Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

---

Ricerca e sviluppo di un sistema di telemedicina per il monitoraggio remoto del rischio embolico nei subacquei e delle fluttuazioni motorie nella malattia di Parkinson

Tesi di Dottorato di:
Lorenzo Maurizi

Tutor:
Prof.ssa Paola Pierleoni

Coordinatore del Curriculum:
Prof. Francesco Piazza

XVI ciclo - nuova serie

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

---

Ricerca e sviluppo di un sistema di telemedicina per il monitoraggio remoto del rischio embolico nei subacquei e delle fluttuazioni motorie nella malattia di Parkinson

Tesi di Dottorato di:
Lorenzo Maurizi

Tutor:
Prof.ssa Paola Pierleoni

Coordinatore del Curriculum:
Prof. Francesco Piazza

XVI ciclo - nuova serie

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

A tutti quelli che mi hanno sostenuto in questo lungo percorso!!

Ringraziamenti

Ancona, Gennaio 2018

Lorenzo Maurizi

Sommario

Il tema di ricerca è principalmente inquadrato, ma non solo, nell'ambito dello studio di soluzioni ICT per il monitoraggio remoto di parametri fisiologici, sia su persone con deficit motori, che su persone sane in ambito sportivo e non. Tali soluzioni si basano sulla progettazione di sistemi di video-conferenza che, grazie alla tecnologia multipiattaforma WebRTC, permettano ad ogni soggetto di dialogare con il proprio medico o specialista di riferimento, inviando, se necessario, opportuni report forniti da dispositivi biomedicali custom, specifici per ogni contesto di utilizzo. Lo scopo di questa prima fase di tesi è stato quindi quello di analizzare ed implementare un sistema di comunicazione peer-to-peer tra client browser in grado, non solo di realizzare la normale video/conferenza ormai già consolidata, ma anche di inviare offline e online dati acquisiti da sensori biomedicali. Realizzata la piattaforma, sono stati individuati diversi campi applicativi che abbiano come base di funzionamento l'utilizzo del sistema di monitoraggio remoto, precedentemente sviluppato. Il primo contesto applicativo è nato grazie alla collaborazione con il DAN, "Divers Alert Network", organizzazione mondiale, che opera da anni nella prevenzione e sicurezza dei subacquei durante tutte le loro campagne di immersione. Lo scopo è stato quello di realizzare dispositivi in grado di analizzare eco Doppler audio e identificare in real-time il numero di bolle gassose contenute in essi ed il relativo grado di rischio e livello di Spencer associato. I dispositivi sono in grado di inviare i report generati, mediante il sistema peer-to-peer realizzato, ad una centrale di pronto intervento, dove un operatore, da una veloce analisi del report fornito, è in grado di determinare il tipo di intervento da attuare sul subacqueo (somministrazione di ossigeno, camera iperbarica o altro) e la tempistica di intervento in loco. Successivamente, il campo applicativo sul quale impiegare i precedenti studi, si è spostato su soggetti affetti da deficit motori come tremore, freezing e fluttuazioni, dovuti principalmente alla presenza del Parkinson, malattia neurodegenerativa invalidante causata da una riduzione dei livelli di dopamina. A partire da dati grezzi provenienti da sensori inerziali, come accelerometro, giroscopio e magnetometro, sono stati realizzati algoritmi di data fusion, che hanno permesso di rilevare, in maniera del tutto automatica, manifestazioni dei sintomi del Parkinson. Opportune GUI permettono di generare report ed inviarli, con la tecnologia WebRTC, al team medico, che è in grado di personalizzare la terapia farmacologica in relazione alle manifestazioni

e al progredire della malattia stessa.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Il mondo dell'ingegneria dell'informazione ha prodotto, negli ultimi decenni, moltissime innovazioni che, con l'abbassamento dei costi e la miniaturizzazione dei componenti, hanno garantito un loro facile e diffuso utilizzo in tutti i settori, passando dal mondo industriale a quello domestico. Il principale mezzo di comunicazione, il cui esponenziale sviluppo ha permesso di creare il cosiddetto "villaggio globale", è internet, rete di reti, dove da tutto il mondo sistemi di produzione differenti possono comunicare, scambiandosi grandi moli di dati. Se però da un lato l'evoluzione di internet ha portato alla possibilità di usufruire, dalla propria postazione, di un numero di servizi sempre maggiore, ad esempio si pensi alla banca on-line, dove informazioni sensibili viaggiano da una parte all'altra del mondo, dall'altro è stato necessario affrontare nuove problematiche legate alla sicurezza e privacy dei dati, che viaggiano in questa grande infrastruttura. Uno dei settori che ha beneficiato dello sviluppo di internet è quello della "Telemedicina", ovvero lo scambio di dati utili alla diagnosi, al trattamento e alla prevenzione delle malattie in generale, quando la distanza è un fattore critico [1]. La Telemedicina pone al centro del sistema il paziente ed il suo stato di salute; operatori sanitari, dislocati in diverse parti del territorio, potranno erogare svariate prestazioni grazie allo scambio di informazioni corrette, sicure e di alta qualità. Le informazioni scambiate saranno utili al monitoraggio remoto di parametri e segnali vitali dei pazienti [2]. Altro sviluppo non trascurabile è quello della sensoristica, che ha permesso la realizzazione di dispositivi indossabili sempre più piccoli, impiegati in un gran numero di applicazioni, comprese quelle per il monitoraggio della salute, della forma fisica e della sicurezza dell'uomo. In campo biomedicale questi dispositivi possono sia monitorare parametri fisiologici che essere usati come sistemi di rilevamento intelligenti, individuando cadute ed avvertendo tempestivamente persone di riferimento. L'obiettivo del lavoro di ricerca, condotto nel corso dei tre anni di dottorato, ha riguardato lo studio e l'implementazione di un sistema di telemedicina, di facile utilizzo, in grado di fornire supporto medico real-time sia in ambito sanitario che sportivo. Il sistema fonda il suo sviluppo su WebRTC, una collezione di standard, protocolli e API, che permet-

tono l'instaurazione di una comunicazione peer-to-peer tra medico e paziente, in grado di scambiare contenuto audio, video e dati. Questo stile architetturale, opposto a quello Client-Server, su cui è basata la stragrande maggioranza del Web, consente di ridurre al minimo i tempi di latenza, i costi di gestione e di garantire una facilità di utilizzo anche a chi non ha un'elevata padronanza delle tecnologie dell'informazione e degli strumenti Web in particolare. Inizialmente il lavoro si è incentrato sullo sviluppo del sistema in grado di consentire una comunicazione Web real-time tra due peer, simulando il monitoraggio remoto di un paziente tramite sensori indossabili, atti a rilevare parametri fisiologici come l'elettrocardiogramma. Attraverso l'utilizzo di una Board Arduino, i dati raccolti dai sensori sono stati elaborati ed inviati wired al pc del paziente che, a sua volta, sfruttando le apposite API del WebRTC, si fa carico di pacchettizzarli ed inoltrarli al browser del medico, a cui è affidato il compito della post-elaborazione e visualizzazione. Il sistema di Telemedicina realizzato è stato poi usato come strumento di inoltro, sia di dati grezzi provenienti da sensoristica o schede di acquisizione di vario genere, che di report sullo stato di salute complessivo del soggetto in esame, a dei team medici di supporto. Il primo caso di studio si è incentrato sull'implementazione di algoritmi di signal processing, in grado di identificare bolle gassose in segnali Eco Doppler audio. A partire da dei segnali audio, registrati mediante analisi eco Doppler su subacquei nella fase di post emersione, sono state messe a punto delle tecniche di analisi sia nel dominio del tempo che della frequenza, al fine di estrarre features notevoli in grado di caratterizzare la presenza di una bolla gassosa o di una loro sequenza chiamata "Shower". L'algoritmo è stato implementato su una scheda embedded commerciale, consentendo un'acquisizione audio ed una elaborazione dello stesso in real-time. I risultati ottenuti, sia in numero di bolle o shower presenti che nel grado di rischio del subacqueo, espresso mediante scala di Spencer (SS) o scala di Spencer estesa (ESS), sono stati inviati con il portale WebRTC al team medico di riferimento che, valutando i dati ricevuti del subacqueo, ha deciso la terapia a cui sottoporlo (somministrazione di ossigeno, camera iperbarica o altro). Al fine di realizzare un algoritmo con un'elevata affidabilità, i risultati prodotti dall'elaborazione di un determinato file audio, sono stati confrontati con quelli prodotti manualmente (annotazioni manuali) da più medici iperbarici sul medesimo audio. Vista la non uniformità sia in numero di bolle annotate che nella forma dell'annotazione stessa, si è deciso di realizzare, congiuntamente all'algoritmo di rilevazione bolle, un software in grado sia di facilitare l'attività di annotazione bolle da parte dei team medici che di fungere da strumento di training per gli esaminatori stessi. Come secondo caso di studio sono stati implementati algoritmi di sensor data fusion per la realizzazione di applicazioni nel campo della riabilitazione motoria. Partendo da un'unità di sensing composta da sensori MEMS, quali accelerometro

triassiale, giroscopio triassiale, magnetometro triassiale e barometro, un'elaborazione dei dati grezzi, in uscita dagli stessi, ha permesso di determinare il movimento del sensore nello spazio e l'altitudine del segmento corporeo sul quale viene applicato. L'integrazione delle velocità angolari, prodotte in uscita dal giroscopio dopo un ben definito intervallo di tempo, permette di ottenere l'orientazione del sensore. Questa però risulterà affetta da un errore di bias, che rende indispensabile l'uso combinato anche degli altri due sensori, accelerometro e magnetometro. Anche i dati forniti in uscita da questi due ulteriori sensori, da soli non risulterebbero utilizzabili poichè non esenti da errori, dovuti ad esempio ad accelerazioni non gravitazionali percepite dal sensore o a variazioni di campo magnetico prodotte da apparecchiature elettroniche nelle vicinanze del sensore stesso. Da qui l'esigenza di realizzare un algoritmo che elabori i dati di tutti i sensori a disposizione in maniera combinata, al fine di monitorare h24 persone affette da malattie degenerative come il Parkinson, e capace di rilevare automaticamente e quantificarne in maniera più oggettiva possibile le fluttuazioni (tremori, freezing, discinesie, ecc.); il tutto finalizzato al follow-up della malattia ed alla personalizzazione e verifica dell'efficacia della terapia stessa. L'algoritmo realizzato permette anche una correlazione accurata con il relativo UPDRS (Unified Parkinson's Disease Rating Scale), scala standardizzata, usata negli studi clinici per la valutazione della gravità e della progressione della malattia stessa. Il dispositivo inerziale utilizzato, dotato di interfaccia Bluetooth, ci ha permesso l'inoltro di dati grezzi e dei risultati delle loro elaborazioni, a smartphone o pc, contattando direttamente l'API WebRTC per il monitoraggio da remoto da parte del team medico dell'evoluzione della malattia e dei sintomi ad essa associati.

Capitolo 2

Web Real Time Communication

La fruibilità delle informazioni Web è consentita dall'utilizzo di browser cioè software in grado di interpretare codice Hyper Text Markup Language (HTML) restituito dal servizio Web contattato. L'HTML può essere definito come un linguaggio di "markup" che descrive la formattazione di pagine Web, senza però consentire interazione con l'utente ed utilizzando particolari elementi chiamati "tag". Con lo sviluppo tecnologico sono state prodotte versioni sempre nuove dell'HTML, fino ad arrivare alla versione attuale HTML5 dove, oltre ad esser stati aggiunti nuovi tag, è stato introdotto l'uso di APIs (Application Program Interfaces), in grado di consentire una maggiore interazione tra pagina Web ed utente, sfruttando il linguaggio JavaScript. Il WebRTC può essere definito come una collezione di standard, protocolli e nuove API JavaScript che permettono la condivisione peer-to-peer (P2P) di audio, video e dati. La nascita del WebRTC non è finalizzata solo all'abilitazione della comunicazione real-time tra browser, ma è anche progettata in modo da poter esser facilmente integrabile in sistemi di comunicazione già preesistenti come quelli Voice Over IP (VOIP), SIP Clients e la rete di telefonia pubblica (PSTN). Gli standard del WebRTC sono tuttora in fase di sviluppo ed integrazione da parte sia del World Wide Web Consortium (W3C) che dal gruppo di lavoro dell'Internet Engineering Task Force (IETF). Mentre W3C si occupa della definizione delle API [3], necessarie alle varie applicazioni per interagire con funzioni di comunicazione real-time implementate dai browser, all'IETF è affidato il compito di definire i protocolli usati nelle comunicazioni web tra browser o altri endpoint. Le caratteristiche, fin qui descritte del WebRTC, oltre a quelle classiche dei servizi e-Health precedentemente realizzate [4], hanno incentivato il suo uso per sviluppare un servizio di videoconferenza real-time in peer-to-peer tra un medico ed un paziente dove, oltre ad audio e video, vengono scambiati anche dati acquisiti da sensori biomedicali, come proposto anche da Azevedo et al. [5]. I dati saranno acquisiti da un dispositivo prototipo, realizzato ai fini dimostrativi di funzionamento del sistema, in grado di acquisire un segnale biomedicale su porta seriale ed inviarlo al peer destinatario mediante servizio WebRTC.

Figura 2.1: Schema servizio WebRTC

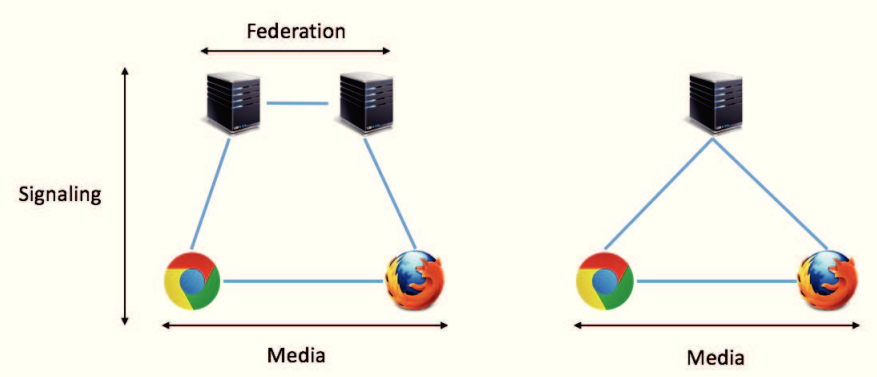

Figura 2.2: Architettura WebRTC trapezoidale (sinistra) e triangolare (destra)

2.1 Architettura WebRTC

L'ormai noto Web ha un'architettura basata sul modello client-server. I browser fungono da client ed inviano una richiesta mediante protocollo HyperText Transfer Protocol (HTTP) ad un server Web. A sua volta, se la richiesta rispetta tutti i canoni di integrità, autenticazione e autorizzazione, verrà gestita dal server che risponderà al client con le opportune informazioni richieste. Il WebRTC invece utilizza il nuovo paradigma architetturale peer-to-peer, garantendo una comunicazione tra i soli browser. In realtà l'utilizzo di un server è necessario almeno nella fase iniziale, dove i clients non conoscono l'indirizzo del peer con cui vogliono comunicare ed è quindi necessario un server che funga da intermediario in questa fase iniziale chiamata "Signaling". I clients utilizzeranno i server non solo per identificare l'indirizzo del peer da contattare, ma anche per scaricare le pagine web necessarie alla loro comunicazione. Se in una comunicazione, i due peer scaricano le medesime pagine da due server differenti, entrambi appartenenti ad un dominio federato, l'architettura avrà uno schema trapezoidale, altrimenti, se scaricate dallo stesso server, l'architettura avrà uno schema triangolare [6].

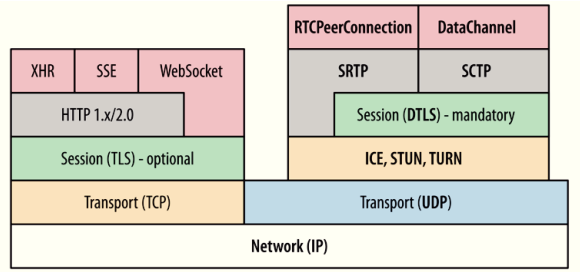

Figura 2.3: Stack protocollare WebRTC *(hpbn.co)*

2.2 Protocolli

A differenza dei più comuni protocolli usati nelle consuete comunicazioni web, l'istaurazione di una comunicazione WebRTC tra due peer necessita dell'utilizzo di differenti protocolli illustrati nella figura 2.3.

2.2.1 UDP

Vista la maggior sensibilità alla latenza, spesso considerata più importante dell'affidabilità nelle comunicazioni real-time, nel livello di trasposto si preferisce l'uso del protocollo UDP. Le motivazioni di tale scelta derivano dal protocollo TCP stesso, che consente un flusso di dati affidabile e ordinato con l'uso di messaggio di ACK. Se durante una comunicazione, un pacchetto viene perso, il TCP bufferizza tutti i pacchetti ricevuti, aspetta la ritrasmissione del pacchetto mancante e solo alla fine, mette a disposizione i pacchetti ricevuti in maniera ordinata alle varie applicazioni. Il protocollo UDP invece, non effettuando questi controlli, mette subito a disposizione alle applicazioni il pacchetto ricevuto.

2.2.2 DTLS

Ogni dato, che transita tra due peers in una comunicazione WebRTC, deve essere opportunamente criptato. Poichè non si ha un'affidabilità e un ordine di ricezione dei pacchetti, utilizzando UDP al posto del TCP, il protocollo Transport Layer Security (TLS) non può essere utilizzato, ricorrendo così alla sua versione modificata, chiamata Datagram Transport Layer Security (DTLS). Il protocollo TLS può essere suddiviso in tre grandi fasi di funzionamento:

- Negoziazione fra i peers coinvolti nella comunicazione del protocollo di cifratura e del protocollo di scambio chiavi;
- Scambio delle chiavi di cifratura ed autenticazione;

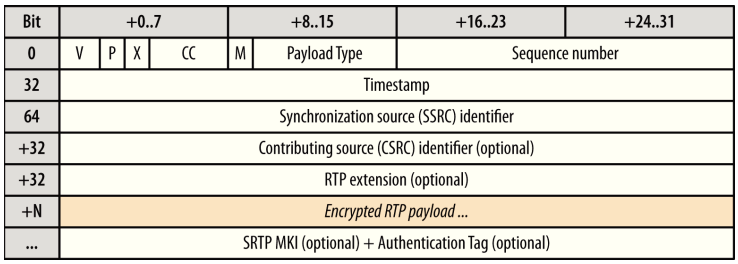

Figura 2.4: Formato pacchetto SRTP *(hpbn.co)*

- Cifratura simmetrica dei messaggi.

Nella fase iniziale, lo scambio di informazioni inerenti ai protocolli da utilizzare per criptare le informazioni inviate tra peers e sulle chiavi usate, avviene attraverso l'uso di una sequenza di handshake prevista nel protocollo TLS. Viste le limitazioni nell'uso dell'UDP, il DTLS si occupa di realizzare un mini-TCP, dividendo l'handshake in più record, ognuno dotato di un offset e di un sequence number. Il DTLS si occupa inoltre anche dei pacchetti persi, introducendo un tempo di ritrasmissione per i record di handshake di cui non si ha avuto risposta.

2.2.3 SRTP e SRTCP

In una comunicazione WebRTC, dove vengono inviati dati di dispositivi esterni di un peer ad un secondo peer, spesso possono essere imposti dei vincoli unicamente sul formato o qualità del dato esterno acquisito. Ciò che invece non viene fatto direttamente, ma gestito dal browser, sono i controlli di congestione e di flusso sul traffico tra peers. Questo è dovuto alla difficoltà del livello applicativo di gestire una comunicazione in un canale con banda fluttuante e latenza. I browsers, attraverso l'uso dei protocolli Secure Real-time Transport Protocol (SRTP) e Secure Real-time Transport Control Protocol (SRTCP), avviano una comunicazione audio video a basso bit rate, andando poi a modificarne la qualità a seconda della banda disponibile. L'SRTP definisce a sua volta un formato standard del pacchetto, che raccoglie frame audio video insieme a dati ausiliari, per la ricostruzione del pacchetto in ricezione. Il formato del pacchetto è mostrato in figura 2.4. Le informazioni di ogni pacchetto sono:

- *Sequence Number*, permette di identificare l'ordine di ricezione dei pacchetti;
- *Timestamp*, tempo di campionamento del primo byte contenuto nel campo payload. Viene usato per la sincronizzazione dei flussi multimediali, come tracce audio e video;

Tabella 2.1: Confronto tra TCP, UDP e SCTP

	TCP	**UDP**	**SCTP**
Affidabilità	Affidabile	Non Affidabile	Configurabile
Consegna	Ordinato	Non Ordinato	Configurabile
Trasmissione	Stream	Messaggi	Messaggi
Controllo di Flusso	Si	No	Si
Controllo di Congestione	Si	No	Si

- *SSRC*, identificatore usato per assegnare in maniera univoca ad ogni pacchetto uno stream;
- *Payload*, campo dati criptato;
- *Authentication Tag*, usato per verificare l'integrità del pacchetto.

Il protocollo SRTCP realizza un canale di feedback tra peers per ogni flusso multimediale, tenendo traccia di statistiche dell'SRTP come il numero di byte / pacchetti inviati e ricevuti e sull'ultimo sequenze number dell'ultimo pacchetto SRTP ricevuto. Queste informazioni scambiate tra peers e il canale di feedback, vengono usate per modificare la rate trasmissiva, la codifica ed altri parametri dello stream.

2.2.4 SCTP

Il protocollo SRTP nasce solo per l'invio di audio video tramite l'opportuna API DataChannel ma, poichè è necessario spesso inviare anche dati generici propri di ogni client, il WebRTC fa ricorso al protocollo Stream Control Transmission Protocol (SCTP), che funziona direttamente con l'IP nel canale sicuro creato con il DTLS. Le maggiori differenze tra SCTP, TCP e UDP sono mostrate in Tab. 2.1. Dalla tabella sopra viene evidenziato come il protocollo SCTP consente di realizzare un canale affidabile, una consegna ordinata dei messaggi ed implementare dei meccanismi atti ad effettuare un controllo di flusso e della congestione. In una comunicazione tra due clients, una connessione SCTP può trasportare più flussi indipendenti, in ogni flusso possono essere inviati più messaggi che a loro volta vengono divisi in blocchi, chiamati chunks. Ogni chunks viene inserito in un pacchetto SCTP e inviato al client destinazione, che assemblandoli, ottiene il messaggio originale. La struttura del pacchetto SCTP è mostrata in figura 2.5. Le informazioni contenute in ogni pacchetto sono:

- *Source port*, porta sorgente;
- *Destination port*, porta destinazione;
- *Verification Tag*, tag casuale di verifica della connessione SCTP;

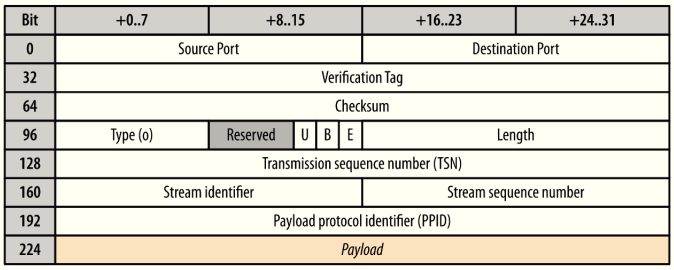

Figura 2.5: Formato pacchetto SCTP *(hpbn.co)*

- *Checksum*, codice di controllo per l"intero pacchetto;
- *Type*, campo che identifica il tipo di chunk, se contiene dati (DATA) o informazioni di controllo (INIT, INITACK, ecc.);
- *Flags*, campi ausiliari, il cui contenuto varia in base al tipo di dato. Ad esempio, per il tipo DATA, U indica se il chunk appartiene ad un canale ordinato o no, i bit B ed E indicano l'inizio e la fine di un messaggio diviso in più chunks;
- *Length*, indica la lunghezza del chunk includendo header e payload;
- *Transmission sequence number (TSN)*, campo utilizzato per verificare la corretta ricezione di un chunk o per indicare eventuali consegne multiple;
- *Stream identifier*, campo per associare lo stream al chunk;
- *Stream sequence number*, numero auto-incrementato univoco per ogni stream, i messaggi frammentati hanno lo stesso numero;
- *Payload protocol identifier (PPID)*, indicano il tipo di dati contenuti nel chunk, ad esempio UTF-8 o binari [7];
- *Payload*, dati inviati al client destinazione.

L'SCTP supporta più flussi contemporanei e ogni messaggio è legato ad un flusso con un identificativo univoco (Stream identifier); a sua volta ogni messaggio contiene un numero di sequenza (TSN) in modo da consentire una ricezione ordinata.

2.2.5 ICE, STUN e TURN

Gli endpoint clients, situati in reti domestiche o aziendali, si trovano confinati all'interno della rete stessa mediante l'utilizzo di dispositivi router o gateway che, spesso, svolgono la funzione di Network Address Translation (NAT). La

NAT è una tecnica attraverso la quale si assegna un indirizzo IP privato ad ogni client della rete considerata, contattando l'esterno mediante l'uso dell'IP pubblico del router. Se un client, all'interno della rete, vuole contattare una risorsa esterna alla rete stessa, tale comunicazione può avvenire mediante l'uso dell'IP pubblico del router. La comunicazione inversa, da una risorsa esterna alla rete ad un client interno, può avvenire solo se è stato effettuato un opportuno mapping tra IP-porta del client interno e IP-porta del router stesso. Il NAT, spesso utilizzato per proteggere i clients e limitare lo spreco di IPv4, non può essere usato nelle reti peer-to-peer, dove gli endpoints devono potersi contattare direttamente senza intermediari. Per risolvere tali impedimenti, sono state sviluppate delle tecniche di NAT Traversal (NAT-T) come ad esempio [8]:

- Universal Plug and Play (UpnP);
- Session Traversal Utilities for NAT (STUN);
- Application Level Gateway (ALG);
- UDP/TCP hole punching.

La tecnica NAT-T, che viene usata nel WebRTC, è il protocollo Internet Connectivity Establishment (ICE), il quale fa uso dello STUN o, nel caso esso fallisca, del Traversal Using Relays around NAT (TURN). In una comunicazione WebRTC i peers coinvolti ignorano la propria topologia di rete e, quindi, potrebbero essere dietro ad una o più NAT, come non esservi affatto. Da qui la necessità di usare l'ICE, protocollo che permette ai peers di scoprire la propria topologia di rete e quindi determinare uno o più percorsi per la loro comunicazione. Prima della comunicazione stessa i due peers si scambiano messaggi, basati sul modello offer/answer, contenenti coppie di indirizzi IP e porte da poter usare per l'invio di audio, video e dati. Tali indirizzi potrebbero essere relativi ad un'interfaccia di rete dell'endpoint stesso, come di una sua NAT o un indirizzo fornito dal TURN server. Ovviamente non è detto che tutte queste coppie garantiscano una comunicazione, quindi l'ICE stesso, attraverso un test sistematico di tutti gli indirizzi, si accerta di quali coppie funzionano correttamente e di quali no. Il protocollo Session Traversal Utilities for NAT (STUN) è un protocollo usato per risolvere problematiche di NAT-T, come determinare l'indirizzo e la porta di un endpoint associato dal NAT, testare la connettività tra due endpoints o come protocollo di *keep-alive* per mantenere attivo il legame dietro la NAT. Diversi sono gli utilizzi di tale protocollo per applicazioni VOIP o SIP [9]. Quando due peers vogliono comunicare tra loro, mediante UDP ma non conoscono i rispettivi IP e porte con cui sono visibili dall'esterno, devono inviare una richiesta di collegamento allo STUN Server che, a sua volta, replicherà con una risposta contenente l'IP pubblico e la porta

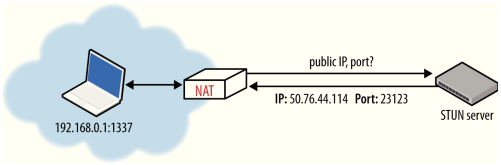

Figura 2.6: Funzionamento STUN Server *(hpbn.co)*

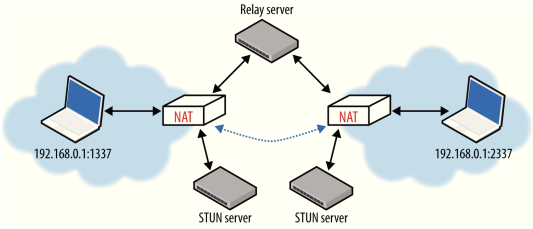

Figura 2.7: Funzionamento con server STUN e TURN *(hpbn.co)*

con cui tale peer viene visto dalla rete pubblica. Ovviamente tale configurazione resta possibile quando l'indirizzo IP dello STUN Server è noto a priori. Nella realtà lo STUN Server non consente di gestire tutte le topologie di NAT ed inoltre alcuni pacchetti UDP potrebbero essere bloccati da Firewall. Per ovviare a tale situazione può essere usato il protocollo Traversal Using Relays around NAT (TURN). Il TURN, estensione dello STUN, permette ai peers di comunicare con uno Server TURN allocando delle porte su una sua interfaccia, utilizzate per ritrasmettere il traffico da un peer all'altro. Ogni peer, dopo una fase iniziale di connessione e negoziazione dei permessi con il Server TURN, invierà i dati che il server prontamente inoltrerà all'altro peer. Tale scenario, per quanto affidabile, ha come svantaggio un cambio di architettura da peer-to-peer a client-server. Il WebRTC utilizza il Server TURN come ultima risorsa a causa degli elevati costi computazionali nel gestire le comunicazioni di tutti i peers. È evidente come l'uso combinato di Server TURN, STUN e ICE è una soluzione efficace per problematiche di NAT-T in comunicazioni multimediali [10]. Nelle attuali comunicazioni WebRTC l'utilizzo dello STUN è di circa il 92% mentre il restante 8% fa uso del Server TURN [6].

2.2.6 Signaling e SDP

In una comunicazione WebRTC ogni client, prima di instaurare una comunicazione peer-to-peer, deve poter conoscere l'indirizzo con cui contattare l'altro interlocutore e negoziare dei parametri di gestione connessione. Questa fase iniziale di *"Signaling"* ha quattro ruoli principali:

- L'identificazione e l'autenticazione dei partecipanti ad una sessione;

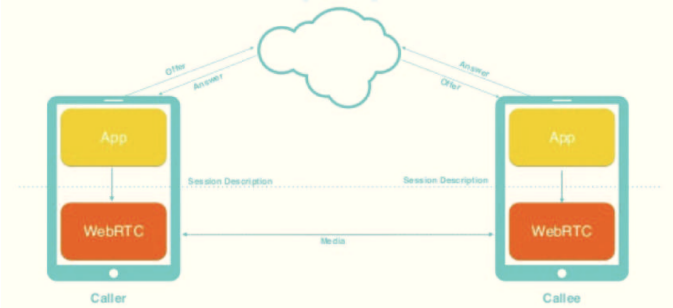

Figura 2.8: Signaling *(slideshare.net)*

- La gestione della richiesta di comunicazione tra i due peers;
- La negoziazione delle capacità dei media e delle impostazioni;
- La gestione della sessione, delle modifiche di parametri e chiusura della connessione.

La gestione della fase di Signaling è demandata all'applicativo web, mentre l'avvio della comunicazione ed il controllo dei media rimangono al WebRTC, poichè ogni applicativo potrebbe preferire un ben determinato protocollo. Il Signaling viene implementato utilizzando un server intermedio che, raggiungibile da entrambi i peers, permette di mettere in contatto i clients registrati. La Session Description Protocol (SDP) è un protocollo utilizzato per descrivere i requisiti di una sessione WebRTC indipendentemente da come poi le informazioni verranno inviate tra i peers coinvolti. L'SDP contiene:

- Il nome e lo scopo della sessione;
- La durata della sessione;
- I media scambiati nella sessione;
- Le informazioni per lo scambio dei media come indirizzi, porte e formati;
- L'utilizzo di banda.

Al fine di instaurare una connessione, gli endpoints devono scambiarsi dei pacchetti SDP per accordarsi sui parametri (codifiche media, IP, porte, ecc.) comuni da usare nella comunicazione. Lo scambio di tali pacchetti è basato sul modello *offer/answer*, dove un partecipante invia un messaggio, contenente la descrizione dei parametri da utilizzare secondo il proprio punto di vista, e l'altro risponde con un secondo messaggio contenente la propria versione dei parametri da utilizzare. Fintanto che non verranno trovati accordi tra i peers, questo

modello di negoziazione può generare anche più offerte da parte del client invitante, poichè entrambi potranno utilizzare solo gli stessi parametri e le stesse codifiche [11]. I pacchetti SDP potrebbero anche contenere gli ICE Candidates, ovvero gli indirizzi IP e le porte UDP, su cui verranno ricevute le informazioni dai browser [12]. Se un utente A vuole instaurare una comunicazione con un utente B, invierà l'offerta SDP, passando per il canale di Signaling e attenderà la risposta SDP da B. Negoziati i parametri e scambiati gli ICE Candidates, la fase di Signaling si può ritenere terminata ed i due peers sono in grado di comunicare in P2P.

2.3 API WebRTC

Le tre API fondamentali del WebRTC sono:

- MediaStream;
- RTCPeerConnection;
- RTCDataChannel.

Queste interfacce permettono di accedere alle risorse locali dell'utente (videocamera e microfono), di gestire i flussi audio video locali, creare i pacchetti SDP di offerta e risposta, gestire gli ICE Candidates, instaurare e gestire un canale per lo scambio di dati generici. Queste API sono in grado di nascondere agli sviluppatori la complessità e la gestione dei protocolli, garantendo con facilità la gestione della comunicazione real-time. Inoltre i più recenti browser implementano algoritmi in grado di togliere echi, adattare la banda, modificare dinamicamente il buffer, controllare automaticamente il guadagno, la riduzione e la soppressione del rumore, rendere le immagini più nitide al solo fine di aumentare la qualità dell'audio video trasmesso e ricevuto. L'architettura complessiva è mostrata in figura 2.9 [13].

2.3.1 API MediaStream

L'API MediaStream si compone di due interfacce fondamentali, la MediaStreamTrack e MediaStream. La prima interfaccia viene usata per identificare i media di una risorsa dell'utente, come il video acquisito dalla webcam o l'audio dal microfono. La seconda interfaccia invece viene usata per raggruppare diversi MediaStreamTrack in una sola entità. Tutte le tracce MediaStream vengono sincronizzate quando elaborate e non al livello hardware, a causa dei differenti clock dei diversi dispositivi [14]. L'API MediaStream viene utilizzata mediante il metodo "navigator.getUserMedia()" che ha tre parametri di ingresso e un output. Il valore di ritorno può esser passato ai tag video della pagina

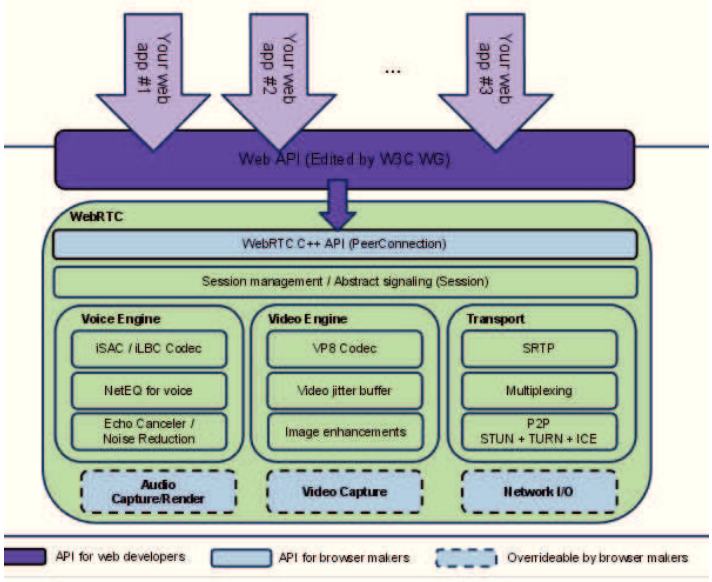

Figura 2.9: Architettura WebRTC *(webrtc.org)*

web per la sua visualizzazione o ad altri oggetti RTCPeerConnection per l'invio ad altri peers. I parametri di ingresso sono:

- *Constraints*, oggetto usato per impostare le caratteristiche dei media, utili per garantire un certo servizio, impostando la risoluzione minima o massima, per risparmiare banda o capacità computazionale [15];
- *function(Stream)*, funzione di call-back con ingresso un oggetto MediaStream in caso successo nell'acquisizione dei media locali;
- *function(Error)*, funzione di call-back con ingresso un oggetto errore in caso insuccesso nell'acquisizione dei media locali.

Ad oggi, per ragioni di sicurezza, il browser puà accedere solo ai dispositivi locali per acquisizioni di audio video, dopo un'autorizzazione esplicita da parte dell'utente (box di conferma dell'uso dispositivi locali). È tuttora in corso l'abilitazione dell'API per l'invio di dati generici, provenienti da periferiche esterne come dvd, usb o sensori esterni [16].

2.3.2 API RTCPeerConnection

L'API RTCPeerConnection è l'interfaccia in grado di creare e gestire la connessione tra i due peers. Le sue principali funzioni sono:

- gestire gli ICE Candidates;

- inviare dei pacchetti STUN di "keep-alive" per mantenere attivo lo stato di connessione tra i peers;
- tenere traccia dei flussi audio, video e dati sia locali che remoti;
- gestire la negoziazione dei parametri tramite lo scambio di pacchetti SDP;
- gestire la rinegoziazione dei parametri di connessione;
- gestire lo stato corrente della connessione.

La Peer Connection, necessaria una per ogni coppia di browser, si occupa dello scambio di dati tra i peers attraverso il metodo "peerConnection()". La funzione accetta come parametro di ingresso un oggetto che contiene tutte le informazioni sull'ICE Candidate, da utilizzare per oltrepassare la NAT. Instaurata la connessione si possono aggiungere indeterminati flussi multimediali per inviare audio, video e dati dall'endpoint locale a quello remoto o viceversa.

2.3.3 API RTCDataChannel

L'API RTCDataChannel permette la realizzazione di un canale bidirezionale per l'invio di dati arbitrari in peer-to-peer. Una volta che viene stabilita la connessione, i peers possono creare uno o più canali, fino al limite teorico di 65536 cadauno, multiplati all'interno della stessa connessione SCTP. RTCDataChannel, parte dell'API RTCPeerConnection, permette di configurare il canale in base alle proprie necessità in affidabilità e ordine di consegna dei pacchetti, poichè basato su UDP, DTLS e SCTP. Il metodo utilizzato per la creazione del canale è "createDataChannel()" che prevede due parametri di ingresso:

- *label*, usato per distinguere i canali tra loro;
- *options*,opzioni usate per la configurazione del canale.

Le possibili configurazioni del canale sono:

- *ordered*, indica se bisogna garantire l'ordine di consegna dei pacchetti;
- *maxRetransmitsTime*, indica il tempo massimo a disposizione per ritrasmettere un messaggio fallito;
- *maxRetransmits*, indica il numero massimo di volte che un messaggio può essere ritrasmesso;
- *protocol*, specifica il protocollo usato;
- *negotiated*, può assumere valore true o false, se true indica che si andranno a rimuovere le impostazioni del canale dati sull'altro peer (con id comune) e le nuove impostazioni saranno specificate nel campo opzioni;

Tabella 2.2: Configurazioni DataChannel

Tipologia	Ordine	Affidabilità	Politica
Ordinato e affidabile	True	True	Non applicabile
Non ordinato e affidabile	False	True	Non applicabile
Ordinato e parzialmente affidabile con contatore pacchetti	True	Parziale	MaxRetransmits Time
Non ordinato e parzialmente affidabile con contatore pacchetti	False	Parziale	MaxRetransmits Time
Ordinato e parzialmente affidabile con temporizzazione	True	Parziale	MaxRetransmits
Non ordinato e parzialmente affidabile con temporizzazione	False	Parziale	MaxRetransmits

- *id*, serve a fornire un identificativo per il canale

La possibilità di configurare la consegna ordinata dei messaggi e l'affidabilità permette di configurare il canale ed il protocollo in base alle proprie esigenze. Ad esempio, avere un canale affidabile e ordinato equivale ad usare il TCP (maxRetransmits > 0 e ordered = true) mentre avere un canale non affidabile e non ordinato equivale ad usare l'UDP (maxRetransmits $= 0$ e ordered = false). Nella tabella 2.2 sono riportate le possibili configurazioni ottenibili.

2.4 Applicazione Peer-to-Peer

L'implementazione del sistema WebRTC ha reso indispensabile definire una serie di strumenti utili alla realizzazione del sistema sia lato server che lato client. Il server, ospitato dal Dipartimento di Ingegneria dell'Informazione dell'Univpm come macchina virtuale Ubuntu Server, è stato usato per la gestione delle fasi di Signaling, piuttosto che per la gestione delle richieste di pagine Web o per l'archiviazione di dati e conversazioni. I dispositivi client invece sono stati computer Linux e Windows sui quali era installato il browser Crhome.

2.4.1 Server-side

Il server WebRTC è stato implementato mediante "Node.js", framework open-source, utilizzato per la realizzazione di applicazioni Web in JavaScript. Esso è basato su JavaScript Engine V8, Runtime di Google utilizzato anche in Crhome. La sua caratteristica principale è la possibilità di accedere alle risorse

in modalità event-driven, ovvero "programmazione ad eventi". Ciò significa che è possibile eseguire azioni al verificarsi di uno o più eventi e non, a differenza della programmazione tradizionale, in cascata una dopo l'altra. Tale modello di funzionamento ci consente uno sviluppo più efficiente delle applicazioni grazie ad una gestione delle chiamate a più basso livello. Le principali librerie di Node.js sono:

- *"https"*, utilizzata per la creazione del server;
- *"express"*, utilizzata per la gestione delle pagine Web;
- *"socket.io"*, utilizzata per le richieste di tipo WebSocket per il Signaling;
- *"mysql"*, utilizzata per la gestione del Database.

Una volta installato il framework, l'applicativo dovrà effettuare una serie di operazioni preliminari come la definizione del path del certificato e quindi della coppia di chiavi pubblica e privata, utilizzate per introdurre maggiore sicurezza mediante il criptaggio delle informazioni scambiate. Inoltre verrà anche definita la porta, dove resterà in ascolto l'applicativo per la gestione delle richieste dei vari clients. Tutto ciò è reso possibile mediante l'utilizzo della libreria *https*. L'utilizzo della libreria *express* ci consente invece di gestire le richieste HTTP dei clients su un certo URL. Per ogni URL invocato, sono realizzate apposite funzioni che possono fare controlli sul Database, reindirizzare ad apposite pagine o semplicemente inviare messaggi di risposta. Da qui l'utilizzo della libreria *socket.io* ci consente di instaurare un canale bidirezionale tra client e server mediante i WebSocket. Il WebSocket è una tecnologia web, che garantisce una connessione tra due entità, nel nostro caso tra il client ed il server, che resta attiva per tutta la durata della sessione, permettendo una comunicazione bidirezionale ed evitando così un numero eccessivo di connessioni HTTP/HTTPS. Importata la relativa libreria, l'applicativo assocerà l'oggetto *socket.io* all' *httpsServer*, utilizzato nella fase iniziale per la definizione del certificato e della porta, ove resterà in ascolto delle richieste dei clients. L'oggetto *io*, output della precedente associazione, permette di gestire gli eventi *"connection"* di ogni cliente, mediante funzioni di call-back che riceve oggetti *"socket"* univoci per ogni client. La comunicazione dal server al client può essere di tre tipi:

- *broadcast*, attraverso l'utilizzo della funzione i *o.emit("evento", "Messaggio per tutti")*, si può inviare un messaggio a tutti i clients connessi al server;
- *esclusione di client*, l'utilizzo della funzione *socket.broadcast.emit("evento", "Messaggio per alcuni")*, permette di inviare un messaggio a tutti i clients connessi al server, ad eccezione di un certo socket/client;

- *messaggio al singolo client*, l'utilizzo della funzione *socket.emit("evento', "Messaggio a singolo client")*, permette di inviare un messaggio al singolo client che ha contattato il server.

Inoltre la libreria *socket.io* permette di creare delle stanze, *room*, ovvero canali dove possono essere aggiunti più clients. Il vantaggio nell'utilizzo della *room* risiede nel fatto che è possibile inviare un messaggio, con una sola operazione, a tutti i clients che partecipano alla room, ma non a tutti quelli connessi al server. La funzione per partecipare ad una *room* è la *join*, mentre per lasciarla si usa la funzione *leave* [17]. L'applicativo realizzato fa uso di un Database per tener traccia degli utenti registrati piuttosto che memorizzare file ed associare il client, che ha inviato tale dato, al medico ricevente. Nel progetto è stato utilizzato MySQL, database della famiglia Relational Database Management System (RDBMS) in grado di memorizzare i dati in tabelle interconnesse tra loro mediante delle relazioni. Le operazioni CRUD (Create, Read, Update e Delete) su uno o più dati possono essere fatte mediante delle interrogazioni scritte in Standard Query Language (SQL). Gli steps per effettuare un'operazione CRUD sono:

- importazione della libreria *mysql*, una singola volta per applicativo;
- definizione dei parametri di connessione al Database come host, username, password e database. Possono essere definiti una singola volta per applicativo;
- connessione al database mediante i parametri sopra definiti e l'uso del metodo *connect*;
- esecuzione dell'operazione CRUD, scritta in SQL, mediante utilizzo della funzione *query*;
- lettura della risposta (dati nel caso di operazione di lettura mentre esito della query nel caso di operazioni di inserimento, aggiornamento o cancellazione);
- chiusura della connessione mediante il metodo *end*.

Le tabelle ed i loro campi sono descritti nella figura 2.10:

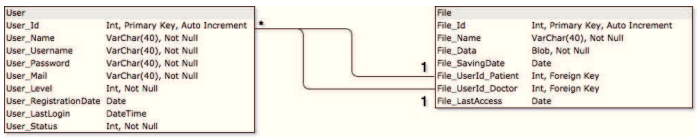

Figura 2.10: Struttura tabelle Database WebRTC

Figura 2.11: Steps per l'acquisizione dei flussi multimediale da webcam e microfono

2.4.2 Client-side

Il server, oltre alla parte ad esso necessaria per lo svolgimento della fase di Signaling e di archiviazione dati, ospita l'insieme di pagine web utili all'utente, medico o paziente, per instaurare la comunicazione peer-to-peer. L'utente, attraverso l'acceso all'URL pubblico del server, verrà reindirizzato, dall'applicativo server-side, alle pagine web di login/registrazione. La pagina di registrazione conterrà un form HTML per l'inserimento di un nuovo record nella tabella *User*, previa verifica dell'esistenza dello stesso. La pagina di login invece verifica la corrispondenza delle credenziali (coppia Username e Password) inserite nel form della pagina con quelle presenti nella tabella *User*. Trovata la corrispondenza, la pagina effettuerà un redirect alla homepage del medico o del paziente a seconda del valore del campo *Privilege* ad essa associato (0-Paziente, 1-Medico). La base di un servizio di videoconferenza è l'acquisizione di flussi multimediali provenienti da dispositivi, integrati o esterni al terminale dell'utente, come webcam e microfono. Il WebRTC permette l'acquisizione di flussi dati utilizzando l'interfaccia MediaStream ed il relativo metodo *getUserMedia*. Tali flussi vengono acquisiti sia nelle pagine di livello medico che in quelle di livello paziente, ed iniziano non appena l'utente autorizza, mediante accettazione di un avviso nel browser stesso, l'utilizzo dall'applicazione di webcam e microfono. Lo schema 2.11 illustra gli steps necessari all'acquisizione di flussi audio video mediante metodo getUserMedia. Ognuna delle due homepage

conterrà le funzioni necessarie all'instaurazione della connessione peer-to-peer medico paziente, l'avvio della videochiamata e della chat.

