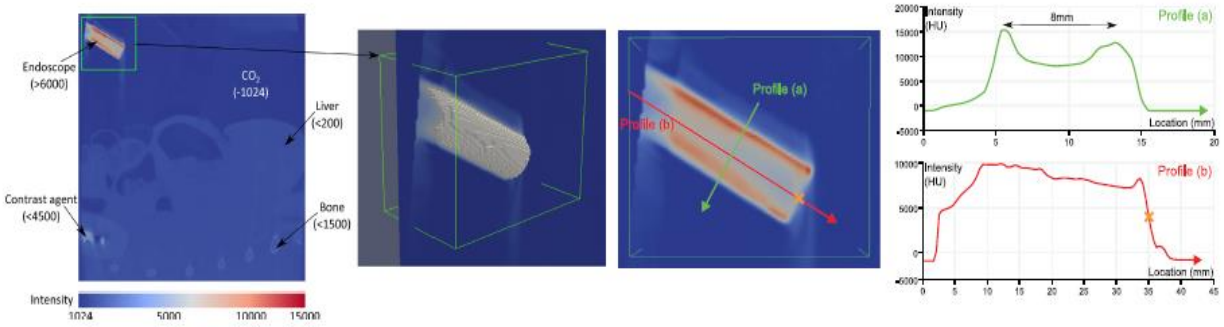


Figura 1.3: La prima immagine rappresenta una fetta di una scansione CT in cui si individua l'endoscopio e l'anatomia di un maiale. L'intensità dei voxel è rappresentata nelle diverse regioni e si nota come i voxel rappresentanti l'endoscopio abbiano un alto valore di contrasto con ogni altra parte del volume. Pertanto risulta facile estrarre la porzione di volume contenente l'endoscopio e procedere alla ricerca della posizione. La seconda immagine rappresenta il volume estratto. La terza e la quarta immagine rappresentano il profilo di intensità (espresso in unità Hounsfield HU): (in verde) lungo la sezione dell'endoscopio, (in rosso) lungo l'asse dell'endoscopio. Si noti come il profilo di intensità rosso cali drasticamente sulla punta dell'endoscopio [13].

Riabilitazione

La riabilitazione è il processo di allenamento mediante il quale si cerca di migliorare o recuperare le funzioni perse a causa di infortuni o malattie [19]. Il problema principale, di questa branca della medicina, è che ogni paziente presenta delle specifiche esigenze e questo porta alla necessità di essere seguito individualmente da parte del personale medico specializzato. Attualmente, per far fronte al problema, si sta cercando di sviluppare sistemi per riabilitazione, a basso costo e che richiedano una bassa supervisione da parte dei fisioterapisti. Alcuni sistemi proposti sono basati sulle tecnologie AR e VR che permettono di integrare esercizi riabilitativi a giochi, rendendo così l'approccio più divertente e il paziente più motivato. Le applicazioni AR si prestano meglio ad essere utilizzate in ambito riabilitativo in quanto più adattabili e regolabili rispetto a quelle VR [14].

Dal Jae Im et al. [15] hanno sviluppato un sistema AR per migliorare le funzionalità degli arti inferiori e l'equilibrio, nelle persone anziane. Le cadute negli anziani hanno un'incidenza molto alta e le lesioni che ne derivano possono essere anche serie. Questo è dovuto principalmente a ritardi motori, diminuzione della potenza muscolare e

diminuzione dell'equilibrio. Il metodo migliore per diminuire la percentuale di rischio di caduta nei soggetti anziani è l'esercizio fisico per gli arti inferiori mirato a migliorare la forza, equilibrio e propriocezione.

In particolare, in questo lavoro, è stato sviluppato un programma di allenamento usando una Kinect Xbox 360 per individuare e tracciare i movimenti del soggetto e le interazioni con gli oggetti virtuali. I pazienti sono stati sottoposti a tre diversi esercizi *balloon game*, *cave game* e *rhythm game*, pensati per migliora la forza degli arti inferiori, l'equilibrio e la propriocezione.

Figura 1.4: Nella figura (A) il soggetto è impegnato con il balloon game, nelle figure (B) e (C) è impegnato con il cave game e nella figura (D) con il rhythm game [15].

Heyn et al. [16] hanno sviluppato un sistema di *mixed-reality* (MR) per studiare gli effetti di questa tecnologia nei programmi di riabilitazione per soggetti che presentano lesioni al midollo spinale (SCI) o disabilità intellettive e dello sviluppo (IDD).

Questi soggetti non sono in grado di eseguire gli esercizi standard a causa della loro condizione e richiedono una strumentazione apposita. L'attrezzatura utilizzata è formata da un *tapis roulant*, *cyclette* e un ergometro per la parte superiore del braccio.

Durante la sessione di allenamento AR viene mostrato, su appositi schermi, un video in prima persona di una passeggiata lungo una pista ciclabile in Colorado. La maggior parte dei partecipanti ha mostrato un livello di coinvolgimento maggiore rispetto alla seduta di allenamento senza AR.

Ortiz-Catalan et al. [17] hanno proposto un metodo innovativo per il trattamento del *phantom limb pain* (PLP), dolore generato dall'arto fantasma, che si basa sull'utilizzo della realtà aumentata. Il PLP è una condizione altamente invalidante che colpisce il 70-80% dei soggetti amputati. Consiste nel percepire ancora l'arto amputato ed avvertire una sensazione dolorosa provenire da questo. Il dolore percepito, in alcuni soggetti, raggiunge intensità molto alte, quasi insopportabili, anche diverse volte nell'arco di una giornata compromettendo seriamente la qualità della vita di chi ne è affetto e delle persone a lui vicine. Le terapie standard, che si utilizzano per trattare questa condizione, sono tutte varianti della *mirror therapy*. Questa terapia consiste solitamente nel nascondere l'arto amputato del paziente dietro uno specchio, che riflette invece l'arto sano dando la percezione al paziente di osservare l'arto invalidato, ancora presente e sano. Inoltre i movimenti vengono percepiti come se fossero propri dell'arto amputato e questo sembra alleviare il dolore percepito. Purtroppo lo scarso realismo della tecnica fa sì che, l'efficacia del trattamento, sia altamente variabile.

Gli autori hanno sviluppato un sistema AR che offre un maggior grado di realismo ed inoltre considera gli sforzi fatti dal soggetto nel cercare di muovere l'arto fantasma, prelevando il segnale EMG direttamente dalla muscolatura residua del moncone. Il segnale EMG prelevato viene usato per muovere un arto virtuale mostrato in un apposito schermo. In particolare, il soggetto viene ripreso da una camera e grazie alla presenza di un marker, si è in grado di sovrapporre l'arto virtuale sul moncone, dando l'impressione al paziente di avere ancora il braccio. Per migliorare l'esperienza della seduta di riabilitazione, è stato sviluppato anche un gioco in cui, mediante specifici movimenti, il soggetto si trova a dover pilotare una macchina. Il sistema è stato testato su un unico paziente, refrattario alle terapie convenzionali da più di quarant'anni, che ha riportato un'importante diminuzione del dolore sperimentato.

Figura 1.5: In (A) abbiamo gli elettrodi di superficie e il marker posizionati sul moncone. In (B) si può osservare l'ambiente catturato dalla camera e mostrato sullo schermo del PC con l'aggiunta del braccio virtuale nella posizione indicata dal marker. In (C) il paziente sta pilotando una macchina. In (D) il paziente sta eseguendo dei movimenti indicati dal braccio nello schermo [17].

1.2 Biofeedback

Il *biofeedback* è una tecnica che consiste nel misurare segnali fisiologici e mostrarli al paziente in real-time in una forma comprensibile, solitamente audio o video, al fine di aiutarlo ad acquisire il controllo dei processi fisiologici stessi [18]. Il soggetto cerca così di modificare il segnale di feedback andando a modificare la risposta fisiologica. Inoltre, un dispositivo utilizzato per fornire il biofeedback permette di visualizzare o ascoltare le funzioni fisiologiche che normalmente non possono essere percepite, come ad esempio il grado di attivazione muscolare.

Le applicazioni cliniche del biofeedback includono il monitoraggio dell'attività elettrica cerebrale, pressione sanguigna, frequenza cardiaca, temperatura della pelle e l'attività elettrica muscolare rilevata in superficie [19]. In generale, per fornire il biofeedback è necessario anzitutto prelevare il segnale di interesse. Normalmente per questa tecnica si utilizzano sensori non invasivi, i quali permettono di andare ad individuare il segnale biomeccanico o elettrofisiologico cercato. Una volta acquisito, il segnale viene elaborato per poter estrarre le informazioni utili (ad esempio parametri) che infine vengono convertite in una forma più comprensibile rappresentate il feedback. Solitamente i segnali acquisiti vengono convertiti in segnali visivi o audio che sono facilmente interpretabili, ma possono essere utilizzati anche per produrre determinati effetti come la movimentazione di una macchinina in un ambiente virtuale [17] oppure nell'ambiente reale [18].

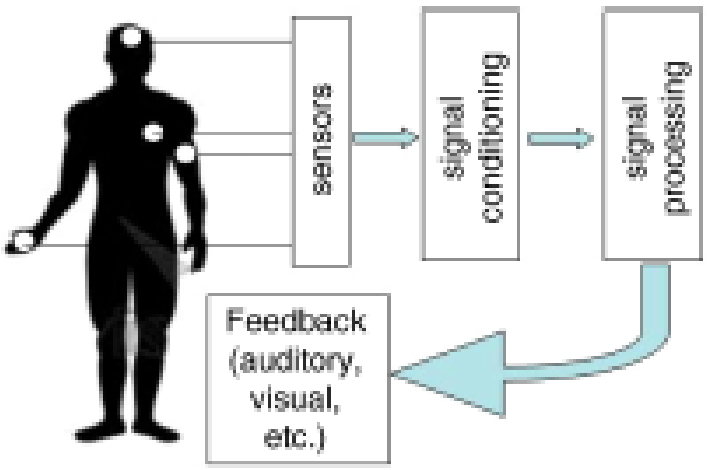


Figura 1.6: Diagramma di flusso rappresentante di un sistema di biofeedback generico [18].

È stato dimostrato [18] [19] che l'utilizzo del biofeedback, da solo o in combinazione con altre terapie, è efficace per trattare disordini come ipertensione, incontinenza, molte condizioni muscolo-scheletriche e nella riabilitazione post incidente cardiovascolare (*cardiovascular accident* - CVA) come ad esempio attacco cardiaco o infarto miocardico.

La tecnica del biofeedback applicata al segnale sEMG è la forma di feedback più comunemente utilizzata ed ampiamente investigata in ambito riabilitativo [18] [19]. La tecnica consiste nel fornire al soggetto informazioni relative all'attività muscolare e le applicazioni più comuni sono la riduzione dell'attivazione muscolare, nei casi in cui risulta eccessiva o inappropriata come la spasticità, e acquisire o riacquisire la capacità di controllare la muscolatura scheletrica. Il biofeedback viene fornito solitamente mediante segnali acustici o stimoli visivi. Ad esempio quando l'intensità del segnale EMG supera una certa soglia, si attiva un segnale acustico che deve essere ridotto cercando di modificare il segnale prodotto dall'attivazione muscolare. In altri casi invece viene mostrato il segnale EMG trasformato in colore, indicativo dell'intensità della contrazione, che deve essere modificato in base al tipo di esercizio che il soggetto sta svolgendo.

1.3 Elettromiografia

L'elettromiografia è una tecnica che permette di studiare i segnali elettrici generati dai muscoli scheletrici. Tali segnali inoltre vengono rilevati sulla superficie cutanea [20]. I segnali elettrici prelevati prendono il nome di segnali elettromiografici (EMG) e sono una rappresentazione del potenziale elettrico generato dalla depolarizzazione della membrana esterna della fibra muscolare. In particolare il segnale EMG è generato dall'attività elettrica delle fibre muscolari che si attivano durante la contrazione [20]. L'elettromiografia è un valido strumento utilizzato per studiare il movimento, valutare i meccanismi che coinvolgono la fisiologia neuromuscolare e diagnosticare i disturbi neuromuscolari [21].

I campi di applicazione dell'elettromiografia sono numerosi, fra cui citiamo quelli dalla neurologia, ergonomia, analisi del movimento e del cammino e infine biofeedback in riabilitazione [22].

In neurologia, il segnale EMG trova applicazione nella valutazione di soggetti che presentano malattie del sistema nervoso centrale (SNC). Queste malattie spesso sono accompagnate da cambiamenti delle abilità motorie [22]. In particolare possono verificarsi paresi (debolezza) o paralisi (perdita totale della capacità di contrazione del muscolo a piacimento), ma anche a riflessi e tono muscolare potenziati (spasticità). Quest'ultimi sono dovuti alla perdita della capacita di inibizione del circuito spinale locale in seguito all'interruzione delle connessioni nervose centrali [22].

In ergonomia, il segnale EMG viene utilizzato per valutare gli effetti, dell'applicazione di carichi ciclici, sul sistema muscolo-scheletrico dei lavoratori. Esistono diversi disturbi muscolo-scheletrici correlati al lavoro, che interessano varie strutture anatomiche come il sistema mano-braccio (sindrome del tunnel carpale, epicondilite), spalla e collo [22]. L'elettromiografia permette di quantificare il coinvolgimento muscolare di determinate regioni anatomiche in determinate task motorie studiarne gli effetti [22].

Nell'analisi del movimento e del cammino, il segnale EMG trova importanti applicazioni ed è stato utilizzato anche prima dell'introduzione delle moderne tecniche opto-elettroniche. In particolare sono stati svolti molti studi atti a determinare la sequenza temporale delle attivazioni muscolari durante il ciclo del cammino, sia in soggetti sani che in soggetti patologici. Altri studi invece hanno investigato le caratteristiche del segnale elettromiografico in determinate patologie come l'emiplegia (paralisi della metà destra o sinistra del corpo in seguito a lesioni organiche) [22].

Per quanto riguarda l'ambito riabilitativo, il segnale EMG viene fornito come feedback visivo al paziente. Per le applicazioni e i principi si rimanda al paragrafo sul biofeedback.

Vengono riportati qui di seguito le due grandezze utilizzate in questo lavoro per ricavare le informazioni relative all' ampiezza del segnale EMG di superficie, che sono il valore rettificato medio (*average rectified value*-ARV) e il valore efficace (*root mean square* RMS).

L'ARV è definito nel seguente modo [21]:

$$ARV = \frac{1}{n}\sum_{i=0}^{n}|s(i)|$$

Dove:

- *n =* numero di campioni totali contenuti nell'epoca
- *s(i) =* rappresenta il valore del campione *i-esimo*

rappresenta la media del valore assoluto del segnale EMG in un determinato periodo temporale (epoca).

L'RMS è definito nel seguente modo [21]:

$$RMS = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(s\ (i))^2}$$

Dove:

- *n =* numero di campioni totali contenuti nell'epoca
- *s(i) =* rappresenta il valore del campione *i-esimo*

rappresenta la radice quadrata del valor medio del quadrato del segnale EMG calcolato in un determinato intervallo temporale (epoca).

1.4 Obiettivo del lavoro

Quanto detto fino ad ora, suggerisce l'importanza del biofeedback nelle procedure di riabilitazione: il biofeedback permette di avere un riscontro su come il sistema elettrofisiologico o biomeccanico sta rispondendo al trattamento fisioterapico, ad esempio se la muscolatura si sta rilassando o meno, e in particolare dà la possibilità al paziente stesso di acquisire la capacità di controllare il sistema monitorato, aumentando l'efficacia del trattamento.

Riconosciute le potenzialità della realtà aumentata in ambito clinico e non essendo presente nessun sistema che permetta di visualizzare le informazioni relative all'attivazione muscolare in realtà aumentata direttamente sopra il distretto muscolare d'interesse, viene proposto un sistema innovativo che sfrutta tale tecnologia e permette di realizzare quanto proposto.

In particolare l'obiettivo è quello di aumentare l'efficacia della tecnica del biofeedback in EMG, coinvolgendo maggiormente il paziente nelle procedure riabilitative, fornendo un feedback visivo più intuitivo e immediato. Inoltre, fornire al fisioterapista un nuovo strumento che mostri le informazioni relative all'attività muscolare direttamente sull'arto del paziente in modo da poter monitorare l'attività muscolare durante la procedura riabilitativa e valutare se l'esercizio venga svolto correttamente o il trattamento sia efficace.

2. Progetto e sviluppo del sistema

In questo capitolo viene riportato il lavoro di progettazione e realizzazione del sistema.

Anzitutto viene mostrata l'architettura del sistema e messi in evidenza i vari componenti (Sistema di prelievo del segnale sEMG, marker sovrapposti agli elettrodi o alle matrici di elettrodi, sistema di ripresa video, unità di elaborazione e sistema di visualizzazione). Poi si passa a definire l'architettura del software mettendo in evidenza i quattro macro blocchi funzionali che lo costituiscono (Acquisizione segnale EMG, acquisizione segnale video, elaborazione dei segnali e biofeedback).

Successivamente i quattro macro blocchi vengono descritti singolarmente in dettaglio: dopo aver illustrato i passi principali seguiti all'interno di ciascun blocco funzionale, vengono definite le specifiche tecniche di progetto richieste per ogni blocco e descritte le soluzioni adottate al fine di rispettarle.

Viene inoltre approfondita l'importanza dei marker nelle applicazioni di AR, fornendo alcuni esempi di sistemi di marker, ovvero un gruppo di marker e di un algoritmo in grado di riconoscerli, proposti in letteratura e viene approfondito il processo di riconoscimento marker utilizzato dalla libreria ArUco, riportando i passi principali che vengono seguiti per andare a ricavare le coordinate degli angoli nell'immagine.

Infine vengono proposte le classi implementate per realizzare il sistema di realtà aumentata.

