ALMA MATER STUDIORUM — UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA
SCUOLA DI INGEGNERIA E ARCHITETTURA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

REALIZZAZIONE DI UN PROTOTIPO DI SCHERMO PERVASIVO ADATTATIVO

Elaborato in
Fondamenti di informatica A

Relatore
Mirko Viroli

Presentata da
Simone Costanzi

Sessione I°
Anno Accademico 2013/2014

Ai miei genitori,
Paolo e Luciana

INDICE

INTRODUZIONE

I recenti progressi nelle tecnologie hardware e di comunicazione stanno lentamente cambiando il modo di concepire l'elaborazione informatica e le modalità d'interazione fra l'uomo ed i computer. In breve, ci troveremo a vivere in ambienti popolati da una serie di dispositivi "intelligenti" che comunicano e cooperano tra di loro per assisterci nello svolgimento delle nostre comuni attività.

Esistono sempre più scenari applicativi le cui fasi operative vengono monitorate tramite un sistema basato su una rete eterogenea di dispositivi quali sensori, palmari, smartphone, etichette RFID.

A questo tipo di scenari ci si riferisce quando si parla di Ubiquitous (o Pervasive) Computing: scenari in cui capacità di calcolo e di elaborazione assumono caratteristiche di ubiquità, si diffondono nella nostra realtà e penetrano all'interno degli oggetti che ci circondano.

Piuttosto che un fenomeno tecnologico, l'Ubiquitous Computing è una visione che ci spinge a riconsiderare il ruolo dei computer e della computazione nel nostro mondo quotidiano. E' una visione alla ricerca di una qualità fondamentalmente nuova dell'utilizzo del calcolatore.

Considerando due casi estremi di un continuo di possibili esempi: il processo di produzione e trasporto dei vini di qualità e il supporto automatico dei visitatori di un museo. Nel primo caso, il processo va controllato a partire dalle condizioni ambientali dell'uva nel vigneto, alla maturazione del vino nelle botti, fino al trasporto su camion ed al mantenimento delle bottiglie nelle cantine; nel secondo caso possiamo immaginare di dotare i visitatori del museo di un dispositivo portatile, che reagisce ai vari spostamenti dell'utente e ai suoi cambi di contesto, segnalati da appositi sensori.

Questi scenari applicativi richiedono impianti molto diversi tra loro: le reti di sensori wireless possono essere la soluzione ideale per il monitoraggio di parametri quali la temperatura e l'umidità nel vigneto o la qualità della luce nelle sale del museo; l'applicazione di etichette RFID, nel caso dei vini di qualità, può essere appropriata per l'identificazione delle bottiglie durante e dopo il trasporto, mentre nel caso del museo può consentire di riconoscere l'arrivo del

visitatore in una sala allo scopo di trasmettere al suo dispositivo le informazioni relative alle opere che sono presenti in quella sala.

La "vision" dell'elaborazione pervasiva prevede quindi sensoristica e dispositivi mobili (interconnessi da una o più reti) in grado di reagire, in modo trasparente all'utente, alle situazioni contingenti e al cambiamento del contesto operativo.

Obiettivo di questa tesi è la realizzazione di un sistema che si avvicini a queste caratteristiche.
In particolare si è realizzato una versione prototipo di uno *schermo pervasivo adattativo*, in grado di riconoscere una persona qualora questa gli porga attenzione e successivamente adattare i suoi contenuti interattivi (attraverso interfaccia gestuale) in base alla determinata persona. Si è poi simulato il funzionamento di questo sistema calandolo nel contesto accademico.

Uno strumento di notevole importanza in questo sistema realizzato è il sensore *Kinect.*

Il 4 novembre 2010 Microsoft mette sul mercato Kinect. Quella che all'inizio sembrava una semplice periferica di gioco, avrebbe in realtà rivoluzionato poco dopo il modo di interagire con numerosi dispositivi tecnologici.
Questa tecnologia è basata su un nuovo paradigma di interazione uomo-macchina in grado di rendere obsolete periferiche attualmente in uso come mouse e tastiera, ovvero l'*interazione naturale (NI).*
Il concetto su cui si basa NI è quello per cui l'utente interagisce con il determinato dispositivo attraverso i sensi umani, cercando quindi di rendere l'interazione più facile e intuitiva.
Grazie ai driver open source (*OpenNI*) rilasciati dalla società *PrimeSense* appena un mese dopo l'arrivo del Kinect è stato possibile utilizzare questo importante strumento come dispositivo di input del nostro sistema, sfruttando molteplici sue potenzialità.

STRUTTURA E CONTENUTI

Primo capitolo
Vengono esposti i concetti principali di Pervasive ed Ubiquitous Computing.