Instaurazione Connessione Peer-to-Peer L'avvio di una comunicazione WebRTC inizia con il login al servizio da parte del medico, che rimarrà in attesa di essere contattato da parte di un paziente richiedente. Lo stato disponibile sarà indicato dall'aggiornamento del record del medico nella tabella *User*, ponendo il campo *User_Status* a 1. Il servizio web inoltre creerà una *room* alla quale il paziente, che effettua login e vede il medico disponibile, si unirà. Una volta che il paziente si autentica, sul proprio terminale verrà visualizzato l'elenco con tutti i medici disponibili a prestare assistenza. La selezione del medico fa si che, tramite il metodo *socket.emit("RichiestaChiamata", InformazioniChiamata)*, venga inviato al server un oggetto JSON contenente informazioni sul paziente da assistere, il medico richiesto ed un flag di accettazione richiesta del medico, inizialmente impostato a false. La struttura è di seguito riportata:

var InformazioniChiamata = {
patient: IdPatient,
doctor: IdDoctor,
answer: false
}

Ricevuta la richiesta da parte del client, il server inoltra la variabile JSON al medico e fa partecipare il paziente alla *room* creata inizialmente dal medico. Questa operazione viene effettuata per togliere il medico dall'elenco di quelli disponibili cosicchè altri client, possano contattare medici differenti. La funzione, utilizzata per ricevere la richiesta di chiamata nella pagina web del medico, è la *socket.on*. Il medico può così scegliere se accettare o meno la richiesta di videoconferenza dal client. Effettuata la scelta, la variabile JSON con valore del campo *answer* pari a *true* nel caso di accettazione e *false* nel caso di rifiuto, sarà inoltrata al paziente passando per il server. Nel caso di rifiuto di teleconferenza il paziente verrà anche rimosso dalla *room* del medico. La fase iniziale di comunicazione è raffigurata in figura 2.12. Ricevuta la risposta affermativa dal medico, è affidato al client il compito di iniziare la fase di Signaling con l'elaborazione ed invio al medico del proprio pacchetto SDP mediante l'interfaccia RTCPeerConnection. Le funzioni utilizzate sono la *createOffer* per creare l'offerta e la *setLocalDescription* per impostare l'SDP come locale. Per inviare l'SDP al medico verrà invece utilizzata la funzione *socket.emit*. Lato medico, ricevuto l'SDP dal paziente, verrà impostato come offerta remota e verrà elaborata una propria offerta, considerata locale, che sarà inviata al paziente. Il paziente, ricevuto l'SDP dal medico, lo imposterà come

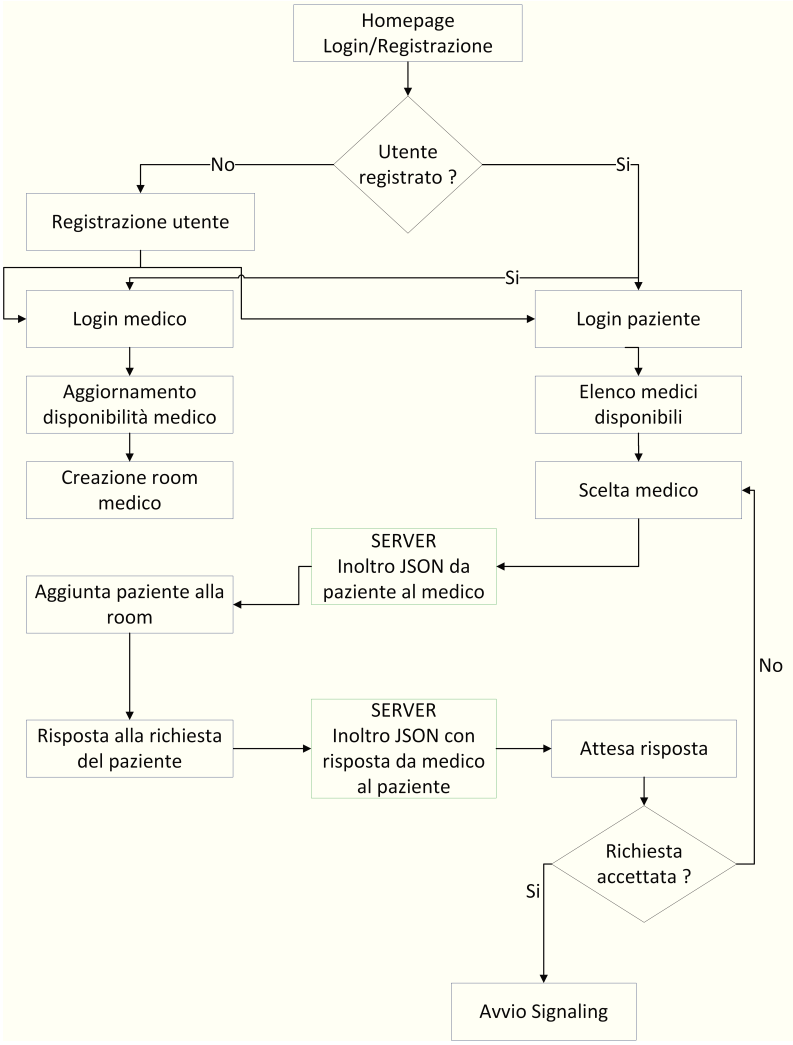

Figura 2.12: Fase iniziale di comunicazione

offerta remota. Scambiati i pacchetti SDP, sia il medico che il paziente avranno i parametri della sessione, locali e remoti. Simile allo scambio dei pacchetti SDP, avverranno anche gli scambi degli ICE Candidate mediante l'interfaccia RTCPeerConnection che prevede due metodi, *oniceCandidate* per registrare l'ICE locale e addiceCandidate per registrare gli ICE ricevuti dall'altro peer.

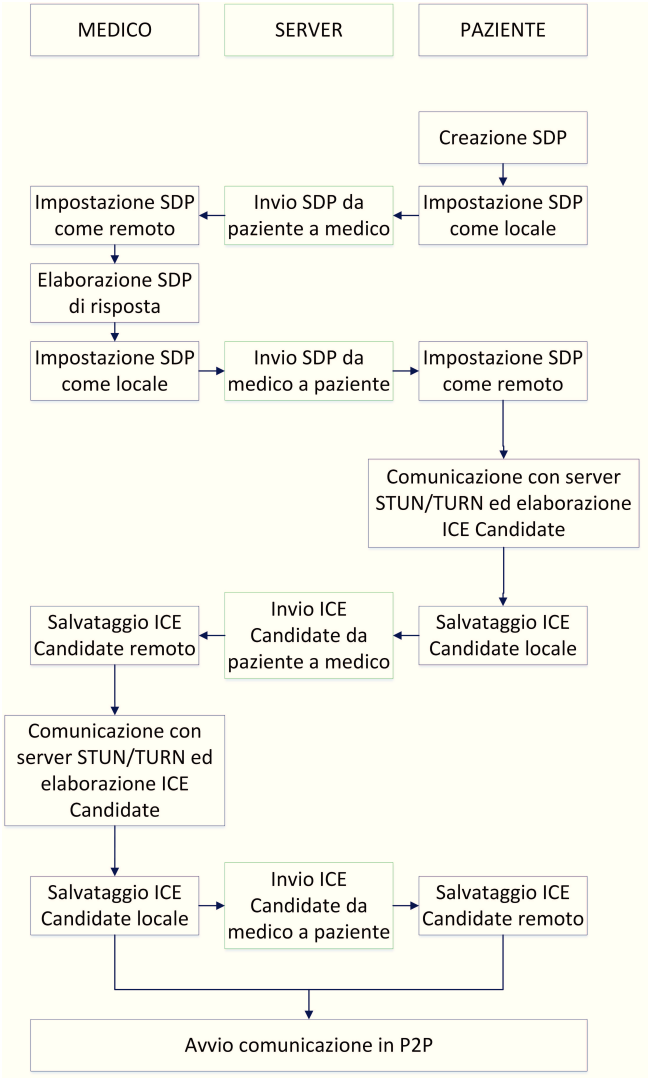

Figura 2.13: Signaling

Ogni ICE Candidate, ottenuto dopo aver contattato i server STUN/TURN, contiene le coppie IPv4/IPv6 e porta UDP sulle quali è possibile contattare il peer stesso. Terminata la fase di Signaling, raffigurata nello schema 2.13, è possibile iniziare la comunicazione dei flussi multimediali in peer-to-peer [18].

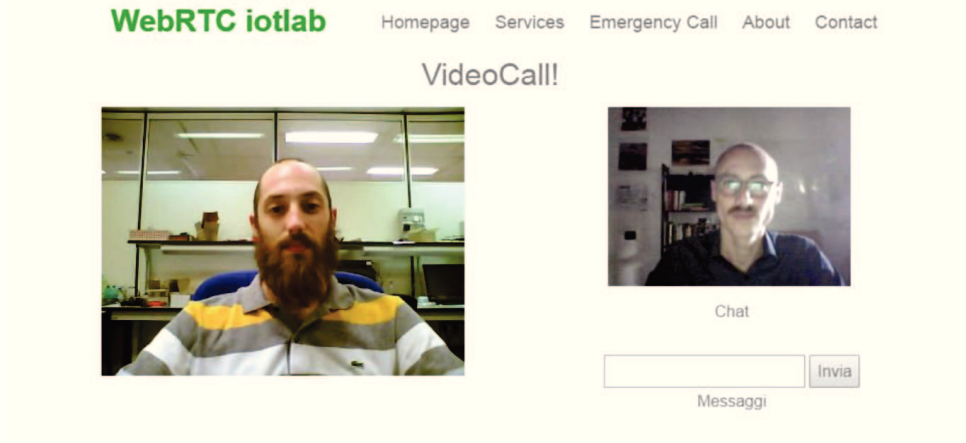

Figura 2.14: Videochiamata P2P

Realizzazione Videochiamata Per gestire il flusso multimediale, che un peer riceve dall'altro, è necessario utilizzare la funzione *onaddstream* che ci consente di visualizzare l'audio e video ricevuto, all'interno di un secondo elemento video HTML del peer ricevente. Combinando i flussi audio video locale e quello remoto, è possibile realizzare il servizio di videochiamata P2P. In figura 2.14 è mostrato un esempio di comunicazione.

Realizzazione Chat L'ultima interfaccia utilizzata è la RTCDataChannel che ci consente di realizzare un canale peer-to-peer, parallelo a quello del flusso audio/video, da usare per inviare dati arbitrari. Il canale viene creato da uno dei due peer, ad esempio dal paziente, mediante la funzione *createDataChannel* dell'oggetto RTCPeerConnection. Il secondo peer è in ascolto di eventi di creazione del canale, che verificandosi, permetterà l'avvio della funzione *ondatachannel*. Sia la funzione *ondatachannel*, che la *onopen* usata per aprire il canale, fanno parte dell'oggetto RTCDataChannel. Realizzato il canale, i due peers possono inviarsi messaggi con la funzione send e riceverli con la *onmessage*. Tutti i messaggi saranno poi accodati in un campo text, visibile nel browser di entrambi i peers. Un esempio di chat tra medico e paziente è illustrata in figura 2.15.

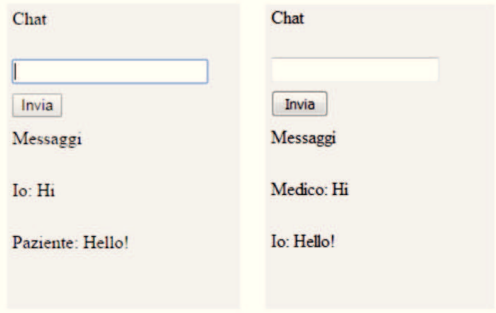

Figura 2.15: Esempio chat in P2P

2.5 Invio di un segnale biomedicale

In questa sezione viene descritta la realizzazione di un sensore indossabile al fine di dimostrare come anche segnali biomedicali, acquisiti da dispositivi esterni, possano essere usati per una comunicazione P2P con le API WebRTC. Gli steps svolti sono stati:

- Realizzazione di un dispositivo prototipo in grado di acquisire ed elaborare l'elettrocardiogramma del paziente mediante board Arduino e sensori indossabili;
- Invio dei dati elaborati da Arduino al computer mediante porta seriale;
- Frammentazione dei campioni ed invio degli stessi al browser del medico mediante interfaccia DataChannel;
- Ricezione dei campioni sul browser del medico e visualizzazione dell'ECG.

2.5.1 Realizzazione Prototipo

Il prototipo realizzato nasce da un progetto preesistente, realizzato all'interno del laboratorio di telecomunicazioni del Dipartimento di Ingegneria dell'Informazione dell'Univpm, che fonda le sue basi sul circuito applicativo della Texas Instruments [19] [20] e sul lavoro proposto da Gao et al. [21] e da Fulford-Jones [22]. Il prototipo fa uso dell'amplificatore operazionale CMOS OPA4340 e dell'amplificatore CMOS INA321, usati in campo biomedicale ed entrambi prodotti dalla Texas Instruments. Al fine di realizzare un prototipo nel più breve tempo possibile, atto a dimostrare le potenzialità del servizio WebRTC, si è scelto come unità di elaborazione Arduino, che ospita il microcontrollore ATmega328P. Arduino è stato anche scelto per la facilità con cui è possibile acquisire segnali analogici, elaborarli ed inviarli tramite Bluetooth al device, dove è attivo il servizio WebRTC. Come frequenza di campionamento si è scelto 100

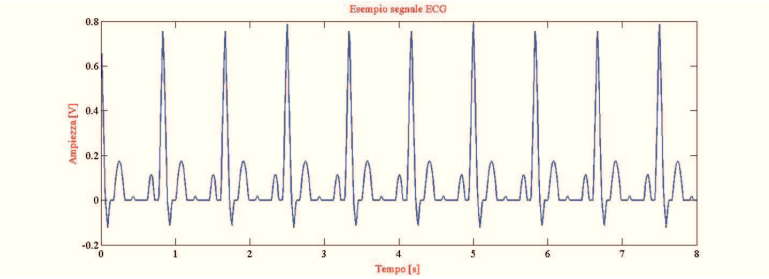

Figura 2.16: Segnale ECG acquisito dal prototipo

Hz, che risulta più che soddisfacente per un prototipo senza nessuna valenza medica [23]. Esso, inoltre, mediante il suo ADC a 10 bit, converte il segnale in digitale per poi elaborarlo al fine di ottenere l'HR istantaneo. Per ottenere l'HR è stata implementata una versione scalata dell'algoritmo QRS proposto da Pan et al. [24], utilizzando finestre scorrevoli di 2s, overlap del 25% ed un filtraggio passabanda per attenuare il rumore. Un esempio di HR acquisito ed elaborato è proposto in figura 2.16. Il segnale acquisito è stato poi inviato al computer (ricevuto tramite un Dongle Bluetooth), in cui è attivo il canale WebRTC con il medico, mediante il serial Port Protocol (SPP), che permette di virtualizzare una connessione seriale su tecnologia Bluetooth. Il parametro di velocità utilizzato è stato 1115200 baud/s. I dati ricevuti dall'applicazione, ogni 10ms, saranno del tipo: *"t: tempoAcquisizione - v: valoreAcquisito"*. Uno dei principali problemi riscontrati in questa fase è stata l'attuale limitazione del WebRTC che, ad oggi, può solamente fare accesso ai dispositivi audio e video locali, senza acquisire dati dalla porta seriale del computer. Sia l'IETF che il W3C hanno deciso di ampliare l'API MediaStream, ma il lavoro è ancora in corso di svolgimento [16]. La soluzione proposta, per aggirare il problema, è stata quindi quella di realizzare un'applicazione Crhome, constrastando così l'idea di realizzare un servizio cross-browser. L'uso dell'applicazione è legato unicamente alla disponibilità, nel dispositivo del medico e del paziente, del browser Crhome. Sebbene questo vincolo può risultare un grande svantaggio nell'uso del servizio WebRTC, risulta la soluzione più semplice riducendo al massimo le operazioni svolte dall'utente poichè soluzioni alternative come l'uso del plug-in esterno jUART, benchè supportato da tutti i browser, richiedono una ben precisa configurazione.

2.5.2 Applicazione Crhome

L'applicazione Crhome, eseguibile direttamente nell'apposito browser, permette di realizzare interfacce interattive per gli utenti [25] con la possibilità di usare i comuni linguaggi HTML, CSS e JavaScript a differenza di quelli utiliz-

zati per sviluppare il sistema operativo [26]. A differenza di un normale sito web, l'applicazione realizzata permette di accedere a risorse locali, previa definizione dei permessi ad essa associati. I principali componenti dell'applicazione Crhome sono:

- *File manifest*, definisce l'applicazione, quello che fa ed i permessi ad essa associati;
- *Script di background*, pagina per la gestione del ciclo di vita dell'applicazione;
- *Launcher.html*, pagina per la realizzazione del front-end dell'applicazione;
- *Risorse aggiuntive*, altre risorse utilizzate come immagini, icone e font.

Il manifest è il file, con estensione json, che viene utilizzato per definire il comportamento dell'applicazione in fase di installazione e funzionamento. Inoltre contiene i permessi usati dall'applicazione che nel nostro caso sono stati:

- *"serial"*, per l'acquisizione di dati proveniente da dispositivi collegati mediante porta seriale;
- *"storage"*, per salvare eventuali informazioni sul dispositivo dell'utente;
- *'videoCapture"*, per l'accesso alla videocamera;
- *"audioCapture"*, per l'accesso al microfono.

Lo script di background, file JavaScript, definito nel manifest, viene utilizzato per gestire, mediante l'API *crhome.app.runtime*, il ciclo di vita dell'applicazione come l'installazione, l'avvio e la chiusura della pagina [27]. La fase di nostro interesse è quella di avvio, dove è possibile specificare, mediante il metodo *crhome.app.window.create*, la pagina da eseguire e le dimensioni ad essa associate. La pagina launcher contiene il codice necessario per realizzare l'interfaccia utente, instaurare la comunicazione P2P, come descritto nel paragrafo 2.4 e gestire i dati acquisiti dalla porta seriale. L'API, necessaria a gestire la comunicazione con la porta seriale è la *crhome.serial* ed i metodi utilizzati sono [28]:

- *"chrome.serial.getDevices(function callback)"*, utilizzata per ottenere l'elenco di tutti i dispositivi connessi alle rispettive porte seriali;
- *"chrome.serial.connect(string path, ConnectionOptions options, function callback)"*, utilizzata per connettersi ad una determinata porta seriale;
- *"chrome.serial,send(integer connectionId, ArrayBuffer data, function callback)"*, utilizzata per scrivere sulla porta seriale;

- *"chrome.serial.disconnect(integer connectionId, function callback)"*, utilizzata per effettuare la disconnessione da una porta seriale;
- *"chrome.serial.onReceive.addListener(function callback_ReceiveData)"*, funzione eseguita ogni volta che l'applicazione riceve un dato da una porta seriale a cui è connessa;
- *"chrome.serial.onReceiveError.addListener(function callback)"*, funzione eseguita ogni volta che si verifica un errore sulla porta seriale, alla quale l'applicazione è connessa.

Una volta instaurata la comunicazione peer-to-peer, come nel classico sito web, il paziente può inviare sulla porta seriale i dati acquisiti da Arduino tramite canale DataChannel. L'invio dati inizia con il click dell'utente su un apposito button che, invocando la funzione *getDevices*, ottiene l'elenco di tutte le porte seriali e dispositivi ad esse connessi. La funzione di call-back associata riceverà un elenco di tutte le porte seriali rilevate, permettendo all'utente di scegliere quella a cui connettersi. Scelta la porta, verrà chiamata la funzione *connect* che, oltre a prendere in ingresso il percorso del sistema della porta seriale scelta, prende un oggetto JSON contenente bitrate, bit di parità e grandezza del buffer di ricezione. La funzione di call-back associata alla *connect*, effettuato un primo controllo sul buon esito della connessione alla porta seriale, fornirà l'id della porta, sulla quale aggiungere un listener all'evento *onReceive*. Esso permette di invocare la funzione *callback_ReceiveData* per ogni nuovo dato ricevuto. Infine questa callback riceverà un oggetto contenente la variabile *connectioId*, intero che rappresenta l'identificativo della connessione e la variabile *data* contenente i dati ricevuti. I dati, sotto forma di ArrayBuffer, verranno letti al fine di estrarre informazioni sul tempo di acquisizione e sul valore del campione. Ogni dato verrà accodato a quelli ricevuti in precedenza cosicchè, dopo un numero ben preciso di campioni ricevuti, verrà creato il pacchetto per l'invio al peer destinatario. Al fine di ridurre le dimensioni del pacchetto, il tempo di acquisizione verrà sostituito con la differenza temporale tra il campione *n* e l' *n-1*. Ogni pacchetto ricevuto sarà formato dai seguenti campi [29]:

- *"Type"*, tipo di informazione contenuta;
- *"NumSample"*, numero campioni contenuti al suo interno;
- *"Id"*, id del pacchetto inviato;
- *"TimeStamp"*, l'istante di invio del pacchetto considerato;
- *"T0"*, timestamp del primo campione inviato nel pacchetto;
- *"Data"*, dati inviati.

Per l'invio dei dati si è scelto di configurare il canale come ordinato e parzialmente affidabile, impostando il numero di ritrasmissioni pari a 0. La configurazione ottenuta è un UDP, realizzando una comunicazione real-time dove le attese, dovute ai tentativi di ritrasmissione pacchetti, vengono evitate. Convertito il pacchetto in stringa, viene inviato mediante la funzione *send* utilizzata nel DataChannel per l'implementazione della chat. Ogni due minuti vengono inoltre inviati pacchetti contenenti l'HeartRate e il numero di battiti per consentire al medico una prima analisi sullo stato del paziente. Dal lato medico, i dati ricevuti vengono bufferizzati e, riempito il buffer, viene realizzata la visualizzazione sull'interfaccia del medico utilizzando la libreria CanvasJS, che permette il rendering grafico dei dati ricevuti. Lo schema riassuntivo dell'invio al peer medico di dati acquisiti da dispositivo esterno è mostrato in figura 2.17. Nello sviluppo dell'applicazione uno dei fattori critici considerato è il ritardo di trasmissione, ovvero il tempo che intercorre tra la trasmissione di un pacchetto dal peer paziente e la sua ricezione da parte del peer medico. Un valore basso del ritardo è un elemento significativo, in applicazioni come la videoconferenza [30], dove è indispensabile una rapida risposta del canale di comunicazione a richieste di trasmissione e ricezione da parte dei peers coinvolti. Gli steps per il calcolo del ritardo di trasmissione sono:

- Il calcolo della latenza media tra client e server;
- La sincronizzazione dei due peers;
- L'invio di un pacchetto *i-esimo* da un peer all'altro, contenente i dati e l'istante di invio del pacchetto;
- La ricezione del pacchetto *i-esimo* dal peer e calcolo dell'istante di ricezione;
- Il calcolo del ritardo di trasmissione tra i due peer.

La latenza è definita come il tempo che impiega un peer a comunicare con il server ed è una grandezza variabile nel tempo. In questo progetto è stata calcolata la differenza tra l'istante temporale in cui un client riceve la risposta dal server, per un pacchetto precedentemente inviato e l'istante di invio del pacchetto stesso, il tutto diviso 2. Questa operazione, svolta da entrambi i peer, viene effettuata 10 volte per poter calcolare la latenza media. Poichè i clients possono essere dislocati in aree non prossime e possono utilizzare browser diversi, è necessaria una fase di sincronizzazione per eliminare i ritardi, dovuti a riferimenti temporali differenti, dei messaggi ricevuti dal server. Non appena si instaura la chiamata, il server invia un messaggio ad entrambi i peer, che sarà ricevuto in instanti temporali diversi a causa della latenza non nulla.

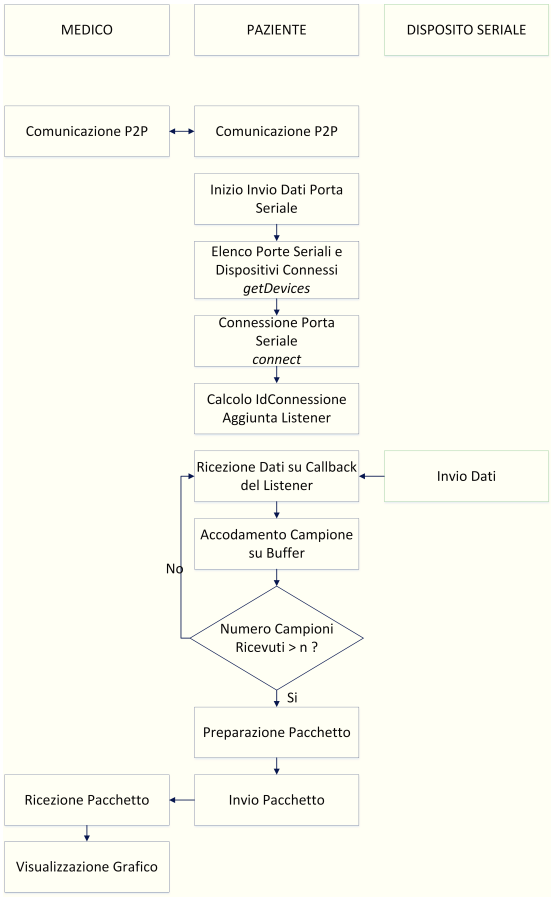

Figura 2.17: Invio dati dispositivo esterno in P2P

Al tempo di ricezione del messaggio acquisito, ogni client vi sottrae la propria latenza media precedentemente calcolata, impostando il valore ottenuto come riferimento temporale per l'inizio della comunicazione. Da qui può essere calcolato il ritardo di trasmissione come la differenza tra il tempo di ricezione del pacchetto *i-esimo* "modificato" ed il timestamp di invio contenuto nel pacchetto ricevuto. Il tempo di ricezione "modificato" altro non è che la differenza tra il tempo di ricezione di un pacchetto e la latenza media del peer. Per motivi di velocità di trasmissione si è realizzato un canale inaffidabile e senza controllo dell'ordine dei pacchetti per l'invio del segnale dell'elettrocardiogramma; risulta però evidente che per fini diagnostici la perdita di un certo numero di pacchetti non è trascurabile. Per tale motivo si è deciso di realizzare

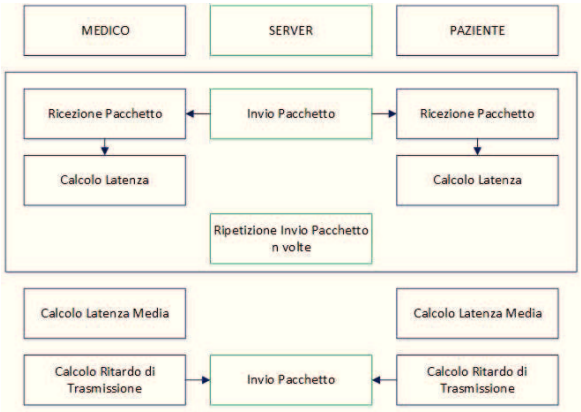

Figura 2.18: Calcolo ritardo di trasmissione

un secondo canale di comunicazione affidabile che verrà utilizzato solo per il recupero dei pacchetti, che nella precedente trasmissione sono andati persi. Il canale può essere utilizzato sia dal peer del medico che dal peer del paziente. Il primo controller è l'id del pacchetto ricevuto con quelli adiacenti e, nel caso non ci sia una progressione unitaria, invierà un messaggio di ritrasmissione del pacchetto mancante nel nuovo canale. Il secondo peer, ricevuto il messaggio, invierà il pacchetto che verrà poi, nel peer ricevente, inserito nella posizione idonea per essere visualizzato. Tutto ciò deve pervenire prima che la porzione di dati debba essere visualizzata sull'interfaccia del medico e quindi la necessità di avere un buffer di dimensioni ragionevoli.

2.6 Test sul canale DataChannel

Il primo test effettuato ha lo scopo di determinare la dimensione massima del pacchetto dati che può essere trasmesso senza dover ricorrere a tecniche di frammentazione dello stesso. Poichè il DataChannel si basa su SCTP, che a sua volta è basato su DTLS, UDP ed IP, la dimensione massima del pacchetto sarà data dalla Maximum Transmitting Unit (MTU), ovvero la dimensione massima di un pacchetto inviato in rete di telecomunicazioni meno i byte di overhead dei protocolli utilizzati. La dimensione della MTU di un pacchetto SCTP, impostata dai client WebRTC, è 1280 bytes, mentre quella degli overhead dei protocolli è mostrata nella tabella 2.3. Considerando le dimensioni minime di overhead di ogni protocollo, si ottiene un overhead complessivo di 76 byte che, tolti alla dimensione massima del pacchetto di 1280 bytes, ci consente di avere 1204 bytes per i dati. Nel nostro caso, poichè il segnale dell'elettrocardiogramma è formato anche da valori con segno negativo, si utilizzerà un byte

Tabella 2.3: Dimensione overhead dei protocolli di rete

Protocollo	**Dimensione [byte]**
IP	20-40
UDP	8
DTLS	20-40
SCTP	28

Tabella 2.4: Risultati al variare del numero di campioni per pacchetto e del numero di pacchetti inviati al secondo

Num campioni/ pacchetto	**Num secondo inviati**	**Rate misurata [byte/s]**	**Dim tot pacchetto [byte]**
1	100	12016	120
2	50	6887	138
20	5	2276	456

per rappresentarne il segno, implicando un byte di differenza nel numero totale disponibile per il campo dati. L'header del pacchetto sarà formato dai campi *Type*, *NumSample*, *Id*, *TimeStamp* e *T0* e con il passare del tempo, avrà dimensioni variabili a causa dell'aumento del *TimeStamp* e di *T0*. Le prove effettuate per calcolare il rate, la dimensione del pacchetto al variare del numero di campione per pacchetto e del numero di pacchetti inviati al secondo, sono mostrate nella tabella 2.4. Ovviamente anche in un canale non affidabile si possono inviare pacchetti di grandi dimensioni, ma in questo caso l'applicazione bufferizza l'informazione, frammenta il pacchetto e lo invia. Supponendo un pacchetto di grandi dimensioni come 120KB e una percentuale di perdita di pacchetti del 1%, il messaggio verrà diviso in 100 pacchetti per essere trasmesso. Se uno dei pacchetti andrà perso, in un canale inaffidabile non verrà ritrasmesso e quindi in ricezione non potrà essere ricostruita l'informazione originale. Per ovviare a tale problema si possono diminuire le dimensioni del pacchetto complessivo al di sotto dei 1204 bytes, contenibili in un unico pacchetto [6], o implementare un secondo canale per la ritrasmissione dei pacchetti persi, come effettuato nel nostro caso. Partendo dalla dimensione massima del messaggio, inviato tramite il DataChannel, si può ottenere il numero massimo di campioni che un pacchetto può contenere, evitando la frammentazione. Dai bytes ottenuti si può togliere l'header e, dividendo per la dimensione del campione, si ottiene 61 come numero di campioni che possono essere contenuti in uno stesso pacchetto senza che questo venga frammentato.
Come seconda prova è stato calcolato il ritardo di trasmissione al variare del numero di campioni contenuti in un pacchetto e quindi della frequenza di trasmissione su DataChannel SCTP. Le prove sono state effettuate tra due peers

Tabella 2.5: Risultati delle prove effettuate

Campioni	**Dim. Pacchetto [byte]**	**Pacchetti per secondo**	**Rate trasmissivo [byte/s]**	**Ritardo Medio [ms]**	**Dev. Stnd.**
1	120	100	12000	45	23
20	462	5	2310	44	52
40	822	2.5	2056	44	46
60	1182	1.67	1970	48	30
80	1542	1.25	1928	69	67
100	1902	1	1902	54	46
120	2262	0.23	1886	46	57

connessi ad internet con normale linea ADSL, dove sono stati inviati pacchetti con numero di campioni variabile da 1 a 120 con passo 20. Poichè Arduino invia un campione ogni 10ms, questi saranno bufferizzati fino ad ottenere il numero desiderato per la composizione del pacchetto. Quindi il peer invierà un pacchetto ogni n*10ms, la cui dimensione sarà data da header più numero di campioni per dimensione del singolo pacchetto. La rate invece sarà calcolata come la dimensione del pacchetto per numero di pacchetti inviati al secondo. La tabella 2.5 mostra la dimensione dei pacchetti al variare del numero di campioni e il ritardo di trasmissione ottenuto. Dalle prove effettuate si evidenzia come il ritardo di trasmissione non cambia in maniera considerevole al variare delle dimensioni del pacchetto. Inoltre esso non è cumulativo in quanto il sistema riesce ad inviare la quantità di messaggi richiesta senza necessità di bufferizzazione. La figura 2.19 mostra le prove effettuate con Id del pacchetto in ascissa e ritardo di trasmissione in ordinata.

Figura 2.19: Ritardo di trasmissione al variare del numero di pacchetti

Come ultima prova sono stati inviati pacchetti di 120 campioni per dieci minuti ottenendo un ritardo medio di 48.34 millisecondi. Dalla figura 2.20 si può notare come il ritardo, eccetto picchi sporadici dovuti alle condizioni della rete, ritorna nell'intorno del ritardo medio calcolato dalle prove di tabella 2.5. Da qui, visto il ritardo inferiore ai 100ms, si può concludere che il sistema mantiene delle buone performance anche per trasmissioni prolungate nel tempo con grande numero di campioni.

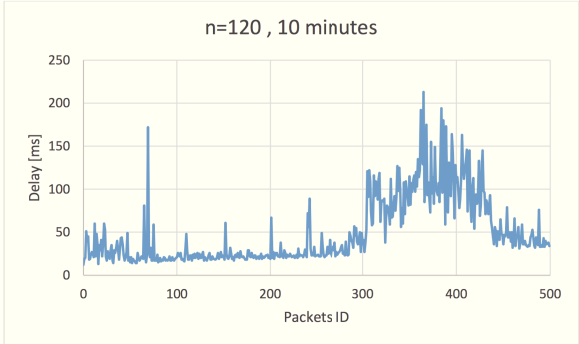

Figura 2.20: Ritardo di trasmissione per prova di 10 minuti con numero di campioni pari a 120

2.7 Conclusioni

In questa prima parte di lavoro è stato realizzato un servizio di teleassistenza basato su tecnologia WebRTC. Dopo un primo studio del suo funzionamento e dei protocolli coinvolti, è stato realizzato un servizio di videoconferenza, che poi si è esteso alla condivisione di dati acquisiti da dispositivi esterni connessi ad uno dei due peer. I test sono stati effettuati con lo scopo di valutare le prestazioni del sistema in termini di dimensioni dei pacchetti e ritardo trasmissivo, andando a confermare le attuali potenzialità del WebRTC. Il problema che rimane ancora irrisolto è la possibilità di condividere dati acquisiti da dispositivi esterni, ad esempio, nel caso in questione, provenienti da una porta seriale, che implica l'utilizzo di plug-in o applicazioni ad hoc. Questo va contro l'idea di base del WebRTC, ovvero utilizzare un servizio *"ready to use"*, facile da usare che possa essere usato da qualsiasi soggetto, anche privo di specifiche conoscenze tecnologiche. Il servizio potrà essere notevolmente migliorato con lo sviluppo di applicazioni smartphone, sia native che mediante l'uso di WebView, disponibili dalla versione 36 del WebRTC [31]. Queste potranno consentire di accedere ai servizi di WebRTC mediante dispostivi mobili, rendendo il sistema, oltre che facile da usare, anche portabile. Un secondo settore, nel quale l'introduzione dei servizi WebRTC porterebbe notevoli vantaggi è quello della telefonia. L'interazione di sistemi eterogenei consentirà una comunicazione peer-to-peer tra SIP client che, snellendo il traffico sulle attuali centraline, garantirà un servizio di chiamate ad alta qualità. In campo sanitario l'utilizzo dei servizi basati su tecnologia WebRTC può portare a notevoli riduzioni del costo della spesa sanitaria, contribuendo al miglioramento della qualità della vita di molte persone. Si stima che il costo della spesa sanitaria nel 2011 sia stato il 17,7% del prodotto interno lordo [2] che non potrà che aumentare a

causa della crescita demografica [32]. Potrà esser di grande importanza per le persone la cui mobilità è un fattore critico e quindi non hanno la possibilità di recarsi presso strutture mediche per effettuare esami e/o visite. Nello studio si è trattato un esempio di invio di un segnale biomedicale prodotto da una board Arduino mentre, in campo pratico, potranno essere molteplici i segnali fisiologici in grado di transitare nel canale DataChannel.

Capitolo 3

Rilevamento di bolle gassose in segnali eco Doppler

Il lavoro di ricerca, presentato in questo capitolo, nasce dalla collaborazione con il DAN - Europe, "Divers Alert Network" - Europe, organizzazione medica, che in ambito internazionale tpromuove la sicurezza dei subacquei. Infatti uno dei settori che negli ultimi anni ha visto un notevole incremento di investimenti, in ricerca scientifica, è quello della tutela dell'attività subacquea. Le risorse sono state impiegate nello studio di nuove tecnologie atte ad individuare tempestivamente la malattia da decompressione ed altre patologie legate alla pratica dell'attività subacquea. La malattia da decompressione (MDD) è una patologia che colpisce soprattutto i soggetti che effettuano rapidi spostamenti in zone con diversa pressione, come ad esempio l'ambiente subacqueo e aereo [33]. Le MDD si possono suddividere in *"decompression sickness"*, ovvero disturbo da decompressione (DS) ed *"arterial gas embolism"*, ovvero embolia di gas arteriosa (AGE). La DS è dovuta ad una inadeguata decompressione che causa la formazione di bolle nei tessuti provocando danni locali, mentre l'AGE si verifica quando le bolle gassose entrano nella circolazione polmonare attraversano le arterie e creano complicazioni serie come l'interruzione del flusso sanguigno nei vasi. L'AGE è comunemente chiamata embolia d'aria. Già nel 1878 la ricerca condotta da Bert [34] evidenziò come causa principale dell'MDD la variazione repentina della pressione esterna con conseguente formazione di bolle a livello sanguigno e tissutale. Normalmente il corpo umano è in grado di gestire la presenza di bolle di azoto, ciò che invece conduce all'MDD è la loro eccessiva presenza, chiamata saturazione. In questa condizione la velocità di comparsa di bolle è talmente maggiore della velocità di dissoluzione che permette alle bolle stesse di penetrare nei tessuti circostanti o altresì impedisce l'irrorazione di organi celebrali, con gravi complicanze neurologiche. Poichè le bolle possono causare seri problemi di salute, è fondamentale nei subacquei un primo trattamento da personale iperbarico qualificato nella fase di post-emersione. Negli anni 70 il Dott. Spencer [35] dimostrò come l'assenza di sintomi non esclude la presenza di bolle e da qui l'esigenza di investigare nuove tecniche di identifica-

zione delle bolle e della loro correlazione con i sintomi. Infatti è evidente che ci possono essere bolle senza DS, ma non ci può essere DS senza bolle [36]. Ad oggi l'identificazione del numero di bolle che circolano nei flussi venosi di un subacqueo, in fase di post-emersione, avviene mediante un'analisi eco Doppler manuale condotta da personale medico. Questi esami, benchè affidabili, sono spesso soggetti ad errori dovuti alla errata valutazione soggettiva dell'esaminatore. Da qui il mio intento di realizzare un algoritmo in grado di elaborare il segnale audio prodotto in uscita da strumentazione eco Doppler ed identificare con precisione le bolle contenute in esso. Successivamente è stato migliorato l'algoritmo effettuando il porting su scheda embedded. Ciò ha consentito al team medico di eseguire l'algoritmo in real-time, parallelamente all'esecuzione dell'analisi eco Doppler. In letteratura non è la prima volta che vengono tentati approcci come questo, producendo anche risultati discreti [37]. Ciò che contraddistingue lo studio fatto è il sito di monitoraggio che, in questo caso, è stato il precordiale. La regione cardiaca, nonostante la presenza di molto rumore prodotto da battiti cardiaci, movimenti di valvole e flusso ematico garantisce, rispetto alle regioni transcraniche, succlavie e femorali, il passaggio di tutte le bolle, che nelle altre regioni risultano filtrate dai polmoni. L'efficacia degli algoritmi è stata testata su 240 eco Doppler audio, registrati in regione precordiale, durante le campagne di acquisizione del DAN. Ad ogni audio sono stati allegati tre report, relativi ad annotazioni manuali condotte da tre "Blind Team" indipendenti. La comparazione dei report ha evidenziato differenze sia sul numero di bolle rilevate che sul loro istante di comparsa, ciò è riconducibile sia ad errori casuali imprevedibili che ad errori dovuti alla diversa percezione di un audio da parte di ogni medico esperto. Parte dello studio è così stata incentrata sulla realizzazione di un software, strumento unico di annotazione manuale delle bolle, in grado di compensare o ridurre gli errori dovuti ad un ascolto soggettivo dell'audio. Il software ha il compito di ridurre errori relativi ad una diversa percezione dello stesso audio, causati principalmente da utilizzo di molteplici player e dai diversi tempi di risposta a stimoli acustici esterni, propri di ogni esaminatore. Esso inoltre garantisce una struttura standard ed univoca di annotazione bolle, facilmente interpretabile da software terzi.

3.1 Eco Doppler

Nello studio della malattia da decompressione è stato dimostrato che una delle cause scatenanti è l'eccessiva presenza di bolle, spesso non correlate alla comparsa di sintomi evidenti. L'eco Doppler è la tecnica utilizzata per identificare fattori di rischio prima della comparsa della patologia stessa [38] [39] [40] [41]. La sua ampia diffusione è da ricondurre all'utilizzo di strumentazione portabile, poco costosa e facile da usare. Inoltre questo tipo di analisi, essen-

do non invasiva, può essere eseguita da chiunque, previa una semplice fase di training iniziale.

3.1.1 Principio di funzionamento

L'analisi eco Doppler inizia appoggiando la sonda o il trasduttore sopra il corpo del soggetto in esame. Il trasduttore contiene cristalli di quarzo in grado di generare onde ad ultrasuoni, che si propagano al di sotto dei tessuti corporei. Le onde, attraversando i tessuti, incontrano degli "ostacoli" generando degli "echi" di ritorno che, opportunamente elaborati, permettono di formulare diagnosi sulla zona di interesse. L'analisi Doppler è particolarmente adatta alla rilevazione di bolle gassose poichè queste, considerate "ostacoli", produrranno degli echi facilmente distinguibili dagli altri suoni riflessi. Il principio fisico alla base di questa analisi è quello dello stesso "Christian Andreas Doppler", il quale afferma che la frequenza di un suono si modifica (aumento o diminuzione) quando viene emessa da un oggetto in movimento. Se consideriamo una sorgente S ed un ricevitore R, supponendo che S emetta onde acustiche a frequenza costante, l'onda riflessa verrà ricevuta con una frequenza diversa a seconda del moto tra S (sonda) ed R (bolla). Il loro avvicinamento provocherà una frequenza più alta dell'originale, mentre il loro allontanamento una più bassa. In ambito medico diagnostico non è importante la frequenza ricevuta, bensì la differenza tra quella trasmessa e quella ricevuta chiamata "Doppler Shift", con il quale determinare la velocità di movimento del ricevitore [42] 3.1. L'equazione alla base del "Doppler Shift" è la 3.1.

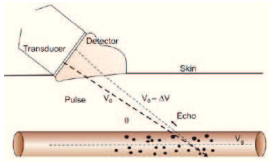

Figura 3.1: Doppler Shift *(jaypeejournals.com)*

$$V = \frac{F_d c}{2 f_0 \cos\theta} \tag{3.1}$$

I componenti della formula sono:

- Fd, frequenza di "Doppler Shift";
- F0, frequenza emessa dalla sonda (2-15 MHz);
- V, velocità oggetto in movimento;
- θ, angolo tra direzione del flusso e direzione degli ultrasuoni;

- c, velocità di propagazione degli ultrasuoni nei tessuti biologici (1540 m/s).

Dalla formula inversa si evidenzia come sia fondamentale l'angolo di inclinazione della sonda che, se non opportunamente posizionata, non ci consente di avere una stima reale della velocità V della particella. Le possibili configurazioni di θ sono:

- θ=90°, nessun segnale Doppler apprezzabile;
- θ=0°, condizione ideale, ma non raggiungibile nella pratica;
- 40°<θ<60°, condizione di lavoro.

Le tecniche Doppler utilizzate sono il "Doppler Continuo" e il "Doppler Pulsato". Nel Doppler continuo si usano metà dei quarzi per inviare ultrasuoni e metà per ricevere gli echi di risposta. La caratteristica di questa analisi è l'invio continuo di energia sonora ad elevata frequenza. Questa tipologia di Doppler è la più utilizzata grazie al basso costo della strumentazione e della possibilità di identificare anche corpuscoli con velocità elevata [43]. Nel Doppler pulsato invece vengono utilizzati tutti i quarzi alternativamente per trasmettere e ricevere. Inizialmente si invia un treno di onde per poi attendere un tempo proporzionale alla distanza tra la sorgente e la regione da monitorare. Passato l'intervallo temporale, verrà inviato un nuovo treno di onde e ripetuta l'intera procedura.
Una bolla è definita come un elemento gassoso in movimento nel flusso sanguigno. Esse sono in grado di riflettere ultrasuoni più intensamente dei globuli rossi circostanti, generando un suono caratteristico riconducibile ad un "cinguettio" o "click" [44]. Questi suoni, ascoltati solo in presenza di bolle, possono avere intensità diverse a seconda del tipo di bolla considerata. Le bolle piccole producono onde riflesse ad alta frequenza, mentre le bolle grandi genereranno onde riflesse a bassa frequenza [45]. Da qui è possibile dedurre come lo spettro di frequenze considerato sarà molto ampio, dovuto, oltre alle numerose bolle di dimensione variabile, anche ai contributi in frequenza dei battiti cardiaci, dei movimenti di valvole e al rumore di fondo. L'onda riflessa giungerà in uscita della strumentazione Doppler come un segnale audio analizzabile foneticamente dall'esperto medico.

3.1.2 Sito di monitoraggio e strumentazione

Al fine di individuare le bolle gassose circolanti nel flusso ematico, è necessario il monitoraggio mediante tecnica Doppler di un vaso di grandi dimensioni. I vasi presi in considerazione sono:

- La vena succlavia (braccio/spalla);

- Le arterie femorali;
- L'arteria cerebrale media (cervello);
- La regione precordiale.

Nella scelta del sito di monitoraggio è necessario prendere in considerazione l'effetto filtrante dei polmoni, particolarmente significativo nelle prime tre regioni. Alcuni studi effettuati, come quello di Brubakk [46], hanno dimostrato che, a seguito del passaggio del flusso sanguigno nei polmoni, parte delle bolle viene intrappolata nei polmoni stessi. Quindi un'analisi condotta a valle dei polmoni non garantirebbe una stima corretta delle bolle presenti nel vaso scelto. L'unica regione in cui si registra il passaggio di tutte le bolle è quella precordiale, che funge da crocevia poichè tutto il flusso sanguigno passa di qui prima di transitare nei polmoni. Un'analisi in questa zona ha come svantaggio la produzione di un audio che non contiene solo le bolle, ma anche suoni aggiuntivi dovuti a battiti cardiaci, movimenti di valvole e rumore di fondo che richiedono un'analisi più complessa.
In accordo con il DAN, il Doppler scelto è stato l' "Huntleigh DF1 Fetal Doppler" con sonda a 2 MHz della "Huntleigh Ltd, UK". I vantaggi della strumentazione Doppler sono, oltre alla lunga durata con batteria di circa 500 minuti, le ridotte dimensioni (140 mm x 70 mm x 27 mm) e il peso non eccessivo (295 g). Al fine di memorizzare le acquisizioni fatte per effettuare un'elaborazione successiva, l'uscita del Doppler è stata collegata all'ingresso di un registratore anch'esso portabile. Il registratore scelto è stato il "Tascam DP-004" della "TEAC America Inc. U.S.A." che, grazie alla sensitività di 92 dB/mW e frequenza di risposta compresa nel range 10-2000 Hz, premette di ottenere elevate prestazioni controllando la saturazione del segnale registrato. Tra le sue caratteristiche si evidenzia una lunga durata della batteria, 500 minuti, piccole dimensioni (155 mm x 33,5 mm x107 mm) e piccolo peso (360 g). I dati acquisiti nel Tascam possono essere ascoltati in real-time, mediante cuffie esterne, e contemporaneamente memorizzati in formato Wave, con frequenza di campionamento di 44 KHz, su scheda microSD (figure 3.2 e 3.3). La strumentazione scelta è stata utilizzata nelle due campagne di acquisizione del DAN all'isola d'Elba e alle Maldive, dal 2014 al 2016, producendo 240 eco Doppler audio file utili, rispettivamente 137 alle Maldive e 103 all'isola d'Elba.

Figura 3.2: Huntleigh DF1 Fetal Doppler *(huntleigh-diagnostics.com)*

Figura 3.3: Tascam DP-004 *(tascam.com)*

3.1.3 Protocollo di acquisizione e Scala di valutazione

Il protocollo di acquisizione, congiuntamente scelto con il DAN, prevede due fasi di registrazione di 45 secondi ciascuna, separate da una fase di 10 secondi, durante la quale il subacqueo esegue delle attività fisiche per favorire il rilascio, nel flusso ematico, delle bolle intrappolate nei tessuti muscolari. La prima acquisizione deve essere effettuata circa dopo 35 minuti dall'emersione poichè il picco di bolle si registra dai 30 ai 60 minuti dall'emersione stessa [47]. Oltre alle normali bolle circolanti, ce ne sono delle altre che rimangono intrappolate nei tessuti, chiamate intra tissutali. In questa fase vengono ripetuti 2 o 3 squat, liberamente ed in base alle proprie condizioni, al fine di favorire il loro rilascio 3.4. Una volta effettuata l'acquisizione, ogni audio registrato è stato

Figura 3.4: Sequenza di Acquisizione

manualmente annotato da parte di tre blind team, ognuno composto da un medico iperbarico, membro del DAN, con almeno 10 anni di esperienza. Le annotazioni sono state effettuate al fine di individuare il numero di bolle presenti nell'audio per assegnare un grado di rischio. Il livello di rischio, basato sulla "Scala di Spencer" o "Scala di Spencer Estesa" viene utilizzato per definire la terapia a cui sottoporre il subacqueo come somministrazione di ossigeno, camera iperbarica o altro. La scala di Spencer divide i rischi del subacqueo,

Grado-0	Nessun segnale di bolle
Grado-0.5	1 o 2 segnali sporadici di bolle, sopra il minuto di registrazione
Grado-1	Fino a 5 segnali di bolle, sopra il minuto di registrazione
Grado-1.5	Fino a 15 segnali di bolle, sopra il minuto di registrazione e con presenza di showers
Grado-2	Fino a 30 segnali di bolle, sopra il minuto di registrazione
Grado-2.5	Oltre i 30 segnali di bolle, sopra il minuto di registrazione e con presenza di showers
Grado-3	Un continuo di segnali bolla, sopra il minuto di registrazione
Grado-3.5	Un continuo di segnali bolla, sopra il minuto di registrazione, con numerosi showers
Grado-4	Un continuo di segnali bolla, sopra il minuto di registrazione, con continui showers

Figura 3.5: Scala di Spencer Estesa

Livello scala di Spencer	Grado
0	0
0.5 ÷ 1.5	LBG
1.5 ÷ 2.5	HBG
2.5 ÷ 3.5	HBG+

Figura 3.6: Corrispondenza grado SSE con livello SS

basandosi sulle bolle contenute nell'audio, in LBG (Low Bubble Grade), HBG (High Bubble Grade) e HBG+ (Very High Bubble Grade) [48] [49]. La scala di Spencer estesa è una scala numerica con steps di 0.5 tra un livello e il successivo, questo consente una stima più completa dello stato di decompressione del subacqueo [50], come mostrato nella figura 3.5. Per ogni valore della scala di Spencer estesa esiste una corrispondenza con la scala di Spencer standard illustrata nella figura 3.6.