2.1 Architettura del sistema

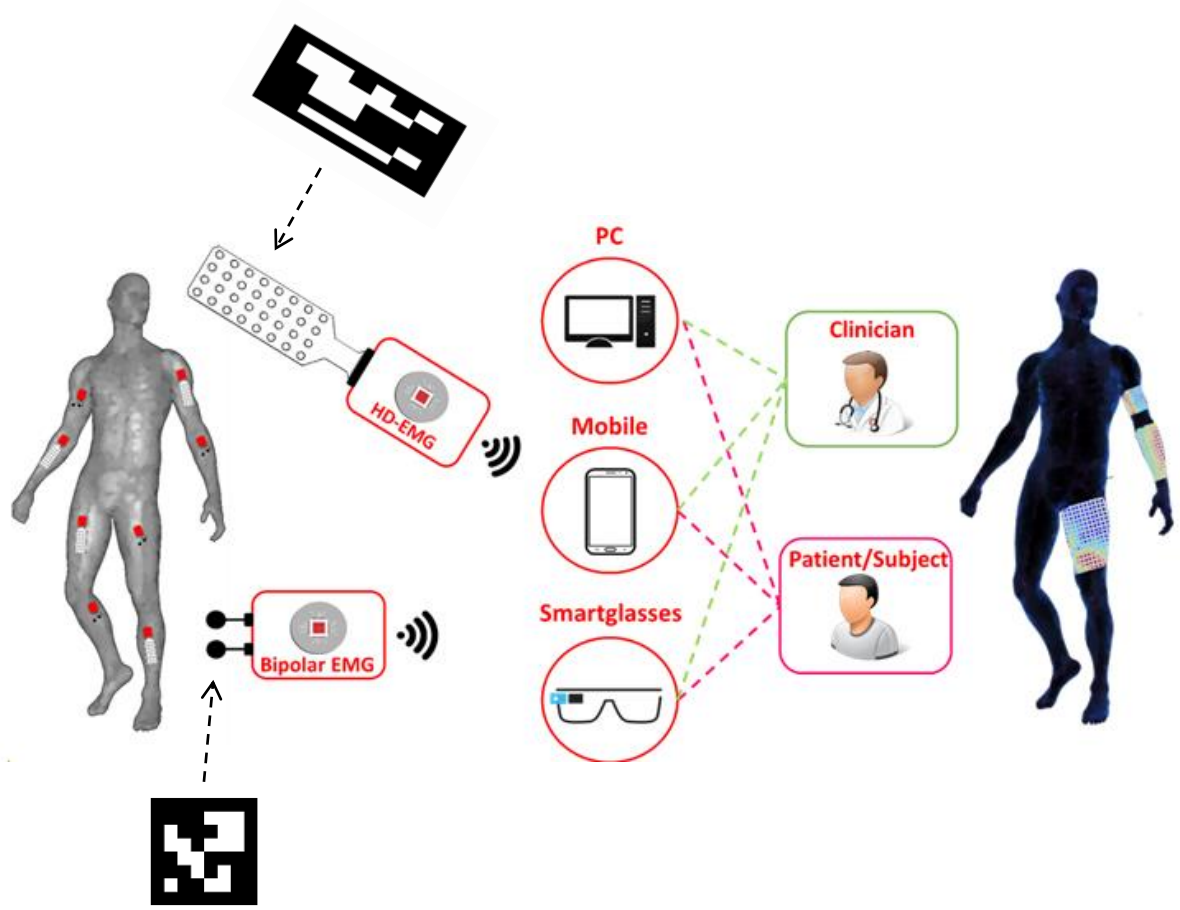


Figura 2.1: Architettura del sistema. Da sinistra verso destra i componenti sono: Sistema di prelievo del segnale sEMG, marker sovrapposti agli elettrodi o alle matrici di elettrodi, sistema di ripresa video, unità di elaborazione e sistema di visualizzazione.

In figura 2.1 viene mostrata l'architettura del sistema sviluppato.

L'architettura del sistema prevede:

1. Sistema di acquisizione segnale sEMG.

   Il segnale EMG può essere acquisito con la tecnica bipolare classica o con tecniche di EMG ad alta densità (HD-EMG) nel caso in cui si sia interessati alla distribuzione spaziale delle attivazioni muscolari. I sistemi utilizzati sono il DuePro (tecnica bipolare classica) e il Meacs (HD-EMG) che sono riportati in figura 2.2.

Il DuePro è uno strumento di prelievo modulare formato da 8 sonde wireless di cui 7 sono identiche fra loro e progettate per prelevare il segnale EMG di superficie, mentre una, detta DUE-Bio, è utilizzata per il prelievo di grandezze biomeccaniche. Tutte le 8 sonde dispongono di due canali che permettono l'acquisizione di segnali bipolari.

Il Meacs è una sonda wireless progettata per acquisire segnale HD-EMG di superficie utilizzando matrici di elettrodi. Le matrici sono formate da 32 elettrodi e permettono di prelevare segnali monopolari.

2. Marker sovrapposti agli elettrodi o alle matrici di elettrodi (figura 2.2).

   I marker, propriamente detti *fiducial marker*, sono dei punti di riferimento artificiali che vengono aggiunti nell'ambiente per facilitare il riconoscimento di oggetti specifici, oppure per effettuare la stima della posa [3]. Per stima della posa si intende definire la trasformazione lineare che permette di passare da un determinato sistema di riferimento, ad esempio quello del marker, ad un altro, ad esempio quello della camera. In particolare, grazie alle caratteristiche dei marker e quindi alla facilità di essere riconosciuti dall'algoritmo di riconoscimento marker utilizzato, si è in grado di trovare agevolmente la posizione degli elettrodi o delle matrici all'interno della scena. Esempi di *fiducial marker* sono riportati in figura 2.9 mentre quelli da noi utilizzati in figura 2.1 2.2 e 2.10.

3. Sistema di ripresa video.

   Il video frame che rappresenta la scena viene acquisito dalle camere integrate nei dispositivi (camera integrata negli smartglasses, camera integrata negli smartphone, webcam integrata nel PC) o webcam esterne collegate al PC.

4. Unità di elaborazione

   L'unità di elaborazione permette di effettuare l'elaborazione del segnale EMG di superficie e del segnale video che vengono acquisiti dai rispettivi sistemi di prelievo. Inoltre permette di creare la scena arricchita dagli elementi virtuali costituenti il biofeedback fornito al paziente. Il sistema creato è multipiattaforma pertanto in grado di funzionare con PC *smartphone* e *smartglasses*.

5. Sistema di visualizzazione

   I sistemi di visualizzazione permettono di fornire al fisioterapista o al paziente il biofeedback ricavato dall'unità di elaborazione. I sistemi di visualizzazione sono quelli integrati nei dispositivi (monitor, smartphone in modalità *see-through*, smartglasses).

Figura 2.2: Sistemi di prelievo utilizzati e relativi marker associati.
A sinistra una sondo del sistema DuePro,a destra una sonda Meacs.

Architettura software

Il sistema AR è stato sviluppato come *plugin* di bluePlot, un software sviluppato all'interno del LISiN che permette di acquisire il segnale EMG interfacciandosi con diversi sistemi di prelievo wireless, tra cui i due riportati sopra. Le parti principali che compongono il sistema (bluePlot + AR *plugin*) sviluppato possono essere racchiuse in quattro macro blocchi che sono: acquisizione segnale EMG, acquisizione segnale video, elaborazione dei segnali ed infine biofeedback. Per quanto riguarda le specifiche di progetto, è stato richiesto che ogni blocco fosse: 1) multipiattaforma, 2) scritto in C++ e 3) integrabile in Qt.

Le relazioni che sussistono fra i vari blocchi sono mostrate nella figura 2.3. All'interno del riquadro rosso, sono riportate le parti che costituisco il sistema di realtà aumentata.

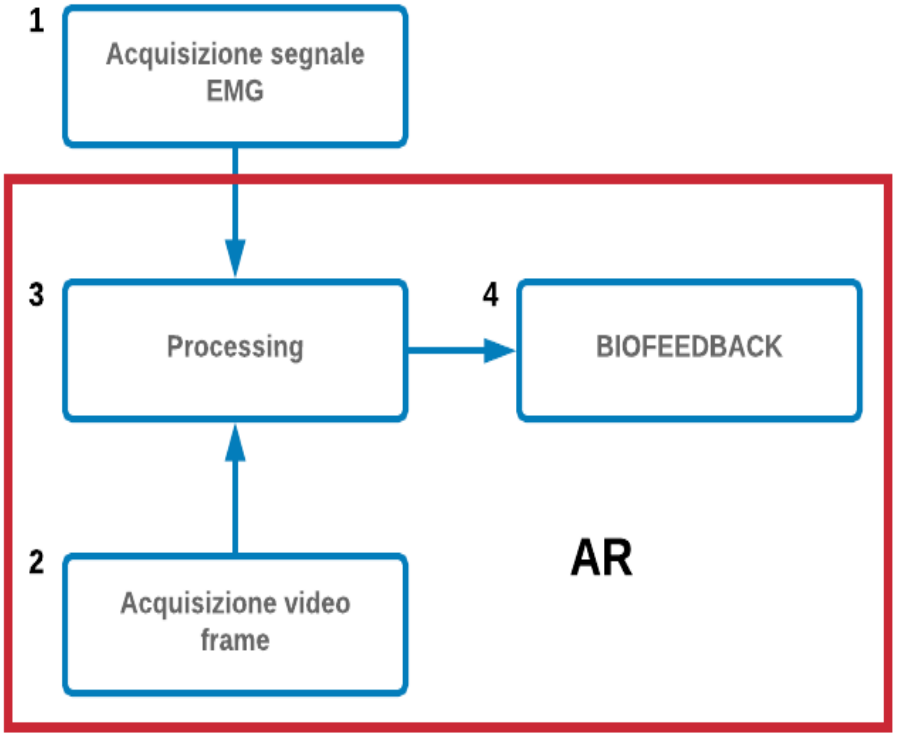


Figura 2.3: I quattro blocchi funzionali costituenti il sistema.

Il primo blocco riceve il segnale prelevato dai dispositivi connessi al software, legge il segnale ricevuto, lo converte in mV e infine lo invia al terzo blocco.

Il secondo blocco invece si occupa della gestione del flusso video. Apre la connessione con la camera presente nel dispositivo in uso, riceve i video frame, ne controlla formato e risoluzione e li invia al terzo blocco.

Il terzo blocco gestisce i segnali EMG e video ricevuti dai primi due. In particolare il segnale EMG viene processato al fine di calcolare l'ARV o RMS in un determinato periodo temporale (epoca) la cui durata può essere impostata a piacimento. Successivamente, sulla base di questo parametro, viene calcolato il colore indicativo dell'intensità dell'attività muscolare rilevata dalle sonde nell'epoca corrente. Il colore ricavato servirà poi per fornire il feedback visivo. Per quanto riguarda il segnale video invece, una volta ricevuto il video frame si passa al riconoscimento dei *fiducial marker* presenti nella scena. Ogni marker presente nella scena e appartenente al dizionario attualmente in uso, viene riconosciuto da ArUco che calcola le coordinate dei vertici di ogni marker nell'immagine e la relativa posa. Ad ogni sonda appartenente al sistema DuePro e connessa al software sono associati due marker differenti, mentre per ogni dispositivo di tipo Meacs connesso, un solo marker viene assegnato. Soltanto i marker attivi, cioè assegnati ad un qualche sistema di prelievo, sono di nostro interesse, mentre eventuali marker presenti nella scena, ma non attivi, vengono scartati.

Il quarto blocco si occupa di combinare i risultati dell'elaborazione dei segnali del blocco precedente per generare il biofeedback. Il video frame acquisito viene modificato andando ad aggiungere l'oggetto virtuale all'interno della scena. Sovrapposto ai marker associati alla sonda di tipo Due, viene mostrato un poligono colorato, mentre per i Meacs viene mostrata una mappa colore. Una volta generato il nuovo frame, si procede al repainting ottenendo così la visualizzazione dell'attivazione muscolare.

2.2 Acquisizione del segnale elettromiografico

L'acquisizione del segnale elettromiografico (EMG) si compone delle seguenti fasi: avvio del sistema di acquisizione del segnale EMG, connessione dei sistemi di prelievo del segnale EMG, attesa del segnale EMG da parte dei sistemi di prelievo, verifica sullo stato del *plugin* di realtà aumentata e infine richiamo del blocco di elaborazione del segnale EMG (figura 2.4).

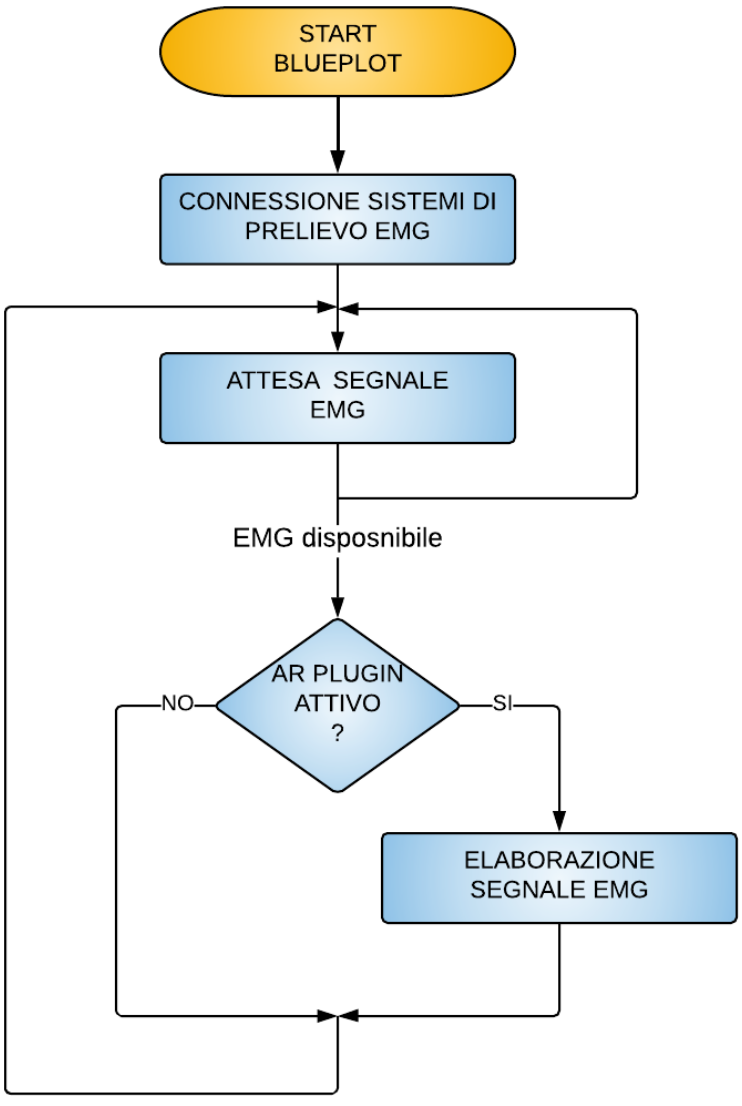


Figura 2.4: Flow chart acquisizione segnale EMG.

Il flow chart dell'elaborazione del segnale EMG è riportata in figura 2.7

Quando il sistema di acquisizione del segnale elettromiografico viene avviato, i dispositivi per il prelievo del segnale EMG, che sono attualmente attivi, si connettono al sistema. Da questo momento, bluePlot rimane in attesa del segnale EMG che viene inviato dalle sonde non appena disponibile.

In particolare, i sistemi di prelievo attivi, acquisiscono il segnale EMG, lo organizzano in pacchetti che hanno una durata temporale di 32 ms e lo inviano al software. La struttura dei pacchetti e la loro lunghezza dipendono dal tipo di sonda che sta inviando il segnale.

Ogni sonda DUE del DuePro è in grado di prelevare 2 segnali bipolari, organizzarli in pacchetti da 128 campioni ciascuno ed inviarli al PC mediante comunicazione wireless. I 128 campioni appartengono per metà al primo canale e per metà al secondo canale. Pertanto avendo una frequenza di campionamento pari a 2KHz, i 64 campioni corrispondono ad un arco temporale di 32 ms.

Il Meacs invece è in grado di acquisire 32 segnali monopolari, organizzarli in pacchetti da 2048 campioni ciascuno ed inviarli al PC mediante comunicazione wireless. I campioni per canale sono 64 e la frequenza di campionamento è di 2 KHz, quindi anche qui un pacchetto corrisponde ad un istante temporale di 32 ms.

Quando il pacchetto è pronto, il sistema di prelievo invia un messaggio al software bluePlot. Il messaggio contiene il pacchetto e tutte le informazioni necessarie a leggerlo, come il nome del dispositivo e il numero di campioni inviati. Ogni sonda invia pacchetti in modo asincrono rispetto alle altre, pertanto, se vengono inviati più pacchetti contemporaneamente, il software crea una coda d'attesa.

Quando il segnale EMG è disponibile, su ogni pacchetto ricevuto dal sistema presente nella coda d'attesa, viene effettuata una verifica sullo stato di connessione della *sensor unit* che lo ha inviato e, nel caso risulti sconnessa, si procede alla sua connessione. La connessione della sonda consiste nel verificare la presenza della *sensor unit* nel database e provvedere a cambiarne lo stato da disconnessa a connessa. Nel caso in cui l'unità sensore non venisse trovata all'interno del database, un messaggio di *warning* viene mostrato a video suggerendo di aggiungere la *sensor* in questione nella lista contenuta nel database.