Secondo capitolo

Data l'importanza che ha il Kinect in questo progetto, viene aperta una panoramica proprio su questo sensore. Vengono descritte le caratteristiche tecniche e vengono dati alcuni cenni principali sull'hardware. Inoltre vengono presentate le librerie OpenNI e NITE che si occupano dell'elaborazione dei dati del sensore, nonché del suo interfacciamento. Infine viene spiegato il funzionamento del sensore e le tecniche utilizzate per il calcolo della mappa di profondità.

Terzo capitolo

E' quello di maggiore importanza in quanto vengono esposti i tre componenti costituenti la nostra soluzione:

1. il primo componente consiste in un prototipo per il riconoscimento dell'attenzione basato principalmente sulla postura del corpo degli utenti individuati dal sensore Kinect. Sulla base delle informazioni fornite da questo strato si è in grado di definire quando una persona voglia interagire con lo schermo.
2. Successivamente al rilevamento di un osservatore il secondo componente si occupa dell'unificazione fra la persona fisica e la persona virtuale rilevata dal Kinect. In altri termini dà la potenzialità al sistema pervasivo di identificare la persona che le sta davanti in modo da potersi adattare ed offrire un contenuto informativo che riguarda quel determinato utente.
3. Un altro componente che sfrutta le potenzialità del dispositivo Kinect è utilizzato in questo caso per offrire una interazione naturale fra l'utente e lo schermo.

Quarto capitolo

Viene presentato e discusso il diagramma a stati per descrivere il funzionamento del sistema, approfondendo alcuni punti di maggiore importanza, come alcuni aspetti relativi all'integrazione dei tre componenti. All'interno di questa discussione vengono anche riportati alcuni shot del sistema rilevabili durante il suo funzionamento.

Infine viene riportato una breve ma esaustiva descrizione del flusso di esecuzione principale.

1 UBIQUITOUS E PERVASIVE COMPUTING

1.1 introduzione

Era il 1991 quando Mark Weiser, direttore scientifico delle ricerche tecnologiche allo Xerox PARK, in un suo articolo d'avanguardia, utilizzò per la prima volta il termine *Ubiquitous Computing* [1]. Nel suo articolo, Weiser annunciava un cambiamento nel modo di concepire l'elaborazione automatica e descriveva scenari in cui i computer, onnipresenti, entravano sempre più a far parte della vita di tutti i giorni.

I *computer* ed il *computing* in generale, infatti, stanno lentamente ed inesorabilmente navigando verso nuovi paradigmi. Negli anni sessanta, alla parola "*computer*" venivano associati grandi e costosi *mainframe*, caratterizzati da un grosso numero di utenti che ne condividevano le risorse. Si parlava di paradigma "*many people per computer*": molti utenti per una sola macchina. Il progresso tecnologico ha poi consentito la realizzazione dei *personal computer* che hanno significativamente modificato il tipo di utilizzo dei sistemi di calcolo, trasformando il paradigma in "*one person per computer*": ogni persona poteva disporre di un proprio calcolatore. Nell'ultimo decennio, la diffusione *tablet*, *smartphone*, dispositivi portatili dotati di microprocessori e di capacità di immagazzinare dati, ha mutato ulteriormente il rapporto uomocomputer, aprendo le porte all'era dei "*many computers per person*": tanti elaboratori per una singola persona (Figura 1).

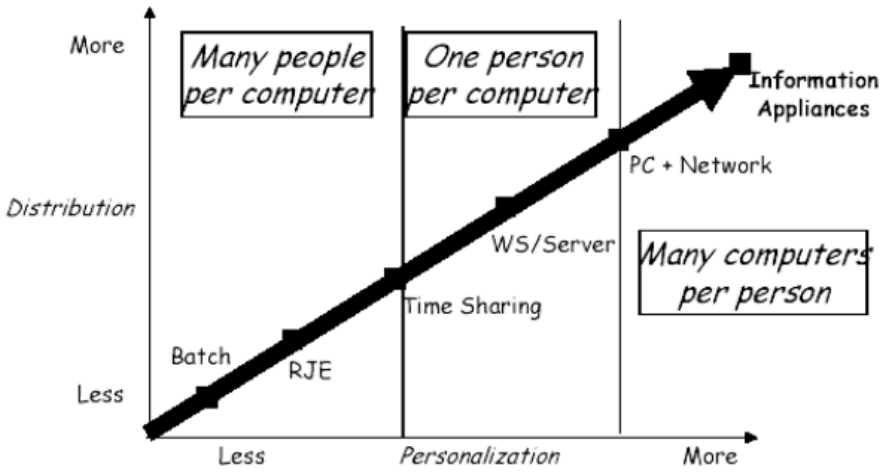

Figure 1.: L'evoluzione del computing dai mainframes ad oggi.

Il proliferare di questi dispositivi intelligenti, sempre più piccoli e meno costosi, insieme con i progressi delle tecnologie di comunicazione, ha indotto Mark Weiser ad immaginare un futuro, non troppo lontano, nel quale i computer entrano a far parte integrante di ogni oggetto della realtà quotidiana, favorendo la comunicazione e l'elaborazione di informazioni in maniera naturale: *everywhere, all the time*.