3.2 Principali tecniche usate per rilevare le bolle gassose

Il segnale audio, acquisito dalla registrazione Doppler, contiene, oltre ad eventi bolla, anche segnali di disturbo dovuti sia ai movimenti di valvole che ai battiti cardiaci che al rumore di fondo. Il disturbo può essere classificato come un *processo non stazionario*, ma *quasi periodico*, mentre gli eventi embolici sono *processi casuali non stazionari* [43] [51] [52] [53]. Benchè in letteratura ci siano già molteplici tecniche in grado di rilevare emboli dai segnali audio, queste si basano su acquisizioni svolte in regioni non precordiali, caratterizzate quindi da segnali di disturbo quasi del tutto assenti e da bolle facilmente identificabili dalla restante parte del segnale. Scelta la regione di monitoraggio, la sezione qui presentata descriverà gli algoritmi presenti in letteratura utilizzati per identificare tutte le bolle presenti in file eco Doppler audio.
Le principali tecniche usate per identificare bolle gassose in segnali eco Dop-

pler audio si basano sulla trasformata di Fourier, la trasformata Wavelet e l'Empirical Mode Decomposition (EMD).

3.2.1 Trasformata di Fourier

Il primo ad un utilizzare la trasformata di Fourier, per identificare bolle gassose in segnali eco Doppler, fu Kisman [43] che, già nel 1977, aveva realizzato un algoritmo molto performante in grado di rilevare gran parte delle bolle gassose contenute nei segnali audio. Il principale svantaggio del suo approccio è legato alla scelta della regione transcranica come sito di monitoraggio, che non consente di acquisire tutte le bolle circolanti nel flusso sanguigno. Inoltre l'acquisizione transcranica richiede personale esperto, con elevata manualità nel posizionamento della sonda Doppler. Un altro approccio da tenere in considerazione è quello di Tufan [54] che scelse di usare l'operatore "Teager Energy" [55] ed un set di filtri passa banda. Tufan, mediante tale operatore, definisce una rappresentazione non lineare di segnali audio per poi, mediante filtri passa banda, incrementare il rapporto tra l'intensità della bolla e il rumore di fondo. Successivamente viene realizzato un algoritmo che, definito un segmento di prova privo di bolle sul segnale audio e una soglia basata sulla potenza in tale segmento, effettua un confronto a soglia su tutto il segnale utile. Le bolle saranno definite come gli eventi in cui la potenza di segmenti del segnale utile supera la soglia precedentemente definita. Anche questo secondo approccio raggiunge buoni risultati, ma risulta sempre limitato alla regione transcranica, priva di rumore, ed inoltre resta vincolato ad una scelta manuale del segmento di prova, non realizzabile automaticamente a priori. Altri ricerche significative basate sull'utilizzo della trasformata di Fourier sono quelle di Girault [56], Zhang [57] e di Skogland [58]. Questo primo approccio, basato sull'utilizzo della trasformata di Fourier o sui filtri selettivi in frequenza, si adatta molto bene solo a segnali con percentuale di rumore di fondo molto bassa. La loro applicazione alla regione precordiale comporta avere un numero elevato di falsi positivi, ovvero segnali di disturbo come movimenti di valvole o battiti cardiaci identificati come bolle. Da qui l'esigenza di investigare nuove tecniche, che operino congiuntamente sia nel dominio del tempo che della frequenza.

3.2.2 Trasformata Wavelet

Un secondo tipo di approccio può essere effettuato mediante l'utilizzo della trasformata Wavelet discreta che consente di effettuare un'analisi tempo-frequenza, decomponendo il segnale in una serie pesata di coefficienti. Una delle prime applicazioni della trasformata Wavelet fu quella di Marvasti-Gilies-Marvasti [59] che realizzarono un algoritmo di rilevazione bolle in regione transcranica. Gli steps dell'algoritmo sono:

- Digitalizzazione delle acquisizioni;
- Annotazioni manuali dei file acquisiti da parte di più medici esperti. Solo i frame con valutazione concorde degli esperti viene definito come evento bolla;
- Filtraggio del segnale per eliminare rumore di fondo;
- 1° Caratterizzazione, stima del flusso ematico attraverso la rimozione degli eventi embolici mediante filtraggio;
- 1° Identificazione bolle, confronto tra segnale originale e segnale filtrato per identificare le possibili finestre contenenti bolle;
- 2° Caratterizzazione, calcolo di parametri significativi come maxPower - maxPower/Threshold - maxVariation;
- 3° Caratterizzazione, scelta dei due parametri più significativi in grado di rilevare con precisione gli eventi bolla;
- Applicazione trasformata Wavelet, calcolo dei parametri ottenuti dalla 3° caratterizzazione sui *"dettagli"* ed *"approssimazioni"* ottenuti dalla trasformata Wavelet;
- Filtraggio del segnale per eliminare rumore di fondo;
- Applicazione dell'algoritmo Fuzzy, con assegnazione di pesi diversi ai parametri scelti;
- Classificazione delle bolle mediante uso di soglie;
- Confronto risultati algoritmo con annotazioni manuali e tecniche FFT.

Il principale vantaggio di questa tecnica è la miglior localizzazione temporale delle bolle rispetto alle tecniche FFT. Essa infatti ci consente di rilevare il 60% delle bolle, sul totale degli audio testati, mantenendo costante il rapporto EBR (Embolic Background Ratio).
Un primo confronto tra la Wavelet e la WDT (Windowed Fourier Transform) venne fatto da Aydin-Markus [60] che compararono parametri significativi come "Half Width Maximum Time" (HWMT), "Half Width Maximum Frequency" ed EBR. Dai risultati ottenuti si evidenzia come la DWT permetta una miglior localizzazione di eventi embolici, anche se essi in questo studio vengono aggiunti artificialmente sul segnale audio.
Una successiva ricerca di Aydin e Serbes [61] ha esteso lo studio della DWT comparando i parametri EBR ed HWMT con quelli ottenuti dalla *"Dual Tree Complex Discrete Wavelet Transform"* (DTCWT). La DTCWT utilizza due

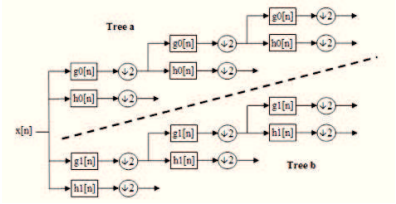

Figura 3.7: Dual Tree Complex Wavelet Transform *(wikipedia.org)*

banchi di filtri, uno superiore con filtri passa basso ed uno inferiore con filtri passa alto. Dato un set di audio campioni, acquisiti in regione transcranica, la DTCWT ha un miglioramento di circa 8dB rispetto la DWT, ma richiede un costo computazionale elevato a causa del numero di livelli necessari prima di ottenere un risultato accettabile. La figura 3.7 mostra la serie di filtri della DTCWT nei due rami superiore ed inferiore.
Rimanendo incentrati sull'uso della DWT, Aydin propose anche un nuovo metodo teorico per identificare bolle gassose in base all'andamento di parametri noti [51]. In questo studio viene valutato l'andamento dell'EBR che indica quanto sia forte il segnale rispetto al rumore di fondo. L'EBR può essere calcolato in due modi:

- P2TR, peak to threshold ratio, eq. 3.2
- TP2TR, total power to threshold ratio, eq. 3.3

$$P2TR = 10 * \log \frac{A_{peak}}{A_{th}} \tag{3.2}$$

$$TP2TR = 10 * \log \frac{A_{tot}}{A_{th}} = \frac{\sum_{t=t_{on}}^{t_{off}} A_{forward}(t)}{A_{th}} \tag{3.3}$$

La rappresentazione dell'EBR è illustrata nella figura 3.8. Per il calcolo della

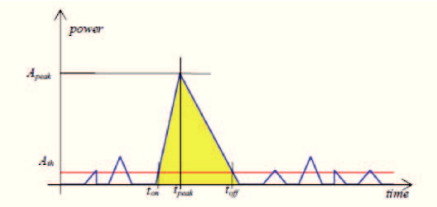

Figura 3.8: Rappresentazione livello A_{peak} ed A_{th}

soglia viene utilizzata l'equazione 3.4 di Donoho-Johnstone [62] [63], dove σ_{fn} e σ_{rn} sono la deviazione standard dell'energia del segnale diretto ed inverso all'

n_{th} scala e C è una costante di rifinitura.

$$A_{th} = C(\sigma_{fn} * \sqrt{\log_2 N} + \sigma_{rn} * \sqrt{\log_2 N}) \tag{3.4}$$

Aydin nella sua ricerca va a valutare le variazioni del parametro P2TR o TP2TR, che subisce un aumento in corrispondenza della bolla gassosa. Un'applicazione pratica della soglia proposta da Aydin è stata realizzata dall'algoritmo di Aydin-Marvasti-Markus [64], che seppur monitorizzando la regione transcranica, raggiunge il 90% di identificazione corretta di bolle su tutti i campioni di audio considerati.
Dallo studio dei segnali eco Doppler si nota come la bolla gassosa sia sovrapposta al segnale originale e quindi risulta difficile una separazione delle due componenti. Un miglioramento può essere raggiunto utilizzando la trasformata Wavelet, anche se tutti gli studi basano le loro fondamenta su un'acquisizione condotta in regione transcranica dove, al massimo, compaiono disturbi dovuti ad artefatti come movimenti della sonda, respiro e parlato. Anche se in letteratura non ci sono applicazione evidenti dell'utilizzo della trasformata Wavelet per rilevare bolle su acquisizioni condotte in regione precordiale, lo stesso Chappell suggerisce che la DWT possa essere una possibile via di ricerca futura [37].

3.2.3 Empirical Mode Decomposition

L'EMD è una tecnica di analisi proposta da Huang [65] con l'intento di rappresentare un segnale, non stazionario, non lineare e adattativo, sia nel tempo che nella frequenza. Questa tecnica empirica permette di scomporre il segnale usando basi di espansione, provenienti da processi non lineari e non stazionari, che derivano dai dati stessi. Il primo a tentare l'identificazione di bolle gassose mediante la tecnica EMD fu Chappell [37], utilizzando un set di dati fornito dal DAN. Ogni audio, di durata massima di 70 secondi, è caratterizzato da due fasi attive di registrazione, separate da una sezione di silenzio. L'idea di base dell'algoritmo di Chappell consiste nello scomporre il segnale in 20 componenti non stazionarie, chiamate IMF. Su ognuna di esse, dopo operazioni preliminari di filtraggio, si vanno ad identificare i picchi del battito cardiaco e la frequenza ad esso associata al fine di determinare l'intervallo temporale di tutti i picchi cardiaci. Dal segnale originale verranno poi eliminate tutte le componenti relative al battito cardiaco cosicchè, dopo la rimozione di segnali indesiderati e artefatti, verranno identificate le bolle mediante un confronto a soglia. Se l'energia media di una finestra di osservazione è maggiore dell'energia media di tutto il segnale, tale finestra verrà identificata come bolla. Prima di identificare la finestra come bolla, se il confronto a soglia dà esito positivo, si va a verificare la periodicità del picco di energia che può essere caratteristico del

movimento di valvole cardiache e quindi non considerato bolla. Questa tecnica è la prima ad usare un set di dati acquisiti in regione precordiale ed è particolarmente adatta per la rilevazione di bolle di grado elevato. Di contro, oltre a non essere performante nell'identificare le bolle piccole, sia singole che multiple in successione, si possono verificare dei falsi positivi dovuti ai movimenti della sonda. Altro svantaggio non trascurabile è il costo computazionale necessario per scomporre il segnale in più componenti, al fine di trovare la componente che contiene le singole bolle. Successivo approccio è quello di Li-aung Yip [66] che tentò di realizzare un algoritmo EMD real-time per identificare bolle intra vascolari. L'algoritmo EMD real-time viene ottenuto come una cascata di operazioni di sifting, mostrato in figura 3.9. Poichè l'algoritmo non dispone

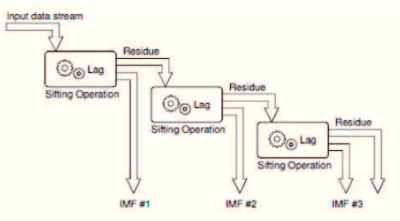

Figura 3.9: Struttura per il calcolo dell'EMD real-time *(articolo Li-aung Yip)*

del segnale intero, il blocco iniziale memorizza il dato prodotto dalla sorgente real-time su un buffer, al fine di avere una porzione sufficiente di segnale da elaborare. Calcolata la IMF1, il primo blocco passa il residuo al blocco successivo, che ripete la memorizzazione su un nuovo buffer fino ad avere dati a sufficienza da elaborare. Calcolata la IMF2, il nuovo residuo viene passato al blocco successivo e l'operazione iterata per tutti i blocchi considerati nell'algoritmo. Il problema che si può verificare applicando questo algoritmo è l' "end-effect", ovvero la discontinuità tra i blocchi IMF, come illustrato in figura 3.10. Una

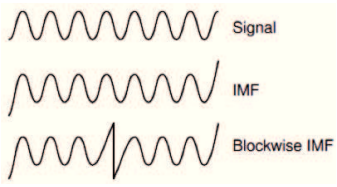

Figura 3.10: "end-effect" tra blocchi IMF adiacenti *(articolo Li-aung Yip)*

prima strategia per la rimozione delle discontinuità consiste nell'impostare un numero fisso di iterazioni di sifting. Se ciò non bastasse, si può ricorrere alle Spline di Hermite, all'estensione dei punti dell'intervallo di interpolazione o al calcolo dell'IMF con più dati del necessario e tagliare i bordi superflui. Que-

sto algoritmo si presta bene per segnali semplici, mentre l'elaborazione diventa onerosa con segnali complessi, dove si raggiungono solo velocità di elaborazione pari al 10% del segnale reale.
Ulteriore metodo considerato in letteratura è quello di Pinle-Yan-Ming [67], che propongono un metodo di calcolo della EMD basato sulla trasformata Wavelet. L'idea di base di questo studio è che un segnale può essere affetto da "intermittenze" che comportano una errata creazione delle IMF. In questo caso prima viene scomposto il segnale mediate la trasformata Wavelet e poi applicata l'EMD per ottenere IMF, prive di modi mescolati.
La letteratura propone svariati metodi di rilevazione di bolle gassose in segnali eco Doppler audio, acquisti in regione transcranica e succlavia. L'unico approccio che tenta di elaborare audio precordiali fu quello di Chappell, il quale però non considera la variabilità della frequenza cardiaca, presente nel nostro caso pre-post sforzo fisico, dovuto al rilascio delle bolle intra-tissutali. Ovviamente come lo stesso Chappell afferma, sono necessarie oltre che al test di un numero sempre maggiore di campioni, delle tabelle standardizzate contenenti le bolle e il grado di rischio assegnato al file testato, con cui confrontare le performance degli algoritmi realizzati.

3.3 Algoritmo automatico di rilevamento bolle gassose

In questo paragrafo verrà descritto l'algoritmo realizzato per identificare bolle gassose in segnali eco Doppler audio, acquisiti in regione precordiale. Il software realizzato è composto da due sotto algoritmi: il primo basato su tecnica EMD, per rilevare la gran parte di bolle e un secondo, basato sul calcolo della densità spettrale di potenza, per confermare le bolle identificate dal primo algoritmo e per individuare i falsi negativi, ovvero le bolle esistenti non rilevate dalla tecnica EMD. In questa fase l'algoritmo è eseguito off-line su audio acquisiti in istanti temporali diversi.

3.3.1 Implementazione EMD

Di seguito l'elenco degli steps eseguiti:

1. Pre-processing audio acquisito;
2. Definizione del numero di operazioni di sifting, necessarie al calcolo delle IMF, ed applicazione dell'algoritmo EMD;
3. Calcolo soglia T_h sul segnale originale;
4. Finestratura del segnale in frame da 10ms;
5. Calcolo del parametro P2TR per ogni frame;
6. Confronto a soglia;
7. Rifinitura del segnale:
   a) Identificazione Shower;
   b) Accorpamento bolle adiacenti;
   c) Calcolo livello di Spencer e gradazione associata.

Gli steps dell'algoritmo sono illustrati nella figura 3.11.

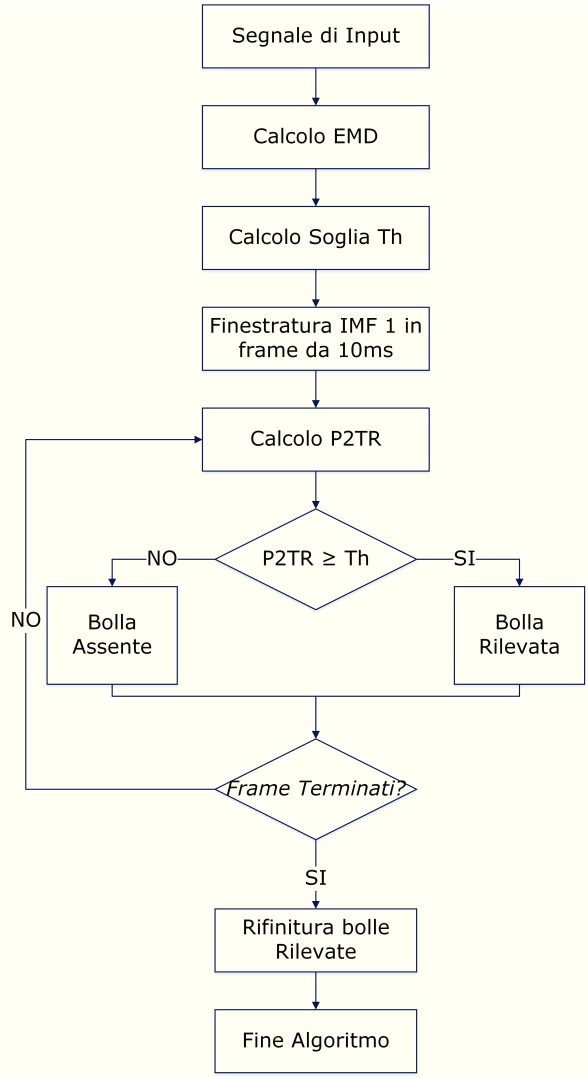

Figura 3.11: Diagramma di flusso EMD

Pre-processing Audio Una volta scelto l'audio da elaborare, vengono eseguite delle operazioni automatiche di taglio del segnale, al fine di eliminare parti che influirebbero negativamente sull'identificazione delle bolle. Le parti rimosse sono il silenzio centrale e i primi 0.5 secondi iniziali e finali dell'acquisizione. La zona centrale viene rimossa poichè, mentre il subacqueo effettua esercizi fisici

per il rilascio delle bolle intra-tissutali, il segnale acquisito sarà pressochè nullo andando ad influenzare negativamente i parametri globali e le soglie. I primi ed ultimi 0.5 secondi dell'acquisizione, sono zone dove viene posta e rimossa la sonda dal corpo del soggetto, con conseguente creazione di picchi di elevata ampiezza, che vanno ad influenzare negativamente nel rilevamento di bolle. La figura 3.12 mostra un esempio di file audio originale. La regione di silenzio

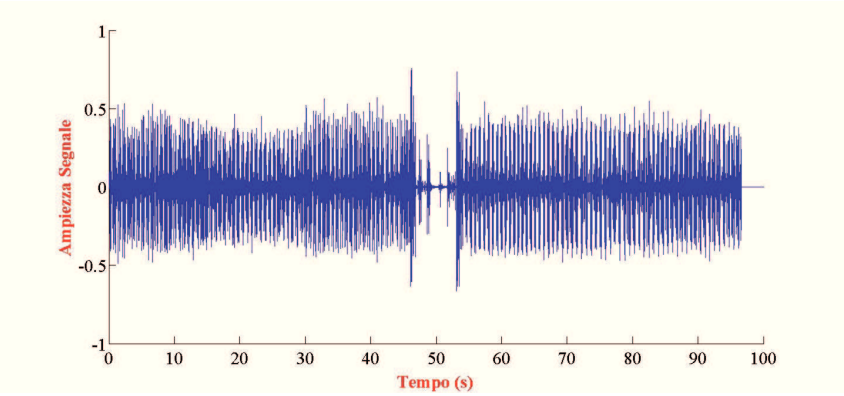

Figura 3.12: File audio originale

centrale viene rimossa individuando i due picchi dovuti al posizionamento e alla rimozione della sonda. Al segnale originale viene sovrapposto un secondo segnale formato da finestre rettangolari di durata 10ms, ognuna di ampiezza pari a quella del campione massimo contenuto, come mostrato in figura 3.13. La prima acquisizione sarà delimitata dal primo campione dell'audio acquisito

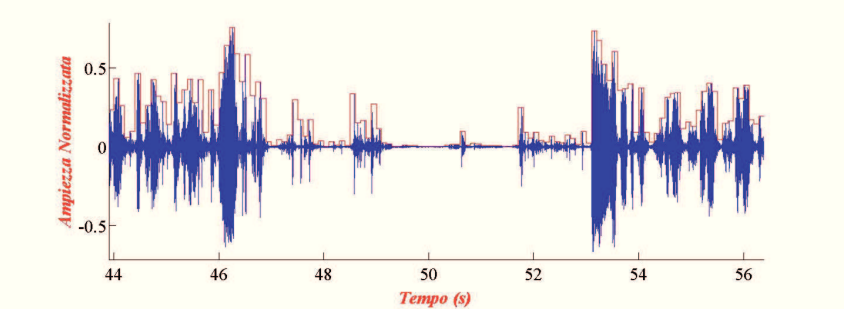

Figura 3.13: Regione di silenzio centrale con finestratura

all'ultimo della finestra, che anticipa quella di picco massimo. La seconda acquisizione è invece delimitata dal primo campione della finestra, che segue la seconda di picco massimo, all'ultimo campione del segnale acquisito. Le finestre di picco massimo sono quelle, la cui ampiezza supera di 0.2 quella media

calcolata su tutte le finestre del segnale. Da un test condotto su tutti i 240 file audio a disposizione si evidenzia come questa condizione sia sufficiente a determinare, in ogni file, solo i due picchi che delimitano le regioni di silenzio. La seconda operazione di taglio viene condotta rimuovendo i campioni corrispondenti ai primi ed ultimi 0.5 secondi di acquisizione che, sperimentalmente, risulta essere la durata del tempo necessario al posizionamento della sonda. Questi campioni non contengono bolle, bensì solo artefatti dovuti al movimento della sonda stessa. Le due acquisizione saranno poi unite per formare un unico segnale da elaborare come mostrato nella figura 3.14.

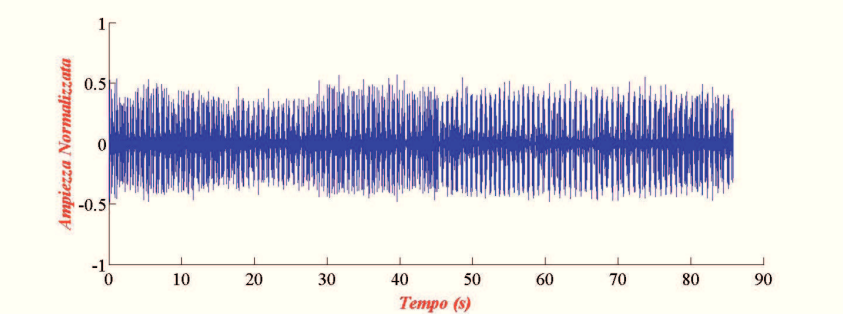

Figura 3.14: Segnale tagliato

Algoritmo EMD Dopo aver tagliato il segnale e rimosso i campioni non utili, viene scelto il numero di iterazioni sifting (S) necessarie per l'applicazione dell'algoritmo EMD. Al fine di valutare il valore ottimale di S, sono state condotte estrazioni di IMF sia con S basso (pari ad 1), che con S alto (maggiore o uguale a 10), su tutti i 240 file a disposizione. Dai test empirici condotti, porre S=1, consente di estrarre funzioni di modalità intrinseca tali da non soddisfare le due condizioni imposte da Huang [65], necessarie per poter considerare il segnale estratto come una IMF utile. La scelta di un S molto grande comporta invece un eccessivo costo computazionale necessario per estrarre le IMF. Inoltre in segnali con gradi di rischio basso (0.5), la scelta di S maggiore di 10 provoca una riduzione sul numero di bolle rilevate che viene portato drasticamente a 0 [68] [45] [69]. Da ciò la scelta effettuata sul numero di iterazioni di sifting è pari a 5, basandoci anche sulla letteratura dello stesso Huang che pone S tra 3 e 5 [65]. Lo stesso Huang afferma che, dato un segnale audio, potremmo estrarre tante componenti IMF fino a che l'ultimo residuo sia una componente monotona. Nell'algoritmo considerato è stato sufficiente estrarre solo la IMF di livello uno, poichè da un'analisi fonetica (condotta dal team medico) è possibile notare l'assenza quasi totale di rumore di fondo, con solamente le componenti dovute a movimenti di valvole, battiti cardiaci e bolle.

Calcolo Soglia In questa fase è stata calcolata una soglia sulla IMF 1, al fine di individuare un limite che permettesse di discriminare un segnale di bolla da uno non bolla. Per il calcolo della soglia viene utilizzata la formula di Donoho-Johnstone [62] [63] [70], che stima la rappresentazione di segnali non lineari. La formula alla base della soglia, utilizzata anche negli studi di Aydin [51], è la 3.5.

$$A_{th} = \sigma_B * \sqrt{2logN} \tag{3.5}$$

I parametri utilizzati nell'equazione di Donoho-Johnstone sono stati:

- N, lunghezza del segnale audio in ingresso;
- σ_B, parametro di rumore stimato mediante la deviazione assoluta mediana standardizzata (MAD) e definita in 3.6.

$$\sigma_B = \frac{median|x(t) - median(x(t))|}{0.6745} \tag{3.6}$$

Al fine di semplificare il calcolo della soglia è stata utilizzata una funzione del simulatore Matlab, che calcola automaticamente la soglia A_{th} dato il segnale di ingresso x(t).

Finestratura IMF1 Una volta definiti i parametri iniziali, eseguita la EMD e calcolata la soglia, viene applicata una finestratura al segnale IMF 1. Ogni finestra rettangolare, senza campioni di sovrapposizione con le finestre adiacenti, ha una durata di 10ms, considerata durata minima di una bolla gassosa [54].

Calcolo P2TR Il parametro utilizzato per distinguere le componenti bolle da quelle di disturbo è l'EBR *("Embolic background Ratio")*, calcolato mediante il parametro P2TR.

$$P2TR = 10 * \log \frac{A_{peak}}{A_{th}} \tag{3.7}$$

Nella formula 3.7 A_{peak} è il campione con massima ampiezza nel frame da 10ms del segnale IMF1, mentre A_{th} è il valore di soglia calcolato nella fase precedente. La figura che descrive il calcolo dell'indice P2TR à la 3.15.

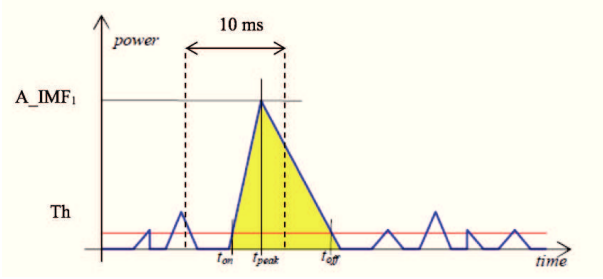

Figura 3.15: Individuazione APeak e ATh su frame da 10ms

Confronto a Soglia L'identificazione delle bolle avviene mediante un confronto a soglia fatto tra il parametro P2TR, calcolato su ogni finestra di 10ms, e la soglia calcolata nelle fasi precedenti. Le condizioni verificate sono:

- P2TR $\geq A_{th}$, parametro P2TR maggiore o uguale alla soglia, questo evento indica la presenza della bolla nel frame considerato;
- P2TR $< A_{th}$, parametro P2TR minore della soglia , questo evento indica che nel frame considerato non è presente l'evento bolla.

Nei file audio ci possono essere bolle che hanno una durata maggiore di 10ms o sequenze di bolle di breve durata. L'identificazione di bolle adiacenti avviene mediante l'utilizzo di un flag che, ad ogni confronto a soglia, verrà posto a *true* in presenza della bolla (confronto a soglia con esito positivo) o a *false* in caso di assenza della bolla (confronto a soglia con esito negativo). Quando si individua una bolla, si va a verificare lo stato del flag, che si riferisce alla presenza di bolla nel frame precedente. Se nel frame corrente viene trovata una bolla e il flag assume valore *true*, i due frame possono essere considerati come bolle adiacenti. Le bolle rilevate verranno memorizzate in una matrice, dove ogni riga conterrà l'istante di comparsa della bolla. Nel caso di più frame adiacenti contenenti bolla, la riga della matrice conterrà tante colonne quante i frame con bolla, ognuna con il proprio riferimento temporale. La fase di controllo a soglia viene effettuata per tutti i frame della IMF 1 dell'audio considerato.

Rifinitura Bolle La fase di rifinitura è formata da una serie di elaborazioni che comprendono:

- Identificazione di Shower;
- Accorpamento Bolle;
- Calcolo livello di Spencer e grado di rischio.

La sequenza che descrive le operazioni di rifinitura è mostrata nel diagramma 3.16.

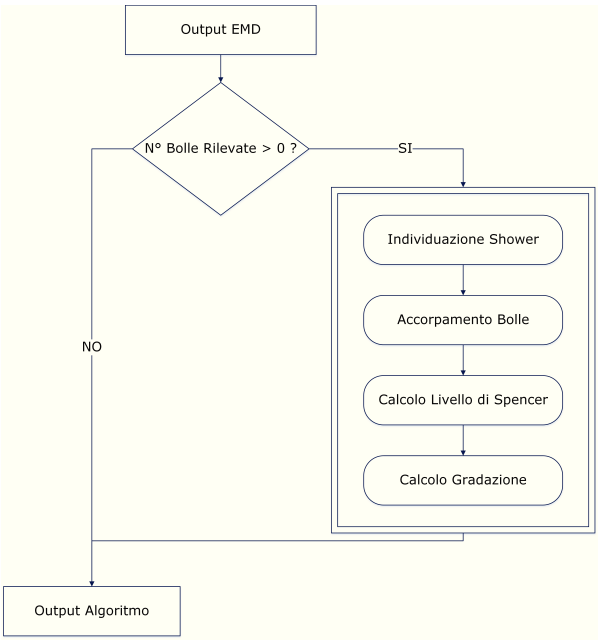

Figura 3.16: Rifinitura Bolle

Identificazione Shower Gli shower vengono definiti come un flusso continuo di bolle, presenti soprattutto dopo il rilascio delle bolle intra-tissutali nella seconda fase di acquisizione. Definiti i frame contenenti bolle, gli shower vengono identificati come quelle coppie di bolle, ognuna nel proprio frame da 10ms, separate da un frame senza bolle.

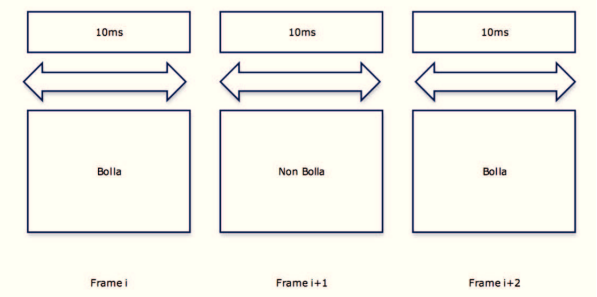

Figura 3.17: Rappresentazione degli Shower

Vista la struttura della matrice contenente bolle, l'idea chiave dell'algoritmo di identificazione shower è di prendere l'ultimo elemento di ogni riga (istante temporale della singola bolla o di fine bolle adiacenti) e confrontarlo con il primo elemento della riga successiva. Se tra i due elementi la distanza temporale è minore di un certo Δ allora le due righe potranno essere considerate shower. Gli steps dell'algoritmo sono:

1. Data la riga i-esima, si cerca l'ultimo elemento non zero;
2. Data la riga i-esima +1, si cerca il primo elemento non zero;
3. Confronto tra istante temporale dell'elemento al passo 1 più un certo gap (20ms) e l'elemento selezionato al passo 2.
   a) Se la condizione dà esito negativo, si reitera il processo alle righe successive fino alla fine della matrice;
   b) Se la condizione dà esito positivo, le operazioni eseguite sono:
      i. Concatenazione delle due righe in un'unica riga;
      ii. Sostituzione delle due righe con quella unificata nella matrice originale;
      iii. Aggiornamento variabili aggiuntive di controllo e reiterazione del processo alle righe successive fino alla fine della matrice.

Il diagramma di flusso dell'algoritmo è illustrato in figura 3.18.

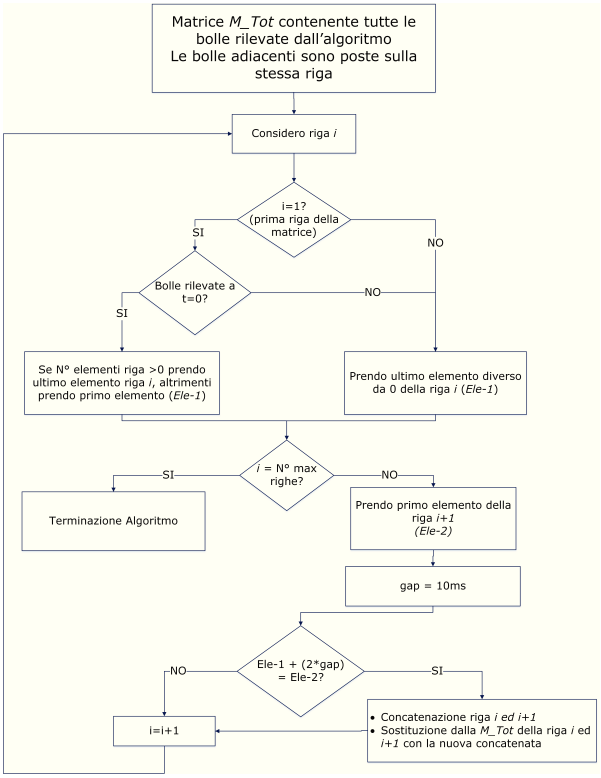

Figura 3.18: Diagramma di flusso per identificare Shower

Accorpamento Bolle La successiva operazione di rifinitura consiste nel raggruppare tutte le bolle rilevate in frame adiacenti in singole bolle. Benchè questa operazione possa esser simile a quella effettuata in fase iniziale di rilevamento bolle, in questa parte viene rifatta, escludendo tutti gli shower precedentemente calcolati. È possibile inoltre specificare il gap di accorpamento, ovvero la distanza temporale che devono avere due bolle, contenute in due frame differenti, affinchè possano essere considerate unica bolla.
Partendo dalla matrice contenente sia bolle che shower *"M_Tot"*, vengono rimossi tutti gli shower al fine di avere solo eventi bolla. Quindi verrà generata una nuova matrice, *"M_Tot2"*, contenente nella prima riga tutti i secondi dell'audio, con passo pari al gap predefinito (10ms). La seconda riga conterrà invece un flag posto ad 1, quando l'istante corrispondente nella prima riga contiene una bolla e 0 quando non la contiene. Per decidere se il flag deve essere posto a

0 o ad 1, partendo dalla matrice *"M_Tot"*, si prende ogni istante di comparsa bolla e si cerca la sua posizione nella nuova matrice *"M_Tot2"*, impostando il corrispondente flag. Terminata tale operazione, nella matrice *"M_Tot2"*, le righe adiacenti con flag ad 1 vengono accorpate come singola bolla. Ogni istante di comparsa della bolla, alla fine dell'algoritmo, verrà arrotondato per eccesso in intero con due cifre dopo la virgola.

Calcolo Livello di Spencer e Gradazione L'ultima operazione di rifinitura si occupa di assegnare un livello della scala di Spencer e un grado della scala di Spencer Estesa. Il quantitativo di bolle necessario ad assegnare un particolare livello dell'SS o grado della ESS è definito nelle figure 3.5 e 3.6. La presenza di shower introduce un innalzamento di 0.5 gradi alla scala di Spencer Estesa.

3.3.2 Implementazione PSD

Con l'intento di andare ad individuare anche le micro bolle, che l'algoritmo EMD non è in grado di rilevare, è stato realizzato un secondo algoritmo da eseguire in seguito al primo, basato sul calcolo della densità spettrale di potenza (PSD). L'idea alla base dell'algoritmo PSD è di avere un incremento di PSD in corrispondenza di una bolla, rispetto ad una PSD di riferimento. Le fasi dell'algoritmo sono:

1. Scelta del segnale e della finestra senza bolle, con cui calcolare la PSD di riferimento (PSD_{rif});
2. Scelta del segnale con bolle e finestratura in finestre di lunghezza fissa. Calcolo delle finestre di valutazione (PSD_{val-i});
3. Confronto a soglia di tutte le finestre: $PSD_i \geq PSD_{rif}$. Se la condizione è verificata, la finestra i verrà identificata come finestra con bolla. Iterazione del confronto per tutte le finestre.

Il diagramma di flusso 3.19 illustra gli steps realizzati dall'algoritmo.

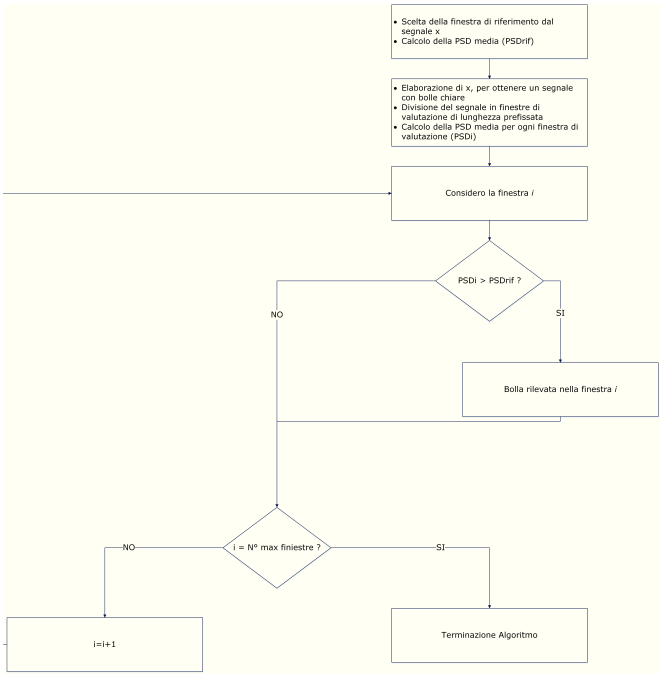

Figura 3.19: Diagramma di flusso algoritmo PSD

Calcolo PSD_{rif} Il primo step, necessario per l'esecuzione dell'algoritmo PSD, è l'individuazione di una porzione del segnale audio, priva di bolle, su cui calcolare la densità spettrale di potenza di riferimento. Le porzioni di segnale prese in considerazione sono state le finestre contenenti singoli battiti cardiaci e quelle prive di bolle. La finestra con singoli battiti cardiaci, presuppone però la conoscenza a priori del suo istante d'inizio, della sua durata ed eventualmente della frequenza associata; il tutto tenendo in considerazione situazioni di bolle sovrapposte al battito cardiaco. La seconda soluzione, che prevede la scelta di una porzione di segnale priva di bolle, non può essere attuata, poichè non è possibile determinarla automaticamente, senza prima un'analisi fonetica da

parte del team medico. Da qui si è deciso di prendere in considerazione tutto il segnale, sia con le zone prive di bolle che con quelle contenenti bolle. Gli steps per il calcolo della PSD_{rif} sono illustrati in figura 3.20.

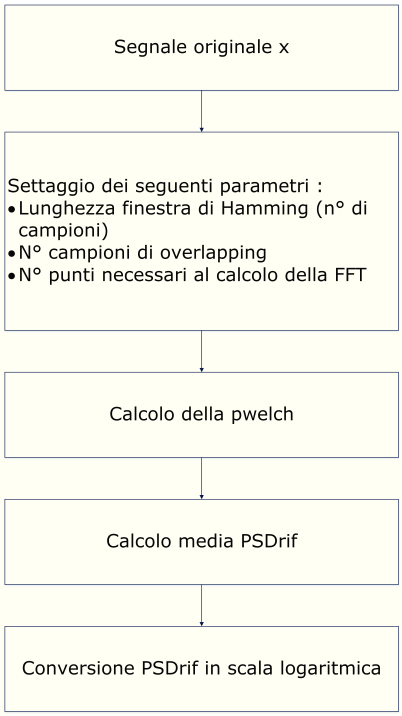

Figura 3.20: Steps per il calcolo della PSD_{rif}

Il grafico 3.21 mostra l'andamento della PSD per le singole frequenze e quella mediata.

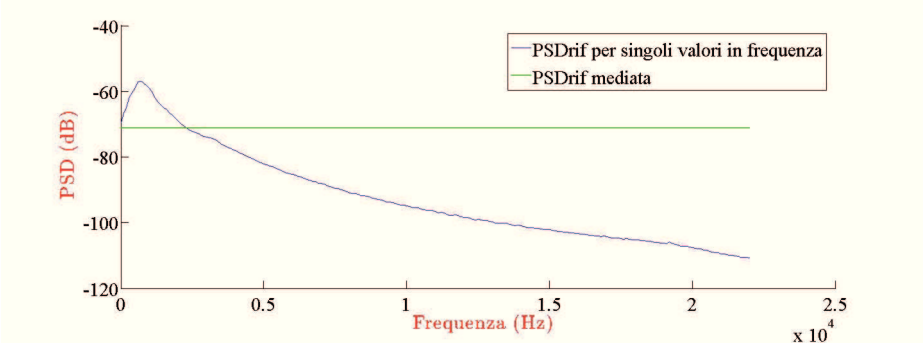

Figura 3.21: PSD_{rif} del segnale originale

Calcolo PSD_{val} Dopo aver individuato la densità spettrale di riferimento, è necessario scegliere la componente del segnale, contenente bolle, da finestrare per il confronto a soglia delle relative PSD. Le basi del segnale, prese in considerazione, sono state il segnale originale x, la IMF 1 e la IMF 3. La IMF 1 è stata scelta poichè gran parte del rumore di fondo risulta rimosso e le bolle sono facilmente distinguibili dai movimenti di valvole e dai battiti cardiaci. La IMF 3 invece è utilizza come segnale di conferma, poichè è presente solo il battito cardiaco, che nonostante subisca una forte attenuazione, risulta ancora perfettamente riconoscibile. I segnali scelti sono così stati divisi in finestre di lunghezza iniziale 10ms, su ognuna delle quali poi è stata calcolata la PSD media. Per la densità spettrale di potenza relativa ad un certo segnale *"s"*, ad una finestra i-esima *"i"* di lunghezza arbitraria *"l"* in millisecondi, è stata definito l'acronimo PSD_{l-s-i}. Le figure 3.22 e 3.23 mostrano esempi di calcolo delle PSD.

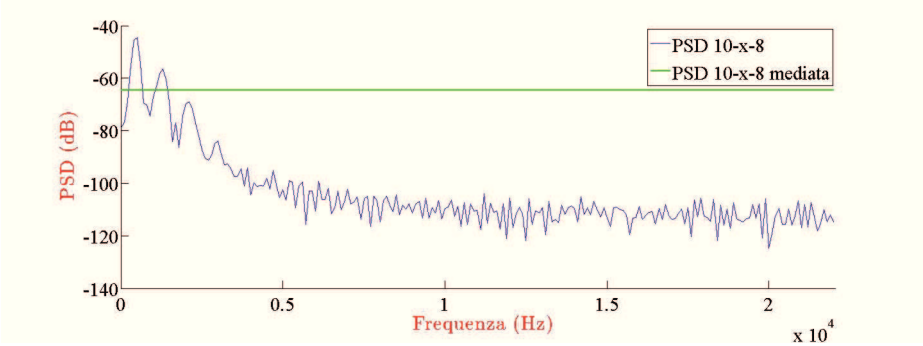

Figura 3.22: PSD della finestra 8 di lunghezza 10ms del segnale x

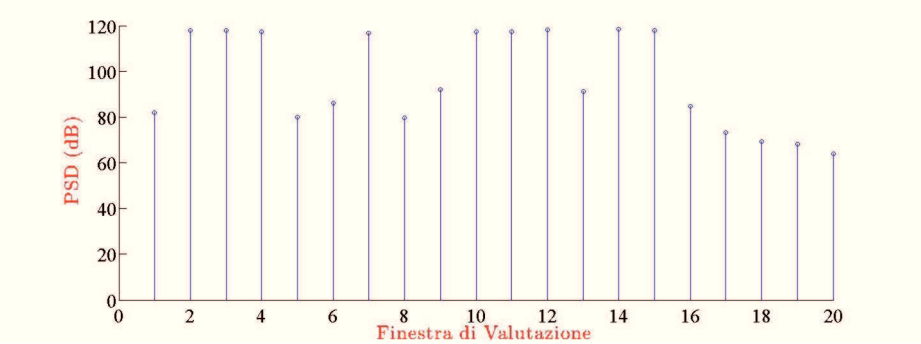

Figura 3.23: Modulo delle PSD medie delle prime 20 finestre da 10ms del segnale originale x

Confronto a Soglia L'ultimo step dell'algoritmo PSD si occupa di un confronto a soglia tra le varie PSD_{val}, delle singole finestre, con quella di riferimento PSD_{rif}. La finestra i-esima, la cui relativa densità spettrale di potenza PSD_{l-s-i} supera la PSD_{val}, viene indicata come finestra con bolla. Le operazioni fatte in questa fase sono raffigurate nel diagramma 3.24.

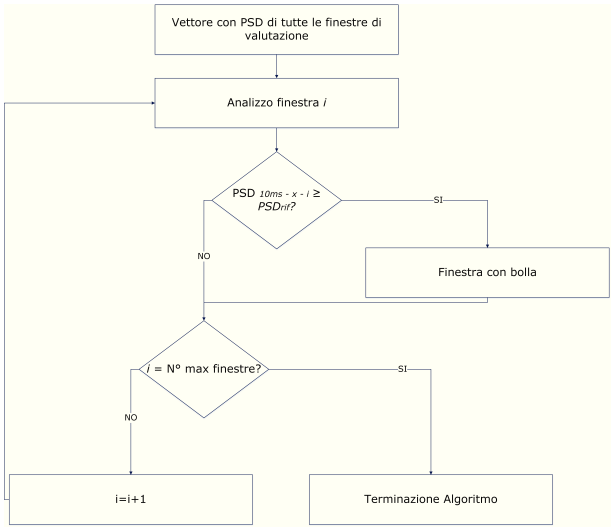

Figura 3.24: Diagramma di flusso delle operazioni necessarie per il confronto a soglia

Il confronto a soglia sarà ripetuto per tutti e tre i segnali, x, IMF 1 e IMF 3. I grafici 3.25, 3.26 e 3.27 mostrano i confronti a soglia delle prime 100 finestre di valutazioni per le tre basi di segnale scelte.