Verificato lo stato di connessione, si passa a salvare i valori all' interno di un'apposita variabile che presenta come numero di righe il numero di canali del dispositivo, mentre per numero di colonne il numero di campioni acquisiti per ogni canale. Durante l'operazione di popolamento della variabile, il segnale contenuto nel pacchetto ricevuto, viene convertito in mV. Di fatto l'ampiezza del segnale EMG inviato al software è espressa in numero di livelli del convertitore A/D e quindi deve essere convertita nei mV corrispondenti. La conversione viene effettuata utilizzando la seguente formula:

$$\frac{(data - offset) * range}{n°liv * gain} = V\ mV$$

In cui:

- *Data* = ampiezza espressa in numero di livelli del convertitore;
- *Offset* = eventuale offset presente nei campioni;
- *Range* = rappresenta la dinamica del convertitore A/D;
- *N°liv* = rappdresenta il numero i livelli del convertitore A/D;
- *Gain* = rappresenta il guadagno del sistema di acquisizione;

Impostati i dati relativi all'ultimo pacchetto ricevuto, si passa ad aggiornare il *plugin* di realtà aumentata. Fatta una verifica sullo stato dell'*AR plugin*, se attivo, viene richiamato il blocco di elaborazione del segnale EMG a cui viene passato l'ultimo pacchetto ricevuto. Il blocco di elaborazione del segnale EMG viene riportato nel capitolo 2 paragrafo 4 sotto "*Elaborazione segnale EMG*".

2.3 Acquisizione del segnale video

Il blocco dell'acquisizione video, come già indicato nella parte introduttiva del capitolo, gestisce il flusso video. Si interfaccia con la camera del device, avvia l'acquisizione dei frame quando il plugin è attivato e blocca l'acquisizione dei frame quando il plugin viene disabilitato. Inoltre, una volta che il frame viene acquisito, ne controlla il formato, la risoluzione e lo "*mappa*" all'interno della CPU così da poter accedere ai dati video e renderli disponibili al blocco del processing che andrà ad eseguire il riconoscimento marker. Con il termine mappare la CPU, si intende copiare il video frame all'interno di una porzione di memoria non riservata e quindi accessibile. Di fatto, la posizione del frame video all'interno della memoria dipende dal tipo di architettura che il device ha.

Il flow chart esplicativo che riporta la struttura del blocco, viene riportato nella figura 2.5 mostrata sotto.

Quando viene attivato il sistema di realtà aumentata, viene emesso un segnale di start che va a richiamare la fase di inizializzazione di questo blocco. In questa fase viene aperta la connessione con la camera, viene impostata la risoluzione e il formato del video frame, vengono caricati i parametri intrinseci ed estrinseci della camera necessari poi al blocco di biofeedback, viene abilitata la ricezione dei frame e infine viene avviato il flusso video.

Non appena un video frame viene acquisito dalla camera e quindi reso disponibile per il processing, vengono controllati anzitutto risoluzione e formato. Questo viene fatto per una questione di sicurezza. Di fatto, dato che la camera durante l'esecuzione del plugin rimane sempre la stessa, anche i formati utilizzati e la risoluzione non devono cambiare. Pertanto nel caso in cui il frame acquisito e presentato non possiede le caratteristiche attese, viene scartato.

Il passaggio successivo consiste nell'andare a mappare il video frame all'interno di una porzione di memoria che sia effettivamente accessibile. In alcuni casi i dati contenuti all'interno del video frame, sono contenuti nella memoria video (ad esempio nei device

che hanno Android come sistema operativo), mentre in altri casi in porzioni di memoria non accessibile direttamente (come nel caso del PC), pertanto è necessario copiare le informazioni in porzioni di memoria "mappata" prima di potervi accedere.

Infine, reso disponibile il contenuto del video frame, viene richiamato il terzo blocco, in particolare l'elaborazione del frame video, che andrà ad eseguire il processo di riconoscimento marker.

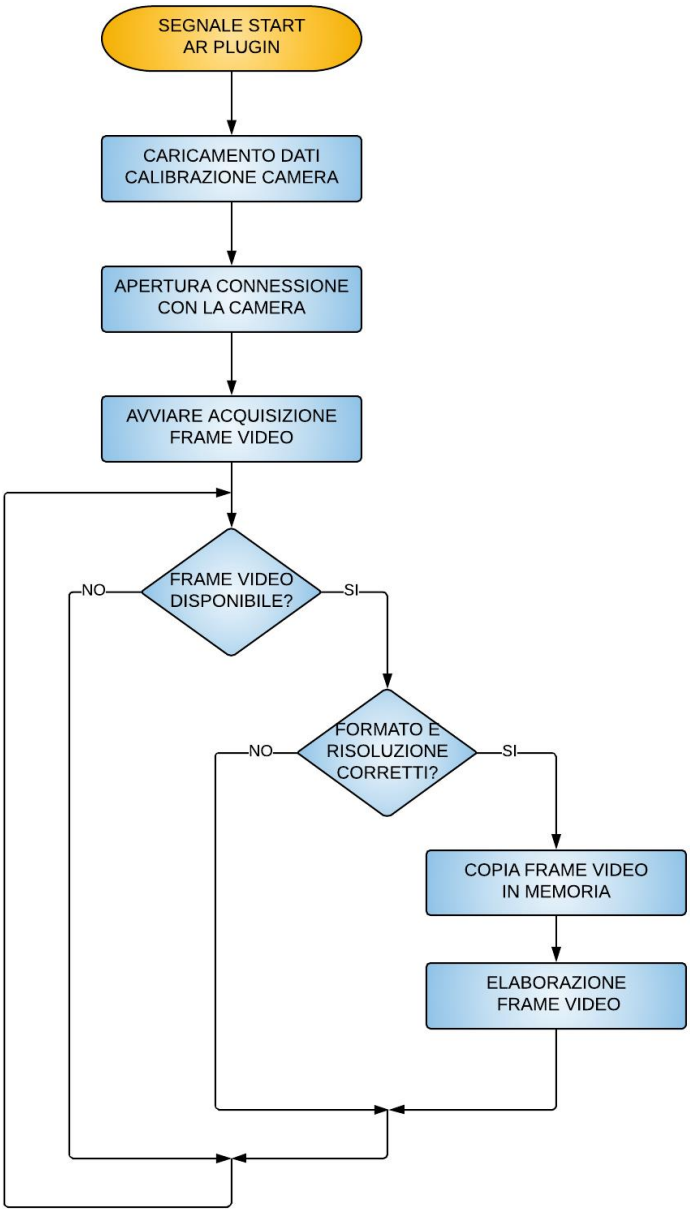


Figura 2.5: Flow chart acquisizione frame video.

il flow chart dell'elaborazione del frame video è riportata in figura 2.11.

Dopo una fase di analisi e di ricerca nella documentazione di Qt, sono state scelte tre classi che hanno permesso di soddisfare le specifiche di progetto e realizzare tutti glistep descritti in precedenza. Le classi in questione sono:

- *QCamera*
- *QAbstractVideoSurface*
- *QVideoFrame*

QCamera permette di interfacciarsi con le camere presenti nei device ed è supportata per diversi sistemi operativi come Windows ed Android che sono quelli di nostro interesse. I metodi principali della classe sono: *QCamera(), start(), setViewFider()* e *stop().* Il primo metodo è il costruttore della classe e permette di creare un'istanza di *QCamera* che, se non diversamente specificato, sarà associata alla camera predefinita in un sistema multicamera. Ad esempio per un cellulare con due camere distinte, frontale e posteriore, l'istanza della classe sarà associata a quella posteriore che è quella di default. Il secondo metodo permette di avviare l'acquisizione dei video frame, mentre il terzo permette di impostare la classe che sarà abilitata alla ricezione dei frame video che nel nostro caso è *QAbstractVideoSurface*. Infine l'ultimo metodo permette di bloccare il flusso video e rilasciare la risorsa camera che si sta utilizzando.

QAbstractVideoSurface è una classe usata dal sistema che genera il video frame, nel nostro caso la camera, per interfacciarsi con la superficie che poi permetterà di rivisualizzarlo. In particolare, se viene ereditata, questa classe abilita la ricezione del video frame acquisito offrendo la possibilità di eseguirci poi il processing desiderato. Inoltre, questa classe contiene e fornisce tutti i formati supportati dal nostro sistema. Al suo interno vi è una lista dei formati che il sistema supporta e che viene fornita alla camera durante la fase di inizializzazione. La camera poi sceglierà il formato più adatto per presentare i video frame. I metodi principali della classe sono: *QAbstractVideoSurface()*, *present(), supportedPixelFormats().* Il primo metodo è il costruttore della classe e crea l'istanza della classe mentre il secondo metodo permette la ricezione dei frame video. Quando un frame viene acquisito dalla camera, questo metodo viene invocato da *QCamera* che inserisce come parametro di input del metodo,

il video frame. Quest'ultimo viene presentato incapsulato all'interno della classe *QVideoFrame*. Il terzo metodo restituisce la lista dei formati supportati dal sistema.

QVideoFrame rappresenta il frame costituente un video. In particolare incapsula i dati relativi al video frame e le informazioni che lo riguardano. Come indicato nella parte iniziale del paragrafo, prima di poter accedere ai dati in esso contenuti, è necessario mappare il frame in una porzione di memoria accessibile. Questa classe permette di effettuare questa operazione e fornire l'indirizzo di memoria relativo alla porzione di CPU mappata in cui sono contenute le informazioni relative ai pixel. I metodi principali della classe sono: *QVideoFrame(), map(), bits(), unmap().* Il primo metodo rappresenta il costruttore della classe, mentre il secondo permette di mappare il frame acquisito. Il terzo metodo restituisce l'indirizzo relativo alla porzione di memoria mappata in cui è contenuto il frame attuale e viene utilizzato per andare a creare un oggetto di tipo *QImage*. Questa *QImage* contiene l'immagine corrispondente alla scena acquisita dalla camera che viene inviata al blocco del processing per effettuare poi il riconoscimento marker. L'ultimo metodo invece permette di rilasciare la risorsa occupata dal frame mappato e che non è più utile.

Durante la fase di test del blocco, si è notato che la funzione *map()* di *QVideoFrame* non era supportata per Android. Per superare questo problema si è osservato che il video frame acquisito in questo caso è contenuto all'interno del buffer della memoria video e presenta un handle di tipo *OpengGL.* Questo ha permesso di effettuare il mapping utilizzando le Application Program Interface di OpenGL che copiano il contenuto di una specifica porzione GPU all'interno della CPU.

2.4 Elaborazione dei segnali

Il blocco di Elaborazione dei segnali si occupa dell'elaborazione del segnale EMG e del segnale video che sono stati acquisiti rispettivamente dal primo blocco e dal secondo blocco (figura 2.6). L'elaborazione dei due segnali avviene in modo asincrono: ciò

permette di creare un flusso video stabile, massimizzando l'interoperabilità dei singoli processi di elaborazione. Di fatto il campionamento del segnale EMG e quello del video, avvengono a frequenze differenti e pertanto, i rispettivi processi di elaborazione vengono richiamati in tempi diversi e renderli sincroni equivarrebbe a rallentarne uno dei due. I risultati dell'elaborazione vengono poi integrati fra loro dal blocco di biofeedback.

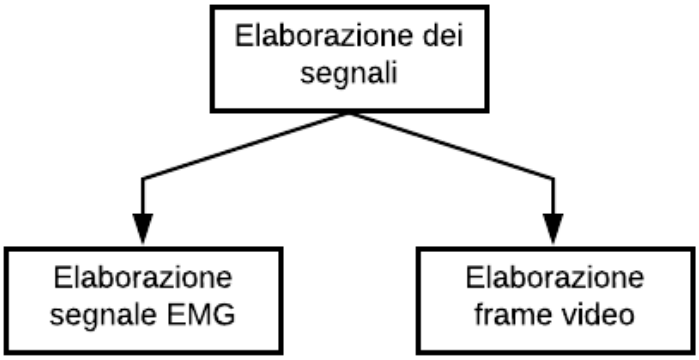


Figura 2.6: Schema del blocco di elaborazione dei segnali.
In figura 2.7 viene riportato il flow chart dell'elaborazione del segnale EMG,
mentre in figura 2.11 viene riportato il flow chart dell'elaborazione del frame video.

Elaborazione segnale EMG

L'obiettivo dell'elaborazione del segnale EMG è quello di andare ad estrarre l'ARV o l'RMS su ogni canale, per ogni epoca di segnale acquisito e calcolare poi il colore corrispondete. Ogni epoca (guardare capitolo 1 paragrafo 3) è composta da un determinato numero di pacchetti dalla durata temporale di 32 ms ciascuno e pertanto non è possibile calcolare i parametri cercati direttamente sui dati del singolo pacchetto.

Ricordiamo infatti che:

$$ARV = \frac{1}{n} \sum_{i=0}^{n} |s(i)|$$

$$RMS = \sqrt{\frac{1}{n} \sum_{i=0}^{n} (s\ (i))^2}$$

Dove:

- *n =* numero di campioni totali contenuti nell'epoca;
- *s(i) =* rappresenta il valore del campione *i-esimo.*

Necessitiamo quindi di un determinato numero di pacchetti affinché i campioni totali siano sufficienti al calcolo di queste due grandezze. Per ovviare al problema, per ogni pacchetto ricevuto, vengono calcolate solamente le sommatorie sul numero di campioni contenuti nel pacchetto stesso e salvate all'interno di un'apposita variabile, che ha per numero di righe il numero di unità sensore connesse, mentre per numero di colonne il numero di canali corrispondenti all'unità sensore. Se il pacchetto ricevuto non è il primo dell'epoca corrente, la variabile non sarà vuota e i valori ricavati dalle sommatorie vengono addizionati ai valori corrispondenti già presente. Inoltre, per ogni pacchetto viene anche incrementato un contatore, che tiene conto del numero totale di campioni attualmente acquisiti. Quando il numero di pacchetti acquisiti è tale da aver coperto un'epoca, si passa a calcolare l'ARV o l'RMS e il colore corrispondente al valore del parametro calcolato, si azzera la variabile che tiene conto della somma delle sommatorie e il contatore in cui sono indicati il numero totale di campioni attualmente acquisiti. Il colore calcolato viene inserito in un apposito vettore il quale verrà letto poi dall'ultimo blocco.

Il flow chart esplicativo viene riportato nella Figura 2.7.

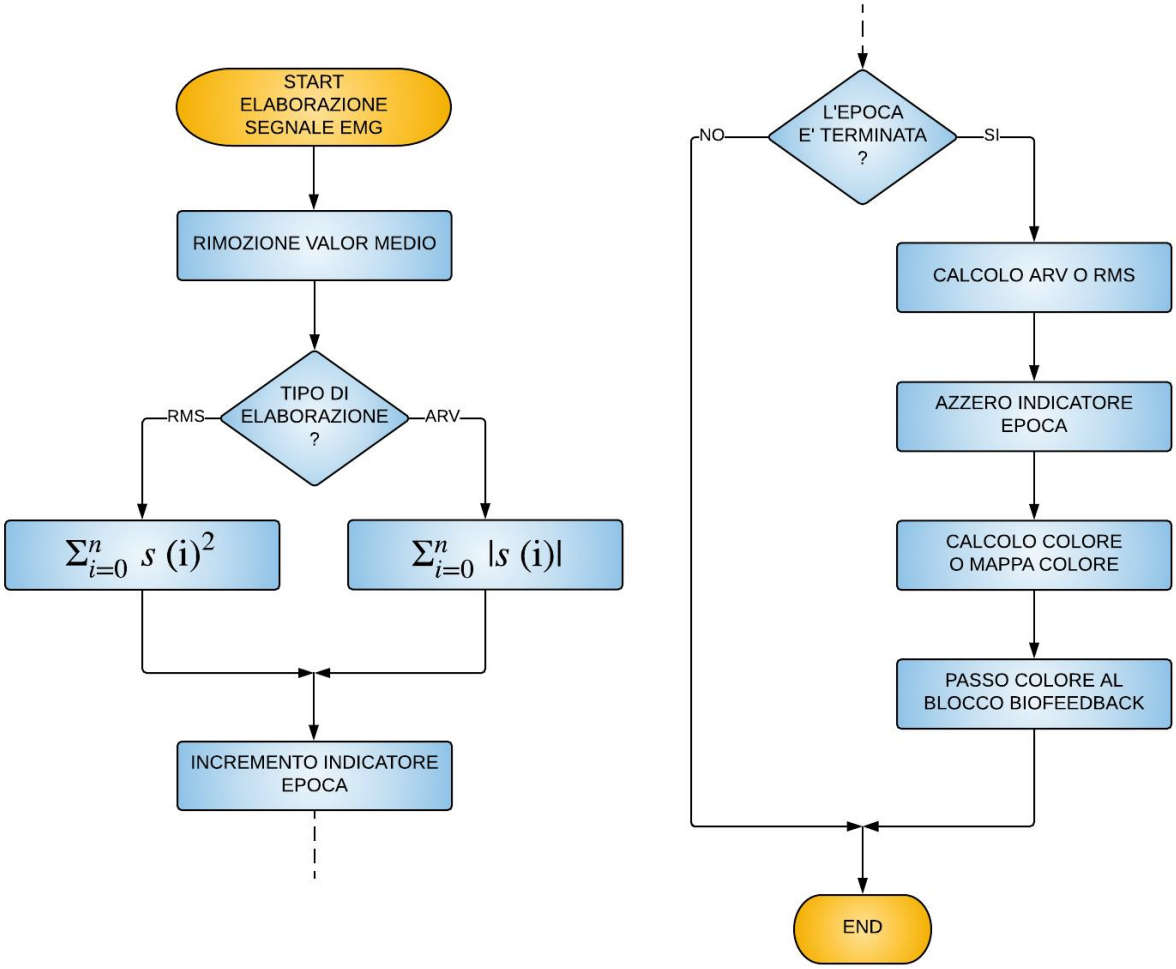


Figura 2.7: Flow chart elaborazione segnale EMG.