L'essenza della sua idea era "*the creation of environments saturated with computing and computational capability, yet gracefully integrated with human users*".

Un sistema di Ubiquitous Computing è caratterizzato da due attributi fondamentali:

- *Ubiquità*: l'interazione con il sistema è disponibile dovunque l'utente ne abbia bisogno;
- Trasparenza: il sistema non è intrusivo ed è integrato negli ambienti della vita quotidiana.

In accordo con questa visione è possibile identificare due dimensioni che forniscono una più chiara definizione degli *ubiquitous system* ed esprimono le relazioni esistenti con le altre aree di ricerca emergenti:

- *Mobilità dell'utente*: esprime la libertà che l'utente ha di muoversi quando interagisce con il sistema;
- *Trasparenza di interfaccia*: riflette lo sforzo consapevole e l'attenzione che il sistema richiede all'utente, sia per operare su di esso che per percepirne i suoi output.

Quindi, nell'ottica di Weiser, *l'Ubiquitous Computing* mira alla realizzazione di un mondo non più vincolato alle scrivanie, ma composto da ambienti dotati di capacità computazionali e comunicative, talmente integrati con l'utente da diventare una "tecnologia che svanisce", utilizzabile in maniera trasparente ed inconscia. Non una realtà virtuale, in cui le persone sono inserite in un mondo generato dai computer, ma piuttosto una virtualità reale che porta i computer a vivere nel mondo reale, insieme con le persone.

Tuttavia, gli scenari dipinti da Weiser 13 anni fa erano anacronistici; la tecnologia hardware necessaria per la loro realizzazione semplicemente non esisteva. E così, i tentativi compiuti allo Xerox PARC fallirono.

Dopo diversi anni, i recenti sviluppi tecnologici hanno dato nuovo impulso alle ricerche *sull'Ubiquitous*

Computing, di recente ribattezzato anche col nome di *Pervasive Computing*, per suggerire il carattere pervasivo con cui l'"intelligenza elaborativa" si diffonde e si manifesta negli oggetti che ci circondano.

Probabilmente, i tempi non sono ancora maturi e la visione di Weiser resta ancora futuristica: "*The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it*". Di sicuro, però, oggi ci troviamo in una posizione migliore, rispetto a quella dei ricercatori della Xerox del 1991, per affrontare le problematiche relative all'*Ubiquitous* ed al *Pervasive Computing*.

Da qualche anno, di fatto, in diverse Università e Centri di Ricerca del mondo sono sorte numerose iniziative che seguono la direzione tracciata da Weiser; ciascun progetto affronta i vari problemi da differenti punti di vista, talvolta contrastanti, ma tutti sono impegnati nello sforzo comune di rendere *l'Ubiquitous* ed il *Pervasive Computing* una realtà.

1.2 L'evoluzione verso il pervasing computing

Il *Pervasive Computing* può essere visto come una nuova forma di computazione altamente dinamica e disaggregata: gli utenti sono mobili e i servizi sono forniti da una serie di componenti distribuiti che collaborano tra loro. Le applicazioni necessarie per supportare queste nuove esigenze sono, da un punto di vista architetturale, non monolitiche bensì costituite da moduli allocati in numerosi nodi della rete.

In tal senso, il Pervasive Computing costituisce una nuova tappa nel percorso evolutivo dell'elaborazione e del calcolo distribuito.

1.2.1 Sistemi distribuiti

Il *Distributed Computing* mira a ripartire dati e capacità computazionali su componenti indipendenti, residenti su macchine diverse, che comunicano tra loro attraverso reti di interconnessione.

Le metodologie attraverso le quali comunicare ed accedere ai dati e agli strumenti di calcolo da postazioni remote, le tecniche sulla replicazione e ridondanza dei dati, che favoriscono la disponibilità e la reperibilità delle informazioni e aumentano l'affidabilità del sistema nel complesso, rappresentano l'oggetto di studio per questa forma di computing.

1.2.2 Sistemi mobili

Con l'introduzione di vincoli e problematiche legate al concetto di mobilità, è stato necessario pensare a nuove soluzioni tecnologiche che hanno portato alla creazione di una nuova forma di *computing*, ossia *Mobile Computing* [2].

In questi nuovi scenari di calcolo distribuito, non si hanno più nodi di rete fissi, con connessioni stabili e veloci, ma nodi costituiti da dispositivi mobili che accedono alla rete e la abbandonano continuamente ed in maniera del tutto imprevedibile, dotati di connessioni precarie e contraddistinte da forti cambiamenti sulle caratteristiche di banda.

Le limitate capacità di calcolo e di memoria dei dispositivi mobili, le esigenze di risparmio energetico, rappresentano ulteriori aspetti di cui il *Mobile Computing* si sta occupando.