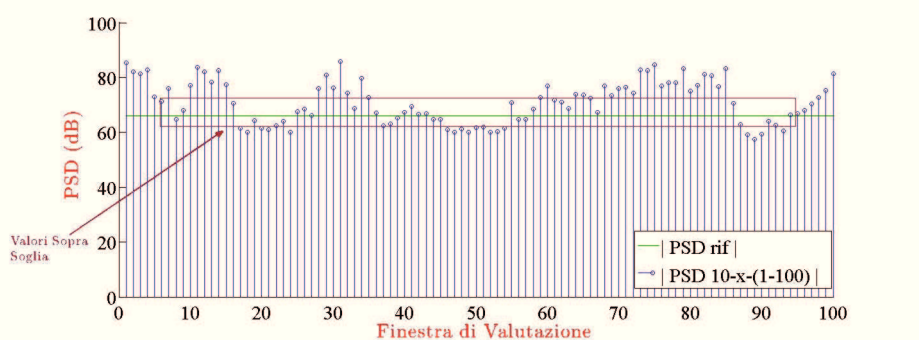

Figura 3.25: Confronto di 100 PSD di valutazione e la PSD di riferimento del segnale x

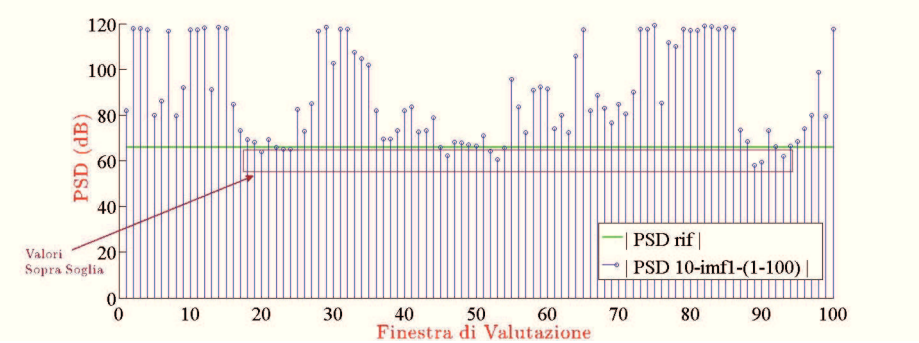

Figura 3.26: Confronto di 100 PSD di valutazione e la PSD di riferimento del segnale IMF 1

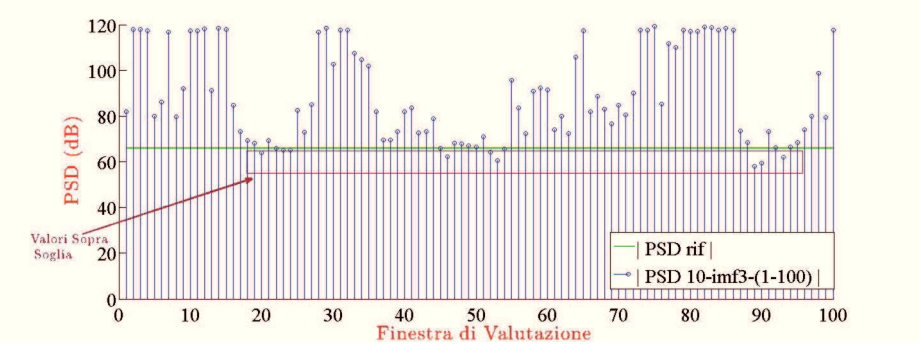

Figura 3.27: Confronto di 100 PSD di valutazione e la PSD di riferimento del segnale IMF 3

Dai test eseguiti si è visto come il miglior segnale base da utilizzare è la IMF 1. Nel segnale originale molti confronti a soglia producono falsi positivi, cioè bolle non esistenti identificate come tali, dovute essenzialmente al rumore di fondo che causa un aumento della densità spettrale di potenza. Il segnale IMF 1 fornisce una buona corrispondenza tra le bolle rilevate dall'algoritmo e quelle dall'analisi fonetica, condotta dal team medico. I test sulla IMF 3 invece producono un elevato numero di falsi positivi, causato principalmente dalla potenza dei battiti cardiaci, i soli ad essere presenti in questo file. Da test empirici si è visto come la lunghezza della finestra ottimale, che garantisce una corrispondenza tra i il numero di bolle calcolate con l'algoritmo EMD e quelle ottenute da un'analisi fonetica, sia tra i 20ms e 50ms. Nel nostro caso la durata della finestra scelta è stata 20ms.

3.4 Algoritmo automatico su scheda Embedded

L'obiettivo del lavoro è stato quello di realizzare una versione embedded dell'algoritmo di rilevazione bolle per una prima stima del livello di Spencer e del grado rischio associato. Il progetto nasce dall'esigenza del DAN di avere un dispositivo pronto all'uso nella fase di post-emersione. Benchè il picco massimo di bolle si registri dai 30 ai 60 minuti dall'emersione stessa [47], spesso il personale a bordo delle imbarcazioni non ha sufficiente esperienza nel riconoscere bolle in eco Doppler audio ed i tempi necessari al ritorno in porto potrebbero essere anche superiori all'ora stessa. In questi casi una prima stima dell'ESS e SS potrebbe essere utile per avvisare i soccorsi agendo tempestivamente prima dell'insorgenza di gravi complicanze fisiche.

3.4.1 Scheda Embedded

Il dispositivo embedded scelto è stata la Raspberry PI, scheda di piccole dimensione, che funge da mini computer su singola board. Inizialmente venne sviluppata dalla *"Raspberry PI Foundation"* per scopi didattici per poi diffondersi a macchia d'olio anche in altri settori come quello domestico, per applicazioni di videosorveglianza [71] o di media player [72], a quello industriale [73]. Nel corso degli anni sono state rilasciate diverse versioni di Raspberry, quella utilizzata (Raspberry PI 2 Model B) monta un processore A7 della famiglia ARM, 512MB di memoria, 4 porte USB, 1 porta HDMI, 1 porta Ethernet e 40 pin di GPIO per l'interfacciamento con altri sottomoduli. Il sistema operativo utilizzato è il Raspbian, versione appositamente customizzata del Debian per Raspberry, che deve essere posta su SD esterna in ingresso alla scheda. Per

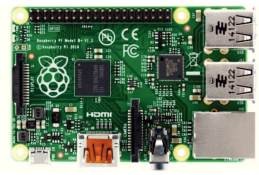

Figura 3.28: Raspberry PI 2 Model B *(raspberrypi.org)*

dotare la scheda di uno schermo, piuttosto che collegare schermi esterni con cavo HDMI, è stato utilizzato uno schermo touch LCD da 7 pollici, collegato tramite cavo piatto alla porta DSI della scheda Raspberry. Poichè la scheda Raspberry non presenta ingressi audio analogici e l'unica uscita audio garantisce una qualità standard, si è deciso di acquistare un shield della *"Wolfson Microelectronic"*, che consente di avere le funzionalità audio richieste ad elevate prestazioni. Le caratteristiche della scheda Wolfson sono un ingresso e due uscite analogiche, due microfoni stereo, un'uscita e un ingresso digitale di tipo

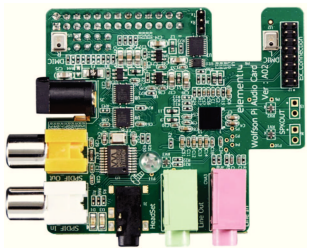

Figura 3.29: Wolfson Audio Card *(farnell.com)*

SPDIF (Sony Philips Digital Interface Format) oltre alla possibilità di collegare direttamente speaker esterni grazie alle uscite dx e sx. Il chipset a bordo della scheda è l'WM5102, progettato appositamente per la gestione di audio integrato nei dispositivi portabili [74]. Il collegamento tra Wolfson e Raspberry avviene attraverso il bus I2C, mentre le librerie utilizzate fanno parte della famiglia *"ALSA Libreries"*, acronimo di *"Advanced Linux Sound Architecture"*. Le librerie, appositamente trasferite su scheda SD, devono essere opportunamente installante dopo il collegamento della scheda. Esse possono supportare fino ad 8 dispositivi audio, ognuno numerato che può essere utilizzato per una apposta funzione come *"playback"*, *"capture"*, *"control"* e *"timer"*. L'ALSA Stream, flusso dati che rappresenta la traccia sonora, è generalmente il PCM (Pulse Code Modulation), cioè campioni convertiti in digitale ed inviati in sequenza. Ovviamente il flusso dati inviato deve essere concorde con l'hardware sottostante (Raspberry) in merito a frequenza di campionamento, profondità di bit, numero di canali e codifica dei campioni. La figura 3.30 mostra il funzionamento delle librerie tra kernel ed applicativo.

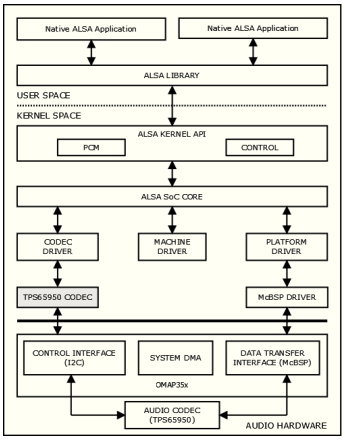

Figura 3.30: Principio funzionamento ALSA Libreries *(alsa-project.org)*

3.4.2 Acquisizione Audio

L'interfaccia audio è quel dispositivo che, mediante l'uso di convertitori A/D, permette al computer di ricevere flussi di bit. Sulle interfacce sono presenti dei buffer che, in maniera circolare, vengono riempiti segnalando al computer la presenza di dati disponibili. In questa prima parte di progetto è stato opportunamente scritto il codice in grado di registrare l'audio in ingresso, esportarlo in formato WAVE ed eventualmente riprodurlo.
Per effettuare la registrazione audio, mediante l'utilizzo delle librerie ALSA, vengono utilizzati degli *handler*, ovvero delle zone di memoria usate come interfaccia di alto livello per il dispositivo. Gli step necessari per la configurazione dell'handler e per l'avvio della registrazione sono:

- Definizione dell'handler;
- Definizione direzione flusso audio (riproduzione o acquisizione);
- Associazione dell'handler al dispositivo;
- Apertura del dispositivo;
- Definizione tipo di accesso (tipologia di dati ricevuti, campione del canale destro intervallato al campione del canale sinistro o prima tutti i campioni di un canale poi quelli dell'altro);
- Definizione formato dei campioni (campione di float che occupa 4Byte, memorizzati in notazione LittleEndian);
- Definizione frequenza di campionamento (96KHz);
- Definizione numero di canali (1 mono o 2 stereo);
- Definizione dimensione periodo (512 frames);
- Associazione parametri all'handler;
- Allocazione memoria nello spazio utente per contenere i dati registrati mediante formula: *dim_mem=dim_float*sample_rate* *secondi_registrazione*num_canali*byte_per_sample* (ad esempio per 10s di registrazione verranno destinati 3,84MByte della memoria);
- Chiamata ricorsiva della funzione di lettura dati (ad ogni interrupt della scheda il pc andrà a prendere i dati dal buffer e li memorizzerà nel vettore allocato al livello utente);
- Dopo ogni lettura viene incrementato il puntatore alla zona corrente di scrittura.

Con *frame* si fa riferimento ad un campione elementare, singolo nel caso di audio mono e multiplo (canale destro e canale sinistro) nel caso di audio stereo. La dimensione del buffer sarà espressa in numero di frame in grado di immagazzinare. Si può definire il periodo un certo numero di frame, in questo caso pari a 512.
Altra funzione implementabile con l'uso delle librerie ALSA è la riproduzione dei campioni audio salvati in memoria utente, in questo caso gli steps da eseguire sono gli stessi dall'acquisizione, andando però a cambiare la direzione del flusso in riproduzione. Iterativamente i dati della memoria utente saranno pressi a blocchi (basati sulla lunghezza del periodo) e passati alla funzione ALSA per l'ascolto sull'apposita uscita audio.
L'operazione successiva che viene eseguita consiste nel salvare i campioni, presenti nella memoria di livello utente, in un file WAVE su scheda SD. I passi utilizzati per scrivere l'audio WAVE, rispettando la sua struttura illustrata nell'appendice, sono:

- Definizione variabile *"struct"* contenente tutti i campi dell'header del WAVE, ognuno con la propria dimensione (long int o short int);
- Riempimento dei campi della struct con i dati dell'audio, considerando la notazione BigEndian o LittleEndian di ogni campo;
- Definizione di una variabile di tipo file e apertura della stessa per scrittura binaria;
- Scrittura sul file dei campi della struct;
- Scrittura sul file del blocco dati salvato in memoria (poichè ogni dato è salvato in tipo float, è necessario dividerlo in 4 byte, ognuno in notazione LittleEndian);
- Chiusura del file e salvataggio con estensione *".wave"*.

Tutte le funzioni sopra descritte (acquisizione, riproduzione e salvataggio WAVE) sono eseguite in sequenza nell'algoritmo di rilevazione bolle, implementato su scheda embedded. Al fine di eseguire singolarmente le operazioni, quindi non legate unicamente all'algoritmo, è stato realizzato un programma main eseguito come bash su schermo touch o da remoto con terminale ssh. La figura 3.31 illustra il main del programma. All'avvio del programma, un menù contestuale mostra le possibili operazioni da eseguire, ognuna con il proprio numero identificativo. Inserito tale valore, verrà chiamata la funzione associata ed eseguite le operazioni sopra descritte.

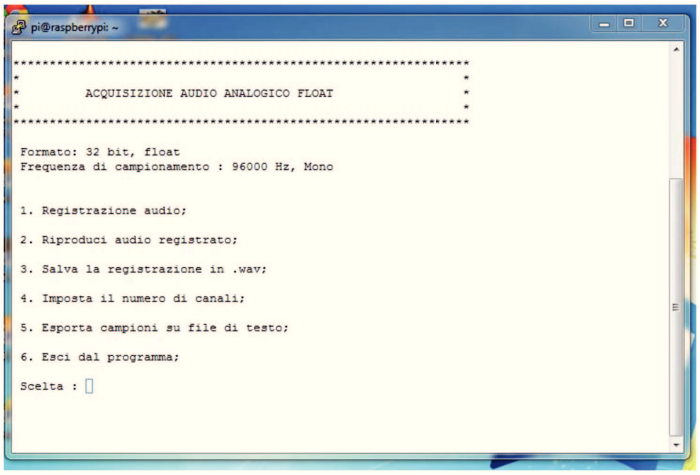

Figura 3.31: Main programma acquisizione audio

3.4.3 Algoritmo Rilevazione Bolle

Lo scopo di questa sezione è stato quello di fare il porting dell'algoritmo EMD, realizzato in Matlab, in una sua versione in linguaggio C, adatta all'esecuzione su scheda embedded Raspberry PI. Le operazioni richieste per l'estrazione della IMF 1 sono state:

- Calcolo dei massimi e dei minimi dal segnale di ingresso x(t);
- Interpolazione dei massimi per ottenere l'inviluppo superiore s(t);
- Interpolazione dei minimi per ottenere l'inviluppo inferiore i(t);
- Calcolo dell'inviluppo medio m(t) = s(t) - i(t) / 2;
- Calcolo della prima componente h1(t) = x(t) - m(t);
- Esecuzione di S operazioni di sifting per ottenere $h_{1k}(t) = h_{1(k-1)}(t)$-$m_{1k}(t)$;
- Calcolo della IMF 1 come $c_1(t) = h_{1k}(t)$;
- Calcolo del residuo r(t) = x(t) - $c_1(t)$.

Per l'estrazione dei massimi e dei minimi si utilizza una variabile chiamata *pendenza*, che mi indica la crescita o decrescita del segnale. Gli steps della funzione realizzata sono:

- Confronto tra valore del campione i-esimo (V_i) e quello del campione i-esimo+1 (V_{i+1});

  - Se $V_i > V_{i+1}$, pendenza = -1, il segnale decresce e mi aspetto di trovare un minimo;
  - Se $V_i < V_{i+1}$, pendenza = +1, il segnale cresce e mi aspetto di trovare un massimo;
  - Se $V_i = V_{i+1}$, pendenza = 0, il segnale ha pendenza nulla, incremento l'indice i finchè non trovo un campione di valore diverso. Le condizioni che si possono verificare sono:
    * Il segnale cresce, poi c'è una zona a pendenza nulla e poi cresce nuovamente: si continua la ricerca del massimo;
    * Il segnale cresce, poi c'è una zona a pendenza nulla e poi decresce: considero la zona a pendenza nulla come massimo;
    * Il segnale decresce, poi c'è una zona a pendenza nulla e poi cresce, considero la zona a pendenza nulla come minimo;
    * Il segnale decresce, poi c'è una zona a pendenza nulla e poi decresce, si continua la ricerca del minimo;

- Itero il confronto per tutti i campioni del segnale.

Per il salvataggio dei massimi e dei minimi sono state create due liste, ognuna contenete il valore del campione, l'indice di comparsa e un puntatore al massimo o minimo successivo.
Al fine di realizzare la funzione di inviluppo superiore ed inferiore, viene utilizzata la funzione pchip, acronimo di *"Piecewise Cubic Hermite Interpolating Polynomial"* [75] [76]. La caratteristica fondamentale di questa funzione è lo *shape-preseriving*, ovvero la possibilità di mantenere la forma della funzione, evitando spigoli o cambi repentini dell'andamento del segnale. Questa funzione, disponibile anche online, può essere generata dal toolbox Matlab di conversione funzioni in linguaggio C. Il risultato della conversione è formato da una serie di file, richiamati da una funzione C che richiede come input le coordinate x e y dei massimi o minimi ed un array di appoggio per calcolare i valori intermedi tra due massimi o minimi. L'uscita invece sarà data da una matrice contenente coppie di coordinate x-y, necessarie per raffigurare l'inviluppo inferiore o superiore. Il calcolo dell'inviluppo medio viene fatto in maniera iterativa, sommando l'elemento i-esimo dell'inviluppo superiore a quello i-esimo dell'inviluppo inferiore e dividendo per due. La prima componente $h_1(\mathrm{t})$ viene calcolata dalla differenza, campione per campione, dell'inviluppo medio al segnale originale x(t). il risultato così ottenuto viene utilizzato per l'iterazione successiva, considerando 5 il numero di operazioni di sifting necessarie per estrarre la prima IMF utile. Dal segnale originale verrà poi sottratta la prima funzione di modalità intrinseca creando il primo residuo. Per il calcolo della soglia si fa sempre riferimento alla formula di Donoho-Johnostone, andando

però a realizzare una sua versione in linguaggio C.
La successiva operazione consiste nell'andare a dividere il segnale in finestre da 10ms che, considerando una frequenza di campionamento di 96KHz, corrispondono a 960 campioni per finestra. Per fare questa operazione è necessario conoscere a priori il numero di finestre ottenibili, mediante divisione della lunghezza del segnale per numero di campioni per finestra. In ogni finestra verrà preso il campione con valore massimo e confrontato con la soglia precedentemente calcolata. Se il valore del campione supera la soglia, nel frame verrà indicata la presenza di una bolla. Una memorizzazione delle bolle, come quella realizzata in Matlab, bolle adiacenti sulla stessa riga, non è possibile ottenerla poichè in C non possiamo cambiare dinamicamente le dimensioni delle matrici. A tal proposito è stato realizzato un nuovo vettore con tanti elementi quante sono le finestre, contenenti strutture dati. Se la finestra i-esima contiene una bolla, la rispettiva struttura dati alla posizione i-esima del nuovo vettore verrà riempita con dei valori non nulli. Per dare una rappresentazione più vicina possibile a quella usata in Matlab, i campi della struttura conterranno il timestamp, l'identificativo di riga e colonna oltre che un puntatore all'elemento bolla successivo. La struttura di memorizzazione è illustrata in figura 3.32. Una volta individuate le bolle, è necessario implementare una tecnica di rileva-

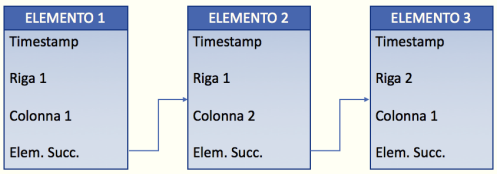

Figura 3.32: Struttura memorizzazione bolle C

zione shower. Lo shower è stato definito come due bolle contenute in finestre indipendenti, separate da una finestra priva di bolle. Gli steps necessari per l'individuazione degli shower sono stati:

- Scorrimento di tutti gli elementi di una riga e memorizzazione del timestamp dell'ultimo elemento (le righe sono indicate dal campo riga nella struct);
- Verifica del successivo elemento puntato che, se NULL, indica la fine della lista;
- Salvataggio del timestamp del primo elemento della riga successiva;
- Calcolo della differenza tra i due timestamp:
  - Se maggiore di 20ms, le due bolle non sono considerate shower e quindi viene iterato il confronto tra le righe successive;

– Se è compresa tra 10 e 20ms, le due bolle sono considerate adiacenti e vengono riposizionate su una stessa riga (cambiando i valori delle variabili riga e colonna della struct). Inoltre viene aggiornato il contatore degli shower.

Al fine di individuare più facilmente quali sono le bolle singole e quali gli shower rilevati è stato definito un nuovo vettore V_t lungo tanto quanto il numero di finestre calcolate. Ogni elemento assumerà valore 1 per finestre con bolla singola, 0 per finestre senza bolla e 2 per gli shower (sia le finestre con bolla che quelle da 10ms che separano le bolle).
L'ultima operazione, da effettuare nell'algoritmo di rilevamento bolle, è assegnare un livello di Spencer e grado di rischio dalla scala di Spencer Estesa, basandosi sul numero di bolle e shower presenti in un file audio. Per ottenere questi due valori, dal vettore finale V_t si va a calcolare il numero di sequenze di 1 e di 2. Le sequenze di 1 indicano bolle su una o più finestre adiacenti, mentre le sequenze di 2 indicano gli shower.

3.4.4 Invio Report con WebRTC

Vista la necessità di avere una prima stima sul grado di rischio del subacqueo in fase di post-emersione e parallelamente avvisare il personale medico di supporto da remoto, è stato implementato un sistema atto ad inviare, al medico remoto, i risultati dell'algoritmo di rilevazione bolle mediante API WebRTC.
La scheda Raspberry PI è stata collegata al modulo Bluetooth HC-05 mostrato in figure 3.33, scheda slave economica che necessita del collegamento dei 4 pin (VCC, GND, TX e RX) sui corrispettivi GPIO della Raspberry. Per utilizzare il dispositivo Bluetooth è stato necessario configurare file di sistema *(/etc/inittb e /boot/cmdline.txt)* al fine di settare parametri come il baud rate, cioè il numero di simboli al secondo che il trasmettitore e ricevitore sono in grado di scambiarsi. Per inviare dati al Bluetooth si utilizza una delle caratteristiche dei sistemi UNIX, ovvero trattare le porte come se fossero dei veri e propri file. Da qui la facilità di aprire il file con l'istruzione *open*, scrivere su di esso (inviare dati con il Bluetooth) con la *write*, leggere dati (ricevuti dal master Bluetooth) con la *read* e chiudere il file con la *close*. Lato ricevitore si è deciso di utilizzare invece dello smartphone un computer portatile, questo perchè almeno una unità è sempre disponibile a bordo delle imbarcazioni, in quanto necessaria per l'upload di profili di immersione, della strumentazione subacquea nella fase di post-emersione. A tale dispositivo è stato collegato un Dongle Bluetooth, come quello utilizzato per testare l'invio dati di ECG mediante il servizio WebRTC. Questa scelta è legata soprattutto all'applicazione Crhome, usata per l'invio dati nel canale DataChannel. Molti smartphone infatti non hanno il browser Crhome e questo limitava l'utilizzo del servizio WebRTC. Altra motivazioni che

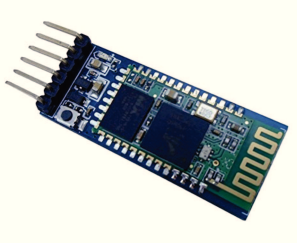

Figura 3.33: Modulo Bluetooth HC-05 *(raspberrypi.org)*

ha comportato l'utilizzo di un singolo computer, piuttosto che più smartphone, è evitare congestione con connessione multiple. Ogni smartphone infatti avrebbe dovuto aprire una connessine peer-to-peer con il medico, rendendo più difficile l'analisi simultanea.
Il software di rilevazione bolle è stato modificato richiedendo un input iniziale, prima dell'acquisizione, di alcuni dati quali nome, cognome, sesso ed età del subacqueo, sul quale si andrà ad effettuare l'acquisizione. I risultati verranno incapsulati ed inviati sul canale DataChannel. Lato client è stata realizzata una variante dell'applicazione Crhome esistente, che non visualizza più il grafico, bensì le informazioni ricevute su apposita textarea. L'interfaccia di comunicazione realizzata è visibile in figura 3.34.

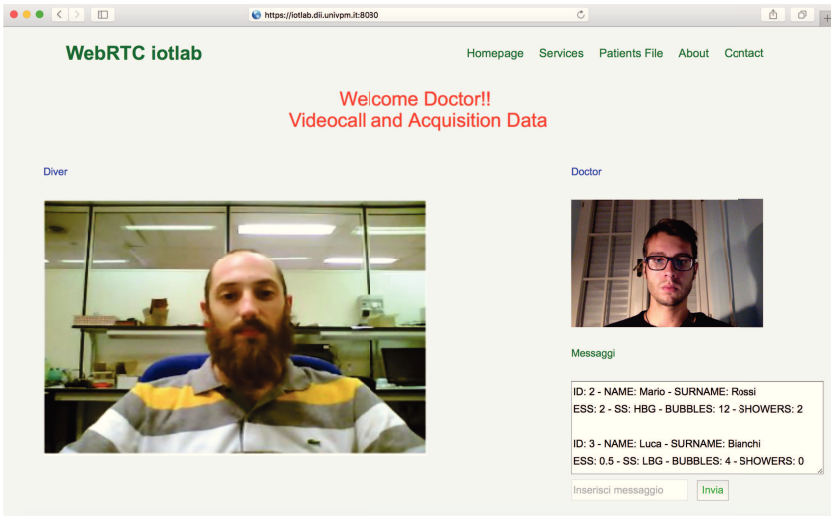

Figura 3.34: Session WebRTC per invio risultati algoritmo rilevamento bolle

3.5 Software Annotazioni Bolle

In letteratura sono presenti un gran numero di algoritmi, implementati con metodi e tecniche differenti, atti a rilevare bolle gassose in segnali eco Doppler audio. Preso in considerazione un file audio campione, per determinare l'efficienza e le performance di un algoritmo sviluppato, è necessario comparare i risultati dello stesso con quelli prodotti da annotazioni manuali da parte di tre Blind team (tre medici esperti indipendenti). Questa comparazione ha lo scopo di verificare se tutte le bolle individuate da un'analisi fonetica, condotta dagli esperti stessi, sono altresì individuate, con lo stesso secondo di comparsa, dall'algoritmo automatico.
Tuttavia il principale problema riscontrato in questa fase è la non corrispondenza dei tre report manuali. Ogni annotazione contiene infatti un differente numero di bolle rilevate, spesso con istanti di comparsa diversi dalle altre annotazioni. Ciò è dovuto principalmente ad una soggettiva interpretazione dell'audio da parte del medico esperto oltre che alla sua possibilità di incorrere in errori accidentali, dovuti a distrazioni nell'annotazione.
È stato cosè sviluppato un software chiamato "Contatore Eventi Bolla" (CEB) che, oltre a velocizzare il processo di annotazione, fornisce una stessa base di riproduzione audio ed una struttura standard ed unificata di tabulazione delle bolle rilevate.

3.5.1 GUI

I moduli principali del software sono:

1. "Settaggi Input Utente", utilizzato per il training dell'esaminatore e per il calcolo del proprio tempo di risposta;
2. "Gestione Input", utilizzato per l'abilitazione dell'input rapido;
3. "Audio", utilizzato per l'ascolto dell'audio e per l'annotazione delle bolle;
4. "Visualizza Txt", utilizzato per la visualizzazione e modifica dei report.

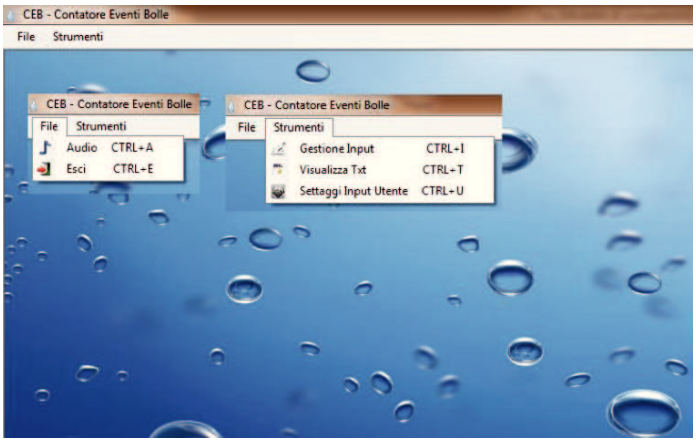

Figura 3.35: Home CEB

Settaggi Input Utente Questo modulo si occupa di verificare quanto rapidamente l'utente esperto, che effettua l'analisi, risponde ad uno stimolo acustico esterno mediante pressione di un tasto. Lo scopo del modulo è di rendere l'analisi più oggettiva possibile, evitando che caratteristiche psico-fisiche del medico influenzino l'istante di comparsa della bolla rilevata. Ciò che si andrà a calcolare è la rapidità di risposta ad uno stimolo esterno, ovvero il tempo che intercorre da quando l'utente percepisce uno stimolo a quando vi risponde compiendo una certa azione motoria. Studi effettuati evidenziano come uno dei metodi più efficaci per valutare la rapidità di risposta a stimoli acustici esterni, è la riproduzione di un suono, compreso nelle frequenze udibili e verificare il tempo che intercorre prima che il soggetto esegua un'azione motoria prefissata (alzare un braccio o premere un tasto) [77]. Nel modulo in esame vengono estratti casualmente 10 suoni campioni tra una popolazione di 100, ognuno di quali è una bolla di dimensioni e durata variabile. Ogni suono sarà riprodotto con un ritardo anch'esso casuale tra 0 e 5 secondi, a cui l'utente deve rispondere con pressione di un apposito tasto preselezionato. Questo esercizio ci permetterà di calcolare il tempo di risposta ad uno stimolo acustico singolo e medio dopo 10 stimoli esterni. Il tempo di risposta così calcolato verrà utilizzato per ottenere il vero istante di comparsa della bolla, senza l'influenza da parte dell'esaminatore che svolge l'analisi. Una funzione non meno importante del tool è la possibilità di caratterizzare l'esperto, ovvero verificare quanto l'orecchio è sensibile a stimoli acustici di diversa frequenza, oltre che allenarlo a riconoscerli in maniera più veloce. Il modulo realizzato è raffigurato nella figura 3.36. Le componenti del modulo sono:

1. Bottone per l'avvio del test;

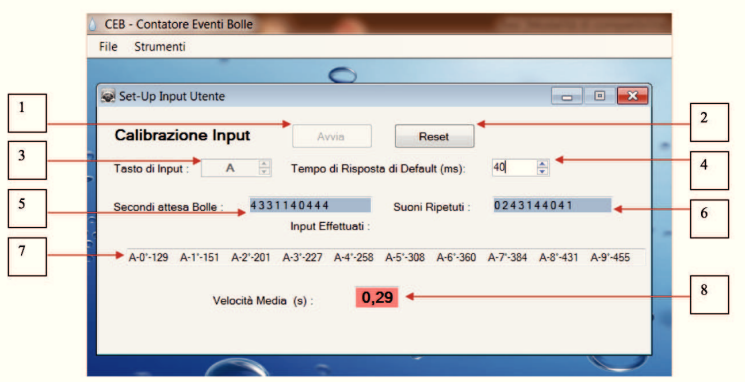

Figura 3.36: Modulo Settaggi Input Utente

2. Bottone per il reset del test;
3. Lista per la scelta del tasto da premere una volta udito il suono;
4. Tempo di risposta di default;
5. Tempi di attesa prima della riproduzione di ogni suono;
6. Indice dell'audio scelto casualmente per la riproduzione;
7. Area di testo con tutti i suoni riprodotti ed i loro tempi di risposta;
8. Rapidità di risposta media calcolata sui 10 suoni.

La rapidità di risposta è un valore variabile da persona e persona, così nel caso l'utente non desideri effettuare il test per il calcolo del suo tempo di risposta, quello di default è impostato a 40ms in accordo con la letteratura [78] [79] [80].

Gestione Input Il modulo "Gestione Input" è la componente del tool che viene utilizzato per agevolare l'attività dell'esperto nel rilevare bolle gassose. Nel modulo "Audio" infatti (discusso di seguito) ogni esperto dovrà prestare attenzione al cursore del mouse, appositamente posizionato sopra il bottone da premere una volta udita la bolla. Questo modulo, che si occupa di intercettare l'input da tastiera, permette al medico di concentrarsi unicamente sull'ascolto (anche ad occhi chiusi) e premere un pulsante da tastiera quando rileva la bolla. La figura 3.37 mostra il modulo realizzato. Le componenti del modulo sono:

1. Scelta del carattere di input usato per indicare la presenza della bolla;
2. Area di testo con gli input effettuati ed i loro timestamp;
3. Contatore degli input effettuati;

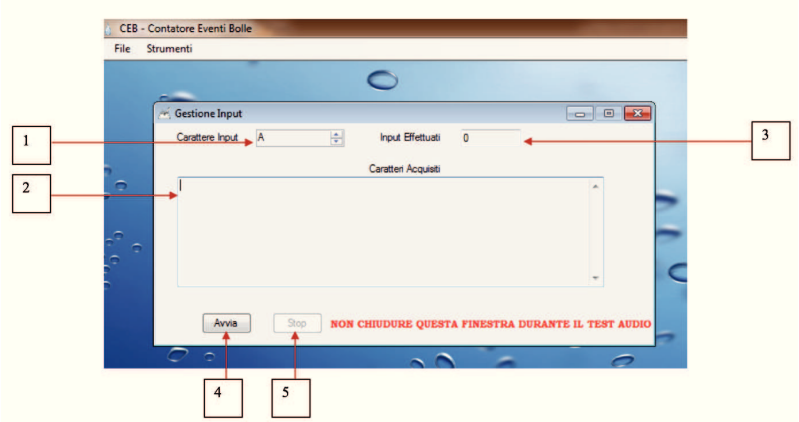

Figura 3.37: Modulo Gestione Input

4. Pulsante avvia lettura buffer tastiera;
5. Pulsante azzeramento modulo.

Audio Il modulo "Audio" è la parte principale di CEB poichè ad esso è affidato il compito di riprodurre un file eco Doppler audio, annotare le bolle gassose su opportuna indicazione del medico e salvare le annotazioni con struttura standard su file di testo. Le azioni necessarie all'analisi di un eco Doppler audio sono:

- Caricamento audio;
- Riproduzione audio;
- Indicazione presenza bolla;
- Salvataggio bolle su file.

La figura 3.38 rappresenta il modulo descritto. Le componenti del modulo sono:

1. Area di testo per la visualizzazione del file corrente riprodotto;
2. Trackbar per la posizione corrente ed eventuale modifica dell'audio riprodotto;
3. Indicatore inizio audio;
4. Indicatore fine audio;
5. Indicatore secondo corrente riprodotto;
6. Barra con bolle individuate;

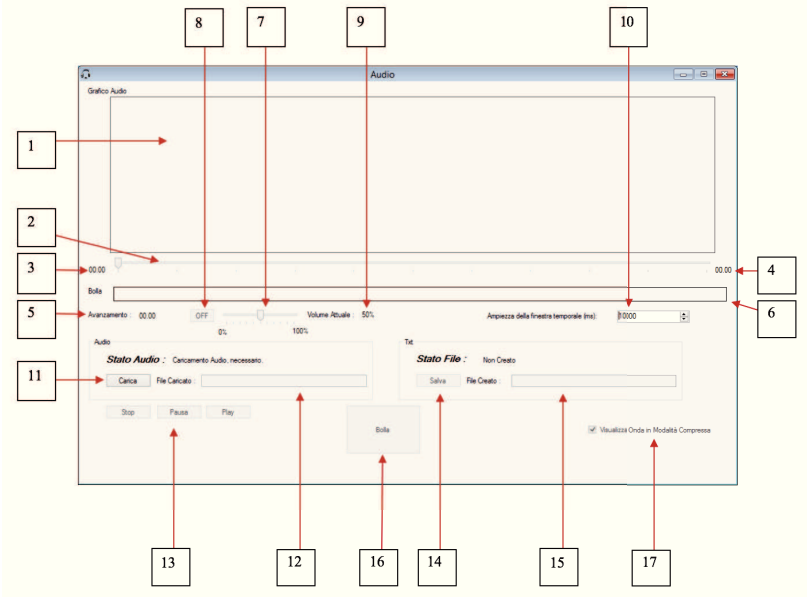

Figura 3.38: Modulo Audio

7. Trackbar per il controllo del volume;
8. Pulsante attivazione/disattivazione volume;
9. Indicatore livello volume;
10. Area modifica ampiezza finestra di visualizzazione;
11. Pulsante caricamento audio;
12. Path audio caricato;
13. Pulsanti stop, play e pausa audio;
14. Pulsante salvataggio bolle rilevate su file;
15. Path file di testo;
16. Pulsante per indicare bolla rilevata;
17. Visualizzazione onda in formato compresso.

La prima azione del modulo è il caricamento dell'audio che consiste, tramite finestra di dialogo, nello scegliere dalle cartelle e sottocartelle del proprio terminale l'audio da analizzare. L'avvenuto caricamento mostrerà un'anteprima di tutto il segnale e abiliterà i controlli necessari per la sua gestione. La figura 3.39 mostra il caricamento di un file audio. Durante la riproduzione dell'audio,

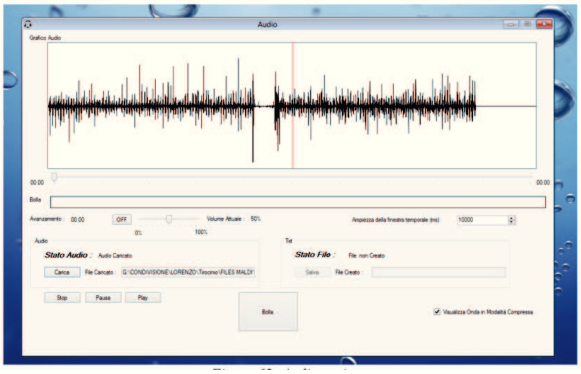

Figura 3.39: Audio caricato

oltre alle normali componenti di gestione, vengono abilitati i controlli per la modifica dell'ampiezza della finestra temporale e per la visualizzazione del file in formato compresso. Ogni audio in formato WAVE contiene un elevato numero di campioni, al fine di rendere una visualizzazione fluida del segnale, viene processato con tecniche di compressione. Spesso quindi la forma d'onda associata può risultare diversa da quella riprodotta da altri player commerciali. La visualizzazione dell'audio in formato compresso o non si può ottenere mediante apposito check del componente 17. La figura 3.40 mostra la riproduzione di un audio in formato compresso e non compresso. Il controllo 10 invece permette

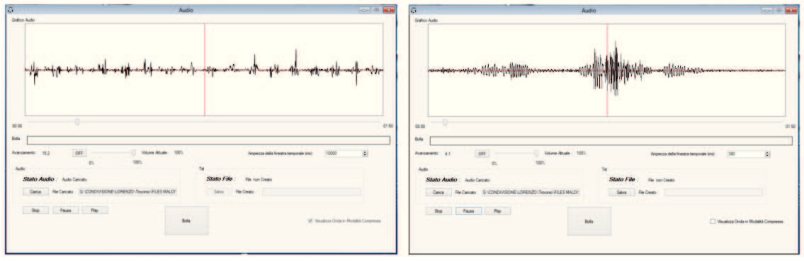

Figura 3.40: Visualizzazione audio in formato compresso e non

di scegliere quanti secondi dell'audio riprodotto devono essere visualizzati nel grafico (di default è 10s). L'utilizzo di questo controllo può agevolare l'esperto a visualizzare la forma d'onda di una bolla rilevata (se la visualizzazione dell'audio è in formato non compresso). La figura 3.41 mostra la riproduzione di un audio con ampiezza di visualizzazione variabile. Ogni bolla rilevata, mediante pressione pulsante o tasto su tastiera (se attivo il modulo "Gestione Input"), verrà visualizzata su schermo da una barra verticale blu in corrispondenza del suo istante di comparsa. All'interno del software, tale istante sarà memorizzato

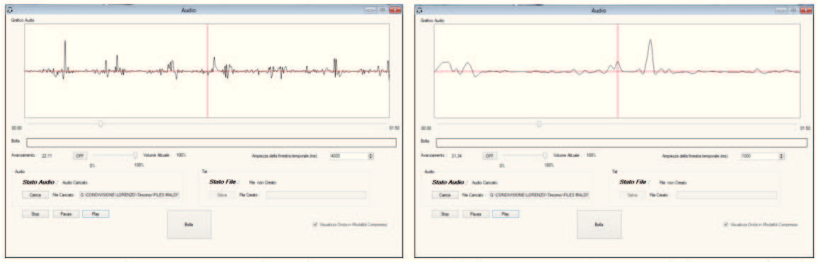

Figura 3.41: Riproduzione con ampiezza di visualizzazione pari a 4 secondi (immagine a sinistra) e 1 secondo (immagine a destra)

partendo dal secondo di rilevazione (secondi e decimi di secondi) meno il tempo di risposta dell'utente calcolato o di default. La figura 3.42 mostra l'identificazione di bolle. Terminata l'analisi dell'audio, tutte le bolle rilevate saranno

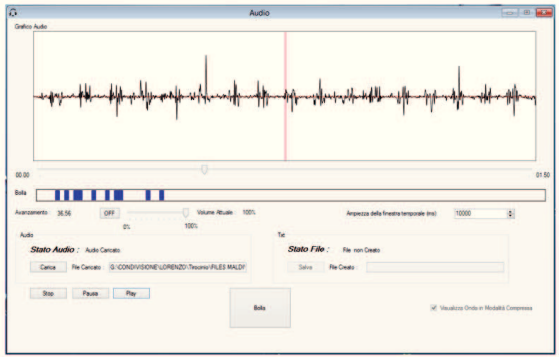

Figura 3.42: Rilevazione Bolle

salvate in un file di testo rispettando una struttura ben definita. Prima del salvataggio vero e proprio, CEB chiede una propria valutazione soggettiva sulla presenza o meno di shower nell'audio corrente. Questo input, memorizzato nel file di testo come apposita stringa, potrà essere usato come ausilio nelle valutazioni future sull'efficacia di metodi automatici di rilevamento bolle.

Visualizza Txt Tutti i report di precedenti analisi potranno essere visualizzati e modificati mediante il modulo "Visualizza Txt". Essi conterranno le bolle tabulate in un certo ordine e con un certo criterio. Le prime righe dei report sono informazioni sulla data di analisi del file e sul numero di bolle annotate, mentre le righe sottostanti contengono le bolle raggruppate per secondo. Tutti gli istanti di rilevazione bolle, salvati in secondi e decimi di secondi, verranno approssimati all'intero più prossimo e le bolle con intero uguale, verranno raggruppate tra loro. I raggruppamenti presenti nel file saranno indicati dalla

coppia *(secondo ; contatore)*. Il *secondo* è l'istante di rivelazione convertito in intero, mentre il *contatore* è un numero intero che è pari ad 1, nel caso di rilevazione di una singola bolla, e maggiore di uno, nel caso di una multi rivelazione per il secondo considerato. *Es. Singola Rivelazione 5;1 Es. Multi Rivelazione 6;2*
Sotto ad ogni coppia *secondo ; contatore*, tutte le bolle sono salvate con tanto di parte decimale. Ad esempio la coppia 2:2 conterrà due bolle, 2.12 e 2,5, che, approssimate all'intero più vicino, vengono raggruppate insieme.
In questo modulo i file di testo potranno essere anche modificati per aggiungere testo o altre informazioni di aiuto all'utente in analisi successive. Il modulo realizzato e i componenti in esso presenti sono illustrati nella figura 3.43. I

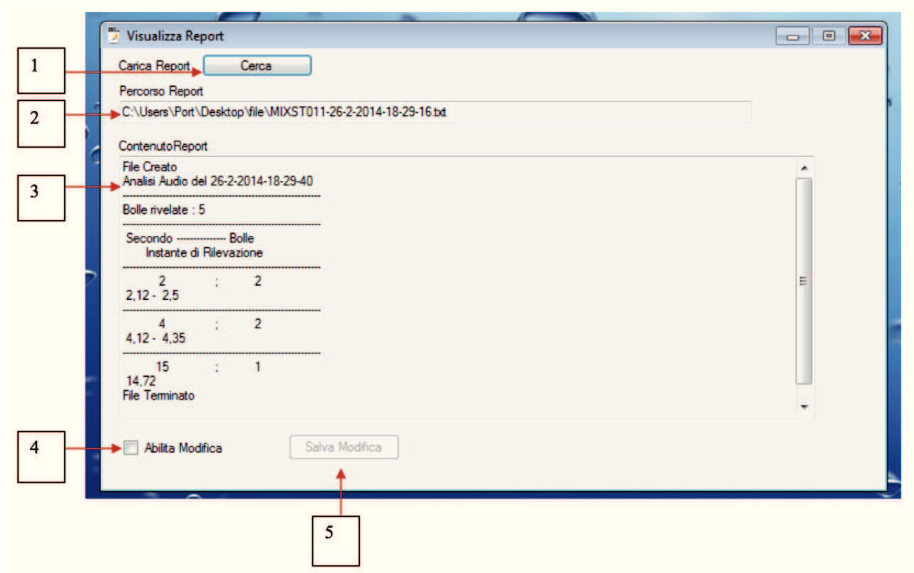

Figura 3.43: Modulo Visualizza Txt

componenti sono:

1. Bottone caricamento report;
2. Path del report caricato;
3. Contenuto report caricato;
4. Check per l'abilitazione delle modifiche sul report caricato;
5. Salvataggio report modificato.

3.5.2 Sviluppo CEB

Il software è stato implementato grazie all'utilizzo della classe "DirectX" [81] [82], un insieme di driver che gestiscono applicazioni ad alte prestazioni, su cui si poggiano i driver specifici delle schede multimediali dei vari costruttori. La figura 3.44 illustra il livello di interconnessione delle librerie DirectX. Le classi

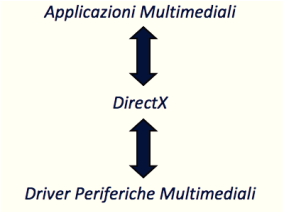

Figura 3.44: Livello funzionamento librerie DirectX

principali della famiglia DirectX sono DirectDraw per la gestione della memoria video, DirectInput per la gestione dei dispositivi di input, DirectPlay per la gestione di comunicazioni di rete, seriali o parallele, DirectAudioVideoPlayback per la riproduzione di audio/video e DirectSound per la riproduzione di suoni ad alta qualità. Il software inoltre è stato progettato in accordo con gli standard ISO/IEC 9126 sulla qualità del software [83] [84].
Il primo modulo realizzato con questa libreria è stato il modulo "Audio", necessario per la riproduzione dell'audio, alla visualizzazione della forma d'onda associata, alla segnalazione della presenza di una bolla e al salvataggio dei report su file di testo. La riproduzione è resa possibile grazie ad istanze delle classi DirectSound, che permettono anche di gestire i controlli associati come le due trackbar. Poichè DirectX non fornisce delle funzioni in grado di visualizzare le forma d'onda di un file WAVE, è stato necessario studiare la struttura del file audio per estrarre solo il blocco dati e visualizzare il contenuto su apposite sezioni del software. Ogni WAVE è composto da una serie di blocchi chiamati *"Chunk"*, l'unico contenente i dati utili è il *"Data-Chunk"*. La struttura completa del file WAVE è descritta nell'appendice A.3. I vari *"Chunk"* differiscono nelle notazioni con cui vengono memorizzati i dati: "LittleEndian" e "BigEndian". Le due notazioni indicano l'ordine in cui vengono memorizzati i byte più significati (MSB) e meno significativi (LSB). Al fine di estrarre tutti i dati in maniera omogenea è necessario convertire i blocchi ed usare una solo notazione, BigEndian. Questo è stato reso possibile mediante la realizzazione dell'apposita classe "EstraiChunkDaWave". Le notazioni LittleEndian e BigEndian sono descritte nell'appendice A.4 mentre il funzionamento della classe è descritto in figura 3.45. Per una gestione più modulare del software è stata creata una seconda classe "GestioneDataWave" con il compito di manipolare il

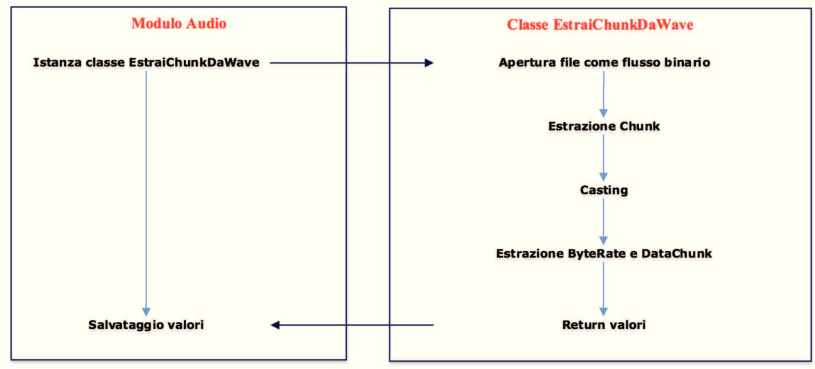

Figura 3.45: Steps estrazione Chunk

flusso dati. La classe è formata da tre metodi principali:

- *"New"*, invocato al momento dell'istanza della classe ed utilizzato per il calcolo dei campioni con la massima e minima ampiezza, necessari per settare l'asse nel grafico. Inoltre si occupa di operazioni ausiliarie come impostare il colore dello sfondo;

- *"Comprimi"*, è il metodo che si occupa di comprimere il flusso dati del file audio. Ogni audio WAVE è dotato di un numero elevato di campioni, necessari per garantire un'elevata qualità di riproduzione. La rappresentazione grafica di questa elevata mole di dati comporta costi computazionali onerosi. Questo metodo si occupa di duplicare il flusso dati, andando ad agire solo su quello dedicato alla visualizzazione della forma d'onda. Questa operazione fa rimanere inalterato il flusso utilizzato per l'ascolto dell'audio, garantendo sempre una riproduzione ad elevata qualità. Per realizzare la compressione, cercando di mantenere una forma d'onda fedele a quella originale, viene estratto un campione ogni k campioni. Considerando un byte rate pari a metà rispetto a quello originale ed una finestra di durata 10millisecondi, il numero di campioni k sarà calcolato dalla formula 3.8.

$$k = RefreshTime * (ByteRate/2) \tag{3.8}$$

  A sua volta il flusso dati dedicato alla visualizzazione sarà reduplicato in due sotto flussi, uno contenente i dati originali non compressi ed un secondo contenente i dati compressi. Tutto ciò al fine di dare la possibilità all'utente di cambiare durante l'analisi dell'audio il tipo di visualizzazione, flusso compresso o non compresso;

- *"GeneraFormaOnda"*, è il metodo che si occupa di creare l'immagine bitmap con i campioni WAVE ricevuti in ingresso e fornirla in ritorno al modulo principale. L'immagine sarà rappresentata da punti di coordinate X-Y in un sistema di riferimento 2D, che mediante operazione di interpolazione, uniti da una linea solida continua. La figura 3.46 descrive il funzionamento della classe.