Il flow chart del blocco di biofeedback è riportato in figura 2.13 e 2.14

L'elaborazione del segnale elettromiografico inizia quando viene richiamato il blocco di elaborazione del segnale EMG dal blocco di acquisizione del segnale elettromiografico. Il primo passo consiste nella rimozione del valor medio. Al valore dei campioni costituenti il pacchetto, viene sottratto il valor medio del canale a cui appartengono.

Successivamente, in base alle impostazioni iniziali, si passa a calcolare le opportune sommatorie e ad incrementare il contatore indicativo del numero di campioni totali attualmente acquisito.

Quando il numero di pacchetti necessario a formare un'epoca è stato raggiunto, vengono calcolati gli ARV o gli RMS sui canali della sonda. Da tali valori si passa a ricavare il colore corrispondente mediante l'utilizzo della seguente normalizzazione:

$$H = 240 - (\frac{EMG_{param}}{A_{max}} * 240)$$

In cui:

- H = Hue cioè tonalità del colore
- EMG_{param} = ARV o RMS
- A_{max} = rappresenta il valore massimo di ARV o RMS in mV

La codifica colore utilizzata è HSV da Hue Saturation Value (tonalità saturazione valore) e permette di comporre i colori andando a combinare i tre parametri. In particolare H varia da 0 a 360 gradi. Questo vuol dire che le tonalità di colore sono distribuite su una circonferenza e modificando questo parametro è possibile modificare la colorazione come mostrato in figura 2.7. S invece rappresenta l'intensità e la purezza della singola tonalità mentre V indica la brillantezza.

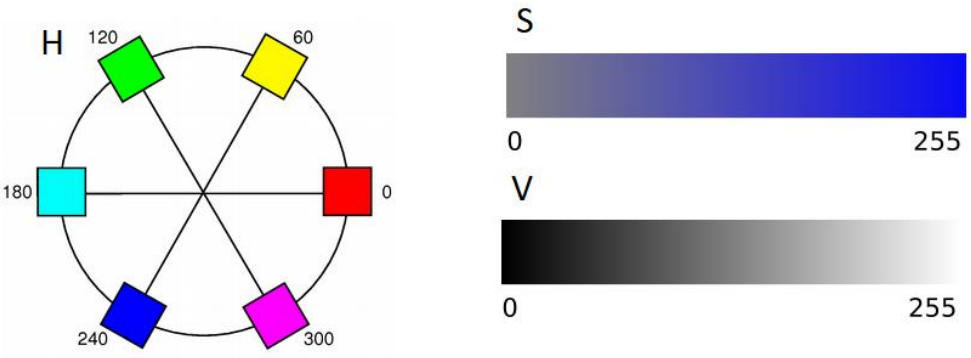


Figura 2.8: Descrizione grafica dello spazio colore HSV.

Ciò che è stato fatto quindi è stato legare H al valore di ARV o RMS calcolato, facendo in modo che un valore basso corrispondesse ad una tonalità vicina al blu mentre un valore alto, prossimo ad A_{max}, corrispondesse ad una tonalità vicina al rosso. S e V sono stati mantenuti pari a 255.

Per quanto riguarda il Meacs, i colori ricavati per ogni canale vengono composti al fine di ricavare la mappa colore. La mappa viene realizzata utilizzando un'apposita funzione già presente in bluePlot.

Una volta terminata la fase di elaborazione, i risultati vengono inseriti in appositi vettori e passati all'ultimo blocco.

Elaborazione del frame video

L'obiettivo dell'elaborazione del frame video è quello di andare ad individuare i marker presenti nella scena, calcolare la loro posa rispetto alla camera e passare i risultati all'ultimo blocco.

Fiducial Marker e dizionario

La stima della posa è un problema che accomuna tutte le applicazioni AR e VR. La stima della posa consiste nell' andare a definire la trasformazione lineare che permette di passare da un determinato sistema di riferimento, ad esempio quello del marker, ad un altro, ad esempio quello della camera. Per poter risolvere questo problema è necessario trovare delle corrispondenze fra dei punti noti nell'ambiente, che costituisce la scena e la loro proiezione nel piano della camera. Per risolvere questo problema vengono utilizzati i *fiducial marker* mostrati in figura 2.9 e 2.10.

I *fiducial marker*, sono dei punti di riferimento artificiali che vengono aggiunti nell'ambiente per facilitare il riconoscimento di oggetti specifici, oppure per effettuare la stima della posa [3]. Grazie alle caratteristiche dei marker, si è in grado di riconoscerli all'interno della scena in maniera semplice, permettendo così di individuare le corrispondenze cercate in maniera veloce e precisa [1].

Un sistema di *fiducial marker*, è composto da un determinato numero di marker, che rappresentano il dizionario, ovvero una lista di marker che possono essere riconosciuti dal sistema, e da un algoritmo in grado di identificarli nella scena. Quest'ultima rappresenta tutto ciò che viene acquisito dalla camera.

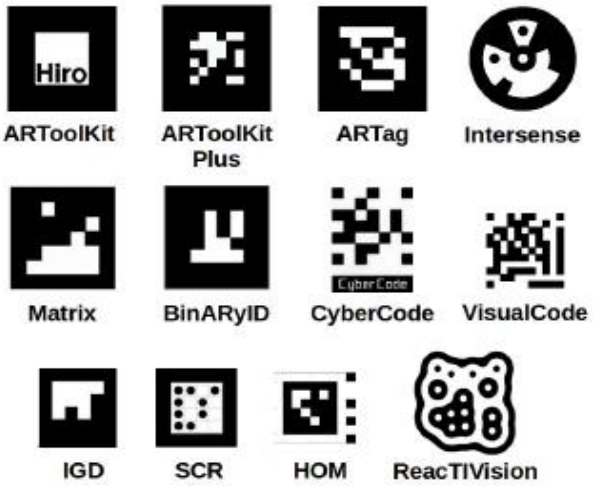


Figura 2.9: Esempi di fiducial marker [3].

In letteratura sono stati proposti diversi sistemi di *fiducial marker* che differiscono per forma, codifica interna e algoritmi che ne permettono il riconoscimento. Nella figura 2.9 vengono mostrati alcuni tipi di marker che possono essere trovati.

Alcuni sistemi impiegano marker planari di forma circolare, il cui identificativo, viene codificato utilizzando settori circolari o anelli concentrici [4][5]. Quest'approccio tuttavia, fornisce solamente un punto di corrispondenza, il centro, rendendo necessario individuare diversi marker prima di poter effettuare la stima della posa.

Altri sistemi invece, sono basati su algoritmi in grado di identificare forme riconducibili a macchie. Un esempio sono i marker a forma di ameba ReacTIVision, la cui forma è ottimizzata mediante l'utilizzo di algoritmi genetici [6].

Tuttavia quelli più utilizzati in applicazioni AR, sono quelli basati su marker di forma quadrata. Questi ultimi infatti, permettono di stimare la posa direttamente grazie ai

quattro angoli che compongono il marker, mentre la regione interna contiene le informazioni necessarie all'identificazione.

Esistono diverse tipologie di marker, fra cui citiamo AR ToolKit [7] i cui marker hanno un design che presenta dei bordi neri molto spessi e un'immagine, contenuta nella regione interna, utilizzata per permetterne l'identificazione. Tutte le immagini valide associabili ad un marker, sono contenute all'interno di un database. L'algoritmo che permette di individuare i marker, utilizza un approccio basato sul riconoscimento di un modello predefinito e una soglia fissa per effettuare il *thresholding* dell'immagine. Questo algoritmo però, mostra dei limiti in termini di falsi positivi e numero di marker riconosciuti, ma con identificativi differenti rispetto a quelli reali [8]. Inoltre, a causa della soglia fissa, è molto sensibile alle variazioni di luminosità.

Altre tipologie di marker, sempre a forma quadrata, differiscono per le modalità di riempimento della regione interna. Una soluzione ampiamente adottata [1], è quella di utilizzare una matrice binaria, da porre all'interno del marker, che lo codifica in maniera univoca. Grazie a questo tipo di codifica, è possibile realizzare anche sistemi di individuazione d'errore, aggiungendo dei bit di ridondanza, nella matrice binaria. Un esempio è il sistema ARTag [3], in cui, la porzione interna, presenta per l'appunto la matrice, mentre l'algoritmo è un *edge-based square detection*. Inoltre, permette di effettuare l'identificazione e la correzione dell'errore fino ad un massimo di due bit.

Il sistema di *fiducial marker* utilizzato in questo lavoro di tesi, è quello proposto nei lavori qui citati [1][2]. Gli autori, hanno sviluppato una libreria, ArUco, che implementa l'algoritmo necessario per individuare i marker, ed altre funzioni importanti come, quelle per effettuare la stima della posa, stampare i marker, creare dizionari ad hoc e mostrare i sistemi di riferimento marker oppure i marker individuati nella scena. Per quanto riguarda le informazioni sulla libreria e sull'algoritmo che permette di ricavare le coordinate dei vertici, dei *fiducial marker*, nel sistema di riferimento camera, si rimanda al sotto-paragrafo successivo, qui di seguito vengono riportate invece, le informazioni sui marker e sul dizionario utilizzato.

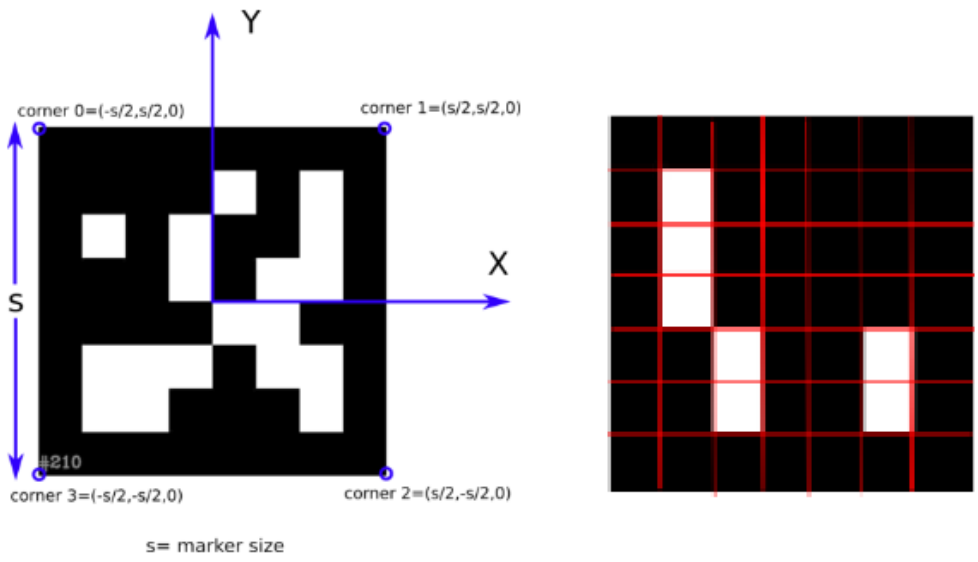


Figura 2.10: Esempio marker ArUco. A sinistra è mostrato il sistema di riferimento Interno del marker, mentre a sinistra viene messa in evidenza la matrice interna. Ogni quadrato delimitato in rosso, rappresenta un bit.

I marker, in ArUco, sono di forma quadrata, presentano un bordo nero esterno di spessore variabile che ne facilita il processo di riconoscimento e una matrice binaria interna che ne va a determinare l'identificativo. Ogni identificativo è unico all'interno di un dizionario. In figura 2.10 viene mostrato un esempio di marker.

Si può notare che l'origine del sistema di riferimento è posizionato al centro del marker stesso e ogni angolo è definito in maniera univoca, grazie alla matrice binaria. Quest'ultimo aspetto, permette di individuare correttamente la posizione del marker nella scena, che di fatto può presentarsi anche ruotato.

Il dizionario, formato dall'insieme di tutti i marker attivi che possono essere riconosciuti dall'algoritmo, viene definito partendo da due parametri:

- Dimensione del dizionario stesso *i*.
- Dimensione dei marker *n*.

La dimensione desiderata, rappresenta il numero totale di marker che compongono il dizionario, mentre la dimensione dei marker, rappresenta il numero di bit utilizzati

per generare la matrice binaria interna. In particolare scelto *n* si ha a disposizione una griglia di *(n+2)x(n+2)* bit da utilizzare per andare a codificare il marker. Le celle esterne vengono colorate a nero, mentre le restanti *n x n* vengono utilizzate per creare la matrice interna. L'algoritmo che si occupa di creare il dizionario, è presentato nell'articolo [1]. L'algoritmo seleziona un numero di marker pari a quello desiderato, all'interno dello spazio formato dalle possibili soluzioni, facendo in modo che siano il più lontano possibile ed abbiano un numero di bit di transizione il più alto possibile. L'identificativo di ogni marker, corrisponde alla posizione occupata all'interno del dizionario e, ad ognuno di essi, è associato il numero intero ottenuto concatenando tutti i bit della matrice interna.

Identificazione dei marker

Nella fase di elaborazione del frame video, abbiamo detto che l'obiettivo è quello di andare ad individuare i marker presenti nella scena, calcolare la loro posa rispetto alla camera e passare i risultati all'ultimo blocco. Tuttavia, non è detto che tutti i marker individuati siano attualmente associati ad una *sensor unit*. È possibile infatti che nella scena sia presente una sonda sui cui elettrodi sono stati posizionati i marker, ma che, per un qualsiasi motivo, sia stata disattivata o non si ha semplicemente interesse a riceve il biofeedback. È quindi necessario che il sistema di processing sia anche in grado di discriminare quali marker siano attualmente associati ad un sistema di prelievo e quali invece no. Per ovviare al problema, viene creata una lista di marker id attualmente associati e attivi che viene usata per confrontare gli id dei marker trovati e scartare quelli indesiderati.

Il flow chart esplicativo che mostra i passi principali del riconoscimento marker viene riportato in figura 2.11.

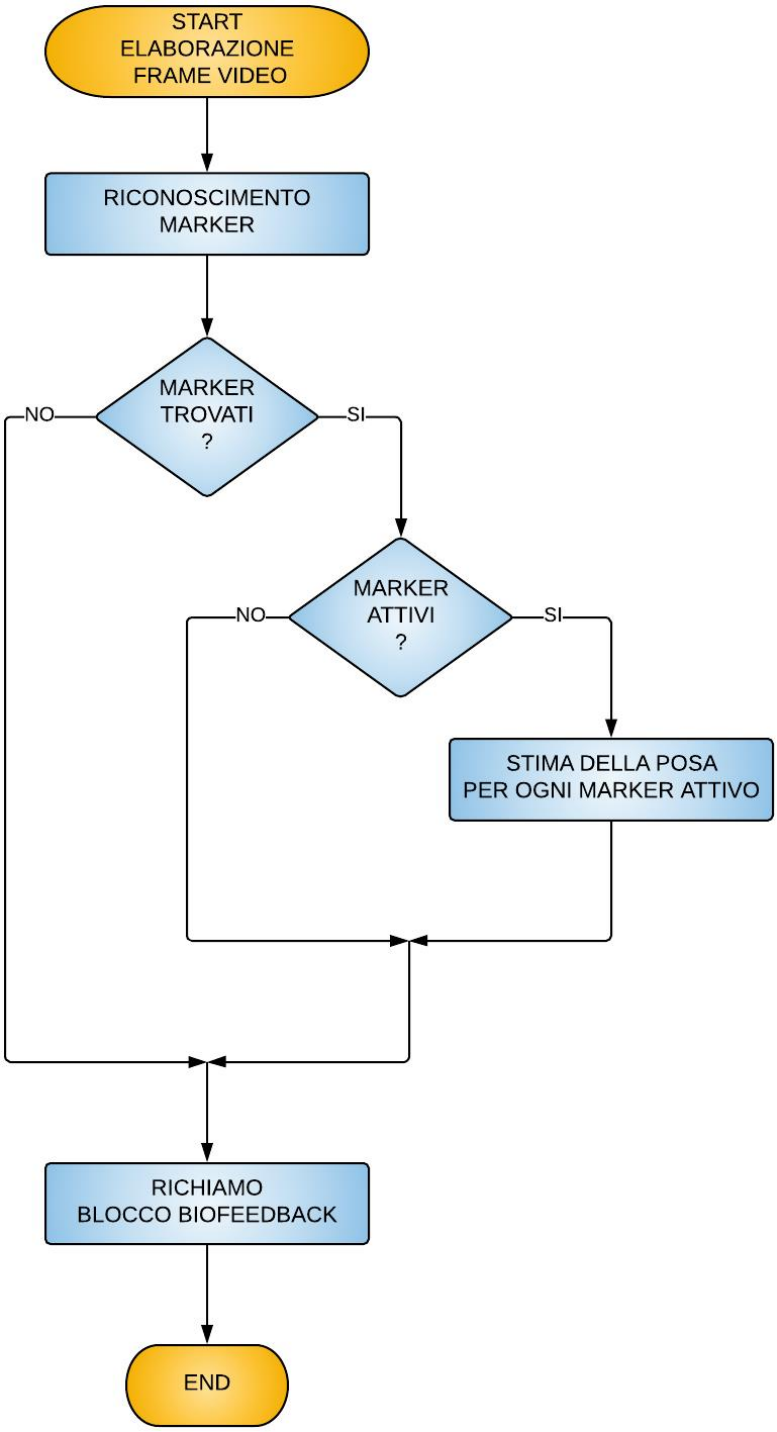


Figura 2.11: Flow chart elaborazione frame video.

Il flow chart del blocco di biofeedback è riportato in figura 2.13 e 2.14.