1.2.3 Sistemi pervasivi

I sistemi di *Pervasive Computing* sono a loro volta anche sistemi distribuiti e mobili. Pertanto, le problematiche inerenti il *mobile* ed il *distributed computing* vengono riprese in questo nuovo paradigma ma, in certo senso, amplificate oltremodo a causa dei requisiti stringenti e dei particolari contesti definiti in questi nuovi ambienti.

1.3 Caratteristiche di un sistema pervasivo

Nei paradigmi di *ubiquitous computing*, servizi ed informazioni sono virtualmente accessibili dovunque, in ogni istante attraverso qualsiasi dispositivo.

Ma considerazioni di carattere amministrativo, territoriale e culturale ci inducono ad analizzare *l'ubiquitous computing* in ambienti discreti, dai confini ben definiti, come per esempio case, uffici, sale convegno, aeroporti, stazioni, musei, etc. In altre parole, è bene considerare il mondo suddiviso in tanti domini pervasivi, piuttosto che vederlo come un unico enorme sistema (Figura 2).

Ogni singolo ambiente è contraddistinto da componenti o unità software che implementano astrazioni di servizi, clienti, risorse o applicazioni ed, in generale, ogni ambiente è costituito da un'infrastruttura fissa e da una

serie di elementi mobili che, in maniera del tutto impredicibile, entrano a far parte del sistema e lo abbandonano continuamente, talvolta migrando fra i diversi domini: tutti i componenti dell'ambiente devono essere in grado di interagire fra di loro.

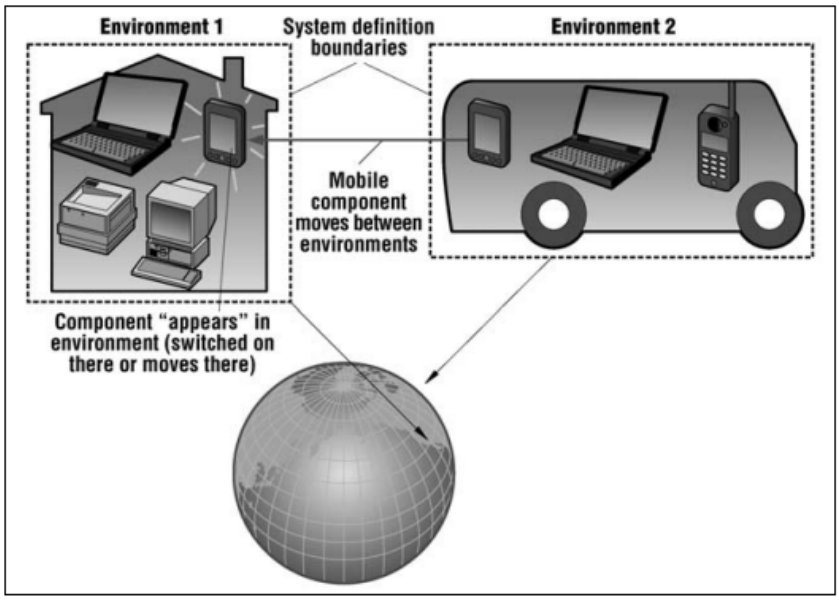

Figura 2.: Il mondo suddiviso in ambienti pervasivi con confini ben definiti

1.3.1 Context Information e Run-time Adaptation

Una prerogativa di un sistema pervasivo è che sia in grado di ottenere informazioni sugli utenti e sullo stato dell'ambiente, come ad esempio, la posizione e l'identità dei singoli utenti e la disponibilità delle risorse.

Un aspetto fondamentale del *pervasive computing* consiste, quindi, nel collezionare dati grezzi da una moltitudine di sorgenti, processare i dati, trasformarli in informazioni di contesto e condividere le informazioni tra le diverse applicazioni in esecuzione sui vari dispositivi, cercando di mantenere la scalabilità, di garantire la sicurezza delle informazioni, inibendo accessi non autorizzati e rispettando la privacy individuale.

In base a queste informazioni, le applicazioni possono adattare il loro comportamento in maniera diversa a seconda dei casi. Possiamo definire due tipi di adattamento: funzionale o strutturale.

Un esempio di adattamento funzionale può essere rappresentato da un'applicazione che fornisce news in un ambiente: il tipo di notizie che vengono fornite potrebbe essere determinato in base a chi si trova nell'ambiente o all'ora della giornata.

Il prototipo di sistema pervasivo oggetto di questa tesi presenta proprio queste caratteristiche: uno schermo che fornisce informazioni, nel nostro caso di natura scolastica, proprio in base a chi gli sta davanti e in base ad una relazione temporale. In particolare vengono forniti gli orari relativi all'anno accademico dell'osservatore in quel determinato giorno.

Un esempio di adattamento strutturale, invece, può essere rappresentato da un'applicazione musicale: a seconda se l'utente è solo oppure no all'interno dell'ambiente, l'applicazione potrebbe utilizzare il sistema audio del laptop dell'utente, oppure quello dell'ambiente stesso.