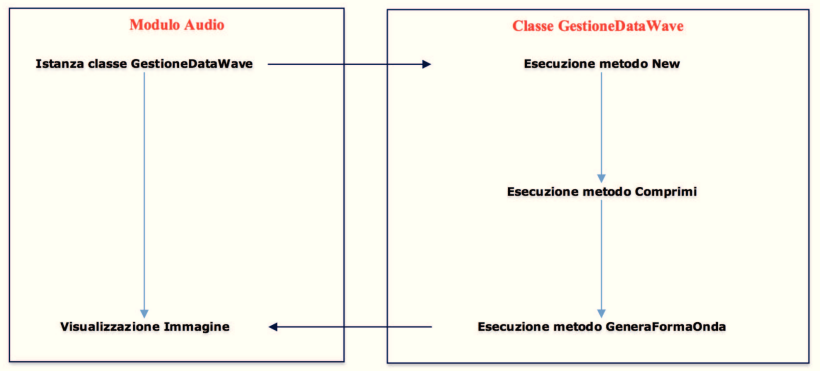

Figura 3.46: Steps generazione Bitmap

Come precedentemente detto, indicare la presenza di una bolla può avvenire o mediante click dell'apposito bottone o mediante pressione del tasto su tastiera. In questo secondo metodo uno dei problemi principali che si può verificare è la multi-rilevazione di bolle. Il tool infatti legge il buffer della tastiera periodicamente e, quando rileva la pressione del tasto, lo segnala come bolla. Se la lettura del buffer viene fatta con una frequenza elevata, ad esempio 1KHz e l'utente ha una pressione del tasto prolungata, variabile da persona a persona, una singola bolla può essere identificata come sequenza di bolle. Per evitare questo problema è stata realizzata una preliminare fase di accorpamento, dove tutte le bolle, che distano un istante di comparsa minore di 0.05 secondi, sono accorpate insieme. La scelta di 0.05 nasce da una valutazione della durata della bolla, che risulta essere molto più grande di 0.05.

3.6 Risultati

In questo ultimo paragrafo verranno presentati i risultati degli algoritmi di rilevazione bolle gassose in eco Doppler audio file, sia offline basati su tecniche EMD e PSD sia real-time basati su EMD. Inoltre verranno presentati i risultati sull'utilizzo del software annotazioni CEB.

3.6.1 Algoritmo offline EMD

Tutti gli algoritmi sono stati testati sul campione composto da 240 file eco Doppler audio acquisiti dal DAN, in rispetto del protocollo concordato, durante le loro campagne di acquisizione alle Maldive e all'isola d'Elba. La tabella 3.1 mostra la comparazione per uno stesso audio del livello di Spencer, prodotto da un'analisi fonetica condotta da medici iperbarici e quello ottenuto dall'algoritmo EMD. Vengono riportati i risultati di solo 24 audio sul totale di 240 elaborati. Da un'analisi dei risultati della tabella 3.1 si evidenzia come non ci

Tabella 3.1: Comparazione risultati annotazioni manuali DAN e algoritmo EMD

	Spencer Level	
$N°$ **File**	**Valutazione manuale**	**Output algoritmo EMD**
1	0	0.5
2	0	0
3	0	0
4	0	0.5
5	0.5	0
6	0.5	0
7	0.5	0.5
8	0.5	0.5
9	1	1
10	1	1
11	1	1.5
12	1	1
13	1.5	1.5
14	1.5	2
15	1.5	2
16	1.5	1
17	2	2.5
18	2	2
19	2	2
20	2	2
21	2.5	2.5
22	2.5	2.5
23	3.5	3.5
24	4	4

sia una puntuale corrispondenza per tutti i file, cosa che invece viene raggiunta se consideriamo una soglia di incertezza pari a 0.5 gradi. Da un'analisi congiunta con i medici iperbarici è stato poi verificato che 0.5 può essere un margine di errore reale nell'annotazione manuale di file audio. Da qui considerando una

soglia di errore di 0.5, la corrispondenza nel livello di Spencer è illustrata in tabella 3.2.

Tabella 3.2: Corrispondenza livelli di Spencer con l'uso della soglia

Livello di Spencer	**Totale Corrispondenza (N° di File)**	**Totale Corrispondenza con Soglia (N° di File)**	**Non Corrispondenza (N° di File)**
0	14	3	1
0.5	10	2	0
1	10	1	0
1.5	11	3	1
2	9	2	3
2.5	12	2	2
3	1	2	1
3.5	5	2	1
4	8	4	0

Utilizzando le annotazioni fornite dal DAN, che includono, oltre al livello di Spencer, anche l'istante di comparsa della bolla, sono state calcolate specificità e sensitività. La specificità è definita come rapporto tra veri negativi (bolle non rilevate dal DAN e dall'algoritmo EMD) più falsi positivi (bolle non rilevate dal DAN ma rilevate dall'algoritmo) e veri negativi. La sensitività è stata invece calcolata come rapporto tra veri positivi (bolle rilevate sia dal DAN che dall'algoritmo EMD) e la somma di veri positivi e falsi negativi (bolle rilevate dal DAN, ma non dall'algoritmo EMD). La specificità ottenuta è stata del 22%, mentre la sensitività dell'80%.

3.6.2 Algoritmo offline PSD

L'algoritmo PSD, basato sull'idea che zone con bolla hanno un'energia spettrale maggiore di zone senza bolla, è stato realizzato per confermare la presenza delle bolle rilevate con l'algoritmo EMD. Come parametri utilizzati si è scelto il segnale IMF 1 come base di valutazione, mentre la lunghezza delle finestre è stata posta a 20ms. La scelta di questi valori rappresenta un buon compromesso tra veri positivi, ovvero bolle presenti e rilevate dall'algoritmo, e falsi positivi, ovvero bolle non presenti, ma rilevate dall'algoritmo. Per tutti i file testati, 240 audio forniti dal DAN, tutte le finestre che l'algoritmo EMD indica come contenenti bolla, effettivamente hanno una densità spettrale di valutazione che supera quella di riferimento.
L'algoritmo PSD riesce ad identificare più finestre, oltre alle finestre corrispondenti con l'algoritmo EMD, la cui densità spettrale supera quelle di riferimento.

Da un'analisi fonetica condotta da blind team indipendenti queste zone possono essere classificate come finestre con bolle.
Calcolate le bolle con algoritmo PSD e relativo livello di Spencer, la tabella 3.3 mostra la corrispondenza tra livello di Spencer calcolato con l'algoritmo PSD e quello ottenuto da un'analisi fonetica condotta dal team del DAN. Da qui le

Tabella 3.3: Comparazione risultati annotazioni manuali DAN e algoritmo PSD

	Spencer Level	
N° **File**	**Valutazione manuale**	**Output algoritmo PSD**
1	0.5	0.5
2	0	0
3	0	0
4	0	0.5
5	0.5	0.5
6	0.5	0.5
7	0.5	0.5
8	0.5	1
9	1	1
10	1	1
11	1	1.5
12	1	1
13	1.5	1.5
14	1.5	1.5
15	1.5	2
16	1.5	1.5
17	2	2.5
18	2	2
19	2	2
20	2	2
21	2.5	2.5
22	2.5	2.5
23	3.5	3.5
24	4	4

nuove percentuali di corrispondenza tra i livelli di Spencer, ottenuti da annotazioni manuali e con l'algoritmo PSD, con e senza soglia di 0.5, sono 87,2% senza soglia e 92,4% con soglia.

3.6.3 Algoritmo EMD su scheda embedded

Il porting dell'algoritmo su scheda embedded Raspberry PI permette di analizzare l'audio appena acquisito, senza ricorrere all'utilizzo di dispositivi ausiliari, per la registrazione dell'audio e computer per l'analisi con strumenti di

calcolo quali Matlab. La figura 3.47 mostra l'output dell'algoritmo eseguito da terminale remoto. La tabella 3.4 mostra la comparazione tra i risultati ottenuti

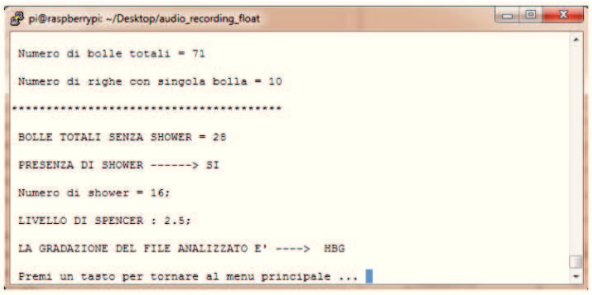

Figura 3.47: Risultati algoritmo di rilevazione bolle su scheda embedded

(10 dei 240 file testati) con l'algoritmo in Matlab e quello in C in termini di bolle e shower rilevati. Da un confronto su tutti i 240 file testati, il 95% di

Tabella 3.4: Risultati algoritmo EMD Matlab e algoritmo EMD C

	Algoritmo Matlab			**Algoritmo C**		
N° **File**	*N°* **Bolle**	*N°* **Shower**	**SS**	*N°* **Bolle**	*N°* **Shower**	**SS**
1	60	3	HBG	60	4	HBG
2	6	0	LBG	8	0	LBG
3	102	6	HBG	98	8	HBG
4	4	2	LBG	4	1	LBG
5	32	4	LBG	36	4	LBG
6	4	0	LBG	5	0	LBG
7	0	0	0	0	0	0
8	1	0	LBG	1	0	LBG
9	145	26	HBG	136	26	HBG
10	2	0	LBG	2	0	LBG

essi ha una corrispondenza completa tra il numero di bolle e shower rilevati oltre al relativo livello di Spencer. I file non corrispondenti sono dovuti a errori sui confronti a soglia, dove differenze sulle cifre significative portano a diverse indicazioni sulla presenza o meno della bolla.
L'algoritmo su scheda embedded impiega 300 secondi, cioè circa 5 minuti nel fornire un risultato utile. Questo è dovuto essenzialmente alle mole di dati che è necessario processare visto che un file audio in formato WAVE occupa circa 40MB. L'80% del tempo è legato al calcolo della EMD che basa il suo funzionamento sulla funzione pchip convertita con il toolbox Matlab. In questo caso, questa conversione crea diversi file con funzioni accessorie che rallentano notevolmente l'esecuzione dell'algoritmo.

3.6.4 Software CEB

Al fine di valutare le prestazioni del tool, sono stati condotti due tipi di test, test delle prestazioni e test sulla sua usabilità [85]. Per valutare le performance di CEB, tre blind team (medici del DAN) sono stati scelti per annotare uno stesso audio più volte, ognuna con un diverso player: CEB, Windows Media Player, QuickTime e Audacity. Tra due annotazioni successive dello stesso audio è intercorso un tempo di 7 giorni al fine di evitare che un esperto ricordasse istanti di comparsa delle bolle. Tutti gli esperti hanno annotato i 240 file a disposizione, ma per ognuno di essi l'ordine di annotazione dei file è stato scelto in maniera casuale. Questo tool non è stato sviluppato con l'intento di confrontare le performance dei vari player, bensì per dimostrare che ognuno di essi introduce variabilità nella percezione delle bolle. Le tabelle 3.5, 3.6 e 3.7 mostrano le bolle rilevate da ogni esperto, utilizzando player differenti.

Tabella 3.5: Comparazione Player Blind Team 1

N° **File**	N° **Bolle (WMP)**	N° **Bolle (QT)**	N° **Bolle (Audacity)**	N° **Bolle (CEB)**
1	15	14	12	15
2	12	12	11	12
3	7	8	8	7
4	7	9	8	9
5	17	18	16	18
6	24	22	21	24
7	6	6	7	6
8	4	4	4	4
9	13	15	13	14
10	31	26	27	28

Tabella 3.6: Comparazione Player Blind Team 2

N° **File**	N° **Bolle (WMP)**	N° **Bolle (QT)**	N° **Bolle (Audacity)**	N° **Bolle (CEB)**
1	16	12	10	15
2	11	12	10	12
3	6	8	8	7
4	8	9	8	9
5	18	18	16	18
6	24	20	20	23
7	6	4	8	6
8	5	4	5	4
9	13	15	12	14
10	30	28	26	28

Tabella 3.7: Comparazione Player Blind Team 3

N° **File**	N° **Bolle (WMP)**	N° **Bolle (QT)**	N° **Bolle (Audacity)**	N° **Bolle (CEB)**
1	16	14	12	15
2	12	11	11	12
3	6	6	8	7
4	7	7	8	9
5	16	18	16	18
6	24	21	22	24
7	6	5	7	6
8	4	4	5	4
9	14	15	14	14
10	31	28	27	28

I risultati sui 240 file testati hanno dimostrato il 62% di convergenza sul numero di bolle rilevate mediante l'uso di CEB. Del 38% delle annotazioni non corrispondenti, il 35% ha una differenza sul numero di bolle minore dell'1% mentre l'altro 3% compreso tra 1% e 1,7%.
Il test di usabilità si è svolto ponendo ad ogni utente 15 domande circa la sua efficienza e facilità di utilizzo. Il 96% degli utenti ha fornito un punteggio medio di 4/5.

3.7 Conclusioni

In questo capitolo è stato realizzato un sistema in grado di rilevare bolle gassose in segnali eco Doppler audio. Il suo impiego non è legato unicamente al settore subacqueo, bensì trova ampia collocazione anche in ambito sanitario. Gli emboli sono particelle gassose che si muovono nel flusso sanguigno, in grado di emettere un suono caratteristico facile da distinguere dalle normali particelle ematiche. La loro presenza è un fondamentale indicatore del rischio di comparsa di malattie cardio vascolari. In ambito sanitario vengono spesso monitorizzati i pazienti sottoposti ad interventi di chirurgia piuttosto che quelli soggetti a trombosi, dove la concentrazione di emboli è notevole. Benchè in ambito clinico la loro natura e pericolosità è nota, risulta invece difficile andare ad individuarli all'interno di uno spettro Doppler. Lo studio condotto ha permesso la realizzazione di un algoritmo in grado di rilevare bolle gassose in segnali eco Doppler audio. In letteratura sono molteplici gli approcci simili, ma tutti scelgono come sito di monitoraggio la zona transcranica o succlavia, dove non tutte le bolle vengono rilevate poichè intrappolate dall'effetto filtrante dei polmoni. L'innovatività dell'algoritmo proposto è in primo luogo la scelta della regione precordiale come sito di monitoraggio. Nonostante in questa zona ci

siano elevate componenti di rumore, l'audio acquisito contiene tutte le bolle circolanti. L'algoritmo è stato anche sviluppato su scheda embedded al fine di fornire al DAN, ente europeo promotore della sicurezza subacquea, un primo strumento di ausilio e stima del livello di Spencer, scala utilizzata per associare la terapia al subacqueo in relazione al numero di bolle presenti. La valutazione delle performance degli algoritmi automatici è basata sul confronto dei risultati prodotti con quelli provenienti da annotazioni manuali. Una comparazione di più risultati manuali, prodotti dai più esperti, ha portato spesso risultati non corrispondenti sia in numero di bolle che nel loro istante di comparsa. Questo ha reso impossibile valutare l'efficienza sia dell'algoritmo proposto che di quelli presenti in letteratura. Da qui è stato sviluppato CEB, un software in grado di facilitare l'annotazione di eco Doppler audio mediante l'utilizzo di librerie, che garantiscono una riproduzione dell'audio uniforme, indipendente dal player e dall'hardware utilizzato. Esso inoltre permette di calcolare il tempo di risposta dell'esaminatore a stimoli acustici esterni, identificando in maniera oggettiva l'istante preciso di comparsa della bolla. Le bolle rilevate da un'annotazione manuale con CEB saranno poi salvate in una struttura standard e ben definita, facile da interpretare sia da personale medico che da algoritmi automatici. Altro vantaggio del software CEB è il suo modulo di training capace di allenare l'orecchio di medici, e non, nel riconoscere tempestivamente bolle gassose.

Capitolo 4

Sviluppo di algoritmi per la diagnosi dei sintomi del Parkinson

In questo capitolo verranno presentati strumenti in grado di rilevare e classificare i principali sintomi della malattia di Parkinson come tremore, freezing e fluttuazioni. Il progetto svolto nasce dalla collaborazione con la casa di cura Villa dei Pini di Civitanova Marche (MC) e il reparto di Neurologia degli Ospedale Riuniti Marche Nord di Pesaro (PU). Il lavoro si pone come base di un monitoraggio continuo (h24) di soggetti affetti dalla patologia ed è utilizzabile sia per avvalorare la diagnosi medica che per ottenere una personalizzazione e verifica dell'efficacia della terapia stessa. I dati raccolti da dispositivi inerziali indossabili, poco ingombranti e ad elevate potenzialità, sono stati processati con tecniche di data-fusion in grado di fornire report sull'evoluzione della malattia in apposite interfacce grafiche. Il sistema WebRTC, proposto nel capitolo 2, è stato utilizzato come strumento di file sharing per trasferire sia real-time che off-line i dati acquisiti dai sensori inerziali al medico remoto. Nel caso di trasmissione real-time saranno inviati i singoli campioni acquisiti dal dispositivo, mentre nel caso off-line verrà trasmesso il singolo file dell'acquisizione svolta.

4.1 Il Parkinson

4.1.1 Natura del Parkinson

La malattia di Parkinson (MP) è un disturbo neurodegenerativo la cui causa è ancora ignota, anche se vengono spesso attribuite le cause a fattori genetici o ambientali. Essa è associata alla diminuzione dei livelli di dopamina, dovuta alla distruzione dei neuroni pigmentati nella substantia nigra (o sostanza nera), uno dei nuclei che costituiscono i gangli della base del cervello. In condizioni normali i neuroni pigmentati liberano dopamina nel corpo striato, che andrà ad inibire l'acetilcolina prodotta dai neuroni colinergici. Nel corpo striato infatti si realizza la connessione tra corteccia motoria e midollo spinale, grazie all'a-

cetilcolina, che determina il segnale che dovrà ricevere il muscolo. Al contrario la dopamina viene prodotta dai neuroni GABA inibitori, che si localizzano dal corpo striato alla sostanza nera. La distruzione dei neuroni dopaminergici si traduce in una minor produzione di dopamina, con conseguente maggior produzione di acetilcolina e perdita di equilibrio tra i neurotrasmettitori, che causa un deterioramento delle attività motorie.
Il Parkinson è una malattia con un decorso lento e progressivo, che porta alla compromissione totale del movimento. Da una stima mondiale si pensa che circa 10 milioni di persone anziane sono affette dalla malattia che attualmente è la seconda più diffusa in tutto il mondo [86]. Da alcuni studi effettuati [87] risulta che in Italia i malati di Parkinson siano più di 200.000. L'età media di insorgenza della malattia è 55 anni ed il tasso di incidenza aumenta con l'età. In merito al genere il tasso di incidenza negli uomini è il doppio rispetto alle donne. Quest'ultime hanno un ritardo nell'età di comparsa dovuto alla presenza di estrogeni, ormoni femminili che esercitano una funzione protettiva dall'insorgenza e progressione della malattia. Tuttavia le donne sono più soggette ad effetti indesiderati del farmaco a causa del minor peso corporeo, delle massa grassa e difficoltà di assorbimento gastrico dei farmaci. Lo studio dello statunitense Goldam et al [88] afferma che l'esposizione ad alcuni solventi ed idrocarburi aumenta il rischio di comparsa della malattia. In particolare l'esposizione al tricloroetilene, al percloroetilene e al tetracloruro comporta un rischio di comparsa di 6, 10.5 e 2.3 volte maggiore rispetto alla non esposizione. Tali solventi sono spesso presenti in molte sostanze usate nella vita quotidiana come benzina e vernici. Lo studio di Willis et al [89] afferma che le donne affette da MP hanno un tasso di mortalità inferiore rispetto agli uomini soprattutto se ispaniche o asiatiche, mentre risulta elevato nei soggetti di razza caucasica. L'incidenza tenderà poi ad aumentare in soggetti con altre disabilità, come la demenza, e che vivono in zone industrializzate con forte emissione di metalli.

4.1.2 Disturbi motori

I principali disturbi motori della malattia sono:

- *Bradicinesia*, sintomo più comune che colpisce circa l'80% delle persone affette da MP. Esso à caratterizzato da una lentezza nei movimenti con conseguente riduzione dei movimenti stessi (ipocinesia). Tipicamente le persone affette da ipocinesia hanno marcata riduzione delle espressioni facciali, riduzione della rotazione del tronco, passi corti, minore oscillazioni delle braccia e spesso un lato risulta essere più colpito dell'altro.

- *Tremore*, è il secondo sintomo più diffuso poichè colpisce il 70% delle persone affette da MP [90]. Spesso è il primo sintomo evidente, segno

dell'insorgenza del Parkinson. È un movimento oscillatorio e ritmico di uno o più segmenti corporei, dovuto all'attivazione alternata di gruppi muscolari in antagonismo tra loro. Il tremore può essere di tipo fisiologico o patologico. Il fisiologico è caratterizzato da un'intensità molto leggera e da una frequenza compresa tra 8-12Hz. Il tremore patologico si divide in tremore a riposo, tremore posturale e tremore cinetico. Il tremore a riposo è caratteristico della mano e meno frequente su piedi e volto. Si evidenzia maggiormente in situazioni di stress emozionale, mentre scompare del tutto o quasi durante il sonno o azioni volontarie. Ha una frequenza caratteristica nel range 3-6Hz. Il tremore posturale si verifica quando si mantiene un arto in una posizione fissa ed opposta alla forza di gravità. Ha una frequenza caratteristica nel range 6-9Hz. Il tremore cinetico si verifica quando il soggetto compie azioni volontarie con movimento degli arti ed ha una frequenza caratteristica nel range 9-12Hz.

- *Rigidità muscolare*, è causata da una riduzione del movimento della parte del corpo interessata a causa di una eccessiva contrazione dei muscoli. La rigidità colpisce prima i muscoli del collo e delle spalle per poi andare al viso e agli arti.

- *Instabilità posturale*, è il sintomo più invalidante della malattia poichè è legata al malfunzionamento dei riflessi posturali. Spesso compare nelle fasi finali della malattia, dove i pazienti, nonostante posture caratterizzata da una flessione in avanti, sono soggetti a frequenti cadute all'indietro. L'instabilità posturale causa difficoltà nel movimento ed incapacità di vivere autonomamente in ambiente domestico. La ricerca di Koller sostiene che circa il 40% degli affetti dalla malattia può andare incontro anche a cadute con frequenza settimanale [91].

- *Freezing*, è l'incapacità improvvisa dei pazienti di compiere un passo e di staccare i piedi da terra. I soggetti durante queste manifestazioni sono coscienti di non riuscire a staccare i piedi ed avanzare. Il blocco può durare da qualche istante a decine di secondi e ripetersi anche più volte al giorno. Il movimento viene ripreso grazie a stimoli visivi o acustici esterni piuttosto che con ad un cambiamento della modalità di avanzamento.

- *Festinazione*, è un progressivo ed involontario aumento della velocità di marcia spesso associato ad una riduzione dell'ampiezza dei passi. Essa può esser causa di cadute del paziente[92].

- *Shuffling*, è una camminata caratterizzata da brevissimi passi con piedi striscianti al suolo.

I disturbi sopra elencati sono spesso motivo di una riduzione della qualità della vita del soggetto affetto da MP poichè si hanno difficoltà evidenti nello svolgere

attività di vita quotidiana come camminare, cambiare posizione e cambiarsi. Questi disturbi portano spesso ad un cambio di postura caratterizzata dalla schiena flessa avanti chiamata "stooped posture" come figura 4.1.

Figura 4.1: Stooped Posture *(wikipedia.com)*

4.1.3 Disturbi non motori

I disturbi non motori causati dalla malattia di Parkinson si possono distinguere in:

- *Disturbi neuropsichiatrici*, sono disturbi del linguaggio, della cognizione, dell'umore, del comportamento e del pensiero. I disturbi più frequenti sono la depressione, la psicosi, l'ansia e gli attacchi di panico. Nel 40% dei pazienti compare almeno un episodio di depressione legato spesso ai sensi di colpa, impotenza e tristezza. I sintomi legati alla depressione vengono validati attraverso tre scale, la "Beck Depression Inventory", la "Hamilton Depression Rating Scale" e la "Geriatric Depression Rating Scale".

- *Disturbi del sonno*, è il disturbo non motorio più diffuso negli affetti dalla MP. È caratterizzato da insonnie, acinesie notturne, sogni vividi e sonnolenza nelle ore diurne.

- *Disautonomie*, sono disturbi legati agli apparati gastro-intestinali e genito-urinari.

- *Disturbi sensitivi*, ovvero sensazioni di bruciore, formicolio o riduzione della sensibilità all'olfatto.

- *Disturbi generali*, disturbi non legati al movimento ma spesso alla posizione scorretta assunta.

4.1.4 Terapia farmacologica

La terapia farmacologica ha lo scopo di ripristinare la dopamina nei gangli, al fine di riportare l'equilibrio dei segnali neuronali. I farmaci attualmente disponibili permettono un miglioramento dei sintomi, ma non l'arresto della distruzione neuronale. Il principio attivo più utilizzato è la "Levodopa" (L-DOPA) che dallo studio di Stowe [93] risulta essere il più efficace. Lo scopo della Levodopa è di riattivare i recettori dopaminergici, con riduzione immediata della maggior parte dei sintomi, ad eccezione del tremore che si riduce dopo un'assunzione prolungata del farmaco. La copertura di una dose di levodopa va dalle 4 alle 6 ore e si riduce a seguito dell'utilizzo prolungato del farmaco. Generalmente la dose inizia a fare effetto dopo i 70-90 minuti dall'assunzione del farmaco [94]. Oltre alla levodopa esistono altri farmaci definiti gli agonisti della dopamina, come gli inibitori delle catecol-O-metiltransferasi o delle monoaminossidasi di tipo B [95]. Gli agonisti non agiscono come la levodopa, ma cercano di stimolare i neuroni a reagire. Gli inibitori catecol-O-metiltransferasi cercano di prolungare l'effetto della dopamina bloccando l'enzima che la distrugge, mentre i monoaminossidasi cercano di prevenire la disgregazione della dopamina naturale e di quella assunta. Questi farmaci hanno un elevato numero di effetti collaterali come allucinazioni, sonnolenza e ipotensione.

4.1.5 Effetti indesiderati della terapia farmacologica

I sintomi del Parkinson possono essere controllati dalla terapia farmacologica nella fase iniziale della malattia stessa. In questa fase infatti, il soggetto è in grado di svolgere le normali attività in maniera del tutto autonoma. Con il progredire della malattia le singole dosi di levodopa non sono più sufficienti ed è quindi necessario un aumento del dosaggio o della frequenza di assunzione. Il trattamento prolungato del farmaco porta ad effetti indesiderati non trascurabili come le *fluttuazioni motorie*, le *discinesie* e le *fluttuazioni non motorie*. Le fluttuazioni motorie sono delle alternanze di fasi "ON" e "OFF" durante l'arco della giornata. Nella fase ON il paziente è in grado di muoversi e camminare speditamente senza alcun segno della malattia, le fasi OFF invece sono momenti in cui i sintomi come tremore, rigidità e bradicinesia, compaiono in maniera piiù o meno grave, rendendo impossibile l'autonomia del soggetto. Le fluttuazioni motorie si possono classificare a loro volta in fluttuazioni prevedibili e non prevedibili [91] [93] [96]. Le fluttuazioni motorie prevedibili, spesso indicate con il termine "wearing-off" o di fine dose, sono fluttuazioni legate alle somministrazioni del farmaco. Il paziente si accorge che l'effetto del farmaco sta per finire dalla ricomparsa del tremore, freezing, festinazione, tachifemia (eloquio incomprensibile) e lentezza dei movimenti. Non appena si riassumerà la dose successiva questi sintomi andranno scomparendo. L'identificazione

delle fluttuazioni avviene attraverso visite neurologiche e questionari specifici come l'UPDRS ("Unified Parkinson's Disease Rating Scale") per il tremore, festinazione e tachifemia [97] e il FOGQ ("Freezing of Gait Questionnaire") per il freezing [98]. Le fluttuazioni motorie non prevedibili, spesso definite come "on-off" o "off-improvvisi", non hanno nessun legame con l'assunzione del farmaco, bensì si verificano in soggetti con malattia in uno stadio avanzato, che assumono regolarmente più farmaci antiparkinsoniani. Questo tipo di fluttuazioni si verifica generalmente nel primo pomeriggio, a seguito del pasto, che rallenta l'assorbimento di levodopa, consentendo al malato di rimanere in "off" per diverse ore [98]. Anche in questo caso le fluttuazioni motorie imprevedibili vengono valutate con la scala UPDRS.
Le discinesie sono dei movimenti involontari "molli" che disturbano i classici movimenti volontari. In genere si manifestano con dei movimenti involontari della lingua, mandibola o con torsioni involontarie delle braccia e delle mani, con dita che assumono posizioni improvvise. Si manifestano principalmente durante la fase di "inizio dose", ovvero fase in cui c'è la massima concentrazione di levodopa. La levodopa non causa discinesie nei pazienti normali, bensì in quelli parkinsoniani, poichè le cellule dello striato di quest'ultimi non ricevono più da tempo l'afferenza della sostanza nera e quindi diventano ipersensibili. Un esempio tipico delle discinesie si può vedere nei pazienti che ricevono più levodopa di quanta realmente necessiterebbero. Oltre alla classica discinesia ne esiste un secondo tipo chiamata distonia di fine dose, che si verifica quando i livelli di dopamina nel sangue sono bassi o si riducono troppo rapidamente. Questa è caratterizzata da torsioni del collo, braccia, gambe, mani o tronco, visibile soprattutto di notte o di prima mattina, con durata variabile da alcuni secondi ad eventi prolungati. La valutazione delle discinesie fa uso sia della scala UPDRS che della specifica scala UDysRS ("Unified Dyskinesia Rating Scale").
Oltre ai disturbi motori esistono altri tipi di disturbi classificati in fluttuazioni non motorie che, nonostante non siano collegate all'ambito motorio, possono essere in alcuni casi più disabilitanti di quelle motorie stesse [99]. Le fluttuazioni non motorie fanno riferimento a disturbi neuropsichiatrici che esasperati nella fase "off", si manifestano sotto forma di stati d'ansia, irritabilità, depressione, attacchi di panico con sospiri ripetuti, grida [100], affaticamento e riduzione dell'energia [101]. La fase "on" è invece caratterizzata da un umore elevato, euforia e sensazione di benessere [102]. La valutazione delle fluttuazioni non motorie avviene per mezzo della scala UPDRS, della "Beck Depression Inventory" o della "Parkinson's Disease Sleep Scale". Dallo studio delle fluttuazioni non motorie è evidente come esse abbiano un grande impatto sulla qualità della vita della persona, al pari delle fluttuazioni motorie. È quindi necessario, come emerge dalla letteratura [103], una scala unificata per la valutazione delle

fluttuazioni non motorie.

4.2 Scale di valutazione

Le più diffuse scale di valutazione usate nella malattia di Parkinson sono l'UPDRS ("Unified Parkinson's Disease Rating Scale") e la EDSS ("Expanded Disability Status Scale").

4.2.1 UPDRS

Nel corso degli anni sono state sviluppate svariate scale cliniche per la valutazione del Parkinson, ognuna delle quali era però incentrata su un particolare sintomo [104]. Nel 1987, da parte di un comitato di esperti, è stato necessario elaborare una scala di valutazione unificata, chiamata "Unified Parkinson's Disease Rating Scale" o UPDRS [105], ad oggi adottata come base standard di valutazione della gravità e progressione della malattia, insieme alla sua versione più recente chiamata "Movement Disorders Society-Unified Parkinson Disease rating Scale" (MDS-UPDRS) [106]. L'UPDRS è costituita da quattro parti:

- Parte I, esperienze non motorie della vita quotidiana;
- Parte II, esperienze motorie della vita quotidiana;
- Parte III, valutazione motoria;
- Parte IV, complicanze motorie dovute alla terapia farmacologica.

Ogni sezione è costituita da una serie di domande a cui deve essere attribuito un valore numerico compreso tra 0 e 4. Il valore 0 indica che il soggetto svolge l'ambito definito dalla domanda in maniera "Normale", mentre i valori successivi indicano la presenza di difficoltà, la cui gravità è definita in 1 "Minima", 2 "Lieve", 3 "Moderata" e 4 "Grave". Nel caso non sia possibile rispondere ad alcune domande, dovrà essere assegnato il valore UR ovvero "Unable to Rate". Il questionario può essere compilato sia dal medico o infermiere in base alle risposte fornite dal paziente che dal paziente stesso basandosi sulle proprie osservazioni.
Parte I: Aspetti Non-Motori della vita di tutti i giorni (nM-EDL)
In questa parte viene valutato l'impatto dei disturbi non motori sulla vita di tutti i giorni mediante 13 domande, 6 riferite alla sezione IA e 7 alla sezione IB. La sezione IA deve esser compilata dal valutatore poichè riferita ad aspetti più complessi come compromissione cognitiva, allucinazioni, umore depresso o ansioso e apatia. La parte IB invece viene compilata dal paziente, poichè riferita a disturbi del sonno, disturbi sensitivi e di affaticamento.

Parte II: Aspetti Motori delle Esperienze della vita quotidiana (M-EDL)
Questa parte deve essere compilata accuratamente dal soggetto poichè vengono trattati argomenti di vita quotidiana come il vestirsi, il mangiare, l'alzarsi dal letto, l'eloquio, il tremore e il freezing.
Parte III: Esame Motorio
Questa parte viene compilata dall'esaminatore in base alle proprie osservazioni mediche. Si valuta se il paziente è sottoposto a terapia farmacologica e la fase "on" o "off" in cui si trova. Inoltre in questa sezione devono esser inserite informazioni legate all'esame motorio del paziente come la mobilità degli arti, i tremori e la stabilità posturale.
Parte IV: Complicanze Motorie
In quest'ultima parte vengono valutate le complicazioni motorie come fluttuazioni e distonie, con l'intento di dividere in modo chiaro i disturbi legati alla malattia da quelli legati alla terapia farmacologica, dividendo la fase di "on" da quella di "off".
Poichè l'impatto della MP è molto ampio, il solo utilizzo dell'UPDRS non è sufficiente a caratterizzare lo stato clinico del soggetto in esame [106]. L'UPDRS dovrebbe essere correlata da altri strumenti legati alla valutazione e quantificazione delle funzioni motorie, utili in fase di riabilitazione.

4.2.2 EDSS

L'EDSS, acronimo di "Expanded Disability Status Scale", è la principale scala usata per la valutazione del grado di severità di malattie neurologiche e per verificare il danno anatomico subito dal tessuto nervoso. Essa venne proposta dall'americano Kurtzke [107]. Il punteggio totale della scala EDSS viene determinato da due fattori: la capacità di deambulazione e i punteggi relativi ad otto sistemi funzionali (Piramidale, Cerebrale, Tronco encefalico, Sensitivo, Sfinterico, Visivo, Cerebrale, Altri). In ogni sottosistema viene usata una sotto-scala che valuta lo stato funzionale dei principali sistemi che possono essere colpiti dalla malattia. Ogni sistema funzionale riceve un punteggio compreso tra 1 e 5 che indica la gravità crescente. Al sistema "Altri" non viene dato un punteggio, bensì viene usato per inserire indicazioni particolari su un ben determinato problema come, ad esempio, l'impossibilità di deambulare. La figura 4.2 mostra tutti i punteggi usati dalla scala.

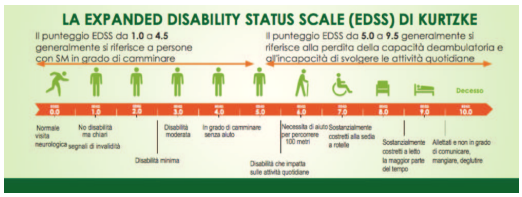

Figura 4.2: Scala EDSS *(notiziemediche.it)*

4.3 Analisi del cammino

Nella malattia di Parkinson la caratterizzazione del cammino è molto importante per valutare il progredire della malattia stessa e l'esito più o meno positivo delle terapie riabilitative. Ad oggi le valutazioni delle caratteristiche motorie si basano sull'uso di cronometri e sulla valutazione soggettiva del medico, mentre il paziente svolge determinate attività. Questo porta ad una difficile valutazione soprattutto nelle prime fasi della malattia, dove è difficile discriminare i soggetti malati, commettendo spesso errori nell'uso dei protocolli definiti. I sensori inerziali possono essere un valido strumento per caratterizzare dei parametri della camminata, anche attraverso un diretto confronto tra quelli dei soggetti sani e quelli dei soggetti parkinsoniani. Il ciclo del passo, spesso chiamato anche "Gait Analysis", è costituito da una serie di eventi periodici che corrispondono all'intervallo temporale tra due contati inziali successivi dello stesso piede. La figura 4.3 mostra il ciclo del passo. Ogni ciclo del passo è

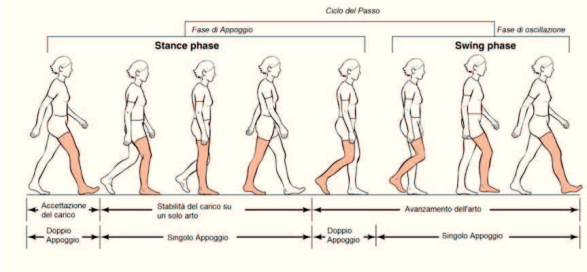

Figura 4.3: Ciclo del passo *(drespositopodologo.com)*

diviso in due fasi:

- *Appoggio (Stance)*, periodo durante il quale il piede è a contatto con il suolo e corrisponde al 60% del ciclo del passo;
- *Oscillazione (Swing)*, periodo durante il quale il piede si trova sollevato da terra per l'avanzamento dell'arto e corrisponde al 40% del ciclo del passo.

Le fasi di Stance si possono dividere in:

- *Doppio appoggio inziale*, entrambi i piedi sono a contatto con il suolo;
- *Singolo appoggio*, il piede destro è a contatto con il suolo, mentre il piede sinistro è in fase di oscillazione;
- *Doppio appoggio terminale*, entrambi i piedi sono di nuovo a contatto con il terreno.

Lo scenario descritto è equivalente per la parte sinistra, con doppio appoggio iniziale per la parte destra corrispondente al doppio appoggio terminale per la parte sinistra. Questo poichè vi è una simmetria precisa tra fase di stance e fase di swing. La durata dell'appoggio del piede a terra, variabile da soggetto a soggetto, è inversamente proporzionale alla velocità della camminata.
Il ciclo del passo può inoltre essere ridiviso in ulteriori otto eventi, cinque nella fase di stance e tre in quella di swing. Gli eventi sono:

- *Contatto tallone (heel strike)*, fase iniziale del ciclo del passo che si verifica quando il centro di gravità è nella posizione più bassa (fase si stance);
- *Piede a contatto con il suolo (foot-flat)*, fase in cui la pianta del piede è completamente a contatto con il suolo (fase di stance);
- *Appoggio intermedio (midstance)*, fase in cui il piede oscillante supera il piede di appoggio. È la fase in cui il centro di gravità è nella posizione più alta (fase di stance);
- *Distacco tallone (heel off)*, fase in cui il tallone perde il contatto con il suolo (fase di stance);
- *Distacco dita (toe off)*, fase ultima di stance, in cui il piede si stacca da terra (fase di stance);
- *Accelerazione*, fase in cui il piede non è più a contatto con il suolo. Si attivano i muscoli flessori dell'anca per accelerare la gamba in avanti (fase di swing);
- *Oscillazione intermedia (midswing)*, fase in cui l'arto si sposta da una posizione posteriore del corpo ad una anteriore (fase di swing);
- *Decelerazione*, fase in cui l'azione dei muscoli rallenta la gamba e stabilizza il piede in preparazione del successivo foot-flat (fase di swing).

4.4 Strumenti per l'identificazione delle attività motorie

Negli ultimi anni uno dei settori sul quale si è concentrata maggiormente la ricerca scientifica, è lo sviluppo di strumenti utili all'identificazione automatica delle singole attività motorie. Questo è dovuto al loro impiego in molteplici settori, dal quello del benessere a quello sanitario. In ambito wellness o sportivo sono nati prodotti come i Google Glass, mini computer indossabili, o il rilevatore di attività Fitbit. Nel settore sanitario gli strumenti più utilizzati per l'"Activity Detection" sono il sistema stereofotogrammetrico e i dispositivi inerziali. Il primo, basato sull'uso di telecamere ad infrarossi, permette di identificare la posizione di specifici marker posti in zone caratteristiche del corpo umano. Benchè questa tecnica abbia un elevato grado di efficienza nel riconoscere le singole attività, il suo uso è limitato in laboratorio o in ambiente controllato. Questo è dovuto all'elevato costo ed ingombro del sistema, oltre che al suo non semplice utilizzo. I dispositivi indossabili sono un'ottima alternativa ai sistemi da laboratorio poichè economici, poco ingombranti e facili da utilizzare. Il loro funzionamento è basato su sensori inerziali in grado di catturare il movimento del segmento corporeo su cui vengono posti. In letteratura esistono diversi esempi di utilizzo di sensori inerziali, in particolare dell'accelerometro, come quello di Najafi [108], di Foerster [109] o di Kern [110], utilizzati per identificare sia attività statiche che dinamiche. Alcuni studi come quelli di Intille [111] o di Ermes [112], hanno ottenuto buoni risultati con l'impiego contemporaneo di più dispositivi. Questo approccio però non può essere utilizzato nella vita quotidiana a lungo termine, poichè la presenza di più dispositivi limita il movimento. Altri studi, come quello di Karantonis [113] o Sung [114], con l'utilizzo di un singolo dispositivo posto sul petto o sulla vita, hanno ottenuto buoni risultati anche se non paragonabili con gli studi precedenti.
Oltre alle tecniche di data-fusion, basate sull'elaborazione dei dati grezzi acquisiti dai sensori, ne esistono delle altre basate sull'utilizzo di classificatori. Queste ultime si basano sul concetto di apprendimento, ovvero da un insieme di dati noto a priori, vengono apprese delle caratteristiche, mediante le quali è possibile fare delle predizioni in maniera induttiva su dei nuovi dati presi in input. L'apprendimento si può dividere in due categorie:

- *Apprendimento supervisionato*, a sua volta divisibile in addestramento e predizione. L'addestramento è l'ottimizzazione di una serie di variabili, in base ad un set di dati di training di cui si conosce l'appartenenza ad una determinata classe. La predizione è la classificazione di un nuovo set di dati basandosi sulle informazioni raccolte dai dati di training.

- *Apprendimento non supervisionato*, classificazione di dati di input, in base alle loro similitudini, senza l'utilizzo di dati di training iniziali. Gli algoritmi sviluppati con questa tecnica hanno una complessità maggiore rispetto all'apprendimento supervisionato e richiedono un'ulteriore fase di validazione.

Nell'activity detection la classificazione mediante l'apprendimento supervisionato è basata su tre steps:

- *Estrazione di features*, per ogni attività nota, identifica a priori, vengono calcolate delle caratteristiche al fine di ottenere un vettore di caratteristiche di tutte le attività;
- *Classificazione*, generazione di un modello di classificazione basato sul vettore di caratteristiche precedentemente calcolato (ogni attività avrà associato un certo andamento delle caratteristiche);
- *Predizione*, generazione di un vettore di risposta per dati di cui non è nota la classificazione (si identifica a quale classe appartengono i nuovi dati presi in input).

Le caratteristiche, nel dominio del tempo, usate in letteratura per identificare le attività sono la magnitudo del segnale [115], la media, la deviazione standard e le correlazioni tra i campioni dei tre assi [116]. Invece nel dominio della frequenza viene calcolata la trasformata di Fourier [117] o la trasformata Wavelet [118]. I tipi di classificatori esistenti sono il "Decision Tree", il "k-nearest neighbors" (k-NN), il "Navie Bayes", le "Support Vector Machine" (SVM), le "Reti Neurali" e la "Linear Discriminant Analysis" (LDA). Il decision tree è una tecnica di classificazione ad albero, dove ogni nodo rappresenta un test effettuato sui dati in ingresso. Il soddisfacimento o meno della condizione porta a percorrere un ramo piuttosto che un altro, andando così a verificare una successiva condizione. Dopo aver testato n condizioni si arriverà all'ultimo nodo dell'albero, la foglia, ovvero l'assegnazione di una classe ai dati in ingresso testati. Le SVM permettono di ricavare una funzione di decisione che, mediante addestramento su un certo set di training, permette di separare tutte le features in n regioni, ognuna associata ad una classe. Si deve trovare la funzione in grado di massimizzare la distanza tra i vettori di ogni classe. Il classificatore naive bayes viene utilizzato quando all'interno di una classe le caratteristiche sono indipendenti le une dalle altre. Esso richiede di conoscere a priori le probabilità condizionali delle classi. Nella sua fase di predizione, le features dei nuovi dati vengono classificate secondo la più grande probabilità di appartenenza ad una classe piuttosto che ad un'altra. Il k-NN è un algoritmo utilizzato per classificare dati, basandosi sulla vicinanza delle loro caratteristiche con quelle di dati noti a priori e classificati. Lo spazio è diviso in n regioni

in base alle posizioni dei dati di training e per considerare i vicini più vicini al dato in input considerato, si calcolano le distanze tra i punti. La classe viene scelta in base alla molteplicità della distanza minima tra le caratteristiche dei dati in input e quelle di training. Le distanze che possono essere considerate sono:

- *Distanza Euclidea*, misura la distanza tra due punti nel piano Euclideo;
- *Distanza di Minkowsky*, distanza Euclidea elevata ad un generico esponente λ;
- *Distanza matrice quadrata*, distanza tra i punti e una matrice quadrata simmetrica da loro generata;
- *Distanza di Chebychev*, massima distanza delle features tra due punti.

La rete neurale è un modello matematico formato da "neuroni" artificiali che formano un gruppo di interconnessioni di informazioni. Le reti neurali sono sistemi adattativi che cambiano comportamento in base a delle informazioni interne o esterne e possono essere divise in tre livelli:

- Livello di ingresso (I), vengono ricevute le informazioni e adattate al livello dei neuroni di calcolo;
- Livello nascosto (H), è il livello di calcolo e può essere composto da più sottolivelli;
- Livello di uscita (O), vengono ottenute le informazioni di risposta dai neuroni e i risultati vengono forniti in uscita al sistema.

L'LDA è un metodo di classificazione che suppone che ogni classe generi dei dati diversi basati su diverse distribuzioni gaussiane. Esso comprende una fase di allenamento, stimando i dati di distribuzione gaussiana per ogni classe e predizione, andando a trovare la classe con minor costo di errori di classificazione. In letteratura i miglior risultati di classificazione di attività sono stati raggiunti con l'uso dell'SVM [118] e del k-NN [119].

4.4.1 Sistemi di rilevazione della MP

Nello studio della MP e dei sintomi ad essa correlati i dispositivi più utilizzati sono la Kinect, il sistema Stereofotogrammetrico, l'elettromiografia, la piattaforma di forza e i sistemi inerziali. La Kinect composta da una telecamera RGB, una di profondità con proiettore ad infrarossi, una monocroma CMOS e 4 microfoni per l'audio, permette l'interazione con la console senza controllore, ma mediante movimenti del corpo, comandi vocali o interazione con oggetti presenti nell'ambiente circostante. Lo svantaggio di questa strumentazione è il

costo non molto economico.
La stereofotogrammetria, costituita da due o più telecamere nel visibile o infrarosso, è in grado di determinare la posizione di più marker messi sulla cute del soggetto rispetto ad un sistema di riferimento prescelto. L'elaborazione delle immagini acquisite dalle telecamere è un'operazione molto laboriosa, poichè include fasi di sogliatura, individuazione dei marker, calibrazione e ricostruzione. Nonostante la sua elevata affidabilità è una strumentazione molto costosa ed utilizzabile solo in laboratorio.
L'elettromiografia di superficie è una tecnica che permette di verificare l'attività muscolare mediante l'applicazione di elettrodi sulla cute del paziente, in prossimità dei muscoli da analizzare. I dati acquisiti da prove di cammino del paziente parkinsoniano vengono confrontati con quelli acquisiti su soggetti sani, al fine di valutare il cambiamento dell'attività muscolare in presenza della MP. Anche l'uso di questa strumentazione può esser effettuato solo in ambulatorio e in presenza di personale esperto, a causa del costo della strumentazione e della lunga fase di analisi e confronto dei risultati.
La piattaforma di forza permette di valutare i cambiamenti posturali e le problematiche di equilibrio del soggetto in esame. Questa tecnica permette di analizzare sintomi della malattia come le torsioni del busto o la perdita di equilibrio. In genere è poco utilizzata a causa del suo eccessivo costo.
I dispositivi indossabili sono dei sistemi miniaturizzati, che grazie all'uso di accelerometro, giroscopio, magnetometro e barometro, permettono di acquisire informazioni sul movimento dell'arto sopra il quale vengono posti. Il loro grande utilizzo è legato alla loro economicità, portabilità, non invasività e facilità di utilizzo. I principali sintomi analizzati mediante sistemi inerziali sono il tremore, il freezing e le fluttuazioni motorie.