L'elaborazione del frame video inizia con la chiamata, da parte del blocco di acquisizione video, del riconoscimento marker. Nel *riconoscimento marker* viene

eseguito il processo di riconoscimento dei marker presenti nella scena. In particolare vengono calcolate le coordinate dei quattro vertici, di ogni marker presente nella scena, nel sistema di riferimento camera e inoltre, viene fornito un vettore contenente tutti gli id dei marker individuati.

Terminata la fase di riconoscimento marker, si verifica che effettivamente siano stati individuati marker e che, se individuati, siano attivi. Di fatto il passaggio successivo è oneroso dal punto di vista computazionale [1] e dovendo essere un sistema real time, è necessario evitare sforzi computazionali inutili.

Verificata la presenza di marker attivi all'interno della scena, si procede ad effettuare la stima della posa. Qui sono necessari i parametri intrinseci ed estrinseci della camera al fine di ricavare il vettore traslazione e il vettore rotazione. Questi due vettori contengono rispettivamente la posizione dell'origine del sistema di riferimento marker rispetto al sistema di riferimento camera e le rotazioni espresse secondo la notazione di Rodrigues. Quest'ultima notazione descrive la trasformazione del sistema di riferimento di partenza in quello d'arrivo secondo una rotazione attorno ad un'asse generico (e non x, y, z) di un certo angolo. Il vettore rotazione, fornito dal processo di stima della posa, contiene sia l'angolo che l'asse di rotazione e sono rispettivamente il modulo di tale vettore e il versore di tale vettore. Il vettore rotazione viene convertito nella matrice di rotazione, utilizzata poi nel blocco successivo.

Una volta che la stima della posa è terminata, viene richiamato l'ultimo blocco passando il vettore traslazione e la matrice di rotazione.

Per rispettare le specifiche di progetto è stato necessario ricercare librerie di computer vision, scritte in C++ e che fossero in grado di effetture marker detection. Dopo un'attenta ricerca, si è individuata ArUco. ArUco è una libreria OpenSouce estremamente veloce, scritta completamente in C++, che permette di riconoscere i marker presenti nell'ambiente e di calcolare la loro posizione in modo quanto più possibile affidabile. Questa libreria si appoggia ad OpenCV (Open Source Computer Vision Library) una libreria open source per computer grafica.

OpenCV è composta da diversi moduli, ognuno dei quali svolge funzioni diverse, che posso essere raggruppati in due gruppi differenti: *Main modules*, *Extra modules.* I *Main modules* sono i moduli che rientrano nella release ufficiale di OpenCV, mentre gli *Extra modules* no. Per poter utilizzare gli *Extra modules,* è necessario ricompilare OpenCV ed aggiungereli esplicitamente nella fase di compilazione. La libreria da noi scelta, fa parte degli *Extra modules,* pertanto per poterla utilizzare è stato necessario ricompilare il tutto.

La compilazione della libreria è stata fatta usando CMake, una famiglia di strumenti open source e multipiattaforma progettati per creare, testare e "confezionare" software. Per poter utilizare la libreria su qualsiasi sistema operativo, è stato necessario ricompilarla diverse volte cambiando i compilatori. Ad esempio per Windows è stato selezionato MinGW, mentre per Android la toolchain specifica per l'architettura selezionata.

A questo punto è stato possibile integrare ed importare ArUco in Qt rispettando le specifiche. Le funzioni della libreria che hanno permesso di effettuare il riconoscimento marker e la stima della posa sono rispettivamente: *detectMarkers()* e *estimatePoseSingleMarkers().* Mentre quella che ha permesso il passaggio dal vettore rotazione alla matrice di rotazione è la funzione *Rodrigues(…)* contenuta in OpenCV.

Algoritmo di riconoscimento dei marker

Vengono riportati qui di seguito i passi principali seguiti dall'algoritmo di *marker detection* implementato in ArUco.

Il processo di riconoscimento marker, può essere diviso in due passi principali:

1) *Individuare i candidati marker,*
2) *Determinare l'Id dei marker*.

1) *Individuare i candidati marker* è il primo passo del riconoscimento marker. L'obiettivo è analizzare l'immagine ed individuare tutte le forme geometriche riconducibili a quadrati e che per l'appunto rappresentano i possibili candidati

marker. Anzitutto viene applicato un thresholding adattivo per andare a segmentare l'immagine, poi vengono estratti tutti i contorni e, infine, viene applicata, sui contorni individuati, un'approssimazione poligonale [1]. L'approssimazione poligonale, permette di scartare tutti i contorni che non sono approssimabili con quattro vertici. Sappiamo infatti che i marker in uso presentano un profilo quadrato o rettangolare e pertanto, tutto ciò che non può essere rappresentato come tale, viene scartato. Ciò che si ottiene è quindi l'insieme di tutti i possibili candidati su cui poter ricercare i marker.

2) *Determinare l'Id dei marker* è l'ultimo step dell'algoritmo di riconoscimento. L'obiettivo è analizzare la regione interna ai contorni individuati nello step precedente per determinare, se presente, l'identificativo del marker. Il primo passo consiste nell'estrazione del codice binario contenuto nella porzione di immagine interna ai contorni. Anzitutto viene effettuata una trasformazione prospettica per riportare il presunto marker nella forma canonica eliminando così, la distorsione [1]. Successivamente viene applicato il metodo di Otsu [9], che permette di separare i bit bianchi da quelli neri, ottenendo un'immagine binaria. A questo punto, la porzione di immagine che è stata sottoposta al thresholding mediante Otsu, viene divisa in una griglia che presenta un numero di celle pari al numero di bit contenuti nei marker in uso. In ogni cella, vengono contati il numero di pixel bianchi e il numero di pixel neri, così da assegnare il colore al bit in base alla maggioranza delle occorrenze. Terminata l'estrazione dei bit, si passa ad identificare l'id del marker. In particolare si va a verificare che il codice estratto, appartenga ad uno dei marker presenti nel dizionario in uso.

Nella figura 2.9 sottostante, vengono riportati tutti i passi seguiti dall'algoritmo di riconoscimento marker.

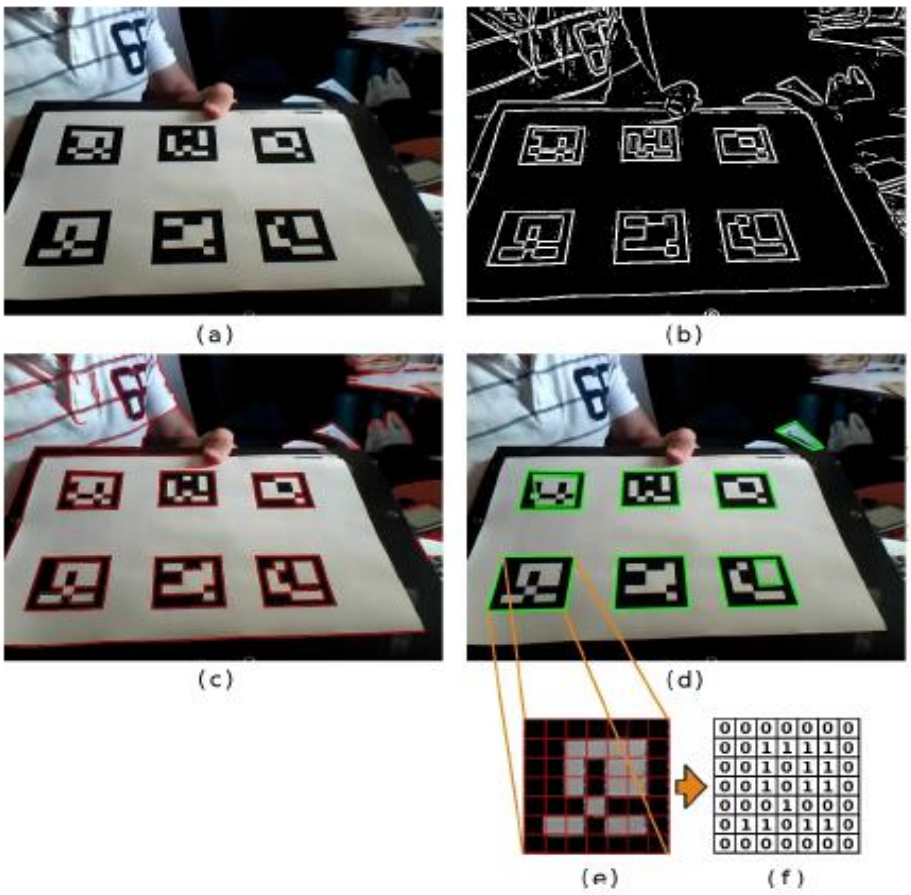


Figura 2.12: Elaborazione dell'immagine per effettuare il riconoscimento dei marker. (a)Immagine di partenza. (b)Immagine sottoposta al thresholding adattivo. (c)Estrazione dei contorni. (d)Approssimazione poligonale e rimozione dei contorni irrilevanti. (e)Esempio del marker dopo la trasformazione prospettica. (f)Assegnazione dei bit per ogni cella[1].

2.5 Biofeedback

Il blocco del biofeedback è l'ultimo del sistema AR ed ha come obiettivo quello di generare l'immagine arricchita del feedback visivo, indicativo dell'attivazione muscolare registrata. Questo blocco riceve le informazioni che sono state estratte dal blocco precedente di processing dei segnali EMG e video, le combina e procede al rendering della scena. Nelle posizioni indicate dai marker, vengono aggiunti gli elementi virtuali, che sono i poligoni colorati per gli elettrodi bipolari e le mappe colore per le matrici di elettrodi. Questi rappresentano il feedback visivo.

Nelle figure 2.13 e 2.14 viene riportato il flow chart esplicativo che mostra i passi principali che vengono eseguiti.

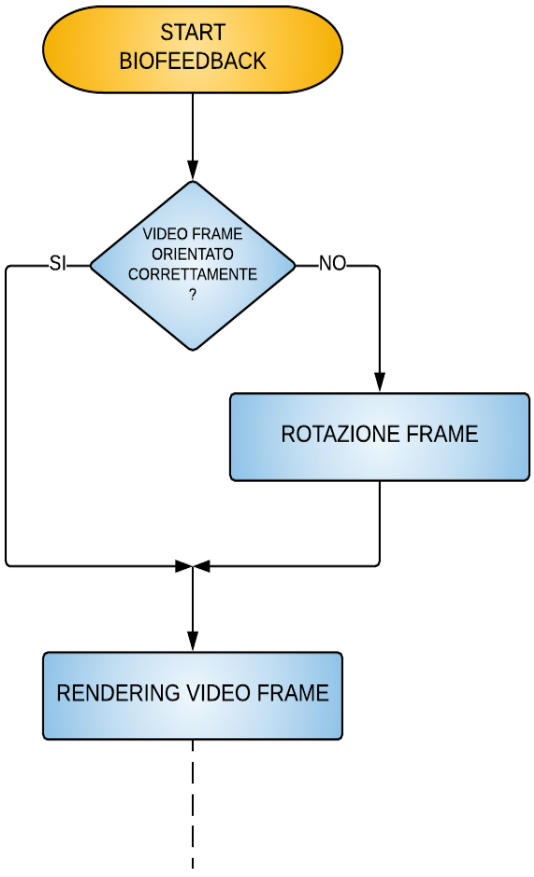


Figura 2.13: Flow chart biofeedback parte 1.

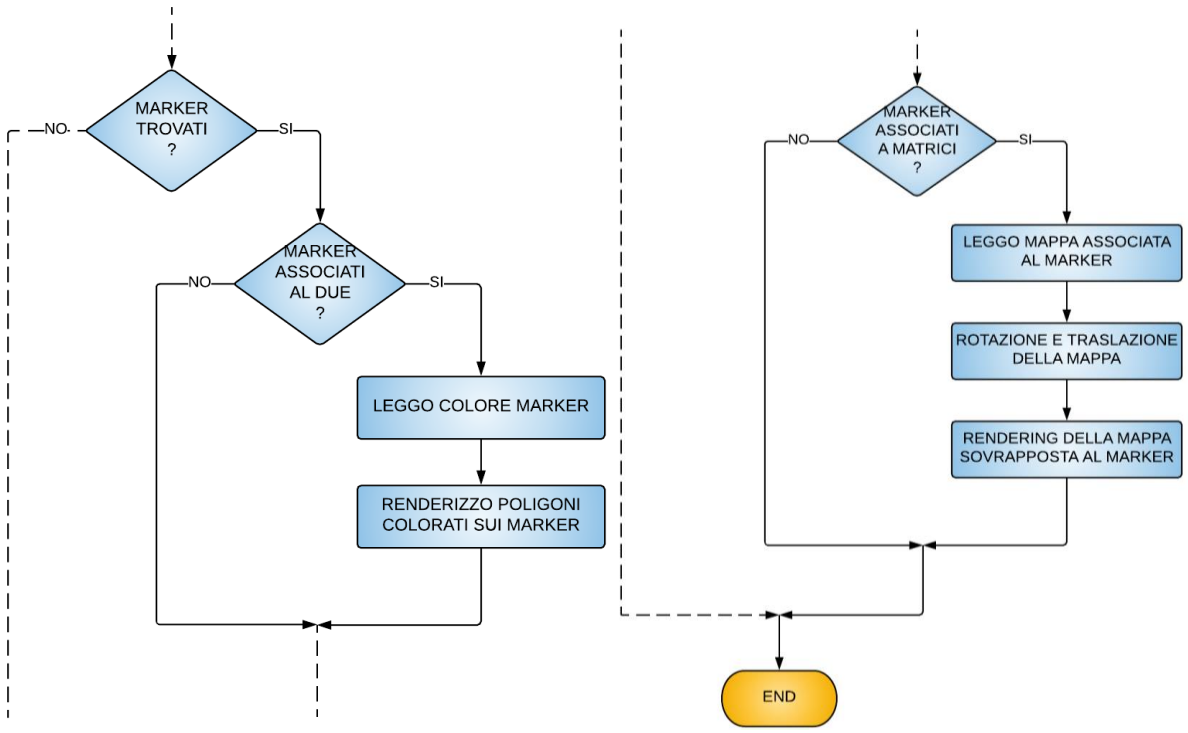


Figura 2.14: Flow chart blocco biofeedback parte 2.

Il quarto blocco viene richiamato ogni qualvolta il processing del video frame termina. Anzitutto si verifica che il frame sia orientato correttamente. Se si sta utilizzando un dispositivo mobile, è possibile che durante l'esecuzione del programma, l'orientazione del dispositivo cambi. Il cambio dell'orientazione del dispositivo, comporta una rotazione dello schermo, che deve essere applicata anche al frame al fine di mostrarlo in maniera corretta. Applicata la rotazione, si procede al rendering dell'immagine che rappresenta la scena corrente.

Si passa ora ad associare i colori e le mappe ai rispettivi marker. Quindi, come prima cosa, viene verificato se effettivamente sono stati trovati dei marker nella fase di processing del segnale video. Se i marker sono stati trovati, si verifica se sono associati ad una *sensor unit* e se si, a che tipologia.

Nel caso venga trovata una corrispondenza fra i marker individuati nella scena e quelli associati alla *sensor unit* di tipo Due, si procede a leggere il colore corrispondente al

marker. I colori sono contenuti in un vettore, la cui posizione all'interno dello stesso corrisponde alla posizione occupata dal marker all'interno del vettore dei marker. Tale vettore contiene gli identificativi dei marker attualmente associati alle sonde di tipo Due. In questo modo è possibile associare in maniera corretta il colore al marker. Letto il colore, si procede a disegnare sul marker corrispondente un poligono colorato. Questo step viene eseguito per tutti i marker trovati nella scena che attualmente sono associati alle sonde per il prelievo bipolare.

Nel caso invece venga trovata una corrispondenza fra i marker individuati nella scena e quelli associati alle sonde per il prelievo di segnale HD-EMG, si procede a leggere la mappa colore associata al marker, ruotarla e traslarla nella pozione corretta e infine mostrarla a video.

Dopo aver ricercato nella documentazione di Qt, sono state individuate due classi che hanno permesso di soddisfare le specifiche di progetto e realizzare tutti gli step descritti in precedenza. Le classi in questione sono:

- *QPainter*
- *QBrush*

QPainter è una classe di Qt, che permette di disegnare qualsiasi oggetto 2D. Utilizza funzioni altamente ottimizzate che sono in grado di disegnare semplici linee, come oggetti più complessi quali immagini, pixmap, poligoni irregolari e quant'altro. Normalmente il painting viene fatto in un sistema di riferimento standard, ma è possibile cambiarlo applicando opportune matrici di trasformazione. Questa classe quindi dà la possibilità di effettuare il rendering della scena, disegnare i poligoni colorati, posizionare e disegnare la mappa colore sopra i marker. I metodi principalmente utilizzati in questo lavoro sono: *QPainter(), drawImage(), drawConvexPoligon(), setBrush().* Il primo metodo è il costruttore della classe e permette di creare l'istanza di QPainter che è utilizzata poi per effettuare le operazioni di disegno. Il secondo metodo permette di disegnare le immagini date come input,

nella regione, della superficie utilizzata per effettuare il rendering, indicata. È importante sottolineare che la superficie utilizzata per effettuare il rendering degli oggeti potrebbe essere tutto lo schermo, come una porzione solamente di quest'ultimo. Pertanto il sistema di coordinate potrebbe coincidere con quello dello schermo o meno, a seconda dei casi. Il terzo metodo permette di disegnare poligoni convessi a partire dal numero di vertici e le rispettive coordinate, che sono riferite sempre alla superficie su cui poi viene effettuato ilpainting. L'ultimo metodo permette di impostare un'istanza dell'oggetto QBrush, che contiene le opzioni per il riempimento della porzione di superficie racchiusa fra i bordi del poligono disegnato.