Per realizzare quanto appena detto, occorre necessariamente avere una percezione dell'ambiente ragionevolmente accurata, e disporre di meccanismi per il rilevamento e la correzione di informazioni di contesto inattendibili o contrastanti.

1.3.2 Task Recognition e Pro-activity

Un sistema di *pervasive computing* dovrebbe essere capace, a partire dalle informazioni di contesto collezionate, di elaborare sullo stato corrente dell'ambiente e sulle intenzioni dell'utente e modificare dinamicamente il proprio comportamento per assisterlo nelle sue attività.

Diversamente dai sistemi di *computing* convenzionali in cui il comportamento del computer è principalmente composto di risposte all'interazione con l'utente, il *pervasive computing* mira alla realizzazione di un modello in cui i dispositivi sono la parte attiva nell'interazione con l'utente. Quindi se la tecnologia corrente si basa su persone che indicano ai computer ciò che devono fare, la nuova generazione di tecnologie dovrebbe essere basata su computer capaci di comprendere quello che le persone stanno facendo e quello che esse desiderano.

I modelli che tengono conto dell'esperienza, del passato, rappresentano un importante strumento per la

caratterizzazione di un sistema pervasivo, perché favoriscono la *pro-activity* del sistema, ossia consentono di determinare, in accordo con i precedenti comportamenti dell'utente, le azioni ottimali che l'utente stesso deve eseguire in determinate situazioni.

1.3.3 Resource Abstraction e Discovery

Le risorse di un sistema pervasivo dovrebbero essere rappresentate in maniera astratta (magari attraverso le caratteristiche anziché con i nomi) cosicché possano facilmente essere selezionate in base a requisiti di tipo generale oltre che specifico. Potrebbe essere necessario definire alcune convenzioni di nomi per evitare che differenti risorse descrivano se stesse adoperando gli stessi termini offrendo, però, servizi diversi.

Dovrebbero essere previsti anche dei meccanismi per scoprire, interrogare ed interagire con le risorse nell'ambiente, per consentire l'introduzione di nuovi componenti senza onerose operazioni di configurazione e di riconfigurazione dei componenti esistenti.

1.3.4 Eterogeneity e Service UI Adaptation

Un ambiente pervasivo è caratterizzato da una moltitudine di dispositivi eterogenei, in numero variabile, come sensori, tablet, samrtphone.

Alla riduzione delle dimensione dei dispositivi dell'ambiente corrisponde una maggiore crescita del numero di dispositivi connessi ed una intensificazione delle interazioni uomo-macchina.

Lo sviluppo tradizionale fornisce servizi tipicamente distribuiti e istallati separatamente per ogni classe di dispositivo e famiglia di processore. Gli scenari di *computing* pervasivo, invece, portano alla conclusione che distribuire ed installare servizi per ogni classe e famiglia diventa ingestibile, specialmente in un'area geografica molto estesa.

E' importante, quindi, che i servizi forniscano agli utenti interfacce che possano adattarsi alle caratteristiche del dispositivo *client*, senza stravolgere le funzionalità del servizio stesso. Ad esempio, un servizio adibito al controllo dell'illuminazione in una stanza è libero di fornire un'interfaccia utente di tipo grafico per un *client* PDA, ma deve necessariamente

garantire anche una rappresentazione testuale della stessa UI per un utente che adopera un smartphone, senza, peraltro, dover cambiare l'implementazione del servizio stesso.

1.4.5 Security e Privacy

Un sistema pervasivo generalmente gestisce grosse quantità di informazioni, molte delle quali acquisite tramite sensori e riguardanti gli utenti.

Dal punto di vista dell'utente, è desiderabile che vengano rispettati i principi di privacy e che sia garantita la sicurezza di queste informazioni.

Ad esempio, risulta fastidioso sapere che la stazione di polizia più vicina possa conoscere in quale stanza ci si trovi nella propria casa, attraverso i rilevatori di moto del sistema d'allarme, o quanto alcool si sta consumando, deducendo questa informazione dal sistema che gestisce l'inventario degli alimenti.

D'altra parte, in molte situazioni, una certa perdita di privacy può essere tollerata, come ad esempio in situazioni di pericolo.

Per aggiungere sicurezza alle informazioni, i servizi nell'ambiente non dovrebbero consentire accessi non autorizzati: ad esempio, a casa propria, non dovrebbe essere possibile per un ospite, aumentare il riscaldamento o controllare il funzionamento del forno.

1.3.6 Fault-tolerance e Scalability

Gli ambienti pervasivi costituiscono sistemi "perennemente" attivi. Pertanto, un componente che subisce un guasto non deve compromettere il funzionamento generale dell'intero sistema, né richiedere una complessa tecnica di gestione.

I componenti che cadono dovrebbero automaticamente ripartire, laddove possibile, magari adoperando, ad esempio, memorie di stato persistenti che consentano di effettuare *resume* rapidi ed efficaci.