4.5 Sensori inerziali

Negli ultimi anni i *Micro Electro Mechanical System* (MEMS) hanno avuto un ampio sviluppo grazie alle loro ridotte dimensioni e crescente precisione e accuratezza. Questi sono dei microsistemi intelligenti in grado di inglobare, in substrati di dimensioni ridotte, sensori altamente performanti come accelerometro, giroscopio, magnetometro e barometro. Di seguito verrà descritto il funzionamento dei sensori utilizzati per l'analisi delle attività motorie.

4.5.1 Accelerometro

Il principio di funzionamento dell'accelerometro consiste nel sospendere una massa m ad un supporto elastico k e valutare lo spostamento prodotto sotto l'effetto di una forza esterna F. La massa sottoposta ad una accelerazione si

sposterà, rispetto alla sua posizione di riposo, di un valore proporzionale all'accelerazione ricevuta. La variazione di posizione sarà poi convertita in un segnale elettrico acquisibile dagli attuali strumenti di misura. Al fine di ridurre le oscillazioni non volute il sistema adotta uno smorzatore b, ottenendo un sistema chiamato Massa-Molla-Smorzatore (MMS). La figura 4.4 mostra il principio di funzionamento dell'accelerometro. La seconda legge di Newton è

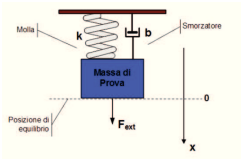

Figura 4.4: Principio funzionamento accelerometro *(notiziemediche.it)*

in grado di descrivere la deflessione che subisce la massa sotto l'effetto di una forza esterna 4.1.

$$m\frac{\partial^2 x}{\partial^2 t} + \frac{\partial x}{\partial t} + kx = ma \tag{4.1}$$

La variabile t è il tempo, a l'accelerazione, m è la massa, k è la costante elastica della molla e b è lo smorzamento. Nel dominio di Laplace è possibile definire il legame esistente tra accelerazione e deflessione mediante la formula (con s variabile di Laplace) 4.2.

$$\frac{x(s)}{a(s)} = \frac{1}{s^2 + \frac{k}{m}s + \frac{b}{m}} \tag{4.2}$$

La frequenza naturale ω_n e la sensibilità S sono invece legate delle formule 4.3 e 4.4.

$$\omega_n = \sqrt{\frac{k}{m}} \tag{4.3}$$

$$S = \frac{m}{k} + \frac{1}{\omega^2} \tag{4.4}$$

Dalle due equazioni si può notare come l'aumento del rapporto m/k determina un aumento della sensibilità e conseguente diminuzione della frequenza naturale, fattore chiave nella definizione della banda di lavoro. Un ulteriore applicazione dell'accelerometro è come strumento di *tilt sensing*, ovvero strumento di misurazione dell'inclinazione di un corpo sul quale viene posto. Per realizzarlo viene utilizzata solo l'accelerazione statica, supponendo che l'unica forza che agisce sul corpo sia la forza peso. Se si considera un accelerometro con asse x parallelo all'orizzonte, tutta la forza di gravità è applicata ad un unico asse. Se invece l'accelerometro viene fatto ruotare di un angolo θ rispetto

all'orizzonte, la forza che agisce sull'asse x è data dalla formula 4.5.

$$A_x = 1g \sin(\theta) \tag{4.5}$$

La formula inversa che ci fornisce l'inclinazione dell'accelerometro è la 4.6.

$$\theta = \sin^{-1} A_X \tag{4.6}$$

La figura 4.5 descrive le due situazioni illustrate. Se invece si considera un

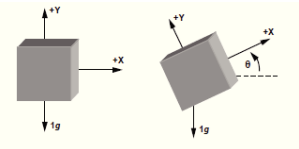

Figura 4.5: Applicazione di tilt sensing con accelerometro mono assiale *(settorezero.com)*

accelerometro tri-assiale, con forza di gravità agente su tutti e tre gli assi, la situazione è quella di figura 4.6. Ipotizzando una condizione iniziale con asse

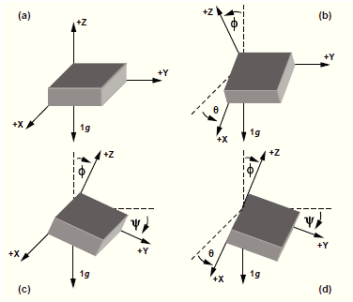

Figura 4.6: Applicazione di tilt sensing con accelerometro tri-assiale *(settorezero.com)*

x ed y paralleli all'orizzonte, le formule necessarie al calcolo degli angoli θ, ψ e ϕ sono date dalle equazioni 4.7, 4.8 e 4.9.

$$\theta = \tan^{-1}(\frac{A_x}{A_x^2 + A_y^2}) \tag{4.7}$$

$$\phi = \tan^{-1}(\frac{A_y}{A_x^2 + A_z^2}) \tag{4.8}$$

$$\psi = \tan^{-1}(\frac{A_z}{A_y^2 + A_x^2}) \tag{4.9}$$

L'angolo θ è l'angolo di beccheggio, ϕ di rollio e ψ l'angolo formato con l'asse z. Viste le risposte lente degli accelerometri, questi potranno essere usati per stimare le inclinazioni di un corpo libero solo in condizioni statiche. In condizioni dinamiche invece, l'errore introdotto dall'accelerometro dovrà essere compensato dall'uso del magnetometro.

4.5.2 Giroscopio

Il giroscopio è in grado di misurare il movimento di rotazione di un corpo in uno spazio inerziale e attorno ad un dato asse. Considerando un corpo di dimensioni trascurabili in moto con velocità v rispetto ad un sistema di riferimento non inerziale, se il sistema non inerziale ruota con velocità angolare Ω rispetto al sistema inerziale, sul corpo agisce una forza F chiamata forza di Coriolis e definita dall'equazione 4.10.

$$F_{coriolis} = -2m\Omega \wedge v \tag{4.10}$$

La figura 4.7 illustra lo scenario descritto. La realizzazione del giroscopio com-

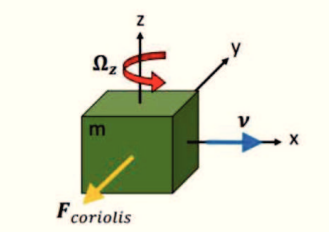

Figura 4.7: Forza di Coriolis

prende due masse, una di drive, posta in oscillazione a frequenza propria lungo x, mediante pettini capacitivi con controllo di retroazione, e una di sens, eccitata lungo la direzione y dalla forza di Coriolis quando viene applicata una velocità angolare al giroscopio. La forza di Coriolis, applicata alla massa m,

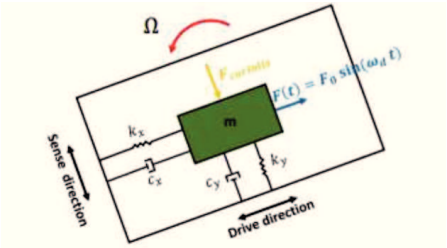

Figura 4.8: Principio di funzionamento del giroscopio

genera uno spostamento della stessa in una direzione perpendicolare alla velocità v. Lo spostamento viene visto elettricamente come una variazione di carica elettrica tra i pettini capacitivi, che circondano la massa (condensatori). Questi valori di capacità vengono opportunamente convertiti ed acquisiti dal sistema di misura. Una delle caratteristiche principali del giroscopio à la necessità di una fase iniziale di calibrazione, al fine di eliminare errori sistematici come il bias. Il bias, definito come uscita media del giroscopio senza nessuna rotazione applicata, è un valore costante che provoca una traslazione dei valori della velocità angolare nel tempo. Esso può essere calcolato andando a prelevare l'uscita del giroscopio, su un numero elevato di campioni, quando al suo ingresso non viene applicata nessuna rotazione. L'uscita reale sarà ottenuta sottraendo al valore misurato il valore del bias. Oltre al bias il magnetometro può essere anche affetto da disturbi termici. La velocità angolare, calcolata dalla derivata della posizione angolare nel tempo, è data dalla formula 4.11.

$$\dot{\theta} = \frac{\partial \theta}{\partial t} \tag{4.11}$$

Al livello teorico il relativo integrale è in grado di fornirci le posizioni angolari della massa. Nei sistemi digitali, vista l'impossibilità di calcolare l'integrale automaticamente, si deve ricorrere a sue approssimazioni mediante sommatorie di un numero finito di valori. Il giroscopio è in grado di calcolare l'orientazione nello spazio di un corpo rigido, a partire dalla stima delle sue velocità angolari intorno agli assi x, y e z. Se le velocità angolari cambiano troppo rapidamente, si genererà un ritardo sulla stima dell'angolo. Inoltre il giroscopio è caratterizzato da un offset non nullo, che tende a produrre un output anche in assenza di rotazioni in ingresso. Dalle varie integrazioni, necessarie per calcolare l'angolo, l'offset cresce rapidamente scostandosi dallo zero. L'offset, detto drift, crea degli errori non trascurabili nella stima degli angoli yaw, pitch e roll dopo appena pochi minuti di acquisizione.

4.5.3 Magnetometro

Il magnetometro è un sensore in grado di fornire intensità e verso del vettore campo magnetico H, mediante tre unità poste ortogonalmente tra loro. Il valore misurato, benchè influenzabile da latitudine e longitudine, può ritenersi costante, se utilizzato localmente in assenza di disturbi elettromagnetici esterni. Al suo interno resistenze, poste lungo le linee di flusso del campo magnetico, subiscono una variazione in presenza di un campo magnetico esterno. La resistenza è un conduttore di corrente avente elevata permeabilità magnetica e formato da una particolare lega di nickel-ferro. In presenza di un campo magnetico esterno il vettore di magnetizzazione subisce una rotazione α, proporzionale all'intensità del campo magnetico applicato. Questa influenza viene

registrata da una variazione della resistenza al passaggio di corrente illustrata nella formula 4.12.

$$R - R_0 = \Delta R_0 \cos \alpha \tag{4.12}$$

Nella formula precedente R_0 è il valore della resistenza in assenza di campo magnetico, α l'angolo di rotazione del vettore di magnetizzazione all'interno del materiale e ΔR_0 è una costante che dipende dal materiale. La figura 4.9 illustra il funzionamento del magnetometro. Uno dei principali problemi dei

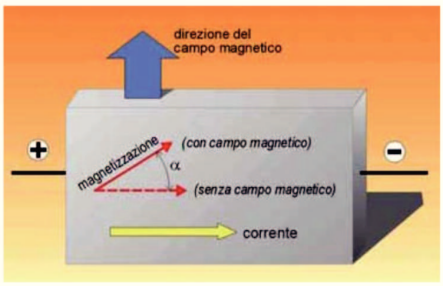

Figura 4.9: Principio di funzionamento del magnetometro

magnetometri è la loro sensibilità a tutti i campi magnetici e non solo a quello terrestre, come a quelli indotti o prodotti da batterie. I campi magnetici esterni costanti possono esser trattati attraverso una fase iniziale di calibrazione. Stesso approccio non può esser fatto però per sorgenti magnetiche casuali e momentanee. Il magnetometro viene spesso utilizzato in combinazione con altri sensori come accelerometro e giroscopio, usati per determinare l'angolo di imbardata mediante considerazioni trigonometriche 4.13, 4.14 e 4.15.

$$M_x = m_x \cos \phi + M_Z \cos \phi \tag{4.13}$$

$$M_y = m_x \sin \theta \sin \phi + m_y \cos \theta - m_z \sin \theta \cos \phi \tag{4.14}$$

$$\Psi = atan2(M_x M_y) \tag{4.15}$$

Nelle formule M_x, M_y e M_z rappresentano le componenti di campo magnetico lungo i tre assi, utili per definire il vettore campo magnetico terrestre e calcolare l'angolo azimutale tra la direzione del sensore ed il polo Nord magnetico terrestre. Poichè il polo Nord magnetico è posto al sud geografico, è necessario sommare un angolo definito di declinazione Ω_{nE}, descritto nella formula 4.16.

$$\Psi_N = \Psi_m + \Omega_{nE} \tag{4.16}$$

4.5.4 Barometro

Il barometro è lo strumento utilizzato per misurare la pressione atmosferica, ovvero il peso dell'aria che circonda la terra ed agisce sopra i corpi che incontra. L'atmosfera terrestre può esser formata da zone di bassa ed alta pressione. Le prime sono caratterizzate da una minor massa d'aria al di sopra del punto di misura, mentre le seconde da una maggior massa atmosferica. La pressione atmosferica tende a diminuire con l'aumentare dell'altitudine. Generalmente è misurata ad una latitudine di 45° al livello del mare e ad una temperatura di 0∘C, corrispondente ad una colonna di mercurio alta 760mm. Ovviamente esistono diverse unità di misura della pressione atmosferica come l'atmosfera dove 1 atm = 760mm Hg, il Torr dal suo inventore Torricelli con 1 atm = 760 torr, il Pascal con 1 atm = 101325 Pa e il bar, unità standard del sistema internazionale, con 1 atm = 1013.25 bar. Il barometro, sensore in grado di misurare la pressione atmosferica, è formato da una piccola cavità al cui interno viene fatto il vuoto. Una parete della cavità è chiusa con un sensore di deformazione in grado di deformarsi al variare del peso dell'aria. La deformazione subita verrà trasformata in segnale elettrico mediante trasduttori piezoresistivi o capacitivi ed acquisito dal sistema di misura.
Nell'activity detection il barometro viene utilizzato come strumento utile all'identificazione di variazioni di quota del soggetto lungo la direzione z. Data la pressione atmosferica e la temperatura, l'altitudine viene calcolata con la formula 4.17.

$$h = (T + 273.15)/0.0065(1 - P/P_0)^{0.19} \tag{4.17}$$

Nella formula T è pari alla temperatura misurata al momento dell'acquisizione ed espressa in centigradi, P è la pressione misurata in Pascal e P_0 è la pressione atmosferica al livello del mare. Il principale problema nell'uso del barometro è il forte disturbo presente nelle acquisizioni, causato sia da rumore termico che da rumore di quantizzazione, che rende necessario un uso combinato del barometro con altri sensori come accelerometri, giroscopi e magnetometri.

4.6 Analisi del movimento

Nello studio delle attività motorie è necessario conoscere l'orientamento del corpo rigido (soggetto in esame) rispetto ad un sistema di riferimento solidale con la superficie terrestre. Una stima dell'orientazione può essere fatta mediante algoritmi di data-fusion su informazioni acquisite dall'unità IMU (Inertial Measurement Unit), che contiene giroscopio triassiale e accelerometro triassiale, o da MARG (Magnetic Angular Rate and Gravity) sensor che, oltre all'unita IMU, aggiunge un magnetometro triassiale.

4.6.1 Angoli di Eulero

Data una terna ortonormale S_t, che rappresenta il sistema di riferimento cartesiano coincidente con la terra e data una seconda terna S_i, che rappresenta il sistema del sensore anch'esso ortonormale, ma non allineato con il sistema S_t, è possibile individuare gli angoli Ψ, Θ e Φ in grado di ruotare il sistema S_i e allinearlo con il sistema S_t. I due sistemi hanno origine coincidente. Questi angoli, indicati con il nome di Angoli di Eulero, permettono di descrivere la posizione di un corpo rigido nello spazio e possono essere descritti come:

- *Yaw*, definito come angolo di imbardata, rappresenta la rotazione dell'angolo Ψ attorno all'asse z;
- *Pitch*, definito come angolo di beccheggio, rappresenta la rotazione dell'angolo Θ attorno all'asse y;
- *Roll*, definito come angolo di rollio, rappresenta la rotazione dell'angolo Φ attorno all'asse x.

La figura 4.10 mostra un esempio di applicazione degli angoli descritti. Le ro-

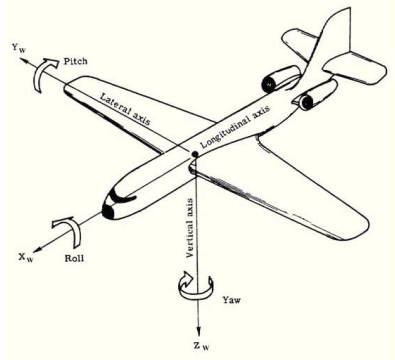

Figura 4.10: Rotazioni angolo Yaw, Pitch e Roll *(giuseppecaccavale.it)*

tazioni intorno all'origine sono rappresentate dalle funzioni $R_z(\Psi)R_y(\Theta)R_x(\Phi)$ con $\Psi \in$ [-π,π], $\Theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ e $\Phi \in$ [-π,π]. $R_z(\Psi)$ è la rotazione di lungo l'asse z, $R_y(\Theta)$ è la rotazione di lungo l'asse y e $R_x(\Phi$ è la rotazione di Φ lungo l'asse x. Data la rappresentazione matriciale di ogni rotazione $R_z(\Psi), R_y(\Theta) e R_x(\Phi)$ e data la rotazione del sistema S_i su S_t, definita da $R^{S_i}_{S_t}$, la matrice di rotazione complessiva è data dalla formula 4.18.

$$\begin{bmatrix} \cos\Psi\cos\Theta & \cos\Psi\sin\Theta\sin\Phi - \sin\Psi\cos\Phi & \sin\Psi\sin\Phi + \cos\Psi\sin\Theta\cos\Phi \\ \sin\Psi\cos\Theta & \sin\Psi\sin\Theta\sin\Phi + \cos\Psi\cos\Phi & \sin\Psi\sin\Theta\cos\Phi - \cos\Psi\sin\Phi \\ -\sin\Theta & \cos\Theta\sin\Phi & \cos\Theta\cos\Phi \end{bmatrix} \tag{4.18}$$

Uno dei problemi, da cui può essere affetta la matrice di rotazione, è il Gimbal Lock; ovvero fissando un particolare valore dell'angolo Θ (nel nostro caso $\Theta = \pi/2$), esistono infinite combinazione degli altri due angoli Ψ e Φ che portano alla stessa matrice di rotazione. Questa problematica porta il passaggio dalla condizione iniziale, dove mediante l'uso della matrice era possibile ottenere una rotazione generica in uno dei tre assi, a quella in cui si perde un grado di libertà.

4.6.2 I Quaternioni

Per ovviare al problema del Gimbal Lock possono essere utilizzati i quaternioni, elementi dello spazio lineare rappresentati come $\hat{q} = (q_1, q_2, q_3, q_4)$. L'equazione con q_1 parte scalare e $w = (q_2, q_3, q_4)$ parte vettoriale, può anche essere scritta come $\hat{q} = (q_1, v)$. Se v è 0 il quaternione viene chiamato reale, se invece q_1 è zero viene chiamato quaternione vettoriale. Vista l'analogia con i numeri complessi è possibile scrivere un quaternione come $\hat{q} = (q_1 + q_2 i + q_3 j + q_4 k$ con i, j e k unità immaginarie.
Alcune proprietà notevoli dei quaternioni sono:

- Il complesso coniugato definito come $\hat{q}^* = (q_1 - q_2 i - q_3 j - q_4 k$;
- La norma del quaternione definita come $|\hat{q}| = \sqrt{\hat{q}^* \hat{q}}$;
- L'inverso del quaternione è definito come $\hat{q}^{-1}$ tale che $\hat{q}^{-1}\hat{q} = 1$ e $\hat{q}\hat{q}^{-1} = 1$;

Considerando un quaternione unitario definito come $u = (u1, u2, u3, u4) = (u1, u) = \cos\Theta + u \sin\Theta$, questo rappresenta la rotazione di un angolo Θ intorno all'asse individuato dal versore $u = (u2, u3, u4)$. Da ciò si può affermare che la rotazione Θ attorno ad un vettore $\overline{U}$ può essere descritta come il passaggio da un sistema di riferimento S_1 ad uno S_2, entrambi con origine coincidente. Il quaternione che descrive questa situazione è calcolato nella formula 4.19.

$$\hat{q}_{S_2}^{S_1} = [q1q2q3q4] = [\cos\Theta/2 - u_x \sin\Theta/2 - u_y \sin\Theta/2 - u_z \sin\Theta/2] \quad (4.19)$$

Le componenti u_x, u_y e u_z sono le componenti del vettore $\overline{U}$ nel sistema di riferimento S_1 e Θ è l'angolo di rotazione attorno all'asse. La matrice che descrive l'orientazione del sistema S_1 rispetto ad un sistema S_2 è descritta nella formula 4.20.

$$R_{S_2}^{S_1} = \begin{bmatrix} 2q_1^2 - 1 + 2q_2^2 & 2(q_2q_3 + q_1q_4) & 2(q_2q_4 + q_1q_3) \\ 2(q_2q_3 - q_1q_4) & 2q_1^2 - 1 + 2q_3^2 & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_3q_4 - q_1q_2) & 2q_1^2 - 1 + 2q_4^2 \end{bmatrix} \quad (4.20)$$

Anche per i quaternioni è possibile descrivere una rotazione come la sequenza di più rotazioni intermedie partendo dallo stesso sistema di riferimento iniziale,

ognuna formata da un quaternione. Le formule utilizzate per il calcolo degli angoli di Eulero dai quaternioni sono la 4.21, 4.22 e 4.23.

$$yaw = atan2(2q_2q_3 - 2q_1q_4, 2q_1q_1 + 2q_2q_2 - 1) \tag{4.21}$$

$$pitch = atan(\frac{2(q_2q_4 - q_1q_3)}{\sqrt{2(q_1q_2q_3q_4)^2 + (q_1q_1 - q_2q_2 - q_3q_3 + q_4q_4)^2}}) \tag{4.22}$$

$$roll = atan(\frac{2(q_1q_2 - q_3q_4)}{\sqrt{2(q_2q_4q_1q_3)^2 + (q_1q_1 - q_2q_2 - q_3q_3 + q_4q_4)^2}}) \tag{4.23}$$

I quaternioni possono essere usati per descrivere la rotazione di un sistema di riferimento S rispetto ad un sistema globale. La rappresentazione di un corpo rigido nello spazio può esser fatta con la matrice di rotazione R alla quale, a sua volta, è possibile associare un quaternione unitario. Ogni quaternione rappresenta una rotazione nello spazio tridimensionale, così come ogni numero complesso unitario rappresenta una rotazione nel piano.

4.7 Protocollo di acquisizione

Nello studio di algoritmi per l'analisi del Parkinson, il primo step fatto è stata la definizione di criteri di inclusione al test e del protocollo di acquisizione. Con criteri di inclusione si fa riferimento ad una serie di scelte, fatte in fase di selezione della popolazione campione, al fine di ottenere un ben determinato target di soggetti. Sono stati scelti tutti i pazienti con malattia conclamata da almeno due anni, con qualsiasi sintomo e in qualsiasi stadio della malattia, purchè abbiano capacità di deambulazione autonoma e senza l'utilizzo di ausili terzi. Ogni valutazione è stata condotta da personale medico qualificato presso il reparto di Neurologia degli Ospedali Riuniti Marche Nord di Pesaro (PU) e la palestra riabilitativa della casa di Cura Villa dei Pini di Civitanova Marche (MC).
I test scelti si possono raggruppare in prove per l'analisi del tremore, per l'analisi del freezing e per l'analisi delle fluttuazioni. Sono stati analizzati 42 soggetti, 30 affetti da tremore e 12 da freezing. Ad ogni soggetto è stato chiesto di effettuare i test per l'analisi delle fluttuazioni. Le prove per l'analisi del tremore sono state:

- *Resting Tremor (RT)*, ogni soggetto rimane seduto con occhi chiusi e braccia distese sulle gambe;

- *Postural Tremor (PT)*, ogni soggetto rimane seduto con occhi chiusi e braccia perpendicolari al corpo;

- *Kinetic Tremor (KT)*, ogni soggetto rimane seduto con occhi chiusi andando alternativamente a toccare con ambo gli indici delle mani la punta del naso;

- *Glass Test (GT)*, dati due bicchieri, uno dei quali riempito con acqua e posti paralleli su una superficie piana ad una distanza di 20cm tra loro, al paziente viene chiesto di prendere con una mano il bicchiere pieno e versare l'acqua nell'altro vuoto. Riposizionare al proprio posto il bicchiere vuoto e ripetere l'operazione per quattro volte.

Tutti i test, ad eccezione del GT, verranno eseguiti per 4 minuti, i primi due senza stimoli esterni, mentre i restanti con stimoli cognitivi. Gli stimoli cognitivi sono scioglilingua o conteggi alla rovescia fatti pronunciare da paziente a voce alta al fine di deviare l'attenzione dall'arto in esame. Le prove scelte per il freezing sono state:

- *Time Up and Go (TUG)*, al soggetto è stato chiesto di alzarsi da una sedia, percorrere un tratto rettilineo di 5m, girare intorno ad un birillo e ritornare al punto di partenza;

- *Attraversamento di un passaggio stretto*, al soggetto viene chiesto di attraversare un percorro rettilineo di lunghezza 6m e delimitato da due parallele equidistanti 40cm;

- *Attraversamento di una porta*, al soggetto viene chiesto, partendo da una posizione seduta, di alzarsi e attraversare una porta sedendosi su una seconda sedia, posta al lato opposto della porta. Le sedie distano 2.5m dal centro della porta;

Per ogni test il numero delle ripetizioni è stato posto a tre.
Le fluttuazioni sono stata valutate con un singolo test, necessario per verificare il passaggio dallo stato ON a quello OFF della malattia, causato dalla perdita di efficacia della terapia farmacologica. Al soggetto è stato chiesto di svolgere le abituali attività di vita quotidiana come sedersi, distendersi, camminare, salire e scendere le scale per almeno 240 minuti. Il passaggio ON-OFF viene valutato attraverso la caratterizzazione della camminata del paziente parkinsoniano, ovvero andando a verificare come durante le fasi dinamiche di camminata, il soggetto varia in maniera considerevole parametri notevoli. I parametri considerati sono stati:

- Il numero di passi in ogni fase dinamica di lunghezza prefissata;

- La lunghezza del passo [m];

- La velocità del passo [m/s];

- L'altezza del piede dal suolo [m];
- L'angolo di inclinazione del busto [◦];
- L'angolo di inclinazione della caviglia [◦].

Nel protocollo di acquisizione è stata prevista una fase di inizializzazione del test della camminata, necessaria per predisporre la strumentazione ad un'acquisizione corretta e ad agevolare la fase di post-processing. La fase di inizializzazione comprende i seguenti steps:

- *Calibrazione*, step di 15 secondi necessario a stabilire un corretto orientamento dei sensori rispetto al sistema di riferimento terrestre. I sensori vengono lasciati per tutta la durata della calibrazione su una superficie piana, al fine di definire il sistema di riferimento terrestre, sistema base al quale riferirsi. Il posizionamento dei sensori sul corpo del paziente, prima della fase di calibrazione, avrebbe comportato adottare come sistema di riferimento di base il corpo del soggetto, sopra il quale sarebbe stato posto il sensore. Da ciò il sistema di riferimento scelto sarebbe stato non univoco, poichè legato alla fisionomia del soggetto stesso.
- *Sincronizzazione*, step di 10 secondi necessario per fornire un segnale di input contemporaneo a tutti i sensori usati nella prova. Questa fase è necessaria poichè i sensori vengono accessi in maniera sequenziale e quindi non inizieranno ad acquisire dati nello stesso istante di tempo.
- *Silenzio*, step di 10 secondi necessario a discriminare con chiarezza le fasi adiacenti di sincronizzazione e posizionamento.
- *Posizionamento*, step di 30 secondi necessario per posizionare correttamente i sensori sul corpo del soggetto con le apposite fasce di velcro.
- *Acquisizione a riposo*, step di 30 secondi dove viene chiesto al soggetto di rimanere fermo in posizione eretta. Questa fase è necessaria per acquisire le caratteristiche del soggetto in esame in fase di quiete.
- *Acquisizione*, step di durata arbitraria dove viene chiesto al soggetto di muoversi liberamente e svolgere tutte le sue abituali attività.

La sequenza delle attività è schematizzata in figura 4.11.

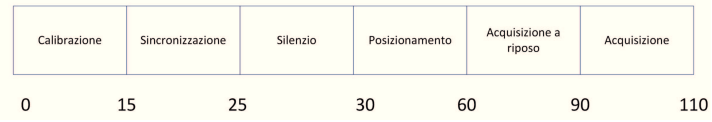

Figura 4.11: Steps fase di inizializzazione

4.7.1 Sensori IMU

Il dispositivo indossabile utilizzato è stato un MARG sensor, dotato di accelerometro, giroscopio e magnetometro triassiale. È stata utilizzato un sensore custom basato sul progetto del sensore NGIMU della x-io technologies. L'elaborazione del segnale e le interfacce grafiche sono state realizzate con il software Matlab. La frequenza di campionamento di ogni sensore è stata posta a 128Hz, al fine di ottenere una quantità di dati sufficiente da elaborare. Per i test del tremore sono stati posti due sensori su ambo i polsi del paziente. In figura 4.12 è possibile vedere le zone di posizionamento e l'orientazione dei sensori. Per

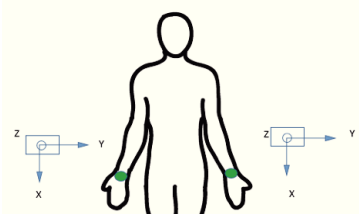

Figura 4.12: Posizionamento ed orientazione dei sensori per i test sul tremore RT, PT, KT e GT

i test sul freezing e sulla camminata sono stati posti tre sensori, due sul collo di ambo i piedi ed uno sul petto del soggetto. La figura 4.13 mostra l'orientazione e il posizionamento dei sensori. Nei test del tremore ogni sensore è stato

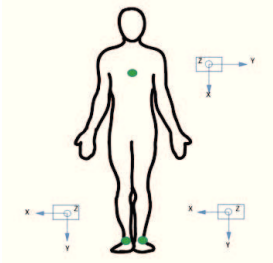

Figura 4.13: Posizionamento ed orientazione dei sensori per i test del freezing e della camminata

impostato con un fondo scala di ± 2g, necessario per ottenere una maggiore

sensibilità nella misura. Nei test del freezing e della camminata è stato scelto invece un fondo scala di ± 8g, per evitare saturazioni del sensore a causa dall'impatto del piede con il terreno.

4.8 Algoritmi di analisi dei sintomi del Parkinson

In questa sezione vengono illustrati gli algoritmi implementati per l'analisi del tremore, del freezing e delle fluttuazioni in pazienti affetti da malattia di Parkinson. Dopo l'esecuzione dei test descritti nel paragrafo precedente, i dati, acquisiti dai sensori e salvati su microSD interna, sono stati estratti ed opportunamente elaborati. Al fine di garantire un'assistenza del paziente continua, anche da remoto, il sistema proposto è stato integrato con il servizio di telemedicina WebRTC che, oltre alla condivisione di audio e video, è utilizzabile sia come servizio di file sharing, per l'invio di file acquisiti in precedenza, che per lo streaming real-time dei dati acquisiti dai sensori.

4.8.1 Analisi del tremore

L'algoritmo per l'analisi del tremore è basato su uno studio del segnale dell'accelerometro e giroscopio nel dominio della frequenza. Il diagramma di flusso dell'algoritmo realizzato è riportato in figura 4.14.

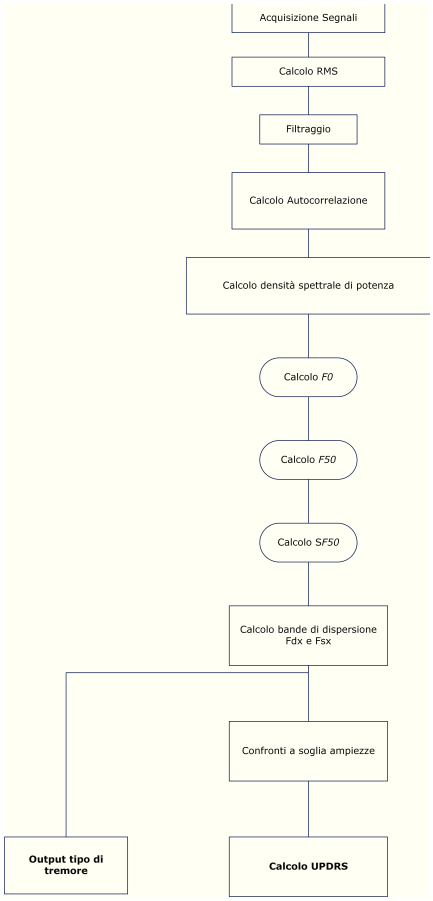

Figura 4.14: Diagramma di flusso algoritmo tremore

L'algoritmo sviluppato può essere sintetizzato nei seguenti steps:

1. Acquisizione dati provenienti dal MARG sensor.
2. Calcolo dell'RMS (Root Mean Square).
3. Filtraggio del segnale con filtri passa banda. Sono stati realizzati filtri con differenti frequenze di taglio al fine di identificare i diversi tipi di tremore, ognuno caratterizzato da una propria banda (tremore a riposo 3-6HZ, tremore posturale 6-9Hz e tremore cinetico 9-12Hz).
4. Calcolo della funzione di autocorrelazione. Il segnale viene finestrato e per ogni finestra viene calcolata l'autocorrelazione, mediante prodotto

della finestra con se stessa shiftata di una certa costante.

5. Calcolo della densità spettrale di potenza mediante trasformata di Fourier (FFT) del segnale autocorrelazione, ottenuto nel passaggio precedente.

6. Modulo della densità spettrale di potenza (PSD) e calcolo della frequenza con ampiezza massima (F0).

7. Calcolo della frequenza centrale F50, caratterizzata da metà della potenza alla sua sinistra e metà alla sua destra.

8. Calcolo della frequenza SF50, banda entro la quale è contenuto il 68% della potenza totale [120]. La frequenza SF50 è chiamata di dispersione ed indica la larghezza di banda del segnale. Poichè SF50 è centrata in F50, una dispersione stretta indica che la stima della severità del tremore è corretta o affetta da errore trascurabile, mentre in presenza di una banda molto ampia la stima sarà affetta da un errore non trascurabile.

9. Calcolo del limite di dispersione sinistro e destro come Fdx=F50 + FS50 e Fsx=F50-FS50.

10. Assegnazione del grado UPDRS mediante confronti a soglia con l'ampiezza del segnale elaborato.

Le figure 4.15 e 4.16 mostrano un esempio di dispersione della potenza del segnale, mentre la tabella 4.1 mostra le soglie utilizzate nell'algoritmo e ricavate da un'analisi empirica dei dati testati.

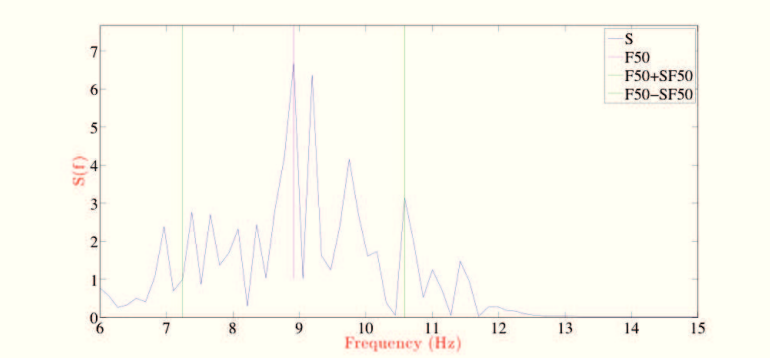

Figura 4.15: Esempio ampia dispersione

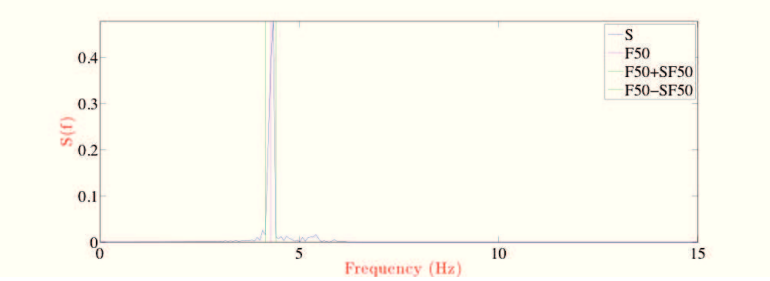

Figura 4.16: Esempio dispersione ridotta

Tabella 4.1: Correlazione grado UPDRS con intensità PSD

UPDRS	**PSD [W]**
0	<2
1	2 <I <35
2	35 <I <200
3	200 <I <300
4	I >300

Al medico è stata fornita una GUI, con algoritmo implementato, mediante la quale è possibile fare:

- Scelta del segnale di input tra accelerometro e giroscopio;
- Scelta del filtro;
- Visualizzazione andamento in frequenza del segnale;
- Visualizzazione tipo tremore e grado UPDRS;
- Salvataggio analisi su file PDF.

L'interfaccia grafica realizzata è mostrata in figura 4.17.

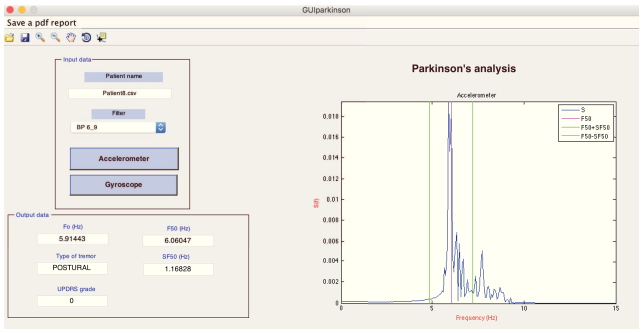

Figura 4.17: Interfaccia grafica per l'analisi del tremore

4.8.2 Analisi del freezing

Lo scopo dell'analisi del freezing è verificare la presenza di blocchi motori durante azioni di vita quotidiana come camminare, attraversare corridoi o porte. L'analisi viene condotta principalmente nel dominio della frequenza, dove è possibile distinguere gli eventi di freeze da quelli non freeze. Il diagramma di flusso dell'algoritmo è riportato in figura 4.18.

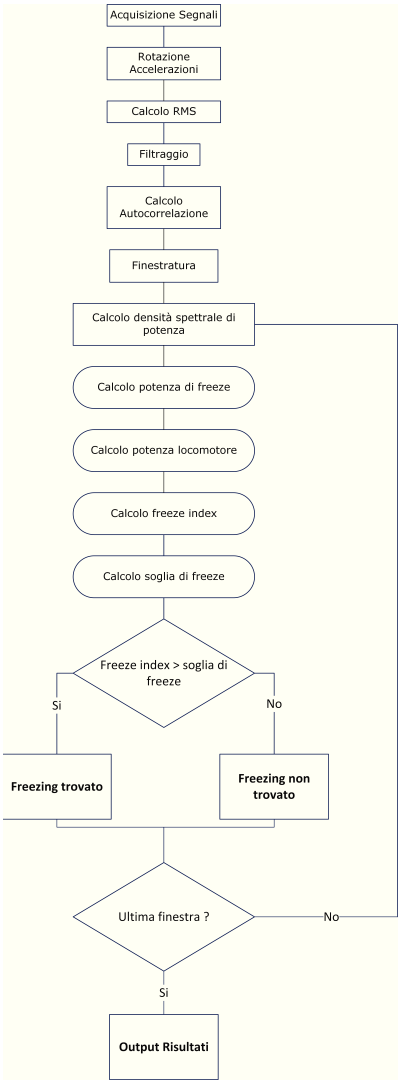

Figura 4.18: Diagramma di flusso algoritmo freezing

Gli steps che compongono l'algoritmo sono:

1. Acquisizione dati provenienti dal MARG sensor.

2. Rotazione delle accelerazioni acquisite nel sistema di riferimento del sensore, al sistema di riferimento terrestre e moltiplicazione per 9.81 m/s^2.

3. Calcolo dell'RMS (Root Mean Square).
4. Filtraggio dei segnali nella banda 0.5-8 Hz.
5. Calcolo della densità spettrale di potenza su finestre di 6 secondi. Il calcolo della PSD viene effettuato mediante la trasformata di Fourier della funzione di autocorrelazione, considerando un numero di campioni di overlap pari al 50% della lunghezza della finestra. La durata della finestra è stata posta a 6 secondi, poichè sufficiente a rilevare eventi di freeze di durata media 3 secondi [121].
6. Calcolo della potenza di freeze come densità spettrale di potenza nella banda 3-8 Hz su finestre di 6 secondi.
7. Calcolo della potenza di camminata nella banda locomotore 0.5-5 Hz su finestre di 6 secondi
8. Se la PSD di ogni finestra è compresa nell'intervallo 0.1-1.5 W, viene calcolato il freeze index come rapporto tra la potenza di freeze e la potenza di locomozione. Si sceglie l'intervallo 0.1-1.5 W poichè si è visto che in finestre con valori di potenza inferiori di 0.1 W il soggetto è fermo, mentre per valori superiori a 1.5 W il soggetto cammina regolarmente.
9. Calcolo in ogni finestra della soglia di freeze come la somma del relativo freeze index e della deviazione standard calcolata rispetto al freeze index medio di tutti i soggetti sottoposti al test [122].
10. Il calcolo dei blocchi motori, mediante un confronto a soglia, viene ottenuto verificando se il freeze index supera la relativa soglia. Il calcolo viene eseguito per ogni finestra.

L'algoritmo è stato fornito al team medico mediante apposita GUI dove è possibile calcolare:

- L'andamento della camminata nel tempo;
- Il numero di blocchi motori rilevati ed il loro istante temporale;
- Il valore del freeze index calcolato su ogni finestra;
- L'andamento in frequenza di un blocco motorio scelto;
- Un'esportazione del test eseguito in formato PDF.

L'interfaccia realizzata è mostrata nella figura 4.19.

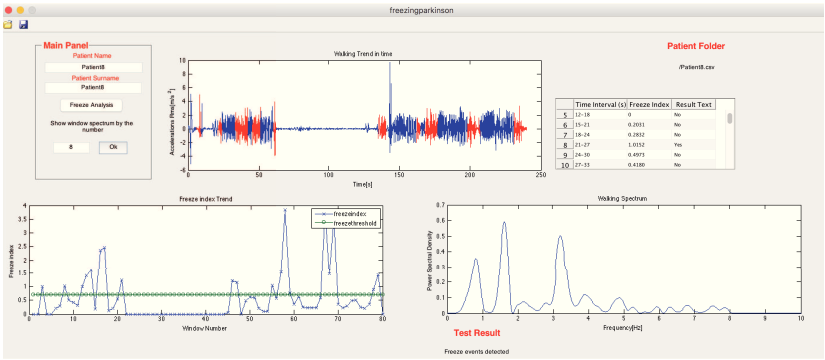

Figura 4.19: Interfaccia grafica per l'analisi del tremore

4.8.3 Analisi delle fluttuazioni

L'analisi delle fluttuazioni viene effettuata per verificare il passaggio da uno stato ON, corrispondente ad una quasi totale assenza dei sintomi della MP, ad uno stato OFF, dove invece i sintomi ricompaiono in maniera più o meno marcata. Le transizioni ON-OFF vengono valutate studiando il peggioramento di parametri caratteristici della camminata. Come definito nel protocollo di acquisizione, i parametri valutati sono stati:

- Il numero di passi in ogni fase dinamica;
- La lunghezza del passo [m];
- La velocità del passo [m/s];
- L'altezza del piede dal suolo [m];
- L'angolo di inclinazione del busto [◦];
- L'angolo di inclinazione della caviglia [◦].

I dati acquisiti dal test hanno bisogno di una prima fase di inizializzazione, necessaria sia per ruotare i segnali nel sistema di riferimento terrestre che per sincronizzarli ad uno stesso istante temporale, momento d'inizio dell'acquisizione vera e propria. Lo schema a blocchi dell'algoritmo è raffigurato in figura 4.20.

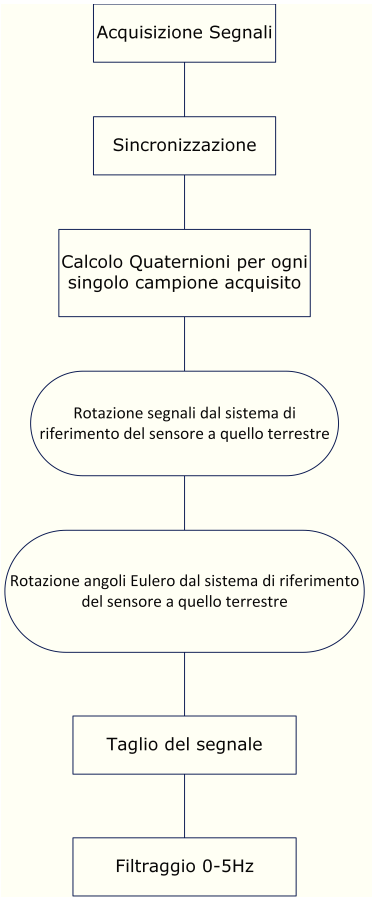

Figura 4.20: Diagramma di flusso fase di inizializzazione

Gli steps svolti sono:

1. Acquisizione dei dati provenienti dai tre MARG sensor;

2. Sincronizzazione dei dati acquisiti mediante allineamento dei campioni con massima ampiezza, individuati nella fase di sincronizzazione;

3. Calcolo dei quaternioni che rappresentano la rotazione dal sistema di riferimento del sensore al sistema di riferimento terrestre;

4. Rotazione dei segnali dal sistema di riferimento del sensore a quello terrestre;

5. Rotazione degli angoli di Eulero dal sistema di riferimento del sensore a quello terrestre;
6. Taglio del segnale per eliminare tutta la fase di inizializzazione;
7. Filtraggio del segnale con banda passante 0-5Hz per la rimozione del rumore.

Terminata la fase di inizializzazione, l'algoritmo per rilevare le fluttuazioni si può dividere in due parti: un primo algoritmo di classificazione, utile per discriminare le singole fasi di attività dinamica, un secondo algoritmo di analisi della camminata, necessario per estrarre i parametri del passo nelle fasi dinamiche.

Algoritmo di Classificazione L'algoritmo di classificazione, mediante l'uso del k-NN, è in grado di generare un modello capace di classificare le attività di nuovi dati, partendo da un set di dati di training in cui le attività sono note a priori. La prima fase dell'algoritmo è chiamata *Addestramento*, mentre la seconda fase è chiamata *Predizione*.
Gli step della fase di addestramento sono:

1. Acquisizione del set di dati di training. L'insieme sarà composto da acquisizione di molteplici pazienti, parkinsoniani e non, ai quali è stato chiesto di fare in maniera accurata e sequenziale delle attività note. Ogni attività è stata estratta dall'acquisizione e trattata separatamente.
2. Divisione delle singole attività in finestre di durata 3.2 secondi. La lunghezza della finestra è tale da contenere almeno un passo di soggetti parkinsoniani [123].
3. Calcolo in ogni finestra di features nel dominio del tempo e della frequenza. Le features nel dominio del tempo sono:
   - Media magnitudo dell'accelerazione X, Y e Z;
   - Media accelerazioni X, Y e Z;
   - Differenza media accelerazione assi X-Z e assi X-Y;
   - σ_{Ax}, σ_{Ay}, σ_{Az};
   - Correlazione accelerazione assi X-Y, Y-Z e X-Z;
   - Media magnitudo giroscopio X, Y e Z;
   - Massimo angolo di inclinazione rispetto all'asse verticale.

   Le features nel dominio della frequenza sono:
   - Valore del picco massimo della densità spettrale di potenza dell'accelerazione verticale e frequenza corrispondente;

- Valore di asimmetria della densità spettrale di potenza sui tre assi;
- Trasformata di Fourier del segnale accelerazione nei tre assi;
- Trasformata di Fourier del segnale giroscopio nei tre assi;
- Trasformata di Fourier del segnale acquisito dal magnetometro.

Le features sono state scelte sia dall'analisi dei segnali che dalla letteratura [124] [125] [126].

4. Calcolo del vettore di risposta numerico. Individuate le features per ogni attività, si genera un vettore di risposta in cui ogni elemento, identificato da un numero, rappresenta una singola attività.

5. Creazione del modello di classificazione con in input sia le features calcolate che il vettore di risposta.

Gli steps della fase di predizione sono:

1. Acquisizione dei dati dai sensori posti sul paziente in esame. Il soggetto non appartiene a quelli utilizzati per la creazione del training set;

2. Divisione dei segnali in finestre di 3.2 secondi e calcolo delle features sopra elencate;

3. Generazione del vettore di riposta, contenente l'attività individuata per ciascuna finestra, dall'algoritmo di predizione. L'algoritmo prende in input le features calcolate sui nuovi dati ed il modello calcolato nella fase di addestramento.