QBrush è la classe che va a definire il riempimento delle forme che vengono disegnate mediante QPainter. È possibile definire il colore del riempimento, lo stile, il gradiente e la texture. Tale classe permette quindi di risolvere i problemi legati al colore e la mappa colore che devono essere sovrapposti al marker. Di fatto, per il primo caso basterà impostare il colore del riempimento sulla base di quello contenuto nel vettore colore, mentre nel secondo caso basterà impostare la mappa colore desiderata come texture da utilizzare nel riempimento. I metodi principalmente utilizzati in questo lavoro sono: *QBrush(), setColor(), setTexture(), setTransform().* Il primo metodo è il costruttore della classe e crea l'istanza di QBrush. Il secondo permette di impostare il colore da utilizzare nel riempimento. Il terzo permette di impostare la texture da utilzzare per riempire il poligono disegnato, quindi per quanto ci riguarda è la mappa colore. L'ultimo metodo invece permette di impostare la matrice di trasformazione applicata alla texture. Quindi permette di orientare l'immagine impostata come texture nella posizione desiderata. Questo metodo viene utilizzato per allineare la mappa colore al marker associato. In particolare la trasformazione deve essere fornita come composizione di rotazioni intorno agli assi elementari, indicando per ogni asse l'entità della rotazione in gradi. Pertanto per ricavare l'entità di questi angoli partendo dalla matrice di rotazione ricavata nel blocco di elaborazione del segnale video, viene utilizzata la funzione *RQDecomp3x3(…)* di *OpenCV*. Questa funzione permette di ottenere gli angoli di rotazione espressi in gradi, attorno agli assi elementari.

2.6 Classi implementate per il sistema AR

Il sistema AR è stato progettato con l'obiettivo di implementare i blocchi descritti sopra e mostrati in figura 2.3 all'interno del riquadro rosso. Il *plugin* di realtà aumentata è formato da 3 classi che sono:

- ***arSettings***
- ***ARPlugin***
- ***ARClass***

Le classi *arSettings* e *ARPlugin* ereditano la classe *Plugin,* presente in bluePlot, pensata per estendere le funzionalità del software in maniera relativamente semplice. Lo schema sottostante mostra le relazioni che ci sono fra le 3 classi e fra la classe principale del software, bluePlot che è *bluePlotmainWin*.

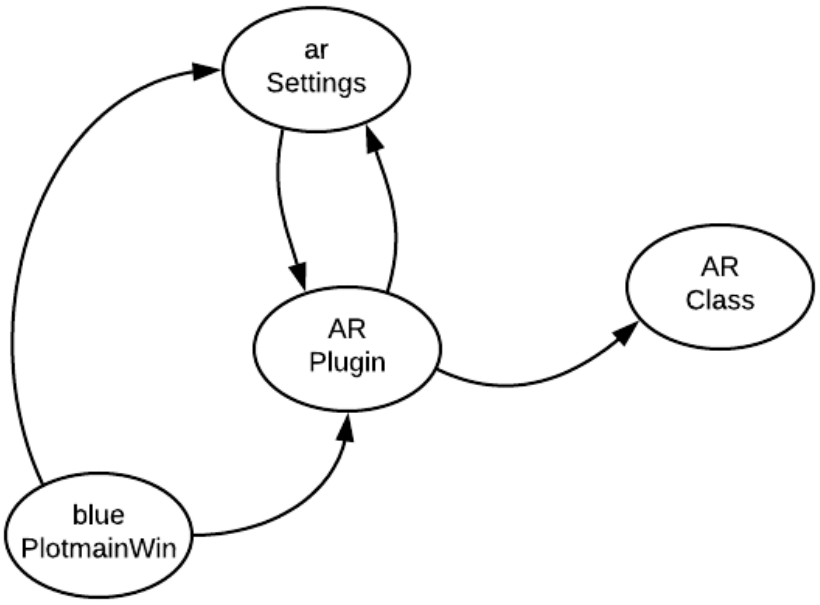


Figura 2.15: schema relazionale tra le classi del sistema.

La classe *bluePlotmainWin* gestisce tutto il programma e pertanto ha anche il compito di istanziare e inizializzare le classi del sistema AR sebbene non tutte in maniera diretta, come nel caso di *ARClass* che viene istanziata e inizializzata da *ARPlugin*.

La classe ***arSettings*****,** gestisce tutte le impostazioni del plugin AR che possono essere modificate dall'utente. Dà la possibilità di:

- Selezionare il tipo di processing da effettuare ARV o RMS;
- Impostare la lunghezza dell'epoca su cui calcolare ARV o RMS;
- Permette di visualizzare le *sensor unit* associate al *plugin,* una lista per quelle di tipo Due e una lista per quelle di tipo Meacs;
- Impostare l'id dei marker associati o associato alle *sensor unit*;
- Impostare il valore massimo di ARV o RMS che corrisponde al fondo scala del codice colore, ovvero quello rappresentato in rosso nel feedback visivo fornito;
- Abilitare o disattivare il *plugin* sulle *sensor unit* selezionate;

Questa classe quindi gestisce la GUI che permette all'utente di interagire con il programma. Inoltre *arSettings* ha il compito di assegnare in automatico gli id dei marker alle varie sensor unit associate al plugin. L'istanza di *arSettings* creata in *bluePlot*, viene passata ad *ARPlugin* il quale, in base ai settaggi, effettua il processing sul segnale elettromiografico.

I metodi principali della classe sono:

1. *Init(…);*
2. *getARSettings(…);*
3. *getARSensorsDueSett(…);*
4. *getARSensorMapSett(…);*
5. *updateAR(…);*

1. Il metodo *init(…)* ha come obiettivo quello di inizializzare la classe, quindi andare a preparare la GUI (grapich user interface) che deve mostrare tutte le impostazioni modificabili dall'utente sopra menzionate. Inoltre all'interno di questo metodo vengono assegnati, in automatico, gli identificativi dei marker associati alle varie *sensor unit*. In particolare ad ogni *sensor unit* di tipo Due vengono associati due identificativi, mentre a quelle di tipo Meacs soltanto uno. Questo metodo viene richiamato dalla classe principale *bluePlotmainWin.*

2. Il metodo *getARSettings(…)* permette di ottenere le impostazioni che l'utente ha scelto dalla GUI. In particolare vengono restituite informazioni sul tipo di elaborazione che deve essere effettuata sul segnale EMG, quindi quale fra ARV e RMS deve essere calcolato, e la lunghezza dell'epoca sulla quale calcolarli. Questo metodo viene richiamato dall'istanza della classe *ARPlugin* per impostare le informazioni sopra indicate.

3. Il metodo *getARSensorDueSett(…)* permette di ottenere la lista di tutte le *sensor unit* di tipo Due attualmente associate al *arplugin* e le relative impostazioni. Le impostazioni comprendono gli identificativi dei marker associati alle sonde e il valore massimo di ARV o RMS che corrisponde al fondo scala del codice colore. Questo metodo viene richiamato dall'istanza di *ARPlugin* per impostare le informazioni sopra indicate.

4. Il metodo *getARSensorMapSett(…)* permette di ottenere la lista di tutte le *sensor unit* di tipo Meacs attualmente associate al *arplugin* e le relative impostazioni. Le impostazioni comprendono gli identificativi dei marker associati alle sonde e il valore massimo di ARV o RMS che corrisponde al fondo scala del codice colore. Questo metodo viene richiamato dall'istanza di *ARPlugin* per impostare le informazioni sopra indicate.

5. Il metodo *updateAR(…)* è un segnale che viene emesso ogni qualvolta l'utente vuole rendere effettive le modifiche effettuate alle impostazioni sopra indicate.

Questo segnale viene intercettato dalla classe *ARPlugin* che procede ad aggiornare le vecchie impostazioni con le nuove.

La classe ***ARPlugin*** si interfaccia con *bluePlot, arSettings* e *ARClass.* Ha il compito di istanziare la classe *ARClass,* di processare il segnale EMG acquisito e calcolarne il colore corrispondente secondo le modalità indicate in settings, associare il colore o la mappa all'id del marker corrispondente e rendere disponibili queste informazioni ad *ARClass* che poi procede a creare e mostrare la scena.

I metodi principali della classe sono:

1. *init(…);*
2. *parseData(…);*
3. *setAROptions(…);*
4. *signalToColor(…);*
5. *updateTimerTimeout(…);*
6. *updateARSettings(…);*

1. Il metodo *init(…)* va ad inizializzare la classe *ARPlugin.* Anzitutto viene allocata la memoria che permette di contenere il segnale EMG acquisito. In particolare viene allocata memoria in base al numero di *sensor unit* attualmente connesse al *plugin* e al numero di canali che ogni sonda presenta. Successivamente viene allocata memoria per il calcolo delle sommatorie dell'ARV o dell'RMS. Vale il discorso fatto nel capitolo 2 paragrafo 4 sotto "Elaborazione del segnale EMG", la memoria allocata è tale da contenere un valore numerico per ogni canale di ogni sonda attiva, indipendentemente dalla lunghezza dell'epoca impostata. Questo perché il calcolo effettivo dell'ARV o dell'RMS viene effettuato solo una volta che l'epoca è terminata, mentre per ogni pacchetto ricevuto viene semplicemente calcolata la sommatoria per ogni canale. Ovviamente viene allocata memoria anche per un contatore che tiene il conto per ogni canale di

ogni *sensor unit* del numero totale di campioni attualmente ricevuti e sommati, dato che sia l'ARV che RMS sono valori medi. Successivamente viene inizializzato un timer che allo scadere richiama il metodo *updateTimerTimeout(…)*. A questo punto vengono impostati le modalità di processing del segnale e l'epoca su cui fare il processing. La fase successiva è creare l'istanza della classe *ARClass*, richiamare *setAROptions(…)* inizializzare l'istanza di *ARClass* richiamando *initARClass(…).*

2. Il metodo *parseData(…)* viene richiamato ogni volta che una sonda attualmente associata al *plugin* riceve un pacchetto di segnale EMG. Qui viene effettuato il primo passo del processing del segnale EMG e cioè vengono calcolate le sommatorie coerentemente con il tipo di elaborazione richiesta. Il risultato viene salvato nella memoria preventivamente allocata nella fase di inizializzazione dell'istanza della classe.

3. Il metodo *setAROptions(…)* va a creare i vettori contenenti gli id dei marker attualmente associati alle *sensor unit* attive. In particolare ne vengono creati due: uno per gli id che sono associati alle sonde di tipo Due e uno per gli id che sono associati alle sonde di tipo Meacs. Oltre ai vettori degli id dei marker, vengono creati anche il vettore dei colori e il vettore delle mappe colore. Il vettore dei colori contiene il colore ricavato dalla conversione dell'ARV o dell'RMS, calcolato sull'epoca corrente, per ogni canale di ogni sonda Due attualmente attiva e che quindi ha associati due marker i cui id sono contenuti nel vettore degli id associati alle sonde Due. Vi è una corrispondenza univoca fra la posizione di un determinato colore, all'interno del vettore dei colori e la posizione di un determinato id all'interno del vettore degli id associati alle sonde Due, in questo modo è possibile associare correttamente il colore al marker. Il vettore delle mappe colore contiene le mappe colore calcolate, grazie ad un metodo già presente in bluePlot, per ogni sonda di tipo Meacs attualmente attiva e che quindi ha associato un marker il cui id è contenuto

all'interno del vettore degli id associati alle sonde Meacs. Vi è una corrispondenza univoca fra la posizione di una determinata mappa all'interno del vettore delle mappe e la posizione di un determinato id all'interno del vettore degli id associati alle sonde Meacs, in questo modo è possibile associare correttamente la mappa al marker.

4. Il metodo *signalToColor(...)* permette di convertire l'ARV o l'RMS nel colore corrispondente, secondo quanto riportato nel capitolo 2 paragrafo 4 "Elaborazione del segnale EMG".

5. Il metodo *updateTimerTimeout(...)* è uno slot che è collegato al segnale di *timeOut* del timer inizializzato in *init()* e viene richiamato ogni qualvolta il segnale sopra citato viene emesso. Qui si verifica se il numero di campioni ricevuti corrisponde a quelli effettivamente contenuti in un'epoca. In caso affermativo si procede al calcolo dell'ARV o RMS, a seconda delle impostazioni, e si richiama il metodo *signalToColor()* nel caso delle sonde di tipo Due, mentre per le sonde di tipo Meacs si richiama il metodo per il calcolo delle mappe colore. Sia il colore che le mappe vengono inseriti nei vettori corrispondenti.

6. Il metodo *updateARSettings(...)* è uno slot collegato al segnale *updateAR(...)* della classe *arSettings* e viene richiamato ogni qualvolta il segnale sopra citato viene emesso. In particolare il metodo viene eseguito ogni qualvolta le impostazioni contenute nella classe *arSettings* vengono modificate dall'utente e devono essere aggiornate nell'istanza di *ARPlugin.*

La classe ***ARClass*** gestisce tutto ciò che riguarda il segnale video. Apre la connessione con la camera, riceve i frame video, li mappa all'interno della CPU, effettua il riconoscimento dei marker all'interno della scena, ruota il frame nella posizione corretta, colora i marker individuati e procede al rendering della scena con l'aggiunta degli elementi virtuali. La classe viene istanziata da *ARPlugin* che fornisce i colori o le mappe da sovrapporre ai marker.

I metodi principali della classe sono:

1. *present(…);*
2. *initARClass(…);*
3. *updateActiveMarkr(…);*
4. *imageWrapper(…);*
5. *markersDetection(…);*
6. *frameRotation(…);*
7. *paintGL(…);*

1. Il metodo *present(…)* questo metodo viene richiamato ogni volta che la camera acquisisce un frame video. Come prima cosa si verifica formato e risoluzione e nel caso questi corrispondano a quelli stabiliti nella fase di inizializzazione, il frame video viene salvato e viene richiamato *paintGL(…)*.

2. Il metodo *initARClass(…)* serve per inizializzare l'istanza di *ARClass* e viene richiamato dall'istanza di *ARPlugin.* Questo metodo serve per passare i puntatori ai vettori contenenti gli id dei marker, i colori e le mappe, creati dall'istanza della classe *ARPlugin* in *setAROptions(…)*, nell'istanza di *ARClass.* Questi vettori verranno poi letti nella fase di rendering per associare correttamente ai marker individuati nella scena i colori o le mappe colore. Inoltre, in questo metodo viene avviata anche la camera che inizia ad acquisire frame video e li invia alla funzione *present(…).*

3. Il metodo *updateActiveMarker(…)* è un metodo che viene richiamato dall'istanza di *ARPlugin* ogni volta che il segnale *updateAr(…)* dell'istanza di *arSettings* viene emesso. In dettaglio questo metodo procede ad aggiornare i puntatori dei vettori che sono stati passati dall'istanza di *ARPlugin* all'istanza di *ARClass* nel metodo *initARClass(…)*. Questo metodo viene richiamato in quanto l'utente potrebbe aver modificato l'id dei marker associati a delle *sensor unit* o semplicemente averne disattivate altre, pertanto è necessario aggiornare i suddetti vettori. È importante sottolineare che l'aggiornamento vero e proprio dei puntatori di memoria, viene effettuato all'interno del metodo *paintGL(…).* Questo avviene perché il metodo *updateActiveMarker(…)* viene richiamato dall'istanza di *ARClass* che "vive" su un thread differente rispetto a *paintGL(…)* che è colui che va a leggere le informazioni contenute nei vettori in questione. Pertanto se non fosse *paintGL(…)* a gestire l'aggiornamento, è possibile che si verifichi la situazione in cui l'istanza della classe *ARClass* richiami il metodo in questione e proceda a deallocare la memoria dove sono contenuti i vettori. Se in contemporanea *paintGL(…)* sta accedendo all'indirizzo di memoria contenente i vettori, si verifica un errore in quanto la memoria è vuota.

4. Il metodo *imageWrapper(…)* viene richiamato dal metodo *paintGL(…)* non appena un video frame viene reso disponibile dalla camera tramite il metodo *present().* L'obiettivo di questo metodo è fornire un'immagine che sia contenuta nella CPU e quindi utilizzabile per effettuare il riconoscimento marker, indipendentemente da dove sia contenuto il frame video acquisito dalla camera. Vale il discorso fatto nel capitolo 2 paragrafo 3 "Acquisizione del segnale video".

5. Il metodo *markersDetection(…)* ha il compito di eseguire il riconoscimento dei marker contenuti nell'immagine rappresentante la scena acquisita dalla camera. Come prima cosa, l'immagine viene convertita in un formato che ci siamo dati come standard (RGB_888 in quanto, in questo modo, non si ha nessun problema di compatibilità di formato con OpenCV) e successivamente si

va a creare un oggetto Mat di OpenCV. Questo è necessario in quanto la libreria ArUco si appoggia ad OpenCV e pertanto utilizza i tipi definiti nella libreria di computer vision. A questo punto viene richiamato il metodo *detectMarkers(…)* della libreria ArUco, che permette di individuare le coordinate, nel sistema di riferimento camera, dei quattro angoli di ogni marker presente nella scena e appartenente al dizionario in uso. Oltre alle coordinate degli angoli, la funzione restituisce anche un vettore contenente tutti gli id dei marker per i quali sono state calcolate le coordinate.

6. Il metodo *frameRotation(…)* permette di orientare correttamente il frame acquisito. In particolare il metodo è stato pensato per i dispositivi mobili in quanto, è possibile che durante l'utilizzo si voglia cambiare l'orientamento dello schermo. Nel caso in cui non fosse ruotato anche il frame, il cambio dell'orientamento dello schermo comprometterebbe ciò che verrebbe mostrato.