Abbiamo visto che gli ambienti pervasivi sono caratterizzati anche da una forte dinamicità: dispositivi possono aggiungersi all'ambiente ed abbandonarlo in qualsiasi momento; alcuni servizi possono cadere e

presentarsene altrettanti nuovi; gli stessi utenti possono entrare ed uscire dall'ambiente secondo la propria volontà.

Il sistema deve garantire scalabilità, ossia essere in grado di gestire e assicurare il suo funzionamento anche in seguito all'aggiunta di componenti; allo stesso tempo, i nuovi dispositivi e servizi introdotti nell'ambiente non dovrebbero interferire con quelli esistenti.

2 KINECT: CONTROLLER E MOTION TRACKER

Numerosi contenuti di questo capitolo sono stati reperiti dalle fonti [12] e [13].

Microsoft Kinect (inizialmente conosciuto con il nome Project Natal), è un accessorio per Xbox 360 sensibile al movimento del corpo umano. Esso rende il giocatore stesso controller della console senza l'uso di alcuno strumento, a differenza dei concorrenti come Nintendo Wii o Sony Playstation (Figura 3).

Figure 3.: Dispositivi a confronto. In questa foto si può vedere il Kinect, il telecomando Wii e il PlayStation Move

Sebbene in origine il dispositivo Kinect fu pensato esclusivamente per Xbox 360, Microsoft prevede di rendere nel prossimo futuro disponibile l'uso della periferica ai PC dotati del nuovo sistema operativo Windows 8. Microsoft ha pero dichiarato di rilasciare gratuitamente, durante l'estate 2011, i driver ufficiali per poter utilizzare Kinect nel proprio Personal Computer, dimostrando così di voler portare quanto prima la tecnologia di Kinect anche sui sistemi operativi Windows attualmente disponibili, favorendo lo sviluppo di varie applicazioni tra il mondo degli sviluppatori di software.

2.1 Storia

Kinect è stato annunciato al pubblico il 1 giugno 2009 durante la conferenza stampa della Microsoft all'E3 2009 (*Electronic Entertainment Expo*) con il nome **Project Natal**, poi rinominato Kinect alla presentazione ufficiale all'E3 2010.

Il 13 giugno 2010 Microsoft ha rivelato per la prima volta il vero nome del dispositivo, ovvero Kinect. Quest'ultimo è in vendita dal 4 novembre 2010 in America e dal 10 novembre in Europa, ed è possibile usarlo su un qualsiasi modello di XBOX 360.

L'hardware di Kinect si basa su tecnologie della 3DV, una compagnia israeliana specializzata in tecnologie di riconoscimento dei movimenti tramite videocamere digitali che Microsoft ha prima finanziato e poi acquisito nel 2009, e sul lavoro della israeliana PrimeSense, che ha poi dato in licenza la tecnologia a Microsoft.

Il software di Kinect è stato, invece, sviluppato internamente dai Microsoft Game Studios e, più precisamente, dai programmatori della Rare, la quale ha dovuto cancellare altri progetti migliori per dedicarsi interamente alla periferica.

Figure 4.: Microsoft Kinect

L'uscita di Kinect ha provocato un grande sommovimento nella comunità di sviluppo libero di software per PC e Mac. Una moltitudine di programmatori e al lavoro sul reverse engineering sulla periferica, allo scopo di trovare nuove modalità di utilizzo di un dispositivo che si configura come il primo di una serie di sistemi che potrebbe davvero portarci ad un futuro alla Minority Report.

2.1.1 Campi di utilizzo

Kinect è uno strumento nato come componente aggiuntivo per la consolle XBOX 360, quindi il contesto principale rimane quello dei videogiochi. Alcuni esempi in commercio che utilizzano Kinect come unico controller sono Kinect Adventures, Kinect Animals ed il gioco di ballo Dance Central.

Il costo relativamente basso insieme alle funizionalità di body-tracking che il dispositivo offre, ha fatto smuovere ed incuriosire la massa di sviluppatori software. Dopo qualche mese infatti il Web si è popolato di una moltitudine di applicazioni non strettamente legate al contesto dei videogames. Tra queste si possono citare programmi di visualizzazione di immagini, di riconoscimento del volto, plugin per software già esistenti e addirittura prototipi di riproduzione di una persona attraverso l'utilizzo di due Kinect.

Grazie a questo sensore è possibile eliminare mouse, tastiere e telecomandi: persone disabili potrebbero utilizzare questi dispositivi per abbattere numerose barriere che impediscono loro l'utilizzo della tecnologia.

Il sensore può essere utilizzato anche nell'ambito della robotica, ad esempio utilizzando la visione artificiale per far muovere degli automi.

Inoltre potrebbe essere utilizzato per far volare un elicottero o per far muovere un piccolo veicolo, evitando ostacoli mediante la creazione di una mappa 3D dell'ambiente. Il dispositivo permette anche di risparmiare enormi budget per la realizzazione un sistema di motion capture. Infine si potrebbero avere applicazioni anche per l'intrattenimento e nel campo della medicina.