I passi dell'algoritmo di classificazione possono essere riassunti nel diagramma di flusso di figura 4.21.

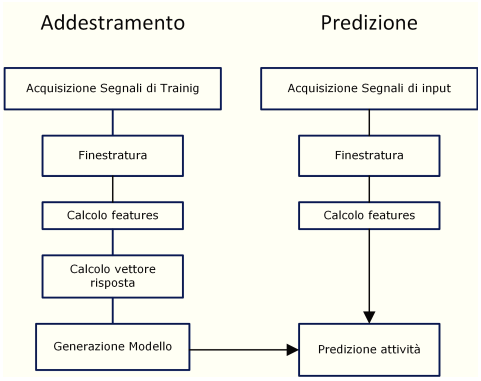

Figura 4.21: Diagramma di flusso algoritmo di classificazione

La predizione viene eseguita impostando k=10. L'intero è utilizzato per definire il numero di vicini da considerare, nell'intorno del valore da classificare, calcolando così la distanza euclidea. Tale distanza è calcolata tra i punti delle features dell'attività da determinare e i punti delle features delle attività note. L'appartenenza ad una attività piuttosto che ad un'altra, viene valutata in base alla molteplicità minima. Se si hanno punti che si riferiscono a diverse attività, si prende quello con distanza minima e molteplicità maggiore. Un esempio di rilevamento corretto di attività è mostrato in figura 4.22.

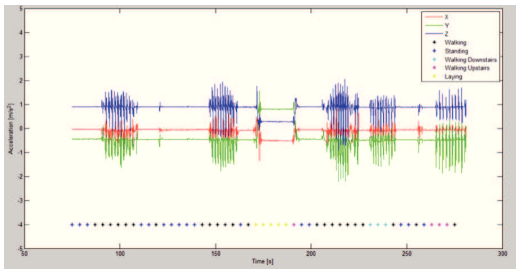

Figura 4.22: Esempio utilizzo algoritmo k-NN per la rilevazione delle attività

Algoritmo di analisi della camminata Una volta individuate le fasi dinamiche, sequenze dove il soggetto svolge attività motorie, è necessario individuare i singoli passi e le caratteristiche relative. Il diagramma di flusso degli step svolti è mostrato in figura 4.23.

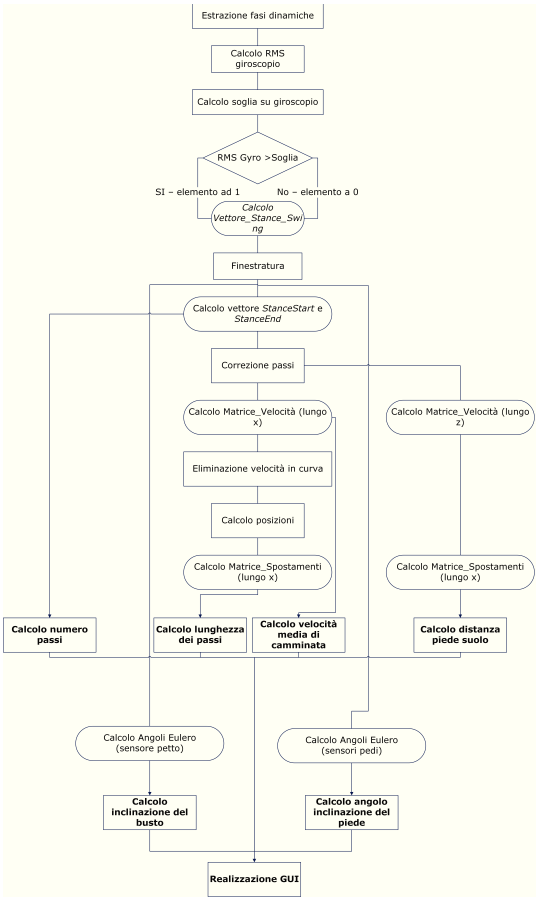

Figura 4.23: Diagramma di flusso dell'algoritmo di analisi delle fluttuazioni

I passi principali dell'algoritmo sono:

1. Elaborazione del segnale acquisito estraendo le sole fasi dinamiche.
2. Calcolo dell'RMS del giroscopio e relativo valore assoluto.
3. Calcolo di una soglia adattativa basata sul valore massimo dell'RMS del giroscopio e sulla media dei valori del giroscopio nelle fasi statiche.
4. Confronto tra RMS del giroscopio e soglia calcolata. Se il valore dell'RMS non supera la soglia si è in una fase di appoggio, altrimenti in una di oscillazione. Il confronto viene fatto campione per campione e i risultati saranno posti in un vettore, *"Vettore_Stance_Swing"*, che conterrà 1 per

fasi di appoggio e 0 per fasi di oscillazione. La figura 4.24 mostra la fasi del passo.

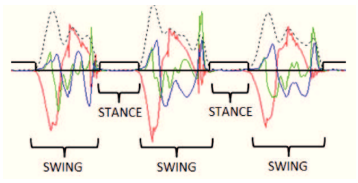

Figura 4.24: Calcolo fase di Stance e di Swing

5. Divisione del vettore con 1 e 0 in finestre di durata 3.2 secondi.

6. Calcolo degli StanceStart e StanceEnd mediante differenza dell'elemento *i+1* del $Vettore\_Stance\_Swing$ con l'elemento *i*. Se la differenza è 1 si è in fase di appoggio, trovando così lo StanceStart, se è -1 si è in fase di oscillazione e si determina lo StanceEnd.

7. Correzioni della durata dei singoli passi eliminando false rilevazioni dovute ad appoggi o oscillazioni troppo brevi.

8. Calcolo delle velocità dall'integrazione dei valori di accelerazione dei sensori posti sui piedi (*"Matrice_Velocità"*).

9. Eliminazione della velocità in curva.

10. Calcolo delle posizioni dei sensori dall'integrazione dei valori di velocità calcolati nei passi precedenti.

11. Calcolo degli spostamenti (*"Matrice_Spostamenti"*) come differenza tra la posizione *i+1* e *i* ottenute dai passi precedenti. La differenza viene fatta tra gli elementi di ogni finestra da 3.2 secondi.

12. Calcolo del **numero di passi** come numero di elementi ad 1 della matrice StanceEnd. Il calcolo viene fatto per ogni finestra da 3.2 secondi. Ogni elemento di StanceEnd ad 1 rappresenta una fase di *toe off*, ovvero distacco delle dita da terra. Per ottenere il numero di passi in ogni sequenza di camminata, si sommano tutti gli elementi ad 1, di tutte le finestre da 3.2 secondi contenute nella sequenza di camminata scelta. L'output totale si ha sommando tutti i passi di tutte le sequenze.

13. Calcolo della **lunghezza del passo** come differenza tra l'elemento *i+1* e l'elemento *i* dei valori della Matrice_Spostamenti. Le lunghezze del passo calcolate all'interno della finestra da 3.2 secondi considerata, vengono tra loro mediate per calcolare la lunghezza del passo media nella finestra

considerata. La media delle lunghezze del passo di tutte le finestre, all'interno di una sequenza dinamica considerata, determina la lunghezza media del passo in quella sequenza.

14. Si determina la **velocità media di camminata** dalla differenza tra l'elemento *i+1* e l'elemento *i* dei valori della Matrice_Velocità per finestre da 3.2 secondi. Si calcola la velocità media in ogni finestra da 3.2 secondi e poi in una sequenza di camminata

15. Si determina lo spostamento nella direzione verticale z del sensore posto sul piede (**distanza del piede dal suolo**) da una doppia integrazione dei dati di accelerazione lungo z. Si determina la media degli spostamenti in ogni finestra da 3.2 secondi (differenza elemento *i+1* ed elemento *i*) ottenendo la media degli spostamenti lungo z in una sequenza di camminata.

16. Calcolo delle medie dell'angolo roll nelle singole finestre di camminata. A sua volta la media dei valori di tutte le finestre (appartenenti ad una certa sequenza) viene utilizzata per trovare il valore medio dell'angolo roll nella sequenza di camminata considerata. L'angolo roll considerato è relativo al sensore posto sul petto grazie al quale è possibile calcolare l'**inclinazione del busto**.

17. Dal valore dell'angolo di pitch delle finestre di camminata si calcola il valore medio dell'angolo in ogni finestra da 3.2 secondi. Si mediano i valori di tutte le finestre (per una certa sequenza) trovando il valore medio dell'angolo pitch nella sequenza di camminata considerata. L'angolo pitch considerato è relativo ai sensori posti sul collo del piede in grado di fornire indicazioni sull'**angolo di inclinazione del piede**.

18. Realizzazione della GUI per la visualizzazione dei parametri calcolati.

L'interfaccia realizzata è mostrata nella figura 4.25.

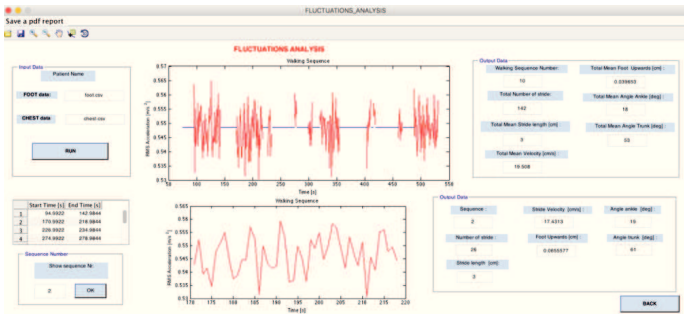

Figura 4.25: Interfaccia grafica fluttuazioni

Mediante GUI è possibile fare le seguenti operazioni:

- Visualizzare l'andamento medio dei parametri su tutta l'acquisizione;
- Visualizzare il numero di sequenze di camminata;
- visualizzare l'andamento medio dei parametri su una singola sequenza;
- Visualizzare l'andamento dell'RMS dell'accelerazione nella sequenza di camminata selezionata;
- Salvare il report in PDF.

4.8.4 Invio acquisizione con WebRTC

Un primo monitoraggio del paziente da remoto è stato reso possibile grazie all'utilizzo del sistema di teleconferenza WebRTC. L'idea nasce dal voler andare incontro ad esigenze di pazienti che, privi di mezzi di trasporto o troppo distanti dal presidio medico, sono impossibilitati nel raggiungere periodicamente, con frequenza anche settimanale, l'ambulatorio medico. Le frequenti visite, anche se di breve durata, sono finalizzate all'individuazione del corretto dosaggio del medicinale, tale da limitare al massimo la comparsa dei sintomi della malattia. In questi incontri il medico cerca di recepire dal paziente informazioni sul tipo di sintomo, frequenza di comparsa ed intensità riscontrata durante le giornate passate. Queste informazioni sono però spesso incomplete limitando la personalizzazione della terapia da parte del medico.
Si è così deciso di dotare il paziente di dispositivi indossabili custom, facili da utilizzare e che consentano un monitoraggio del paziente da remoto, anche real-time. Il paziente, opportunamente formato (o il familiare se le sue condizioni non lo consentano), è in grado di effettuare sia acquisizioni prolungate, anche di intere giornate con invio del file dati al medico, che acquisizioni brevi con monitoraggio remoto real-time. Nell'acquisizione real-time, grazie al Bluetooth presente di default nel sensore inerziale, viene collegato un Dongle Bluetooth al

computer del paziente o del familiare e mediante API DataChannel del servizio WebRTC, vengono inviati istantaneamente i dati acquisiti. Lato medico i dati vengono salvati su file di testo per poter esser pronti all'analisi mediante le GUI degli algoritmi sopra proposti.
Nell'analisi off-line il paziente indossa per tutta la giornata i sensori forniti e invece di inviare i dati in real-time, scarica dalla SD su pc il file con le acquisizioni, caricandolo in upload con il servizio WebRTC. Nello studio del Parkinson la tecnologia WebRTC è utilizzata principalmente come servizio di file sharing. Oltre alla trasmissione dei dati acquisiti, il servizio WebRTC è utilizzato anche come strumento di training, che grazie alla condivisione di audio e video, ha permesso al medico remoto di guidare il paziente nell'esecuzione corretta del test. Tutto ciò fornisce report immediati, inviati anche al paziente, senza necessità di visite ambulatoriali.
Un esempio di invio di file mediante servizio WebRTC è mostrato nella figura 4.26.

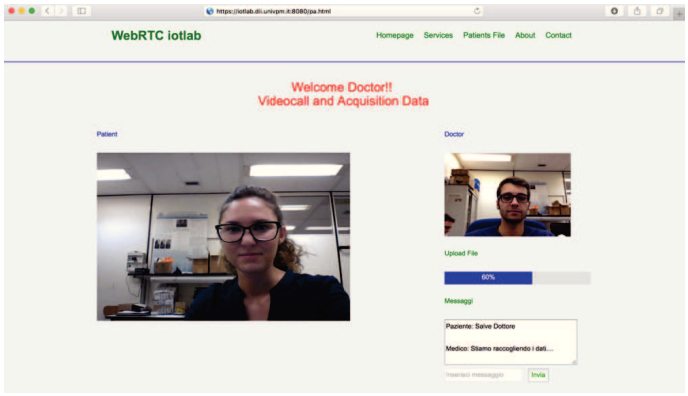

Figura 4.26: Session WebRTC come servizio di file sharing

4.9 Risultati

La bontà degli algoritmi è valutata comparando l'output degli algoritmi stessi con i report prodotti dall'analisi del medico curante. Di seguito vengono riportati i risultati ottenuti divisi per la tipologia di test effettuato.

4.9.1 Risultati tremore

Gli algoritmi realizzati sono stati testati su 30 pazienti affetti da diversi tipi di tremore: a riposo, posturale e cinetico. I risultati, sia sul tipo di tremore che sul livello UPDRS, sono stati confrontati con le valutazioni mediche basate

su scale UPDRS cartacee. Di seguito sono riportati i risultati di 10 pazienti sui 30 sottoposti al test. Le tabelle 4.2 e 4.3 mostrano i risultati per il test a riposo, sia per l'arto destro che per l'arto sinistro.

Tabella 4.2: Risultati algoritmo tremore, test RT, arto destro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	R	R
2	0	0	R	R
3	0	0	R	R
4	1	1	R	R
5	1	1	R	R
6	0	0	R	R
7	1	1	R	R
8	4	4	R	R
9	2	2	R	R
10	3	3	R	R

Tabella 4.3: Risultati algoritmo tremore, test RT, arto sinistro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	R	R
2	0	0	R	R
3	0	0	R	R
4	1	1	R	R
5	1	1	R	R
6	0	0	R	R
7	1	1	R	R
8	2	2	R	R
9	2	2	R	R
10	4	5	R	R

Da una comparazione del valore UPDRS calcolato dall'algoritmo e dal medico, si osserva una puntuale corrispondenza tra i risultati. Oltre al livello UPDRS anche il tipo di tremore è stato correttamente individuato dall'algoritmo, in corrispondenza con quello fornito dalla valutazione medica.
Le tabelle 4.4 e 4.5 sono invece inerenti ai test posturali, anche qui con distinzione tra arto destro e arto sinistro.

Tabella 4.4: Risultati algoritmo tremore, test PT, arto destro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	P	P
2	0	0	P	P
3	0	0	P	P
4	0	0	P	P
5	0	0	P	P
6	0	0	P	P
7	0	0	P	P
8	4	4	P	P
9	4	4	P	P
10	1	1	P	P

Tabella 4.5: Risultati algoritmo tremore, test PT, arto sinistro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	P	P
2	0	0	P	P
3	0	0	P	P
4	0	0	P	P
5	0	0	P	P
6	0	0	P	P
7	0	0	P	P
8	4	4	P	P
9	4	4	P	P
10	1	1	P	P

La tabella 4.4 mostra i risultati del test posturale fatto con il dispositivo inerziale posto sul polso destro mentre la 4.5 relativa al polso sinistro. Dai risultati ottenuti si nota come l'algoritmo è in grado di stabilire il grado di severità e il tipo di tremore per ogni test, con una corrispondenza puntuale tra risultati dell'algoritmo e risultati del medico.
Le tabelle 4.6 e 4.7 mostrano invece i risultati ottenuti per il test cinetico sia su arto destro che su arto sinistro.

Tabella 4.6: Risultati algoritmo tremore, test KT, arto destro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	K	K
2	0	0	K	K
3	0	0	K	K
4	1	1	K	K
5	1	1	K	K
6	0	0	K	K
7	1	1	K	K
8	4	4	K	K
9	1	1	K	K
10	2	2	K	K

Tabella 4.7: Risultati algoritmo tremore, test KT, arto sinistro

	UPDRS algoritmo	**UPDRS medico**	**Tremore algoritmo**	**Tremore medico**
1	0	0	K	K
2	0	0	K	K
3	1	0	K	K
4	1	1	K	K
5	1	1	K	K
6	0	0	K	K
7	1	1	K	K
8	4	3	K	K
9	1	1	K	K
10	2	1	K	K

Quasi tutti i valori forniti dall'algoritmo hanno una corrispondenza completa con i risultati ottenuti dalla valutazione medica, sia in termini di UPDRS che nel tipo di tremore rilevato. Nei test privi di corrispondenza assoluta tra le diagnosi del medico e dell'algoritmo (visibili in rosso nella tabella), il medico riesaminando le registrazioni video delle acquisizioni, ha modificato la sua diagnosi a favore di quella fornita dall'algoritmo.
Come ultima prova viene effettuato il test del bicchiere, finalizzato ad identificare l'insorgenza di eventi di tremore compiendo semplici gesti di vita quotidiana, come versare l'acqua da un bicchiere all'altro senza rovesciarne il contenuto. I risultati ottenuti sono stati confrontati con le valutazioni soggettive del medico fornendo informazioni sia sul tipo di tremore presente che sul relativo grado UPDRS. Le tabella 4.8 e 4.9 mostrano i risultati ottenuti per singolo arto esaminato.

Tabella 4.8: Risultati algoritmo tremore, test GT, arto destro

	UPDRS algoritmo (test GT)	**UPDRS algoritmo (test KT)**	**UPDRS medico**	**Tremore algoritmo (test GT)**	**Tremore algoritmo (test KT)**	**Tremore medico**
1	0	0	0	K	K	K
2	0	0	0	K	K	K
3	0	0	0	K	K	K
4	1	1	1	K	K	K
5	1	1	1	K	K	K
6	0	0	0	K	K	K
7	1	1	1	K	K	K
8	4	4	4	K	K	K
9	1	1	1	K	K	K
10	2	2	2	K	K	K

Tabella 4.9: Risultati algoritmo tremore, test GT, arto sinistro

	UPDRS algoritmo (test GT)	**UPDRS algoritmo (test KT)**	**UPDRS medico**	**Tremore algoritmo (test GT)**	**Tremore algoritmo (test KT)**	**Tremore medico**
1	0	0	0	K	K	K
2	0	0	0	K	K	K
3	1	1	0	K	K	K
4	1	1	1	K	K	K
5	1	1	0	K	K	K
6	0	0	0	K	K	K
7	1	1	1	K	K	K
8	4	4	4	K	K	K
9	1	1	1	K	K	K
10	2	2	2	K	K	K

Nel test del bicchiere i risultati ottenuti mostrano una totale corrispondenza sia con il test cinetico che con la valutazione medica, sia in termini di tipo di tremore che del relativo grado UPDRS.
Da un'analisi dei risultati si può ritenere l'algoritmo affidabile per essere usato come ausilio medico, nella valutazione del livello di tremore e della progressione della malattia di Parkinson. Inoltre, vista la semplicità dei test e la facilità di utilizzo del dispositivo inerziale, il sistema può esser particolarmente indicato per l'analisi in ambito domestico. I frequenti test sono in grado di verificare costantemente la progressione della MP.

4.9.2 Risultati freezing

I test sul freezing sono stati effettuati per verificare l'insorgenza di blocchi motori durante le attività di vita quotidiana. Le attività sono state simulate con tre prove: il TUG, l'attraversamento di un passaggio stretto e l'attraversamento di una porta. La bontà dell'algoritmo è stata valutata dividendo le acquisizioni di ogni prova in finestre di 10 secondi e confrontando i risultati dell'algoritmo (blocco rilevato o blocco non rilevato), con quelli individuati dal medico mediante analisi delle registrazioni video.
Considerando una generica finestra di osservazione, la valutazione delle prestazioni dell'algoritmo sono state effettuate considerando i seguenti parametri:

- *Vero Positivo (VP)*, individuazione del blocco sia dall'algoritmo che da un'analisi visiva del medico;
- *Falso Positivo (FP)*, individuazione del blocco solo dall'algoritmo e non dall'analisi medica;

- *Falso Negativo (FN)*, individuazione del blocco motorio solo dalla visita medica e non dall'algoritmo;
- *Vero Negativo (VN)*, mancata individuazione del blocco motorio sia dall'algoritmo che dall'analisi medica dei video.

La tabella 4.10 riassume i parametri sopra descjabrefritti. L'efficienza dell'al-

Tabella 4.10: Parametri usati per il calcolo di sensitività, specificità ed accuratezza

Output algoritmo	**Analisi medica**		**Totale**
	Blocco motorio	**Non blocco motorio**	
Blocco motorio	VP	FP	T+
Non blocco motorio	FN	VN	T-
	D+	D-	

goritmo è valutata calcolando la sensitività, la specificità e l'accuratezza. La sensitività è la probabilità che un blocco venga rilevato. Si calcola come rapporto tra il numero di finestre, in cui sia l'algoritmo che la valutazione medica hanno individuato un blocco, e il numero di finestre totali contenenti un blocco. La formula che descrive la sensitività è la 4.24.

$$S_e = \frac{VP}{D+} \tag{4.24}$$

La specificità è la probabilità che una finestra di camminata regolare venga identificata come blocco dall'algoritmo. Si calcola come rapporto tra il numero di finestre di camminata regolare, identificate sia dall'algoritmo che dall'analisi medica, ed il numero totale di finestre di non blocco. La formula che descrive la specificità è la 4.25.

$$S_p = \frac{VN}{D-} \tag{4.25}$$

L'accuratezza indica l'efficienza dell'algoritmo e si calcola come rapporto tra le finestre rilevate correttamente dall'algoritmo (sia di blocco motorio che di camminata regolare) ed il numero totale di finestre. La formula che descrive la accuratezza è la 4.26.

$$A_c = \frac{VP + VN}{N} \tag{4.26}$$

I risultati del test TUG sono mostrati in tabella 4.11. I risultati del test di attraversamento di un passaggio stretto sono mostrati in tabella 4.12. Infine come ultimo risultato vengono riportate le prestazioni ottenute dal test di attraversamento di una porta nella tabella 4.13. Da risultati ottenuti l'algoritmo riesce a valutare con elevata accuratezza la maggior parte dei blocchi motori individuati nei test effettuati. Da ciò si può affermare che il sistema proposto

Tabella 4.11: Risultati del test TUG

Parametro	*%*
Sensitività	100
Specificità	97.5
Accuratezza	98

Tabella 4.12: Risultato test di attraversamento di un passaggio stretto

Parametro	*%*
Sensitività	100
Specificità	96
Accuratezza	97

Tabella 4.13: Risultati test di attraversamento di una porta

Parametro	*%*
Sensitività	100
Specificità	98
Accuratezza	90

può essere utilizzato come strumento di ausilio al medico nella diagnosi delle insorgenze di blocchi motori al fine di personalizzare la terapia farmacologica e ridurre al minimo le loro manifestazioni.

4.9.3 Risultati fluttuazioni

L'analisi delle fluttuazioni è realizzata mediante due algoritmi, il primo chiamato di classificazione, necessario per identificare le singole attività, ed il secondo chiamato di analisi della camminata, necessario per il calcolo di parametri caratteristici delle fasi dinamiche di attività. Nell'algoritmo di classificazione sono stati scelti due insiemi di persone, il primo utilizzato come popolazione di training ed il secondo per la verifica delle prestazioni dell'algoritmo. L'insieme di training è stato formato da 40 soggetti, 20 sani e 20 affetti da Parkinson, di genere sia maschile che femminile. Ad ogni soggetto è stato chiesto di svolgere una sola volta, nel miglior modo possibile, le singole attività statiche e dinamiche ovvero, restare fermo in piedi, sedersi, distendersi, camminare, salire e scendere le scale. Ogni attività è stata utilizzata per il calcolo delle features necessarie alla creazione del modello di classificazione. La popolazione di test invece è stata composta da 42 soggetti affetti da MP, diversi dai precedenti e stessi utilizzati per l'analisi del tremore e del freezing. Ogni soggetto, equipaggiato con tre sensori disposti sul petto e sul collo di ambo i piedi, è stato lasciato libero di compiere le attività sopra elencate per almeno 120 minuti.

Nella tabella 4.14 sono riportate le percentuali di accuratezza dell'algoritmo nell'individuare ogni attività e le relative transizioni. Come si può vedere dalla

Tabella 4.14: Accuratezza nell'individuazione delle singole attività

Attività	**Accuratezza (%)**
Camminata	96
Salita Scale	91
Discesa Scale	94
Transizioni in piedi/seduto	100
Seduto	100
Transizioni seduto/in piedi	100
Transizioni in piedi/disteso	100
Distendersi	99
Transizioni disteso/in piedi	100
In piedi	90
Media Totale	97

tabella, l'algoritmo ha un elevato grado di efficienza nel distinguere le singole attività svolte da soggetti Parkinsoniani. L'accuratezza non totale in alcune identificazioni è dovuta sia al soggetto esaminato, che spesso ha una camminata lenta e con piede strisciato sul pavimento, che all'ambiente di misura non omogeneo con scale di altezza e numero diverso.
L'algoritmo delle fluttuazioni ha permesso di identificare i principali parametri della camminata e del passo. La bontà di ogni parametro ottenuto è stata verificata con test manuali. Inizialmente sono state poste sul pavimento due linee, una di inizio e l'altra di fine test, distanti tra loro 10 metri. Ad un soggetto test non parkinsoniano è stato chiesto, partendo con ambo i piedi sulla linea di partenza ed in fase di stance, di percorrere il tratto ed arrivare alla linea di fine test con andatura costante. Il test è stato cronometrato e si è concluso non appena entrambi i piedi si sono posizionati in fase di stance sulla linea di fine corsa. Sul soggetto test sono stati posizionati tre dispositivi inerziali, uno sul petto e due sul collo di ambo i piedi. La distanza percorsa dal soggetto, rilevata dall'algoritmo, è stata confrontata con la distanza tra le due linee (10 metri) riportando un errore di 12 cm. Durante il test sono stati contati i passi compiuti dal soggetto che, confrontati con quelli calcolati dall'algoritmo, ha individuato la differenza di 1 solo passo. Conoscendo la distanza percorsa ed il tempo impiegato è stata manualmente calcolata la velocità di marcia che, rispetto a quella ottenuta dall'algoritmo, distava 0.2 m/s. Durante ogni fase di swing è stato chiesto al soggetto di alzare da terra i piedi di 15 cm. L'algoritmo è stato in grado di calcolare una distanza del piede da terra di 18cm. Il calcolo degli angoli di inclinazione della schiena e dei piedi è stato fatto senza percorrere il tragitto, ma partendo dalla stazione eretta. Al soggetto è stato chiesto

di inclinarsi di un angolo prefissato. L'angolo misurato con la strumentazione è stato confrontato con quello in output dall'algoritmo verificando una differenza di 3.8°. La stessa operazione è stata fatta per ottenere l'inclinazione del piede, dove però la misura del sistema inerziale differisce dal valore misurato di 1.6°. Le tabelle 4.15, 4.16, 4.17 e 4.18 mostrano i parametri calcolati su un soggetto parkinsoniano sottoposto al test.

Tabella 4.15: Parametri della camminata calcolati nei primi minuti di fasi dinamiche 1

	Intervallo temporale (s)	$N°$ **passi**	**Lunghezza passo (cm)**	**Velocità passo (cm/s)**
1	64-72	6	1.5	9
2	72-80	6	1	9
3	80-88	5	1.6	6
4	88-96	6	1.5	8

Tabella 4.16: Parametri della camminata calcolati nei primi minuti di fasi dinamiche 2

	Spostamento piede verso l'alto (cm)	**Angolo inclinazione caviglia** (°)	**Angolo inclinazione busto** (°)
1	0.2	33	34
2	0.6	34	25
3	1	33	36
4	0.8	33	34

Tabella 4.17: Parametri della camminata dopo circa 60 minuti di attività dinamica 1

	Intervallo temporale (s)	$N°$ **passi**	**Lunghezza passo (cm)**	**Velocità passo (cm/s)**
801	6408-6416	3	0.5	4
802	6416-6424	3	0.4	5
803	6424-6432	2	0.3	3
804	6432-6440	2	0.4	4

Tabella 4.18: Parametri della camminata dopo circa 60 minuti di attività dinamica 2

	Spostamento piede verso l'alto (cm)	**Angolo inclinazione caviglia** (°)	**Angolo inclinazione busto** (°)
801	0.2	15	38
802	0.3	18	39
803	0.2	12	38
804	0.4	16	38

Le quattro tabelle rappresentano i parametri della camminata, ottenuti in due momenti temporali diversi. Le prime due sono relative ad una delle prime fasi dinamiche di attività che dista dall'assunzione del farmaco 60 minuti (picco massimo di efficacia del farmaco). In questo arco temporale il soggetto assume un'andatura pressochè normale. Le seconde due tabelle invece sono relative ad una fase dinamica che dista più di 120 minuti dall'assunzione del farmaco. Rispetto alle prime tabelle vi è un calo generale nei parametri che descrivono l'andamento motorio del soggetto. Il paziente rallenta la velocità di marcia e la lunghezza del singolo passo. Inoltre ha il busto leggermente più flesso in avanti e il piede subisce una minor rotazione della caviglia, dovuta all'inizio della fase di strisciamento del piede al suolo.
Questo algoritmo è uno strumento di ausilio al medico nella valutazione del peggioramento motorio del soggetto durante l'arco della giornata. Può essere utilizzato anche nella valutazione della perdita di efficacia della terapia farmacologica e quindi il conseguente inizio della fase OFF. Lo studio delle varie attività e repentine variazioni dell'andamento motorio del soggetto, può essere usate nella personalizzazione della terapia farmacologica, ovvero individuare l'istante esatto di somministrazione della nuova dose del farmaco al fine di evitare la comparsa di fasi OFF.

4.10 Conclusioni

In questo capitolo è stato presentato un sistema di supporto al team medico per la valutazione dei sintomi della malattia di Parkinson e per la personalizzazione della terapia farmacologica. Nel paziente affetto da MP, la scelta della giusta dose di farmaci e del momento esatto di somministrazione è un'operazione assai complessa, effettuata unicamente da personale medico, il cui scopo è quello di limitare al massimo la comparsa di sintomi come tremore e freezing. Le valutazioni sono soggettive, poichè basate sia su questionari UPDRS, standard mondiale di valutazione del paziente parkinsoniano, che su diari giornalieri, compilati dal paziente stesso. Da qui l'esigenza di affiancare alla

valutazione medica soggettiva una valutazione oggettiva basata su strumenti automatici. Gli strumenti utilizzati fanno parte della categoria dei wearable device, dispositivi indossabili che, grazie all'elevata portabilità, basso costo, facilità di utilizzo ed elevate prestazioni, permettono di acquisire informazioni sui movimenti e sulle attività motorie dei soggetti in esame. Mediante algoritmi di signal processing e data-fusion, i dati acquisiti sono stati elaborati per ottenere informazioni sul tipo di evento parkinsoniano, verificatosi sia nell'arco della giornata che durante l'esecuzione del test. In ogni soggetto sono stati posti al massimo tre sensori, fissati con fasce in velcro, sulla zona toracica, sul collo di ambo i piedi e su polsi. I test effettuati hanno avuto lo scopo di valutare l'insorgenza di episodi di tremore, freezing e caratterizzare le fluttuazioni motorie.
Nello studio del tremore è stata realizzata una GUI che fornisce con precisione il livello UPDRS e il tipo di tremore da cui è affetto il soggetto. Dai dati ottenuti c'è una totale corrispondenza tra i risultati forniti dall'algoritmo e quelli ottenuti da una valutazione medica soggettiva. In alcuni casi, quando le valutazioni non erano concordi, il medico ha riesaminato il soggetto e confermato i risultati dell'algoritmo, cambiando così la sua diagnosi.
Nello studio del freezing l'algoritmo con annessa GUI ha permesso l'individuazione di blocchi di freezing sia durante attività di vita quotidiana che nelle sessioni di test. I dati prodotti sono stati confrontati con valutazioni mediche, basate sulle registrazioni video delle prove, confermando l'elevata efficienza dell'algoritmo nell'individuare eventi di freeze.
In ultimo si è andati a verificare la comparsa di fluttuazioni motorie nell'arco della giornata, segno di una perdita di efficacia del farmaco assunto. Come primo step, al soggetto è stato chiesto di svolgere azioni di vita quotidiana e gli algoritmi sviluppati hanno identificato con precisione le singole attività svolte. Successivamente, nelle sole fasi dinamiche è stato caratterizzato il passo andando a calcolare parametri notevoli come numero di passi, lunghezza e velocità del passo, distanza del piede da terra, angolo di inclinazione del busto e del piede. I valori di questi parametri, conformi a quelli reali dal confronto con prove manuali, sono stati ottenuti da acquisizioni superiori alle due ore, valutando come con il passare del tempo, allontanandosi dal momento di assunzione del farmaco, i parametri della camminata tendevano a peggiorare, segno di una perdita di efficacia del farmaco assunto in precedenza.
Le GUI sviluppate, fornite ai medici per valutazioni ambulatoriali dello stato dei propri pazienti, sono state abbinate al sistema WebRTC che ha permesso sia l'invio da remoto dei file acquisiti dai sensori, che l'invio real-time dei singoli campioni. Tale soluzione ha permesso una valutazione medica da remoto, particolarmente utile per soggetti con difficoltà nel raggiungere la clinica medica per visite ambulatoriali.

Capitolo 5

Conclusioni

L'obiettivo di questo lavoro ha riguardato lo studio e lo sviluppo di un sistema di telemedicina basato su tecnologia WebRTC. Il sistema può essere utilizzato in contesti più svariati, dall'ambito medico a quello sportivo. In questo lavoro vengono riportati due esempi di utilizzo dove, grazie all'uso di questa tecnologia, è possibile instaurare una sessione di videoconferenza con un team di supporto remoto per l'invio di informazioni sul grado di rischio embolico nei subacquei, o acquisizioni di sensori inerziali, utili alla valutazione dei sintomi della malattia di Parkinson. Il WebRTC, definito come una collezione di API, standard e protocolli, permette una comunicazione peer-to-peer tra paziente e medico remoto, in grado di scambiarsi contenuto audio, video e dati. Lo stile architetturale peer-to-peer, opposto al classico client-server, permette di ridurre i tempi di latenza, ridurre congestioni unitamente a maggiori garanzie di sicurezza e riservatezza dei dati rispetto a quest'ultima. Inizialmente il lavoro si è incentrato sullo studio di protocolli di comunicazione, usati soprattutto nella fase iniziale di Signaling, per lo scambio di informazioni di configurazione del sistema stesso. Partendo da tale studio sono state sviluppate applicazioni client-side e server-side per lo scambio real-time di audio-video mediante le API MediaStream, RTCPeerConnection e RTCDataChannel. Rispetto ad altri sistemi commerciali la soluzione proposta nasce con l'idea di scambiare in real-time dati acquisiti da dispositivi biomedicali esterni. Poichè WebRTC è utilizzabile solo mediante browser, che ad oggi non permette il dialogo con risorse locali, è stato necessario sviluppare applicazioni Crhome dedicate, piuttosto che l'utilizzo di estensioni browser per l'invio dei dati acquisti dalla porta seriale, risorsa locale alla quale è collegato il dispositivo esterno. Per poter sviluppare e testare soluzioni di trasmissione real-time di dati biomedicali, abbinati allo streaming audio-video, si sono realizzate soluzioni prototipali di dispositivi biomedicali, mediante schede di sviluppo dedicate, in grado, ad esempio, di acquisire il segnale ECG dal soggetto in esame. L'utilizzo di dispositivi sviluppati ad hoc, in alternativa a quelli disponibili sul mercato, è stato necessario al fine di poter efficientare i protocolli di comunicazione e di pacchettizzazione dei dati acquisiti, facendo variare, ad esempio, le frequenze di campionamento, le

sensibilità e le precisioni degli strumenti. Tali sistemi sono stati quindi testati in laboratorio per valutarne le prestazioni, variando il numero di campioni per pacchetto e la durata della comunicazione. I risultati ottenuti, in termini di ritardo trasmissivo e di perdita di pacchetti, hanno dimostrato le buone potenzialità del sistema anche per trasmissioni prolungate e con pacchetti di grandi dimensioni. Una delle peculiarità del sistema proposto è stata la semplicità di utilizzo, non richiedendo l'installazione di software client addizionale, essendo necessario solo un classico browser Internet. Il sistema è stato sviluppato in collaborazione con il DAN-Europe, ente europeo di tutela della sicurezza dei subacquei e ad oggi è utilizzato per l'assistenza remota a subacquei in fase di post-emersione. La collaborazione con il DAN ha permesso la realizzazione di un algoritmo di rilevazione delle bolle gassose in segnali eco Doppler audio. L'algoritmo realizzato ha prestazioni notevolmente superiori rispetto a soluzioni analoghe proposte in letteratura, dimostrandosi in grado di individuare tutte quelle bolle e relativi shower che circolano nel flusso sanguigno e vengono tipicamente rilevati, mediante semplice ascolto, da medici iperbarici altamente specializzati. Tutto ciò è stato altamente avvalorato grazie alla standardizzazione di un ben definito protocollo di acquisizione che, tra l'altro, contempla la scelta della regione precordiale come sito di monitoraggio. In questa regione una registrazione audio permette di acquisire tutte le bolle circolanti rispetto ad altre zone di monitoraggio, dove la maggior parte delle bolle vengono intrappolate dall'effetto filtrante dei polmoni. L'algoritmo ha permesso una diretta correlazione tra numero di bolle acquisite e scala di Spencer, o sua versione estesa, che definisce la terapia a cui sottoporre il subacqueo (esempio somministrazione di ossigeno, camera iperbarica o altro). I risultati prodotti in termini di numero di bolle e livello di Spencer sono stati confrontati con annotazioni manuali prodotte da tre blind teams di medici esperti. Poichè le modalità di annotazione manuale non risultavano in alcun modo definite e standardizzate, ciò ha reso difficoltoso l'utilizzo delle annotazioni prodotte senza la definizione di una precisa metodica. Per tale motivo, si è reso necessario anche lo sviluppo parallelo di un tool software in grado di rimuovere gli errori soggettivi prodotti dagli esaminatori, realizzare la sincronizzazione dei tempi di risposta e standardizzare la struttura dei report ottenuti. Tale software è stato testato da numerosi medici iperbarici e ad oggi è utilizzato dal DAN sia per le annotazioni manuali che come strumento di training. Dai risultati prodotti dall'algoritmo, su più di 200 file testati, vengono rilevate con precisione tutte le bolle presenti nei file e il loro istante di comparsa che corrisponde a quello fornito dai blind teams nei report manuali. Infine, l'algoritmo, per la rilevazione automatica delle bolle e la determinazione del rischio embolico secondo la ESS, è stato implementato su scheda embedded. Tale dispositivo permette inoltre di inviare al team remoto di supporto, tramite opportune interfacce di comunicazione, sia

i report degli eventi e del livello di ESS, che lo streaming del file eco Doppler mediante il servizio WebRTC.
Il secondo studio proposto è un'attività realizzata in collaborazione con l'Istituto di Riabilitazione S. Stefano di Porto Potenza Picena e con la clinica di Neurologia degli Ospedale Riuniti Marche Nord di Pesaro. In questo lavoro sono stati sviluppati e testati algoritmi in grado di identificare episodi di tremore e freezing in soggetti affetti da malattia di Parkinson. Gli algoritmi di data-fusion realizzati, proposti mediante interfacce grafiche, partono dai dati grezzi acquisiti da dispositivi inerziali e forniscono report sui sintomi della malattia e sulle loro fluttuazioni nell'arco della giornata. Uno specifico algoritmo di analisi delle caratteristiche della camminata è stato sviluppato per identificare la perdita di efficacia della terapia farmacologica, mano a mano che ci si allontana dal periodo di assunzione del farmaco stesso. L'algoritmo ha permesso una correlazione diretta tra istante di comparsa dei sintomi e momento di assunzione del farmaco e quindi ha fornito al neurologo dati affidabili per la personalizzazione della terapia, cosìcome richiesto dalla patologia in esame. Il sistema WebRTC è stato ancora una volta utilizzato dal medico per monitorizzare la comparsa dei sintomi della malattia da remoto. Da un'analisi delle prestazioni degli algoritmi realizzati, i sistemi sono in grado di identificare con precisione i tipi di tremore, i livelli UPDRS, scala usata a livello mondiale per la valutazione della progressione della malattia di Parkinson, i blocchi motori, che appaiono sia duranti i test in laboratorio che nelle attività di vita quotidiana, e valutare la comparsa di fluttuazioni motorie mediante calcolo di parametri notevoli del passo e della camminata in generale. Gli output degli algoritmi sono stati confrontati con quelli prodotti da valutazioni soggettive del medico, evidenziando una totale corrispondenza dei risultati. Nei rari casi in cui gli algoritmi hanno fornito un dato o evento non concorde alla valutazione medica, il medico da un riesame dei video del soggetto, ha cambiato la sua valutazione a favore di quella fornita dagli algoritmi.

Capitolo 6

Appendice

6.1 Trasformata Wavelet

Le Wavelet sono delle funzioni matematiche utilizzate per decomporre un dato segnale in diverse componenti in frequenza, per poi studiare ciascuna di tali componenti con una risoluzione pari alla sua scala. Le Wavelet vengono definite come copie traslate e scalate (dette anche Wavelet figlie - "ondine") di una forma d'onda oscillante di lunghezza finita o che decade velocemente (Wavelet madre). La trasformata Wavelet continua di un generico segnale f(t) è espressa dalla funzione di due variabili 6.1, con $\psi_{s,\tau}$ definito dalla formula 6.2.

$$\gamma(s,\tau) = \int_{-\infty}^{+\infty} f(t)\psi^*_{s,\tau}(t)dt \tag{6.1}$$

$$\psi_{s,\psi}(t) = \frac{1}{\sqrt{s}}\psi(\frac{t-\tau}{s}) \tag{6.2}$$

La funzione $\psi(t)$ è detta Wavelet madre, s è il fattore di scala (s>1 corrisponde ad una dilatazione), τ è il fattore di traslazione e $\frac{1}{\sqrt{s}}$ serve per preservare la normalizzazione. In breve la trasformata continua è il prodotto scalare tra f(t) e la Wavelet traslata e dilatata. Il risultato che si ottiene dalla trasformata Wavelet è una matrice contenente valori, che indicano l'intensità di una certa frequenza in un certo tempo. Per poter ottenere questa matrice i principali algoritmi eseguono il prodotto tra la funzione in ingresso e la Wavelet figlia al tempo 0. L'ondina viene così traslata di un tempo τ e viene eseguito nuovamente il prodotto tra essa e la funzione in ingresso. Le operazioni si concludono quando si arriva alla fine della funzione di ingresso. Così facendo si riempie la prima riga della matrice contenente la soluzione. Dopo tale operazione l'ondina viene ri-scalata con una scala maggiore s e viene ripetuto il procedimento precedente. In tal modo si riempie la seconda riga della matrice soluzione. Il numero di riga indica la scala, mentre il numero di colonna il tempo. Il procedimento si ripete iterativamente fino ad arrivare alla scala massima. La trasformata Wavelet continua è ridondante poichè la base, costituita dalla Wavelet dilatata e traslata

in modo continuo, contiene molti più elementi del necessario; inoltre tale tipo di trasformazione è poco adatta ad essere implementata dal calcolatore. Per tale motivo la trasformata viene eseguita su un insieme di Wavelet numerabili, ottenuto selezionando tra le traslazioni e le dilatazioni possibili soltanto quelle definite dalla formula 6.3.

$$\psi_{j,k}(t) = \frac{1}{\sqrt{s_0^j}}\psi(\frac{t - k\tau_0 s_0^j}{s_0^j}) \tag{6.3}$$

Nella formula j e k sono interi, il fattore di traslazione s_0 viene posto maggiore di uno ed il fattore dilatazione τ_0 viene scelto dipendente da s_0. Tale tipo di trasformata prende il nome di Wavelet discreta (DWT), molto utile quando ci troviamo davanti a un segnale quantizzato, cioè conosciuto esclusivamente in precisi istanti di tempo. Il processo di calcolo della DWT, che viene ottenuto con prodotti scalari, ad ogni suo stadio può essere semplificato con operazioni di filtraggio. Le figure 6.1, 6.2, 6.3, 6.4 e 6.5 mostrano esempi di "Mother Wavelet".

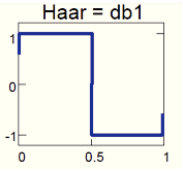

Figura 6.1: Wavelet madre Haar *(wikipedia.org)*

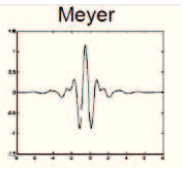

Figura 6.2: Wavelet madre Meyer *(wikipedia.org)*

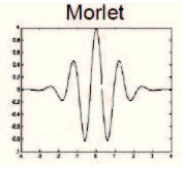

Figura 6.3: Wavelet madre Morlet *(wikipedia.org)*

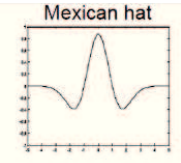

Figura 6.4: : Wavelet madre Mexican hat *(wikipedia.org)*

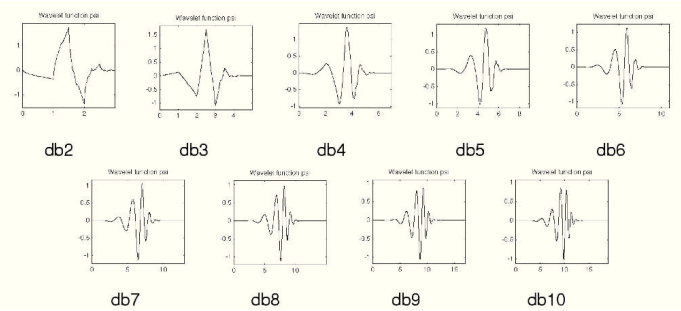

Figura 6.5: Wavelet madre Daubechies *(wikipedia.org)*

I filtraggi necessari sono:

- Filtro passa basso(averaging), per l'approssimazione del segnale;
- Filtro passa alto(differencing), per il mantenimento dei dettagli del segnale;
- Filtro di decimazione(downsampling), per la riduzione della frequenza di campionamento di metà.

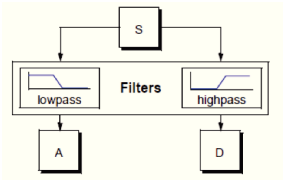

Figura 6.6: Applicazione filtri per il calcolo della Wavelet *(wikipedia.org)*

Come illustrato in figura 6.6, il segnale originale S passa attraverso due filtri complementari e da essi si ottengono le *approssimazioni-A* (in uscita dal filtro passa basso) e i *dettagli-D* (in uscita dal filtro passa alto). Per evitare che la lunghezza totale dei campioni provenienti dal ricongiungimento delle approssimazioni e dei dettagli sia doppia rispetto alla lunghezza del segnale originale,

si pone all'uscita di ognuno dei due filtri un blocco di *downsampling*. Nel fare questo si introduce però *aliasing* nelle componenti del segnale. Il processo di

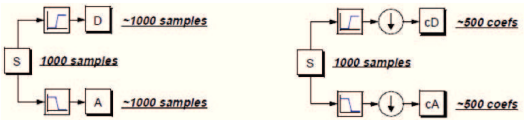

Figura 6.7: Calcolo della DWT mediante applicazione filtri passa alto e passa basso con l'aggiunta di downsampling *(wikipedia.org)*

decomposizione di figura 6.7 può essere iterato, realizzando successive decomposizioni delle approssimazioni ottenute al passo precedente. Questo schema di decomposizione è chiamato *"Albero di decomposizione Wavelet"*. Ad ogni decomposizione corrisponde un dimezzamento dei dati (downsampling) in base al tempo di campionamento Ts. Procedendo per dimezzamenti successivi della scala, si ottiene una ***Decomposizione Multilivello*** del segnale originale in successivi dettagli ed approssimazioni. Ad ogni dimezzamento della finestra i dati disponibili si dimezzano. Se la finestra conteneva N campioni la decomposizione di primo livello ne conterrà N/2, quella di secondo livello N/4, quella di terzo N/8 e così via. La figura 6.8 rappresenta la decomposizione multilivello. La procedura inversa, che permette di ricostruire il segnale originale dai suoi

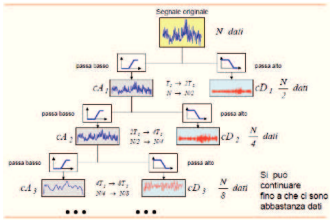

Figura 6.8: Decomposizione multilivello della Wavelet *(wikipedia.org)*

coefficienti, è chiamata Trasformata Wavelet Discreta Inversa (IDWT). Questa trasformata richiede però un sovra campionamento per l'allungamento del segnale con l'aggiunta di zeri tra i campioni e il successivo filtraggio. La scelta di filtri è cruciale nel raggiungimento della perfetta ricostruzione del segnale originale. Una perfetta ricostruzione è infatti qualcosa di possibile, nonostante l'aliasing introdotto dal downsampling delle componenti del segnale, basta scegliere attentamente i filtri per la decomposizione e per la ricostruzione, i quali sono strettamente correlati tra di loro. Perchè ciò avvenga, i filtri di decomposizione passa-basso e passa-alto, insieme con i loro filtri di ricostruzione associati, dovranno formare un sistema generalmente indicato come *"quadrature mirror filters-QMF"*.