7. Il metodo *paintGL(…)* gestisce tutto il flusso del frame video fino al rendering, dopo che la camera lo ha acquisito e reso disponibile in *present(…).* Di fatto questo metodo viene richiamato indirettamente da *present(…)* che dopo aver verificato formato e risoluzione, richiama il metodo *repaint(…)* il quale va a richiamare *paintGL(…).* Come prima cosa viene istanziata la classe *QPainter* discussa nel capitolo 2 sezione 5 "Biofeedback" che permette di disegnare qualsiasi oggetto 2D. Poi viene richiamato il metodo *imageWrapper(…)* e successivamente *markersDetection(…).* A questo punto viene richiamato *frameRotation(…)* e si procede ad effettuare il rendering della scena. Quindi solo l'immagine, rappresentante la scena acquisita dalla camera, viene disegnata dall'istanza di *QPainter* utilizzando il metodo *drawImage(…)*. Prima di passare al rendering degli oggetti virtuali, viene verificato se è disponibile un aggiornamento dei vettori degli id, colori e mappe. In caso affermativo vengono sostituiti i vecchi puntatori con quelli nuovi. A questo punto se i marker sono stati individuati nella scena e c'è corrispondenza con gli id contenuti nei vettori

degli id, si procede ad effettuare la stima della posa mediante la funzione di ArUco *estimatePoseSingleMarkers().* Determinata la matrice di trasformazione, si passa a disegnare gli oggetti virtuali nella scena. Sovrapposti ai marker associati alle sonde di tipo Due, viene disegnato un poligono colorato il cui colore viene letto nel vettore dei colori in corrispondenza della posizione occupata dall'id del marker a cui si sta facendo attualmente riferimento. Lo stesso discorso viene fatto per le mappe colore, che devono essere disegnate sopra i marker associati alle sonde di tipo Meacs. Unica differenza è che la mappa colore deve essere opportunamente ruotata. Utilizzando la matrice di trasformazione ricava sopra, è possibile ruotare la mappa e mostrarla correttamente sovrapposta al marker nella scena.

Come si può notare, le tre classi descritte non coincidono con i 3 blocchi mostrati sopra in Figura 2.1, ma è solo grazie alla combinazione dei loro metodi e delle loro proprietà che si realizzano i blocchi desiderati. Questo perché si è voluto mantenere separato tutto ciò che riguarda il segnale EMG da tutto ciò che riguarda il segnale video. Tale scelta dipende sia da un motivo concettuale di separare i due segnali diversi, ma anche da un motivo pratico legato allo sviluppo della tesi. Di fatti, come primo step è stato realizzato il sistema AR in grado di aprire la connessione con la camera, acquisire i frame, rielaborarli procedendo al riconoscimento marker e visualizzarli. Successivamente, il secondo step è stato quello di integrare il sistema AR con il software di acquisizione del segnale EMG bluePlot. Pertanto i due segnali e di conseguenza le due classi che gestiscono tali segnali, sono state separate fin dall'inizio.

3. Verifica della funzionalità del sistema

Per verificare che il sistema di realtà aumentata fornisse il biofeedback desiderato, sono state svolte due prove sperimentali, in cui è stato chiesto al soggetto opportunamente strumentato di eseguire dei semplici movimenti di contrazione muscolare. Di seguito viene fornita la descrizione completa delle due prove.

3.1 Leg extension

Il primo esercizio svolto è stato la *leg extension*, eseguito mediante l'utilizzo dell'omonimo macchinario, presente in laboratorio. La *leg extension* ha come obiettivo quello di stimolare il muscolo quadricipite femorale che è formato da quattro capi: retto femorale, vasto mediale, vasto laterale e vasto intermedio. In particolare sono stati selezionati il vasto mediale (VM) e vasto laterale(VL), su cui sono stati posizionati gli elettrodi per il prelievo del segnale bipolare mediante due sonde del sistema DuePro. Per il posizionamento degli elettrodi si sono seguite le indicazioni riportate nel manuale di riferimento [20]. Sovrapposti agli elettrodi, sono stati posizionati i marker che permettono di ottenere il biofeedback.

Figura 3.1: Soggetto strumentato elettrodi e marker.

Nella figura 3.1 viene mostrato il soggetto strumentato con elettrodi e marker. Nelle figure sottostanti invece viene riportata la scena acquisita dalla camera, dove si può osservare il biofeedback fornito durante l'esecuzione dell'esercizio.

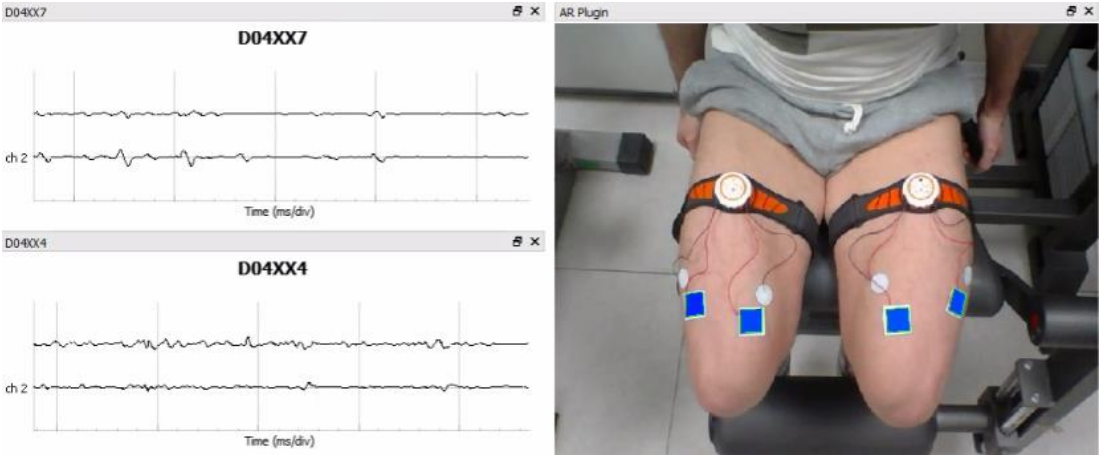
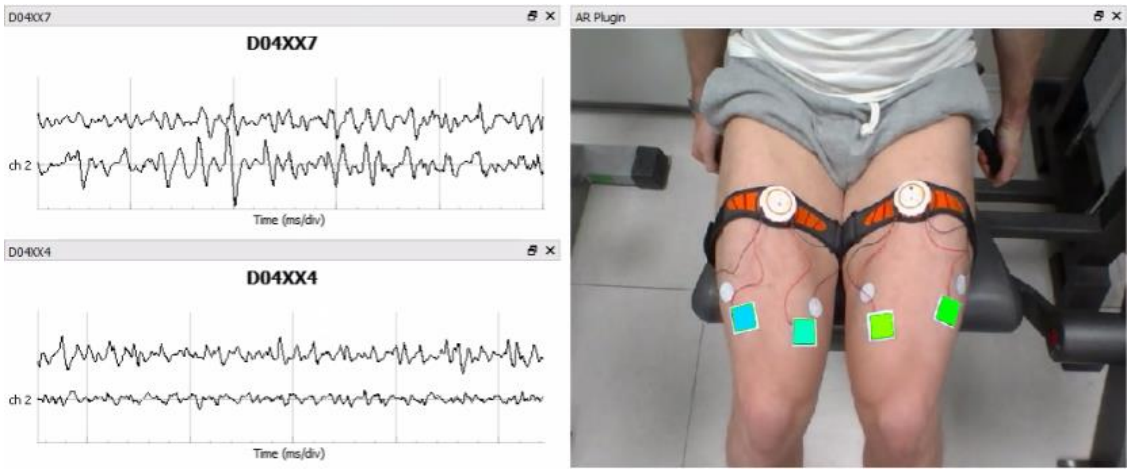


Figura 3.2: Soggetto a riposo.



igura 3.3: Inizio della fase di estensione.

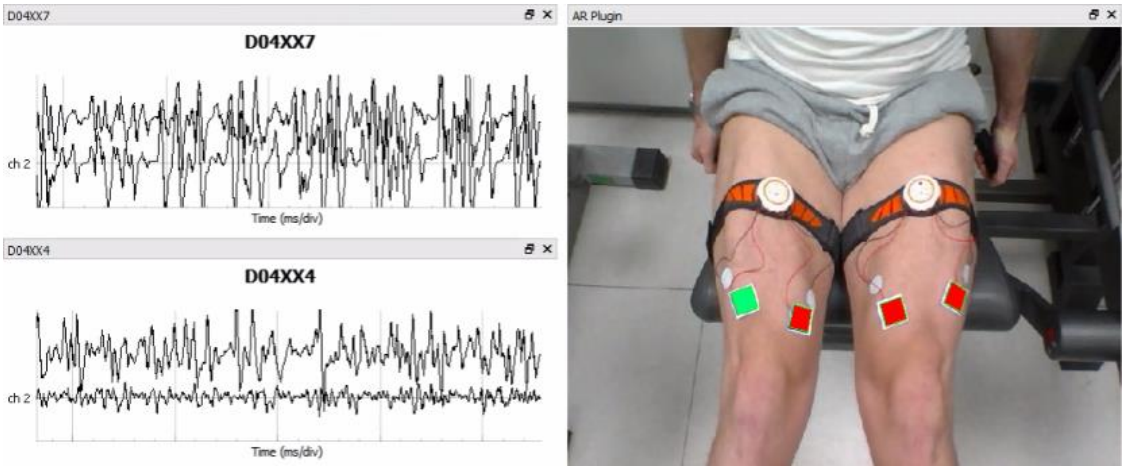


Figura 3.4: Estensione massimale della gamba.

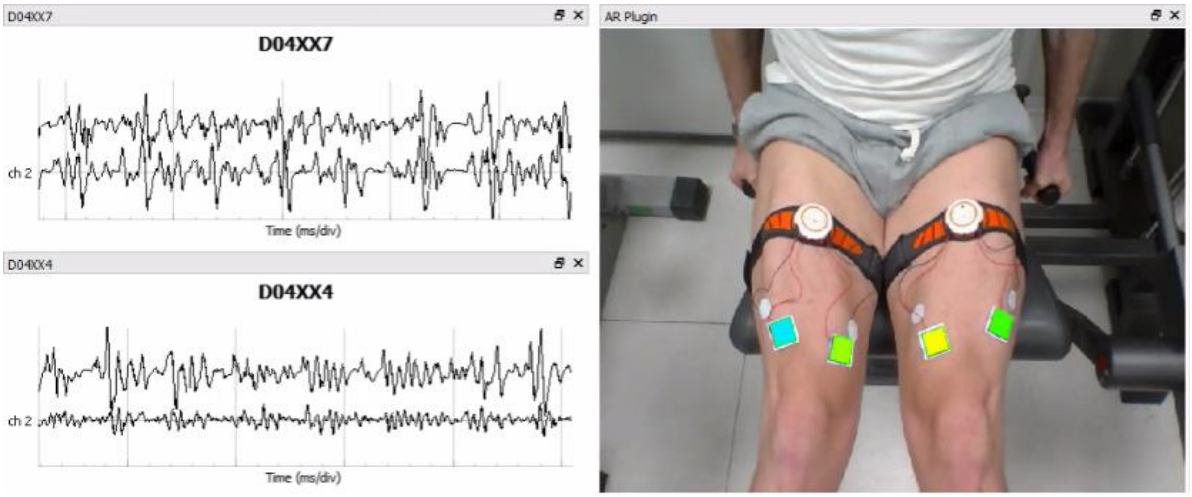


Figura 3.5: Fase iniziale della flessione della gamba.

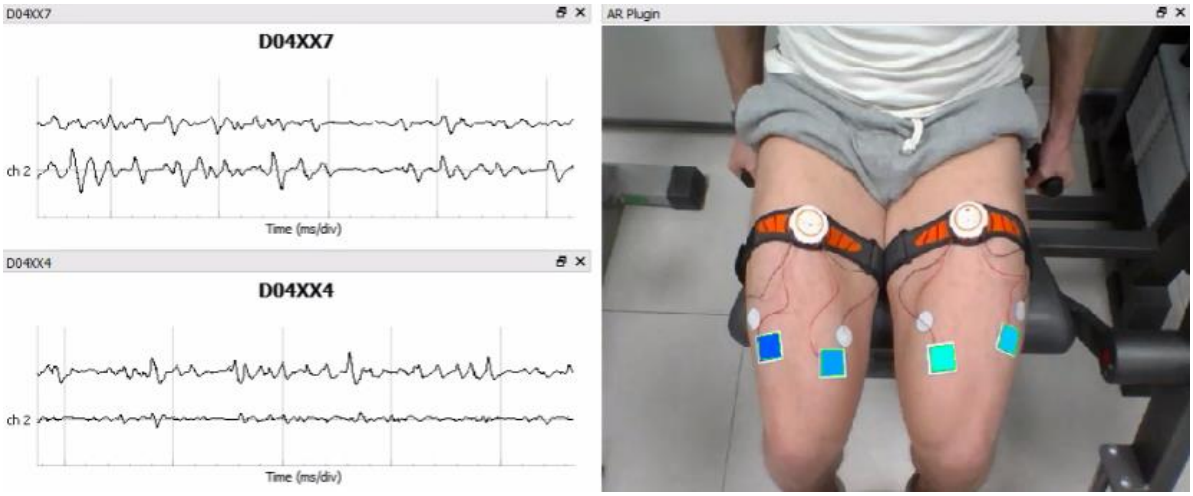


Figura 3.6: Fase finale della flessione della gamba.

La leg extension è stata eseguita impostando un carico pari a 50 Kg. L'informazione relativa all'ampiezza del segnale EMG acquisito, utilizzata per fornire il biofeedback durante l'esecuzione dell'esercizio, è l'ARV. L'intervallo della variabile utilizzata va da 0 mV a 1 mV ed è stato ricavato sperimentalmente modificando l'ampiezza massima rappresentabile fino ad ottenere il feedback visivo desiderato nella fase di estensione massimale della gamba, ovvero la colorazione in rosso dei marker.

Nella Figure 3.2, 3.3 e 3.4, viene riportato il soggetto rispettivamente nella condizione di riposo, fase iniziale dell'estensione della gamba e estensione massimale della gamba. Nelle Figure 3.5 e 3.6, vengono invece riportate rispettivamente la fase iniziale della flessione della gamba e la fase finale della flessione della gamba. Sul lato sinistro delle immagini viene riportata l'intensità del segnale EMG di superficie acquisito dal sistema di prelievo (3.3 mV a divisione), mentre sulla destra viene riportato il biofeedback fornito direttamente sopra la muscolatura interessata.

Nella Figura 3.2 si può notare come in corrispondenza di un'ampiezza del segnale EMG praticamente nulla, venga fornito un feedback visivo di colore blu, indicativo della muscolatura a riposo.

Nella Figura 3.3 è interessante notare come, in corrispondenza della fase iniziale dell'estensione della gamba e quindi ad un'ampiezza del segnale EMG non più nulla, venga mostrato un feedback visivo che presenta una colorazione in verde. In particolare, verde acqua per la gamba sinistra e verde chiaro per la gamba destra. La differenza di colorazione dei marker riportata, corrisponde ad una diversa attività muscolare fra il quadricipite sinistro e il quadricipite destro, ovvero l'attività muscolare del quadricipite sinistro è più bassa rispetto a quella del quadricipite destro. Il segnale EMG riportato sulla sinistra dell'immagine conferma il feedback visivo riportato.

Nella Figura 3.4, in corrispondenza dell'estensione massimale della gamba e quindi all'attività muscolare più intensa, i marker presentano una colorazione rossa, tranne quello posizionato sul VL sinistro che invece presenta una colorazione verdastra. Il feedback visivo quindi riporta un'attivazione muscolare del vasto laterale sinistro più bassa rispetto al resto della muscolatura monitorata. Il segnale EMG riportato sulla sinistra dell'immagine, conferma la bontà del biofeedback in realtà aumentata fornito al soggetto.

Nella Figura 3.5 in corrispondenza della fase iniziale della flessione della gamba, i marker presentano una colorazione verde per il VM sinistro e il VL destro, mentre blu per il VL sinistro e giallo per il VM destro. Globalmente si nota come la colorazione dei marker sta passando dal rosso al blu, indicativo di un'attività muscolare in

diminuzione. Il segnale EMG riportato sulla sinistra dell'immagine conferma il biofeedback fornito dal sistema di realtà aumentata

Nella Figura 3.6 in corrispondenza della fase finale della flessione della gamba, i marker presentano una colorazione prettamente azzurra/celeste, indicativa di un'attività muscolare bassa.

3.2 Curl concentrato manubrio singolo

Il secondo esercizio eseguito è il *curl* concentrato con manubrio singolo. L'esercizio ha come obiettivo quello di isolare il bicipite brachiale che origina da due capi distinti, detti capo lungo e capo breve, per andarsi poi ad inserire con un tendine comune, alla tuberosità del radio. L'esercizio viene svolto da seduti, con i piedi in perfetto appoggio sul terreno. Si impugna un manubrio e si appoggia il gomito sulla parte interna della coscia: il movimento consiste nella flessione dell'avambraccio fino al massimo punto di contrazione muscolare, seguito poi da una lenta estensione. Gli elettrodi per il prelievo del segnale bipolare, sono stati posizionati al fine di prelevare il segnale EMG dal capo lungo e dal capo breve del bicipite brachiale. Il posizionamento degli elettrodi è stato effettuato seguendo quanto riportato nel manuale di riferimento [20]. Sovrapposti agli elettrodi, sono stati posizionati i marker che permettono di ottenere il biofeedback.

Nella Figura 3.7 viene riportato il soggetto strumentato.

Figura 3.7: Soggetto strumentato elettrodi e marker.

Nelle figure sottostanti viene riportata la scena acquisita dalla camera, dove si può osservare il biofeedback fornito durante l'esecuzione dell'esercizio.