La prima cosa che bisogna fare per lavorare con questi sensori è analizzarne l'hardware e il supporto fornito agli sviluppatori. In secondo luogo bisogna scoprire i driver e le funzionalità fornite dalle librerie.

2.2 Hardware

Kinect è dotato di telecamera RGB, sensore di profondità a raggi infrarossi composto da un proiettore a infrarossi e da una telecamera sensibile alla stessa banda. La telecamera RGB ha una risoluzione di 640 x 480

pixel, mentre quella a infrarossi usa una matrice di 320 x 240 pixel.

È presente anche di un insieme di microfoni utilizzato dal sistema per la calibrazione dell'ambiente in cui ci si trova, mediante l'analisi della riflessione del suono sulle pareti e sull'arredamento. In tal modo il rumore di fondo e i suoni del gioco vengono eliminati ed è possibile riconoscere correttamente i comandi vocali.

La barra del Kinect è motorizzata lungo l'asse verticale e segue i movimenti dei giocatori, orientandosi nella posizione migliore per il riconoscimento dei movimenti.

Figure 5.: L'interno del Kinect

Per osservare il meccanismo che permette al Kinect di muoversi è necessario rimuovere la plastica e le 4 viti sotto la base d'appoggio.

Qui sotto è situato il motore che permette alla periferica di ruotare. I componenti non sono molto robusti, escluso il piccolo motore tutti gli ingranaggi sono in plastica e quindi facilmente usurabili.

Dato che le immagini vengono elaborate direttamente sul Kinect, Microsoft ha inserito nella periferica due schede ed una barra di supporto metallico in parallelo, separati da quattro distanziatori metallici. Sul lato è montata una piccola ventola per il raffreddamento, che evita il surriscaldamento e il danneggiamento del dispositivo.

Un particolare difficile da notare è la *cella di Peltier*[1] posta tra l'IR e la barra metallica che svolge il ruolo di sistema di raffreddamento. Tra la telecamera RGB

[1] La cella di Peltier è un dispositivo termoelettrico costituito da molte giunzioni ad effetto Peltier in serie;

e il proiettore IR è situato anche un piccolo LED di stato.

Il dispositivo è dotato di un array di quattro microfoni collegato alla scheda madre con un connettore a cavo unico. L'array di microfoni permette al Kinect di ricevere i comandi vocali.

I microfoni sono tutti e quattro orientati verso il basso, tre sono sul lato destro del dispositivo ed uno sul lato sinistro. L'orientamento dei microfoni non è casuale: la scelta è stata dettata da Microsoft in quanto ritiene che l'orientamento verso il basso sia quello ottimale per la raccolta del suono.

Figure 6.: Componenti del dispositivo Kinect

Per far sì che i comandi vocali funzionino al meglio è necessario calibrare l'array di microfoni ogni qual volta si cambia la disposizione dei mobili nella stanza in cui è montato il Kinect.

Per alimentare la periferica, Microsoft usa ben 12 Watt mentre le porte USB sono in grado di fornire in media 2,5 Watt di potenza. Pertanto Kinect necessita anche di un cavo di alimentazione.

Ora che è stato descritto il sistema visivo ed uditivo del dispositivo bisogna capire come avviene l'elaborazione dei dati.

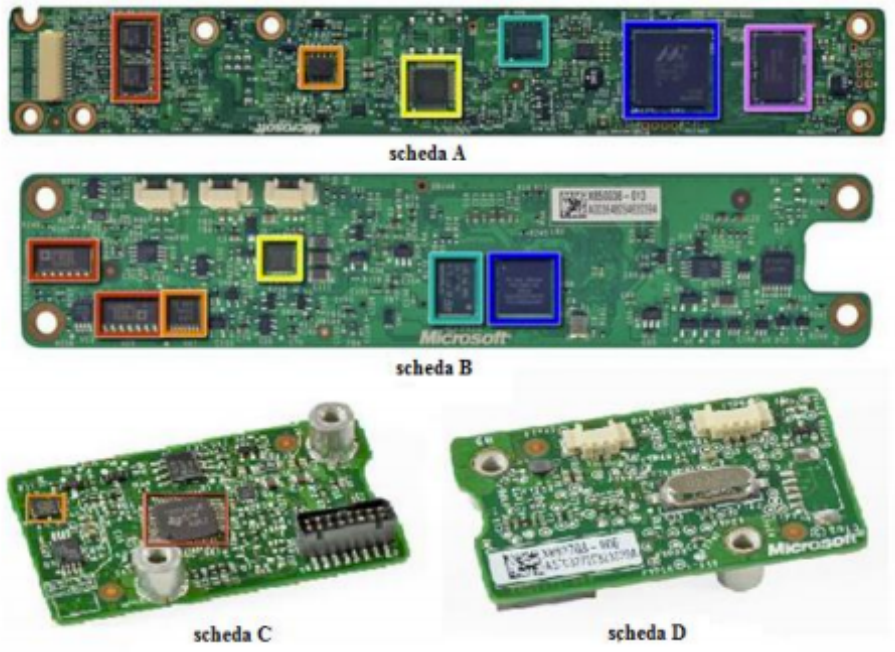

Figure 7.: Schede hardware poste all'interno del Kinect.