6.2 Empirical Mode Decomposition

L'"Empirical Mode Decomposition" (EMD) è un metodo di decomposizione intuitivo, diretto e adattativo, che lavora direttamente nello spazio temporale con una base derivata dai dati stessi (a posteriori). La decomposizione ha un semplice e intuitivo assunto, ovvero, ad ogni istante temporale nel segnale analizzato possono coesistere differenti modalità oscillatorie, aventi frequenza anche significativamente diversa. Ogni componente oscillatoria viene definita attraverso una funzione di modalità intrinseca (IMF), la quale deve soddisfare le seguenti condizioni [65]:

1. Nell'intero segnale il numero di estremi e il numero di volte, in cui la funzione si annulla, debbono coincidere o al massimo differire di uno. Quindi tutti i massimi dovranno trovarsi sopra l'asse temporale e tutti i minimi sotto.

2. Per ogni punto il valor medio tra la funzione di inviluppo definita dai massimi locali e la funzione di inviluppo definita dai minimi locali dovrà essere zero, l'IMF dovrà quindi essere simmetrica.

Soddisfare la condizione 2 risulta tuttavia impossibile, nella pratica sarà sufficiente assumere che il valor medio dei due inviluppi sia circa zero. Il procedimento di estrazione delle IMF viene detto "sifting process". Consideriamo un segnale di partenza x(t), di andamento come quello della figura 6.9. La prima

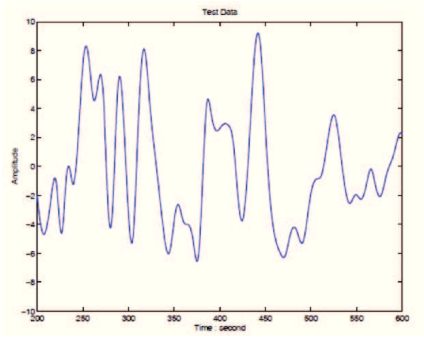

Figura 6.9: Segnale di input x per il calcolo della EMD *(articolo Norden E Huang)*

operazione, che si esegue, è l'identificazione di tutti i suoi estremi locali (massimi e minimi). Successivamente i massimi appena trovati, connessi attraverso una spline cubica, vanno a formare l'inviluppo superiore. La stessa cosa viene fatta con i minimi per trovare l'inviluppo inferiore. Trovati i due inviluppi, si procede con la verifica delle due condizioni di cui sopra per poter affermare

che la IMF trovata sia a tutti gli effetti la IMF giusta. A questo punto viene calcolata la media "m" degli inviluppi dove *inv_min* e *inv_max* rappresentano rispettivamente l'inviluppo inferiore e l'inviluppo superiore come descritto dalla formula 6.4.

$$m = (inv\_min(t) + inv\_max(t))/2 \tag{6.4}$$

La differenza tra il segnale d'ingresso x(t) e la media m è la prima, possibile, IMF estratta e verrà identificata con $h_1(t)$. Teoricamente la funzione $h_1(t)$

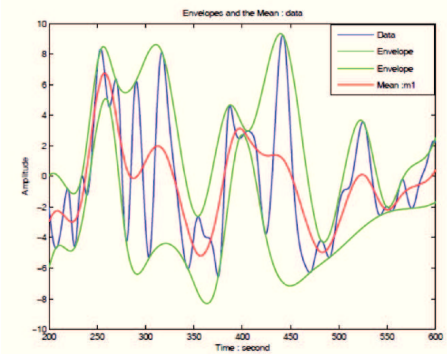

Figura 6.10: Segnale originale (in blu), inviluppo superiore ed inferiore (in verde) e media dei due inviluppi (in rosso) *(articolo Norden E Huang)*

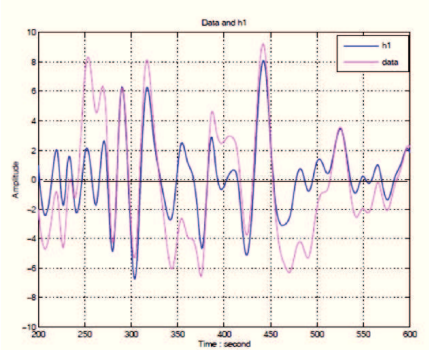

Figura 6.11: Segnale originale (in rosa) e prima IMF estratta (in blu) *(articolo Norden E Huang)*

dovrebbe già essere una IMF, in realtà però il segnale risultante avrà ancora massimi negativi e minimi positivi. Questo perchè la media calcolata con gli inviluppi non corrisponde perfettamente alla media reale, di conseguenza si possono generare nuovi massimi e alcuni già esistenti possono cambiare posizione o

valore. Osservando la figura 6.11 si nota che la $h_1(t)$ non è una IMF perchè, oltre a non esser una funzione simmetrica, tra i 500 e i 550 secondi c'è un minimo non negativo. Considerando che il calcolo della media del segnale, attraverso la media degli inviluppi, rappresenta una approssimazione di quella che in realtà dovrebbe essere una media locale, la quale però non può essere calcolata a causa della natura non stazionaria del segnale. Il processo di sifting viene quindi reiterato per ogni nuova candidata IMF, finchè la k-esima IMF non rispetterà entrambe le condizioni. Nell'iterazione successiva $h_1(t)$ rappresenterà il nuovo segnale in ingresso come descritto dalla formula 6.5.

$$h_{11}(t) = h_1(t) - m_{11} \tag{6.5}$$

Nella formula i campi sono:

- h_1(t), IMF calcolata durante la prima iterazione del sifting process e non selezionata come IMF giusta;
- m_{11}, media degli inviluppi relativi ad h_1(t);
- h_{11}(t), prima IMF prototipo calcolata nella prima iterazione del sifting process.

L'algoritmo sarà ripetuto k volte fintanto che le due condizioni sopra non vengano entrambe soddisfatte. Si avrà così la $h_{1k}(t) = h_{1(k-1)}(t) - m_{1k}$. Se nella k-esima iterazione le condizioni vengono soddisfatte, si seleziona h_{1k}(t) come prima IMF valida e la si indica con $c_1(t) = h_{1k}(t)$. Nella figura 6.12 è riportata l'IMF 1 del segnale test x(t) ottenuta dopo 12 iterazioni. La $c_1(t)$, prima IMF

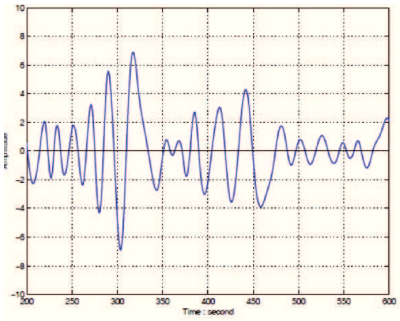

Figura 6.12: Rappresentazione della IMF 1 del segnale x(t) ottenuta dopo 12 operazioni di sifting *(articolo Norden E Huang)*

trovata, sarà quella contenente la componente del segnale x(t) avente periodo più breve, quindi la componente oscillatoria più veloce dell'intero segnale. Si procede allora alla separazione della componente $c_1(t)$ dal resto del segnale x(t) mediante $r_1(t) = x(t) - c_1(t)$. La $r_1(t)$, chiamata *residuo*, conterrà

tutte le componenti oscillatorie del segnale x(t) tranne la più veloce. Il procedimento verrà ripetuto considerando $r_1(t)$ il nuovo segnale di partenza per l'estrazione delle successive IMF. I nuovi residui saranno descritti dalla formula $r_n(t) = r_{(n-1)}(t) - c_n(t)$. Il processo di estrazione delle IMF si conclude al passo n, cioè quando il residuo r_n(t) della n-esima IMF diventa una funzione monotona, dalla quale non potranno più essere estratte altre componenti. Il risultato finale può essere espresso come sommatoria delle n componenti estratte c_n(t), a cui va aggiunto il residuo calcolato all'n-passo r_n(t). La formula che descrive tale sommatoria è la 6.6.

$$x(t) = \sum_{j=1}^{n} c_j(t) + r_n(t) \tag{6.6}$$

Il criterio di Stopping Nella descrizione dell'algoritmo EMD abbiamo supposto che l'estrazione della IMF, come da teoria, avvenisse a seguito del soddisfacimento delle condizioni 1 e 2, relative alla scelta della IMF. Nella realtà è difficile impostare un criterio di stopping del sifting process, sulla base di un valor medio dell'inviluppo superiore ed inferiore pari a zero e su un numero di estremanti pari al numero di passaggi per lo zero della funzione. Si utilizzano allora dei criteri di stopping, i quali altro non sono che delle approssimazioni di ciò che dice la teoria. I criteri maggiormente utilizzati sono due [65]. Il primo, proposto da Huang, è basato su un tipo di convergenza di Cauchy. Più specificamente, il test richiede che il quadrato della differenza, normalizzata, tra due successive operazioni di sifting sia una quantità sufficientemente piccola come illustrato dalla formula 6.7.

$$SD_k = \frac{\sum_{t=0}^{T} |h_{(k-1)}(t) - h_k(t)|^2}{\sum_{t=0}^{T} h^2_{(k-1)}(t)} \tag{6.7}$$

Se questa quantità SD_k è minore di un valore predeterminato, il processo di sifting viene interrotto. Questa definizione, benchè rigorosa, risulta difficile applicarla nella pratica. Ci sono infatti due punti critici da risolvere: in primo luogo, quanto piccola deve essere la quantità SD_k per poter sostenere la validità di una IMF, in secondo luogo, tale criterio non dipende dalle condizioni, a cui deve sottostare una IMF, per essere considerata tale. Il quadrato della differenza potrebbe essere piccolo, ma nulla ci garantisce che la funzione avrà lo stesso numero di passaggi per lo zero ed estremi. Queste lacune hanno spinto Huang a proporre un secondo criterio, alla cui base ci sono il numero di passaggi per lo zero e il numero di estremi della funzione. Nel secondo criterio [65] viene preselezionato un numero S di iterazioni di sifting fisso. Il processo di sifting si fermerà solo dopo un numero S di volte, consecutive, in cui il numero di passaggi per lo zero e il numero di estremi rimangono gli stessi e sono entrambi uguali o

differiscono al massimo di uno. Anche questa seconda scelta ha la sua difficoltà su come selezionare il numero S. In un recente studio sono state utilizzati tanti possibili valori di S e Huang [127] ha stabilito, a livello empirico, che i siftings ottimali si ottengono quando S è compreso tra 3 e 5. Questo secondo criterio con numero di iterazioni fisso è anche il criterio utilizzato da Chappell [37], il quale usa S=3.

Problemi legati alle Spline e ai Bordi Come già illustrato, il processo di estrazione di IMF si basa sulla considerazione che, se la media degli inviluppi superiore e inferiore risulta nulla, risulta simmetrico. Purtroppo anche in segnali costruiti simmetrici, è inevitabile che le spline, utilizzate per l'interpolazione dei massimi e dei minimi locali diano una media non nulla. In particolare c'è il rischio che l'inviluppo sovrastimi o sottostimi il reale andamento del segnale, innescando così l'estrazione di IMF, che non rappresenta modi reali di oscillazione del segnale originario. Un esempio è illustrato in figura 6.13. Un

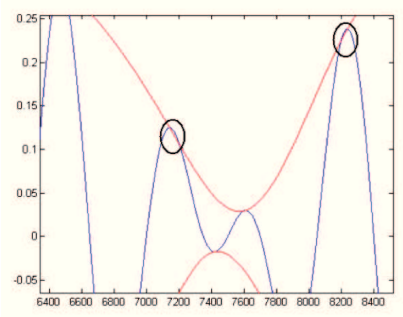

Figura 6.13: Sovrastima o sottostima del segnale *(articolo Norden E Huang)*

ulteriore problema, ancora legato al processo di sifting, ma non direttamente al calcolo delle spline, è l'alterazione che si ha nella rappresentazione di una IMF nei suoi istanti iniziali e finali. Questo è provocato dalla mancanza di punti, in cui calcolare la spline. Soprattutto nei casi in cui la frequenza del segnale da cui estrarre l'IMF è bassa; verso i bordi gli inviluppi non sono vincolati da estremanti, quindi tendono a divergere seguendo l'andamento della spline cubica. La situazione è mostrata in figura 6.14.

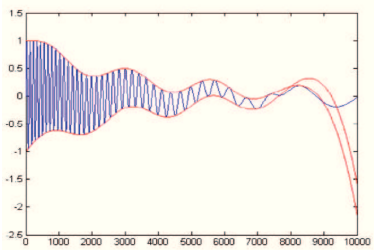

Figura 6.14: Rappresentazione della parte finale di un segnale in cui la spline non segue più il suo inviluppo, ma diverge verso il basso *(articolo Norden E Huang)*

6.3 Struttura WAVE

Il file WAVE ù basato sulle specifiche RIFF (Resource Interchange File Format) introdotte nel 1992 e costituenti un meta-format per i file multimediali che girano in ambiente Windows. La struttura RIFF organizza blocchi di dati in sezioni chiamate "Chunk", ognuna delle quali descrive una caratteristica del file WAVE o contiene i valori dei campioni. All'interno di un file WAVE possiamo distinguere tre grandi blocchi "Chunk":

- Riff Chunk;
- Fmt Chunk;
- Data Chunk.

Il Riff Chunk, composto da 12 byte, è l'intestazione del file audio, che contiene informazioni sul tipo di file e il numero di byte da cui è costituito. Nello specifico è formato da tre sotto-Chunk:

- ChunkId, stringa "RIFF" per formato byte "LittleEndian" o stringa "RIFX" per formato byte "BigEndian";
- ChunkSize, dimensione del file a partire dal byte successivo;
- Format, stringa identificativa "WAVE".

L'Fmt Chunk è un blocco contenente tutte le informazioni sulla codifica utilizzata. Nello specifico la struttura appare come segue:

- SubChunk1Id, contiene la stringa "fmt";
- Subchunk1Size, contiene i byte restanti del blocco, lo standard di questo blocco è 16 bytes per codifica PCM;

- AudioFormat, codifica utilizzata per il segnale, lo standard è la PCM (PCM=1);
- NumChannels, numero di canali (Mono=1 Stereo=2);
- SampleRate, frequenza di campionamento, ovvero il numero di campioni riprodotti al secondo;
- ByteRate, numero di byte riprodotti al secondo; Il ByteRate può essere calcolato con la formula 6.8.

$$ByteRate = \frac{SampleRate * NumChannels * BitsPerSample}{8} \tag{6.8}$$

- BlockAlign, blocco di allineamento; Il BlockAlign può essere calcolato con la formula 6.9.

$$BlockAlign = \frac{NumChannels * BitsPerSample}{8} \tag{6.9}$$

- BitsPerSample, 8 bits =8, 16 bits =16.

Il Data Chunk, è un blocco contenente la lunghezza e il valore dei dati. La struttura appare come segue:

- SubChunk2Id, contiene la stringa “data”;
- SubChunk2Size, numero dei byte nel campo Data;
- Data, i valori reali del file WAVE che rappresentano il suono udibile.

La tabella 6.1 mostra la struttura completa del file WAVE.

Tabella 6.1: Struttura WAVE

Endian	**File Offset (bytes)**	**Field Name**	**Field Size (bytes)**	**Chunk**
big	0-4	ChunkId	4	Riff
little	4-8	ChunkSize	4	Riff
big	8-12	Format	4	Riff
big	12-16	Subchunk1ID	4	Fmt
little	16-20	Subchunk1Size	4	Fmt
little	20-22	AudioFormat	2	Fmt
little	22-24	NumChannels	2	Fmt
little	24-28	SampleRate	4	Fmt
little	28-32	ByteRate	4	Fmt
little	32-34	BlockAlign	2	Fmt
little	34-36	BitsPerSample	2	Fmt
big	26-40	Subchunk2ZID	4	Data
little	40-44	Subchunk2Size	4	Data
little	44-∞	data	∞	Data

6.4 Notazione LittleEndian - BigEndian

I vari blocchi Chunk del file WAVE hanno tra loro una sostanziale differenza, dettata dal modo in cui i byte vengono memorizzati all'interno del blocco stesso. Esistono due tipi differenti di memorizzazione:

- BigEndian, conosciuta anche come left-to-right;
- LittleEndian, conosciuta anche come right-to-left.

Nell'architettura BigEndian i caratteri del blocco sono occupati dal *byte più significativo nel primo carattere* del blocco ($big - end - first$) ed i restanti in sequenza, uno dopo l'altro fino al *meno significativo a destra*. In questa architettura l'indirizzo del multi-byte corrisponde all'indirizzo del suo byte più significativo.

Nell'architettura LittleEndian i caratteri del blocco sono occupati dal *byte meno significativo nel primo carattere* del blocco ($little - end - first$) ed i restanti in sequenza, uno dopo l'altro fino al *più significativo a destra*. In questa architettura l'indirizzo del multi-byte corrisponde all'indirizzo del suo byte meno significativo.

Esempio
Blocco dati formato da 4 byte: 00(MSB)-01-e2-40(LSB)
MSB sta per Most Significative Byte ovvero Byte più significativo mentre LSB sta per Less Significative Byte, cioè Byte meno significativo. Considerando A come il primo indirizzo di memorizzazione byte, le tabelle 6.2 e 6.3 mostrano la scrittura nelle due differenti notazioni.

Tabella 6.2: Esempio architettura "BigEndian"

BigEndian			
00	01	e2	40
A+3	A+2	A+1	A

Tabella 6.3: Esempio architettura "LittleEndian"

LittleEndian			
00	01	e2	40
A	A+1	A+2	A+3

Capitolo 7

Pubblicazioni

7.1 Rivista internazionale

1. Paola Pierleoni, Lorenzo Maurizi, Lorenzo Palma, Alberto Belli, Simone Valenti, Alessandro Marroni, "A software tool for the annotation of Embolic Events in Echo Doppler Audio Signals", Biomedical Informatics Insights, SAGE Publishing, 2017 Dec 7, DOI: 10.1177/1178222617745557

2. Paola Pierleoni, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Luca Pernini, Michele Paniccia, Simone Valenti. "A Wearable Fall Detector for Elderly People Based on AHRS and Barometric Sensor" Sensors Journal, IEEE, 2014. DOI: 10.1109/JSEN.2016.2585667

3. Paola Pierleoni, Alberto Belli, Lorenzo Palma, Lorenzo Maurizi, Michele Paniccia, Simone Valenti, "A Smart Inertial System for Parkinson's Disease Classification and Monitoring", Sensor Journal (IEEE), SOTTOMESSO

4. Paola Pierleoni, Lorenzo Maurizi, Lorenzo Palma, Marco Pellegrini, Alberto Belli, Simone Valenti, Alessandro Marroni, "An EMD based algorithm for bubble detection in echo Doppler audio signals", International Journal of Telemedicine and Applications (Hindawi), SOTTOMESSO

7.2 Congresso internazionale

1. Paola Pierleoni, Luca Pernini, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Simone Valenti, Andrea Mengaroni, Loris Sabbatini, "An Innovative WebRTC Solution for e-Health Services," in e-Health Networking, Applications and Services (Healthcom), 2016 IEEE 18th International Conference on, Sept 2016. DOI: 10.1109/HealthCom.2016.7749444

2. Paola Pierleoni, Luca Pernini, Alberto Belli, Lorenzo Palma, Lorenzo Maurizi, Simone Valenti, "Indoor localization system for AAL over IPv6

WSN," in International Symposium on Personal, Indoor and Mobile Radio Communication (PIMRC), 2016 IEEE 27th Annual International Symposium on, Sept 2016. DOI: 10.1109/PIMRC.2016.7794564

3. Lorenzo Palma, Luca Pernini, Alberto Belli, Simone Valenti, Lorenzo Maurizi, Paola Pierleoni, "IPv6 WSN Solution for Integration and Interoperation Between Smart Home and AAL Systems," in Sensors Application Symposium (SAS 2016), 2016 IEEE, Sept 2016. DOI: 10.1109/SAS.2016.7479840

4. Paola Pierleoni, Luca Pernini, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Simone Valenti, "A Versatile WSN Approach for Smart Environments in AAL Application," International Workshop on Analysis of Biometric Parameters to detect relationship between stress and sleep quality (AnBiPa), Nov. 2016

5. Paola Pierleoni, Luca Pernini, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Simone Valenti, "Highly Accurate Wearable Attitude and Heading Reference System for kinematics assessment and AAL applications," International Workshop on Analysis of Biometric Parameters to detect relationship between stress and sleep quality (AnBiPa), Nov. 2016

6. Paola Pierleoni, Luca Pernini, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Simone Valenti, "Real-time Monitoring of Cardiac and Respiratory Activities Using a Consumer Wearable Device," International Workshop on Analysis of Biometric Parameters to detect relationship between stress and sleep quality (AnBiPa), Nov. 2016

7. Paola Pierleoni, Alberto Belli, Lorenzo Maurizi, Lorenzo Palma, Luca Pernini, Simone Valenti "Performance evaluation of a Pedestrian Navigation System based on an objective experimental method," in Intelligent Solutions in Embedded Systems (WISES), 2015 IEEE 12th International Workshop on, Oct 2015. INSPEC Accession Number: 15665938

7.3 Congresso nazionale

1. Paola Pierleoni, Alberto Belli, Lorenzo Palma, Luca Pernini, Lorenzo Maurizi, Simone Valenti, "An innovative IoT solution for smart environments in AAL," in 6th Forum Italiano dell'Ambient Assisted Living (FORITAAL 2015), May 2015.

Bibliografia

[1] Claudio D Gasparini, Victor Torres-Padrosa, Imma Boada, and Jose L Marzo, "Videoconferencing in ehealth: Requirements, integration and workflow," in *e-Health Networking, Applications & Services (Healthcom), 2013 IEEE 15th International Conference on*. IEEE, 2013, pp. 201–206.

[2] Zhanpeng Jin and Yu Chen, "Telemedicine in the cloud era: Prospects and challenges," *IEEE Pervasive Computing*, vol. 14, no. 1, pp. 54–61, 2015.

[3] Salvatore Loreto and Simon Pietro Romano, "Real-time communications in the web: Issues, achievements, and ongoing standardization efforts," *IEEE Internet Computing*, vol. 16, no. 5, pp. 68–73, 2012.

[4] Julian Jang-Jaccard, Surya Nepal, Branko Celler, and Bo Yan, "Webrtc-based video conferencing service for telehealth," *Computing*, vol. 98, no. 1-2, pp. 169–193, 2016.

[5] João Azevedo, Ricardo Lopes Pereira, and Paulo Chainho, "An api proposal for integrating sensor data into web apps and webrtc," in *Proceedings of the 1st Workshop on All-Web Real-Time Systems*. ACM, 2015, p. 8.

[6] Ilya Grigorik, *High Performance Browser Networking: What every web developer should know about networking and web performance*, " O'Reilly Media, Inc.", 2013.

[7] Randall Stewart, "Stream control transmission protocol," 2007.

[8] Zhou Hu, "Nat traversal techniques and peer-to-peer applications," in *HUT T-110.551 Seminar on Internetworking*, 2005, pp. 04–26.

[9] Dan Wing, Philip Matthews, Rohan Mahy, and Jonathan Rosenberg, "Session traversal utilities for nat (stun)," 2008.

[10] Simon Perreault and Jonathan Rosenberg, "Traversal using relays around nat (turn) extensions for tcp allocations," 2010.

[11] Alan Johnston and Robert J Sparks, "Session description protocol (sdp) offer/answer examples," 2005.

[12] Alan B Johnston and Daniel C Burnett, *WebRTC: APIs and RTCWEB protocols of the HTML5 real-time web*, Digital Codex LLC, 2012.

[13] "Webrtc architecture [online]," .

[14] W3C Editor's Draft 24 June 2016, "Media capture and streams [online]," 2016.

[15] Google H. T. Alvestrand, "Resolution constraints in web real time communications [online]," August 26, 2012.

[16] S. Dutton, "Getting started with webrtc [online]," Febbrary 2014.

[17] "Rooms and namespace [online]," .

[18] M. Khan, "Webrtc for beginners [online]," .

[19] Texas Instruments, "micropower singe-supply cmos instrumentation amplifier," *INA322 Datasheet, Feb*, 2006.

[20] Murugavel Raju, "Heart rate and ekg monitor using the msp430fg439," *Texas Instruments, Dallas, Texas, USA*, 2005.

[21] Zeli Gao, Jie Wu, Jianli Zhou, Wei Jiang, and Lihui Feng, "Design of ecg signal acquisition and processing system," in *Biomedical Engineering and Biotechnology (iCBEB), 2012 International Conference on*. IEEE, 2012, pp. 762–764.

[22] Thaddeus RF Fulford-Jones, Gu-Yeon Wei, and Matt Welsh, "A portable, low-power, wireless two-lead ekg system," in *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*. IEEE, 2004, vol. 1, pp. 2141–2144.

[23] GP Pizzuti, S Cifaldi, and G Nolfe, "Digital sampling rate and ecg analysis," *Journal of biomedical engineering*, vol. 7, no. 3, pp. 247–250, 1985.

[24] Jiapu Pan and Willis J Tompkins, "A real-time qrs detection algorithm," *IEEE transactions on biomedical engineering*, , no. 3, pp. 230–236, 1985.

[25] M. Mahemoff, "Extensions and apps in the crhome web store [online]," 2010.

[26] Crhome, "What are crhome app? [online]," .

[27] Crhome, "Crhome.app.runtime [online]," .

[28] Crhome, "Crhome.serial [online]," .

[29] Paola Pierleoni, Luca Pernini, Lorenzo Palma, Alberto Belli, Simone Valenti, Lorenzo Maurizi, Loris Sabbatini, and Alessandro Marroni, "An innovative webrtc solution for e-health services," in *e-Health Networking, Applications and Services (Healthcom), 2016 IEEE 18th International Conference on*. IEEE, 2016, pp. 1–6.

[30] AGCOM, "Ritardo di trasmissione dati [online]," .

[31] Chrome, "Webview for android [online]," .

[32] World Health Organization and others [Online], "Health and ageing: A discussion paper," 2001.

[33] Richard D Vann, Frank K Butler, Simon J Mitchell, and Richard E Moon, "Decompression illness," *The Lancet*, vol. 377, no. 9760, pp. 153–164, 2011.

[34] Paul Bert, *Barometric pressure*, College Book Company, 1943.

[35] MERRILL P Spencer, "Decompression limits for compressed air determined by ultrasonically detected blood bubbles," *Journal of Applied Physiology*, vol. 40, no. 2, pp. 229–235, 1976.

[36] K David Sawatzky, "The relationship between intravascular doppler-detected gas bubbles and decompression sickness after bounce diving in humans.," 1993.

[37] Michael Chappell, *Modelling and Measurement of Bubbles in Decompression Sickness.*, Ph.D. thesis, University of Oxford, 2006.

[38] RY Nishi, "Ultrasonic detection of bubbles with doppler flow transducers," *Ultrasonics*, vol. 10, no. 4, pp. 173–179, 1972.

[39] Alessandro Marroni, Peter B Bennett, Costantino Balestra, Ramiro Cali-Corleo, Peter Germonpre, Massimo Pieri, and Corrado Bonuccelli, "What ascent profile for the prevention of decompression sickness?," *DA Europe DSL Special Project:"Haldane vs. Hill" EUBS. Brugge, Belgium*, 2002.

[40] A Marroni, R Cali Corleo, C Balestra, P Longobardi, E Voellm, M Pieri, and R Pepoli, "Effects of the variation of ascent speed and profile on the production of circulating venous gas emboli and the incidence of dci in compressed air diving. phase 1.," 2000.

[41] A Marroni, "Patologia da decompressione. una valutazione alla luce delle più recenti acquisizioni," in *Proceedings of the International Symposium" Update on Hyperbaric Oxygen Therapy*, 1995.

[42] Jasminka Brnjas-Kraljevic, Kazuo Maeda, Alin Basgul Yigiter, Zehra Nese Kavak, Ingrid Marton, Badreldeen Ahmed, Berivoj Miskovic, Robin B Kalish, Elena Scazzocchio, M Angeles Rodriguez, et al., "Physical bases of medical ultrasound," *Donald School Basic Textbook of Ultrasound in Obstetrics & Gynecology*, vol. 400, no. 650, pp. 1, 2014.

[43] K Kisman, "Spectral analysis of doppler ultrasonic decompression data," *Ultrasonics*, vol. 15, no. 3, pp. 105–110, 1977.

[44] E Bernd Ringelstein, Dirk W Droste, Viken L Babikian, David H Evans, Donald G Grosset, Manfred Kaps, Hugh S Markus, David Russell, Mario Siebler, et al., "Consensus on microembolus detection by tcd," *Stroke*, vol. 29, no. 3, pp. 725–729, 1998.

[45] Alessandro Marroni, PB Bennett, FJ Cronje, R Cali-Corleo, et al., "A deep stop during decompression from 82 fsw (25 m) significantly reduces bubbles and fast tissue gas tensions," *Undersea & hyperbaric medicine*, vol. 31, no. 2, pp. 233, 2004.

[46] Alf O Brubakk and Tom S Neuman, *Bennett and Elliott's physiology and medicine of diving*, Saunders Book Company, 2003.

[47] Oliver Sykes and James E Clark, "Patent foramen ovale and scuba diving: a practical guide for physicians on when to refer for screening," *Extreme physiology & medicine*, vol. 2, no. 1, pp. 10, 2013.

[48] Peter Germonpré, Jean-Michel Pontier, Emmanuel Gempp, Jean-Eric Blatteau, Stefaan Deneweth, Pierre Lafère, Alessandro Marroni, and Costantino Balestra, "Pre-dive vibration effect on bubble formation after a 30-m dive requiring a decompression stop," *Aviation, space, and environmental medicine*, vol. 80, no. 12, pp. 1044–1048, 2009.

[49] A Marroni, R Cali-Corleo, C Balestra, E Voellm, and M Pieri, "Incidence of asymptomatic circulating venous gas emboli in unrestricted uneventful recreational diving dan europe's project safe dive first results," in *Proceedings of the XXVI Annual Scientific Meeting of the EUBS*, 2000.

[50] Stephen J Payne and Michael A Chappell, "Automated determination of bubble grades from doppler ultrasound recordings," *Aviation, space, and environmental medicine*, vol. 76, no. 8, pp. 771–777, 2005.

[51] Nizamettin Aydin, "Dwt based adaptive threshold determination in embolic signal detection," in *Adaptive Hardware and Systems, 2007. AHS 2007. Second NASA/ESA Conference on*. IEEE, 2007, pp. 214–219.

[52] Robert B Northrop, *Signals and systems analysis in biomedical engineering*, CRC press, 2016.

[53] Merve Gençer, Gokhan Bilgin, and Nizamettin Aydin, "Embolic doppler ultrasound signal detection via fractional fourier transform," in *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*. IEEE, 2013, pp. 3050–3053.

[54] Kadir Tufan, Ahmet Ademoglu, Emre Kurtaran, Gokcen Yildiz, Salih Aydin, and Salih M Egi, "Automatic detection of bubbles in the subclavian vein using doppler ultrasound signals," *Aviation, space, and environmental medicine*, vol. 77, no. 9, pp. 957–962, 2006.

[55] Eivind Kvedalen, "Signal processing using the teager energy operator and other nonlinear operators," *Master, University of Oslo Department of Informatics*, vol. 21, 2003.

[56] J-M Girault, Denis Kouamé, Abdeldjalil Ouahabi, and Frédéric Patat, "Micro-emboli detection: an ultrasound doppler signal processing viewpoint," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 11, pp. 1431–1439, 2000.

[57] Yu Zhang, Hong Zhang, and Nanxiong Zhang, "Microembolic signal characterization using adaptive chirplet expansion," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 52, no. 8, pp. 1291–1299, 2005.

[58] Steffen Skogland, Kåre Segadal, Harald Sundland, and Arvid Hope, "Gas bubbles in rats after heliox saturation and different decompression steps and rates," *Journal of Applied Physiology*, vol. 92, no. 6, pp. 2633–2639, 2002.

[59] Salman Marvasti, Duncan Gillies, Farokh Marvasti, and Hugh S Markus, "Online automated detection of cerebral embolic signals using a wavelet-based system," *Ultrasound in medicine & biology*, vol. 30, no. 5, pp. 647–653, 2004.

[60] Nizamettin Aydin and Hugh S Markus, "Detection of embolic signals using wavelet transform.," in *NSIP*, 1999, pp. 774–778.

[61] Gorkem Serbes and Nizamettin Aydin, "Denoising embolic doppler ultrasound signals using dual tree complex discrete wavelet transform," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*. IEEE, 2010, pp. 1840–1843.

[62] David L Donoho, "De-noising by soft-thresholding," *IEEE transactions on information theory*, vol. 41, no. 3, pp. 613–627, 1995.

[63] David L Donoho and Iain M Johnstone, "Threshold selection for wavelet shrinkage of noisy data," in *Engineering in Medicine and Biology Society, 1994. Engineering Advances: New Opportunities for Biomedical Engineers. Proceedings of the 16th Annual International Conference of the IEEE*. IEEE, 1994, vol. 1, pp. A24–A25.

[64] Nizamettin Aydin, Farrokh Marvasti, and Hugh S Markus, "Embolic doppler ultrasound signal detection using discrete wavelet transform," *IEEE Transactions on Information Technology in Biomedicine*, vol. 8, no. 2, pp. 182–190, 2004.

[65] Norden E Huang et al., "Introduction to the hilbert-huang transform and its related mathematical problems," *Interdisciplinary Mathematics*, vol. 5, pp. 1–26, 2005.

[66] L-a Yip, "Realtime empirical mode decomposition for intravascular bubble detection," *Bachelor Thesis, School of engineering and physical sciences, James Cook University*, 2010.

[67] Qin Pinle, Yan Lin, and Ming Chen, "Empirical mode decomposition method based on wavelet with translation invariance," *EURASIP Journal on Advances in Signal Processing*, 2008.

[68] Peter Germonpré, Virginie Papadopoulou, Walter Hemelryck, Georges Obeid, Pierre Lafère, Robert J Eckersley, Ming-Xing Tang, and Costantino Balestra, "The use of portable 2d echocardiography and 'frame-based'bubble counting as a tool to evaluate diving decompression stress," 2014.

[69] Gabriel Rilling, Patrick Flandrin, Paulo Goncalves, et al., "On empirical mode decomposition and its algorithms," in *IEEE-EURASIP workshop on nonlinear signal and image processing*. IEEER, Grado, Italy, 2003, vol. 3, pp. 8–11.

[70] David L Donoho and Jain M Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[71] Sanjana Prasad, P Mahalakshmi, A John Clement Sunder, and R Swathi, "Smart surveillance monitoring system using raspberry pi and pir sensor," *Int. J. Comput. Sci. Inf. Technol*, vol. 5, no. 6, pp. 7107–7109, 2014.

[72] Brian Trapp, "Raspberry pi: The perfect home server," *Linux Journal*, vol. 2013, no. 229, pp. 1, 2013.

[73] Viren Pereira, Vandyk Amsdem Fernandes, and Junieta Sequeira, "Low cost object sorting robotic arm using raspberry pi," in *Global Humanitarian Technology Conference-South Asia Satellite (GHTC-SAS), 2014 IEEE*. IEEE, 2014, pp. 1–6.

[74] Mark A Wickert, "Using the arm cortex-m4 and the cmsis-dsp library for teaching real-time dsp," in *Signal Processing and Signal Processing Education Workshop (SP/SPE), 2015 IEEE*. IEEE, 2015, pp. 283–288.

[75] C de Boor, "A practical guide to splines, vol. 27 of appl. math," *Sciences. New York, NY: Springer-Verlag*, 1978.

[76] Frederick N Fritsch and Ralph E Carlson, "Monotone piecewise cubic interpolation," *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, pp. 238–246, 1980.

[77] Gioacchino Martinciglio, "Valutazione delle abilità percettive verbali nel bambino prima e dopo impianto cocleare," 2012.

[78] Dominika Dykiert, Geoff Der, John M Starr, and Ian J Deary, "Age differences in intra-individual variability in simple and choice reaction time: systematic review and meta-analysis," *PLoS One*, vol. 7, no. 10, pp. e45759, 2012.

[79] Ellen Gorus, Rudi De Raedt, and Tony Mets, "Diversity, dispersion and inconsistency of reaction time measures: effects of age and task complexity," *Aging clinical and experimental research*, vol. 18, no. 5, pp. 407–417, 2006.

[80] Adam Szyfman, Gregory Wanner, and Leslie Spencer, "The relationship between cellular phone use, performance, and reaction time among college students: Implications for cellular phone use while driving," *American journal of health education*, vol. 34, no. 2, pp. 81–83, 2003.

[81] Kris Gray, *Microsoft DirectX 9 programmable graphics pipeline*, Microsoft Press, 2003.

[82] William Bezdek, Janet Bankhead, and Theresa Trapp, "A real time simulation using linux converted from a unix application," in *2003 AIAA Modeling and Simulation Technologies Conference and Exhibit*, 2003.

[83] Robert W Peach, *The ISO 9000 handbook*, Irwin Professional Publishing, 1995.

[84] Alain Abran, Adel Khelifi, Witold Suryn, and Ahmed Seffah, "Usability meanings and interpretations in iso standards," *Software Quality Journal*, vol. 11, no. 4, pp. 325–338, 2003.

[85] Paola Pierleoni, Lorenzo Maurizi, Lorenzo Palma, Alberto Belli, Simone Valenti, and Alessandro Marroni, "A software tool for the annotation of embolic events in echo doppler audio signals," *Biomedical Informatics Insights*, vol. 9, 2017.

[86] Meg E Morris, "Movement disorders in people with parkinson disease: a model for physical therapy," *Physical therapy*, vol. 80, no. 6, pp. 578–597, 2000.

[87] Angelo Antonini and Giorgio L Colombo, "Il ruolo della spect in associazione a iofupane nella diagnosi della malattia di parkinson: i risultati di un'esperienza," *Farmeconomia. Health economics and therapeutic pathways*, vol. 3, no. 3, pp. 125–133, 2002.

[88] Irene Litvan, Dag Aarsland, Charles H Adler, Jennifer G Goldman, Jaime Kulisevsky, Brit Mollenhauer, Maria C Rodriguez-Oroz, Alexander I Tröster, and Daniel Weintraub, "Mds task force on mild cognitive impairment in parkinson's disease: Critical review of pd-mci," *Movement disorders*, vol. 26, no. 10, pp. 1814–1824, 2011.

[89] Allison W Willis, Mario Schootman, Nathan Kung, Bradley A Evanoff, Joel S Perlmutter, and Brad A Racette, "Predictors of survival in patients with parkinson disease," *Archives of neurology*, vol. 69, no. 5, pp. 601–607, 2012.

[90] RJ Elble and WC Koller, "The physiology of normal tremor," *Tremor, The Johns Hopkins University Press, Baltimore*, pp. 37–53, 1990.

[91] William C Roller, Sander Glatt, Bridget Vetere-Overfield, and Ruth Hassanein, "Falls and parkinson's disease.," *Clinical neuropharmacology*, vol. 12, no. 2, pp. 98–105, 1989.

[92] John G Nutt, Steven T Gancher, and William R Woodward, "Does an inhibitory action of levodopa contribute to motor fluctuations?," *Neurology*, vol. 38, no. 10, pp. 1553–1553, 1988.

[93] Rebecca Stowe, Natalie Ives, Carl E Clarke, Kelly Handley, Alexandra Furmston, Katherine Deane, JJ Van Hilten, Keith Wheatley, and Richard Gray, "Meta-analysis of the comparative efficacy and safety of adjuvant treatment to levodopa in later parkinson's disease," *Movement Disorders*, vol. 26, no. 4, pp. 587–598, 2011.

[94] C David Marsden and JD Parkes, "On-off" effects in patients with parkinson's disease on chronic levodopa therapy," *Lancet*, vol. 1, no. 7954, pp. 292–296, 1976.

[95] MD Muenter and Gertrude M Tyce, "L-dopa therapy of parkinson's disease: plasma l-dopa concentration, therapeutic response, and side effects.," in *Mayo Clinic Proceedings*, 1971, vol. 46, pp. 231–239.

[96] IRA Shoulson, George A Glaubiger, and Thomas N Chase, "On-off response clinical and biochemical correlations during oral and intravenous levodopa administration in parkinsonian patients," *Neurology*, vol. 25, no. 12, pp. 1144–1144, 1975.

[97] C Warren Olanow, Karl Kieburtz, Per Odin, Alberto J Espay, David G Standaert, Hubert H Fernandez, Arvydas Vanagunas, Ahmed A Othman, Katherine L Widnell, Weining Z Robieson, et al., "Continuous intrajejunal infusion of levodopa-carbidopa intestinal gel for patients with advanced parkinson's disease: a randomised, controlled, double-blind, double-dummy study," *The Lancet Neurology*, vol. 13, no. 2, pp. 141–149, 2014.

[98] Lynn K Struck, Robert L Rodnitzky, and Judith K Dobson, "Circadian fluctuations of contrast sensitivity in parkinson's disease," *Neurology*, vol. 40, no. 3 Part 1, pp. 467–467, 1990.

[99] T Witjas, E Kaphan, JP Azulay, O Blin, M Ceccaldi, J Pouget, M Poncet, and A Ali Chérif, "Nonmotor fluctuations in parkinson's disease frequent and disabling," *Neurology*, vol. 59, no. 3, pp. 408–413, 2002.

[100] David E Riley and Anthony E Lang, "The spectrum of levodopa-related fluctuations in parkinson's disease," *Neurology*, vol. 43, no. 8, pp. 1459–1459, 1993.

[101] G Fabbrini, A Latorre, A Suppa, M Bloise, M Frontoni, and A Berardelli, "Fatigue in parkinson's disease: motor or non-motor symptom?," *Parkinsonism & related disorders*, vol. 19, no. 2, pp. 148–152, 2013.

[102] G Giovannoni, JD O'sullivan, K Turner, AJ Manson, and AJL Lees, "Hedonistic homeostatic dysregulation in patients with parkinson9s disease on dopamine replacement therapies," *Journal of Neurology, Neurosurgery & Psychiatry*, vol. 68, no. 4, pp. 423–428, 2000.

[103] Per Odin, K Ray Chaudhuri, JT Slevin, J Volkmann, E Dietrichs, P Martinez-Martin, JK Krauss, T Henriksen, R Katzenschlager, A Antonini, et al., "Collective physician perspectives on non-oral medication approaches for the management of clinically relevant unresolved issues in parkinson's disease: consensus from an international survey and discussion program," *Parkinsonism & related disorders*, vol. 21, no. 10, pp. 1133–1144, 2015.

[104] Christopher G Goetz, W Poeve, O Rascol, et al., "Movement disorder society task force on rating scales for parkinson's disease: The unified parkinson's disease rating scale (updrs): status and recommendations," *Mov Disord*, vol. 18, pp. 738–750, 2003.

[105] Christopher G Goetz, Barbara C Tilley, Stephanie R Shaftman, Glenn T Stebbins, Stanley Fahn, Pablo Martinez-Martin, Werner Poewe, Cristina Sampaio, Matthew B Stern, Richard Dodel, et al., "Movement disorder society-sponsored revision of the unified parkinson's disease rating scale (mds-updrs): Scale presentation and clinimetric testing results," *Movement disorders*, vol. 23, no. 15, pp. 2129–2170, 2008.

[106] Haruko Tanji, Ann L Gruber-Baldini, Karen E Anderson, Ingrid Pretzer-Aboff, Stephen G Reich, Paul S Fishman, William J Weiner, and Lisa M Shulman, "A comparative study of physical performance measures in parkinson's disease," *Movement Disorders*, vol. 23, no. 13, pp. 1897–1905, 2008.

[107] John F Kurtzke, "Rating neurologic impairment in multiple sclerosis an expanded disability status scale (edss)," *Neurology*, vol. 33, no. 11, pp. 1444–1444, 1983.

[108] Bijan Najafi, Kamiar Aminian, Anisoara Paraschiv-Ionescu, François Loew, Christophe J Bula, and Philippe Robert, "Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly," *IEEE Transactions on biomedical Engineering*, vol. 50, no. 6, pp. 711–723, 2003.

[109] F Foerster, M Smeja, and J Fahrenberg, "Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring," *Computers in Human Behavior*, vol. 15, no. 5, pp. 571–583, 1999.

[110] Nicky Kern, Bernt Schiele, Holger Junker, Paul Lukowicz, and Gerhard Tröster, "Wearable sensing to annotate meeting recordings," *Personal and Ubiquitous Computing*, vol. 7, no. 5, pp. 263–274, 2003.

[111] Ling Bao and Stephen Intille, "Activity recognition from user-annotated acceleration data," *Pervasive computing*, pp. 1–17, 2004.

[112] Miikka Ermes, Juha Pärkkä, Jani Mäntyjärvi, and Ilkka Korhonen, "Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions," *IEEE transactions on information technology in biomedicine*, vol. 12, no. 1, pp. 20–26, 2008.

[113] Dean M Karantonis, Michael R Narayanan, Merryn Mathie, Nigel H Lovell, and Branko G Celler, "Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring," *IEEE transactions on information technology in biomedicine*, vol. 10, no. 1, pp. 156–167, 2006.

[114] Michael Sung, Carl Marci, and Alex Pentland, "Wearable feedback systems for rehabilitation," *Journal of neuroengineering and rehabilitation*, vol. 2, no. 1, pp. 17, 2005.

[115] MJ Mathie, Branko G Celler, Nigel H Lovell, and ACF Coster, "Classification of basic daily movements using a triaxial accelerometer," *Medical and Biological Engineering and Computing*, vol. 42, no. 5, pp. 679–687, 2004.

[116] Uwe Maurer, Asim Smailagic, Daniel P Siewiorek, and Michael Deisher, "Activity recognition and monitoring using multiple sensors on different body positions," in *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*. IEEE, 2006, pp. 4–pp.

[117] David Minnen, Thad Starner, Jamie A Ward, Paul Lukowicz, and G Troster, "Recognizing and discovering human actions from on-body sensor data," in *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*. IEEE, 2005, pp. 1545–1548.

[118] Daniel Rodríguez-Martín, Albert Samà, Carlos Pérez-López, Andreu Català, Joan M Moreno Arostegui, Joan Cabestany, Àngels Bayés, Sheila Alcaine, Berta Mestre, Anna Prats, et al., "Home detection of freezing of gait using support vector machines through a single waist-worn triaxial accelerometer," *PloS one*, vol. 12, no. 2, pp. e0171764, 2017.

[119] Kerem Altun, Billur Barshan, and Orkun Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognition*, vol. 43, no. 10, pp. 3605–3620, 2010.

[120] John J Craig, *Introduction to robotics: mechanics and control*, vol. 3, Pearson Prentice Hall Upper Saddle River, 2005.

[121] Arno M Janssen, Moniek AM Munneke, Jorik Nonnekes, Thomas van der Kraan, Alice Nieuwboer, Ivan Toni, Anke H Snijders, Bastiaan R Bloem, and Dick F Stegeman, "Cerebellar theta burst stimulation does not improve freezing of gait in patients with parkinson's disease," *Journal of Neurology*, vol. 264, no. 5, pp. 963–972, 2017.

[122] Steven T Moore, Hamish G MacDougall, and William G Ondo, "Ambulatory monitoring of freezing of gait in parkinson's disease," *Journal of neuroscience methods*, vol. 167, no. 2, pp. 340–348, 2008.

[123] Adil Mehmood Khan, Young-Koo Lee, Sungyoung Y Lee, and Tae-Seong Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer," *IEEE transactions on information technology in biomedicine*, vol. 14, no. 5, pp. 1166–1172, 2010.

[124] Piyush Gupta and Tim Dallas, "Feature selection and activity recognition system using a single triaxial accelerometer," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 6, pp. 1780–1786, 2014.

[125] Hui-Ling Chen, Chang-Cheng Huang, Xin-Gang Yu, Xin Xu, Xin Sun, Gang Wang, and Su-Jing Wang, "An efficient diagnosis system for detection of parkinson's disease using fuzzy k-nearest neighbor approach," *Expert systems with applications*, vol. 40, no. 1, pp. 263–271, 2013.

[126] Ferhat Attal, Samer Mohammed, Mariam Dedabrishvili, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat, "Physical human activity recognition using wearable sensors," *Sensors*, vol. 15, no. 12, pp. 31314–31338, 2015.

[127] Norden E Huang, Man-Li C Wu, Steven R Long, Samuel SP Shen, Wendong Qu, Per Gloersen, and Kuang L Fan, "A confidence limit for the empirical mode decomposition and hilbert spectral analysis," in *Proceedings of the royal society of london a: Mathematical, physical and engineering sciences*. The Royal Society, 2003, vol. 459, pp. 2317–2345.