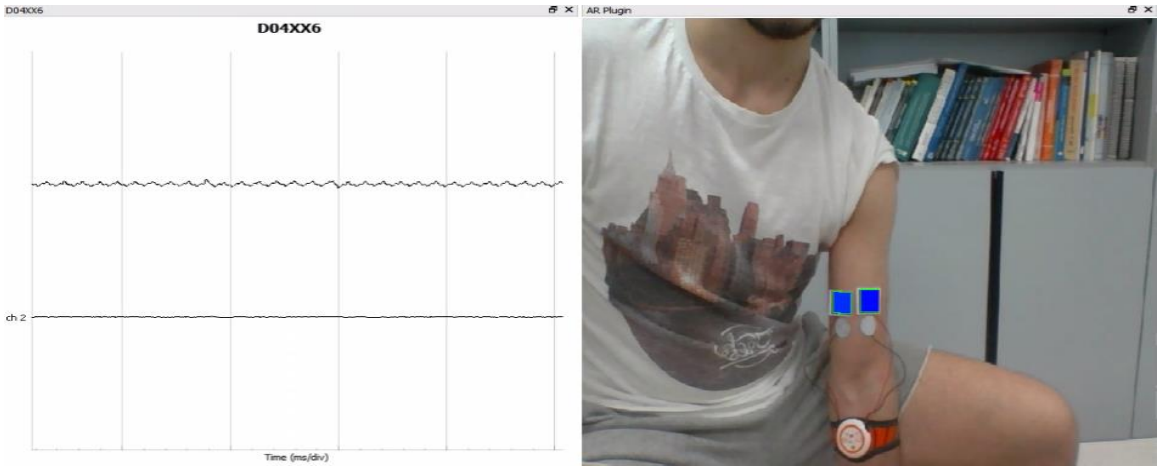


Figura 3.8: Soggetto a riposo.

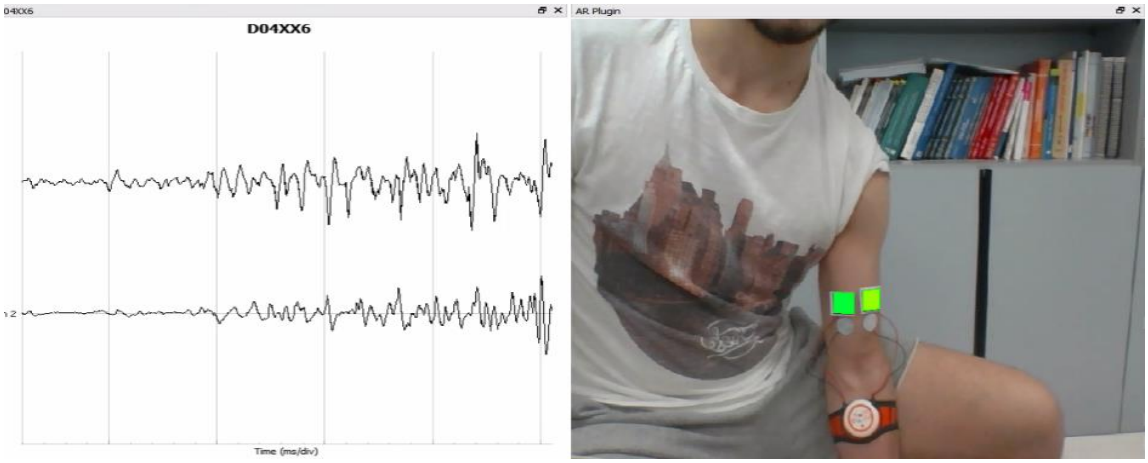


Figura 3.9: Inizio della fase di flessione dell'avambraccio.

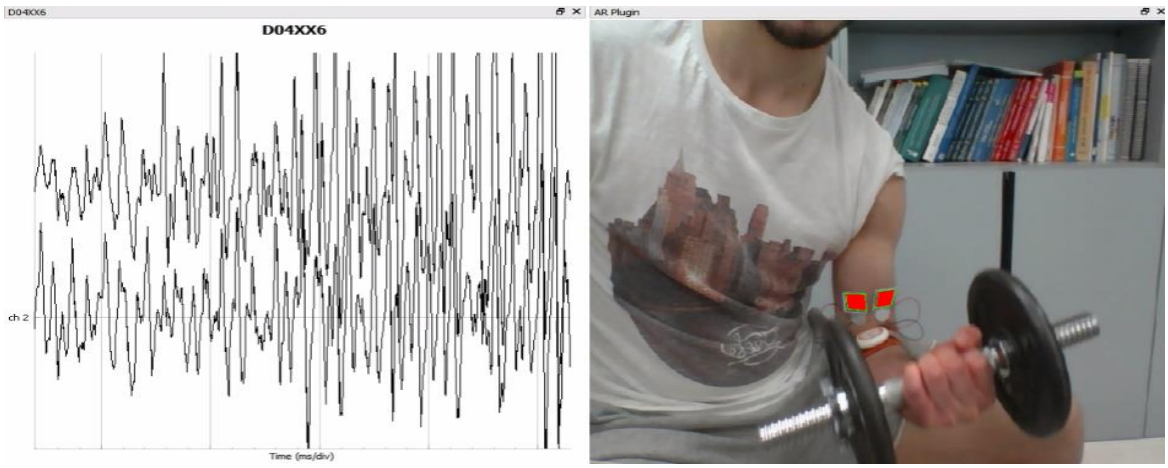


Figura 3.10: Fase finale della flessione dell'avambraccio.

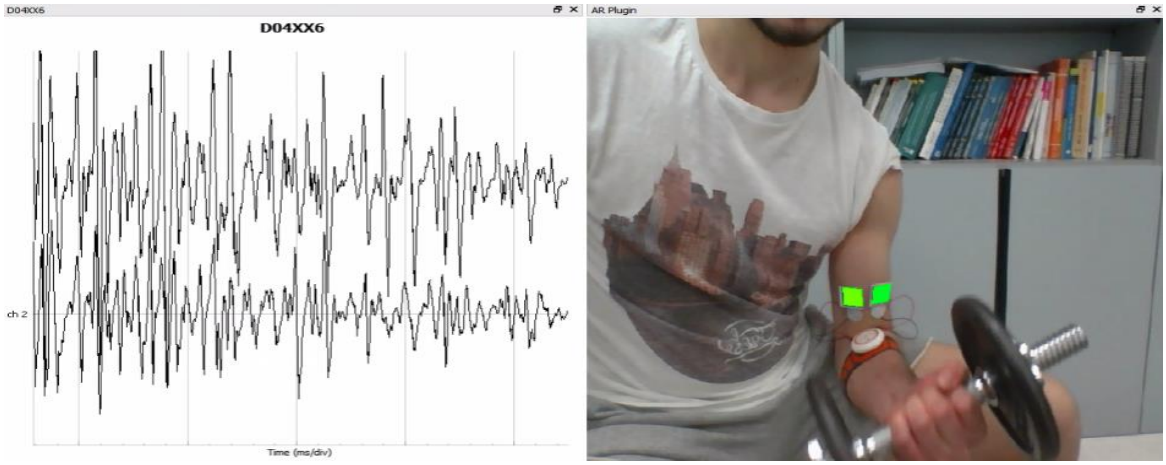


Figura 3.11: Fase iniziale estensione dell'avambraccio.

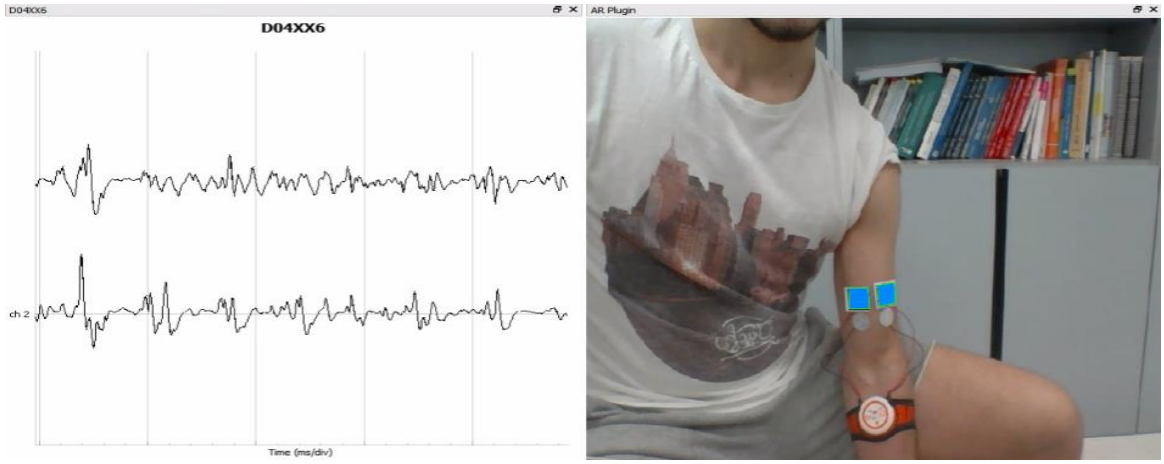


Figura 3.12: Fase finale estensione avambraccio.

Per eseguire l'esercizio di curl concentrato è stato utilizzato un manubrio da 5 kg. L'informazione relativa all'ampiezza del segnale EMG acquisito, utilizzata per fornire il biofeedback durante l'esecuzione dell'esercizio, è l'ARV. L'intervallo della variabile utilizzata va da 0 mV a 0.8 mV ed è stato ricavato sperimentalmente modificando l'ampiezza massima fino ad ottenere il feedback visivo desiderato. Durante la fase finale di flessione dell'avambraccio dove si ottiene un'attivazione muscolare più intensa, il manubrio va a coprire i marker e si perde l'informazione sul feedback visivo. Quindi, ai fini della verifica delle funzionalità del sistema, si è scelto di abbassare la soglia in modo tale da ottenere una colorazione rossa dei marker prima che il

manubrio li copra e quindi prima di raggiungere il termine della fase finale della flessione.

Nelle figure 3.8, 3.9 e 3.10, viene riportato il soggetto rispettivamente nella condizione di riposo, fase iniziale della flessione dell'avambraccio e fase finale della flessione dell'avambraccio. Mentre nelle figure 3.11 e 3.12, viene riportata rispettivamente la fase iniziale dell'estensione dell'avambraccio e la fase finale dell'estensione dell'avambraccio. Sulla sinistra delle immagini viene riportata l'intensità del segnale EMG di superficie acquisito (3.0 mV a divisione) dal sistema di prelievo, mentre sulla destra viene riportato il biofeedback fornito direttamente sopra la muscolatura interessata.

Nella figura 3.8 si può notare come in corrispondenza di un'attività muscolare assente, ovvero in condizioni di riposo, i colori associati al marker siano blu.

Nella figura 3.9 si può osservare come in corrispondenza della prima fase della flessione dell'avambraccio e quindi di un'attività muscolare non nulla, i marker riportino una colorazione verde-gialla. Il segnale EMG riportato sulla sinistra mostra come effettivamente si passa da uno stato di riposo ad uno di attività muscolare, confermando la bontà del feedback visivo

Nella figura 3.10 in corrispondenza della fase finale di flessione dell'avambraccio, quindi in corrispondenza di un'attività muscolare maggiore, il colore riportato sopra i marker è il rosso. Il segnale EMG mostra come all'aumentare della flessione aumenta anche l'attivazione muscolare confermando la coerenza del biofeedback fornito in realtà aumentata.

Nella figura 3.11 si nota che durante la prima fase dell'estensione dell'avambraccio, la colorazione dei marker passa da rosso, al giallo-verde. Questa riduzione nell'attività muscolare è anche ben visibile osservando il segnale riportato sulla sinistra.

Infine, nella figura 3.12 in corrispondenza del termine della fase di estensione dell'avambraccio, il feedback visivo fornito è di colore azzurro indicativo di una bassa attività muscolare. L'intensità del segnale EMG riportata sulla sinistra mostra come effettivamente l'attività muscolare è molto bassa.

4. Conclusioni

Per entrambi gli esercizi si può affermare che il sistema sviluppato è stato in grado di fornire il feedback visivo atteso. Si è verificato che il segnale EMG dopo essere stato acquisito, viene processato al fine di estrarre l'informazione relativa all'ampiezza, ovvero viene calcolato o l'ARV o l'RMS sul segnale, trasformato in codice colore, e viene fornito il biofeedback direttamente sopra al distretto muscolare da cui si sta prelevando il segnale EMG. Il feedback inoltre è coerente con l'intensità del segnale EMG acquisito dalle sonde, il quale può essere visualizzato sulla sinistra delle immagini mostrate nel capitolo 3. In particolare si nota che quando l'attività muscolare è presso che nulla, quindi il muscolo è in una condizione di riposo, si ottiene un feedback visivo di colore blu, mentre quando l'attività muscolare è diversa da zero si ottiene una colorazione del marker che varia da azzurro al rosso come stabilito. Il rosso inoltre si ottiene al raggiungimento, da parte della variabile che riporta le informazioni relative all'ampiezza del segnale elettromiografico, della soglia impostata, ovvero in corrispondenza della massima attività muscolare che si è in grado di codificare in colore prima di raggiungere la saturazione del colore stesso.

Riferimenti

[1] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas e M. Marín-Jiménez, «Automatic generation and detection of highly reliable fiducial markers under occlusion,» *Pattern Recognition volume 47,* pp. 2280-2292, 6 2014.

[2] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas e R. Medina-Carnicer, «Generation of fiducial marker dictionaries using Mixed Integer Linear Programming,» *Pattern Recognition,* vol. 51, pp. 48-491, 2016.

[3] M. Fiala, «Designing Highly Reliable Fiducial Markers,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 32, n. 7, pp. 1317-1324, 2010.

[4] V. A. Knyaz e R. V. Sibiryakov, «The development of new coded targets for automated point identification and non-contact surface measurements,» *3D Surface Measurements, International Archives of Photogrammetry and Remote Sensing,* vol. XXXII, n. 5, pp. 80-85, 1998.

[5] L. Naimark, E. Foxlin, Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker, in: Proceedings of the 1st International Symposium on Mixed and Augmented Reality, ISMAR '02, IEEE Computer Society, Washington,DC, USA, 2002, pp. 27

[6] M. Kaltenbrunner, R. Bencina, reactivision: a computer-vision framework for table-based tangible interaction, in: Proceedings of the 1st international conference on Tangible and embedded interaction, TEI '07, ACM, New York, NY, USA, 2007, pp. 69{74.

[7] H. Kato, M. Billinghurst, Marker tracking and hmd calibration for a video-based augmented reality conferencing system, in: Proceedings of the 2nd IEEE and ACM InternationalWorkshop on Augmented Reality, IWAR '99, IEEE Computer Society, Washington, DC, USA, 1999, pp. 85{.

[8] M. Fiala, Comparing artag and artoolkit plus fiducial marker systems, in: IEEE International Workshop on Haptic Audio Visual Environments and their Applications, 2005, pp. 147{152.

[9] N. Otsu, A threshold selection method from gray-level histograms, IEEE Transactions on Systems, Man and Cybernetics 9 (1) (1979) 62{66.

[10] R. T. Azuma, “A survey of augmented reality,” Presence: Teleoperators and Virtual Environments, vol. 6, no. 4, pp. 355–385, 1997.

[11] T. Sielhorst, M. Feuerstein and N. Navab, "Advanced Medical Displays: A Literature Review of

Augmented Reality," in Journal of Display Technology, vol. 4, no. 4, pp. 451-467, Dec. 2008.

[12] E. Z. Barsom, M. Graafland, M. P. Schijven, "Systematic review on the effectiveness of augmented reality applications in medical trainin," in Journal Article published 23 Feb 2016 in Surgical Endoscopy volume 30 issue 10 on pages 4174 to 4183.

[13] Sylvain Bernhardt, Stéphane A. Nicolau, Vincent Agnus, Luc Soler, Christophe Doignon, Jacques Marescaux, "Automatic localization of endoscope in intraoperative CT image: A simple approach to augmented reality guidance in laparoscopic surgery,Medical Image Analysis," Volume 30, 2016, Pages 130-143.

[14] Y. M. Aung, A. Al-Jumaily and K. Anam, "A novel upper limb rehabilitation system with self-driven virtual arm illusion," 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, IL, 2014, pp. 3614-3617.

[15] Im DJ, Ku J, Kim YJ, et al., "Utility of a Three-Dimensional Interactive Augmented Reality Program for Balance and Mobility Rehabilitation in the Elderly: A Feasibility Study." Annals of Rehabilitation Medicine. 2015;39(3):462-472.

[16] Heyn PC, Baumgardner CA, McLachlan L, Bodine C. Mixed-Reality Exercise Effects on Participation of Individuals with Spinal Cord Injuries and Developmental Disabilities: A Pilot Study. Topics in Spinal Cord Injury Rehabilitation. 2014;20(4):338-345.

[17] Ortiz-Catalan M, Sander N, Kristoffersen MB, Håkansson B, Brånemark R. Treatment of phantom limb pain (PLP) based on augmented reality and gaming controlled by myoelectric pattern recognition: a case study of a chronic PLP patient. Frontiers in Neuroscience. 2014;8:24.

[18] Merletti, Roberto, Dario Farina. “Surface Electromyography: Physiology, Engineering, and Applications, 2016.

[19] Giggins OM, Persson UM, Caulfield B. Biofeedback in rehabilitation. Journal of NeuroEngineering and Rehabilitation. 2013;10:60.

[20] Barbero, Marco. Merletti, Roberto. Rainoldi, Alberto. ATLAS OF MUSCLE INNERVATION ZONES: Understanding Surface Electromyography and Its Applications. SPRINGER VERLAG, 2016.

[21] Kamen, Gary, and David A. Gabriel. Essentials of Electromyography. Human Kinetics, 2010.

[22] Merletti, Roberto. Electromyography: Physiology, Engineering and Noninvasive Applications. Wiley-Interscience, 2010.