La scheda madre ha sei chip. Osservando la scheda A della figura 8, da sinistra a destra sono rispettivamente montati:

- stereo ADC con microfono preamplificato (Wolfson Microelectronics WM8737G);
- N-Channel PowerTrench MOSFET (Fairchild Semiconductor FDS8984);
- controller USB 2.0 hub (NEC uPD720114);
- Pacchetto SAP 6 mm x 4,9 mm – SPI flash (H1026567 XBOX1001 X851716-005 Gepp);
- SoC per il controller dell'interfaccia della macchina fotografica (Marvell AP102);
- SDRAM DDR2 512 megabit (Hynix H5PS5162FF).

Nella scheda B sono montati:

- 2 CMOS Rail-to-Rail amplificatore d'uscita a basso costo (Analog Devices AD8694);
- un campionatore e convertitore A/D 8 bit ad 8 canali, con interfaccia 12C (TI ADS7830I);
- Allegro Microsystems A3906;
- una memoria Flash 1Mb x 8 oppure 512Kb x 16 (ST Microelectronics M29W800DB);

- un processore d'immagini Soc Sensor (PrimeSense PS1080-A2).

Infine la scheda C dispone di un controller audio USB frontale e centrale (TI TAS1020B) e sul lato sinistro della scheda si può vedere un accelerometro (Kionix MEMS KXSD9) che probabilmente è utilizzato come stabilizzatore d'immagine.

Un'ultima cosa importante è il cavo USB proprietario che la periferica Microsoft usa per collegarsi alle normali console. Questo cavo potrebbe non essere riconosciuto dai sistemi operativi su PC, ma ci sono due alternative. La soluzione più semplice è utilizzare un adattatore che di norma viene venduto con il Kinect che Microsoft inserisce nella confezione per questioni di compatibilità con le prime versioni delle console XBOX 360. La seconda soluzione è costruire un adattatore ad hoc.

2.2.1 Caratteristiche tecniche

Il sistema è teoricamente in grado di misurare le distanze all'interno di un area di 2 metri con un margine di errore di 1 cm; questi parametri di precisione vengono forniti direttamente da Microsoft.

Nella seguente tabella sono riportate le caratteristiche tecniche di Microsoft Kinect.

Campo visivo (in gradi)	58° H, 45° V, 70° D
Risoluzione x/y (a 2 m dal sensore)	3 mm
Risoluzione z (a 2 m dal sensore)	10 mm
Range di lavoro	0.8 m – 3.5 m
Interfaccia	USB 2.0
Consumo	2.25 W
Immagine di profondità	320 x 240 pixel
Immagine a colori RGB	640 x 480 pixel
Frame-rate	30 fps
Stream audio	4 canali 16 bit (fc 16KHz)

Table 1.: Caratteristiche tecniche di Microsoft Kinect

2.3 Driver

2.3.1 OpenNI

OpenNI ovvero Open *Natural Interaction* è un *framework* sotto licenza *GNU GPL*[2] indipendente dalle piattaforme di lavoro e multi-linguaggio che definisce le API per scrivere applicazioni che usano le *Natural Interaction*.

L'intento di *OpenNI* è creare uno standard API per svolgere due compiti:

- comunicare con sensori visivi ed audio, per percepire figure e acquisire suoni;
- sviluppare funzionalità software (middleware) per analizzare e elaborare dati video ed audio registrati in una scena.

Il *middleware* permette di scrivere applicazioni che elaborano dati senza doversi preoccupare del sensore che li ha prodotti.
Il framework OpenNI è un livello astratto che fornisce l'interfaccia tra i dispositivi fisici e componenti *middleware* (Figura 9).

Il livello più alto rappresenta il software che fa uso di *OpenNI* implementando le *Natural Interaction*. Il livello centrale (*middleware*) rappresenta l'interfaccia *OpenNI* con le sue capacità di comunicare con i vari sensori e con le funzionalità disponibili (*Skeleton Tracking, Hand Tracking*, ecc.).

Il livello più basso è il livello hardware composto da tutti i sensori che possono inviare dati audio o visivi. Questi componenti sono indicati come moduli ed attualmente quelli supportati dalle API sono:

Moduli per i sensori

- sensore 3D
- fotocamera RGB
- dispositivo audio (un microfono o un array di microfoni)

[2] La GNU General Public License, comunemente indicata con l'acronimo GNU GPL o semplicemente GPL, è una licenza per software libero [fonte: Wikipedia, 2012].

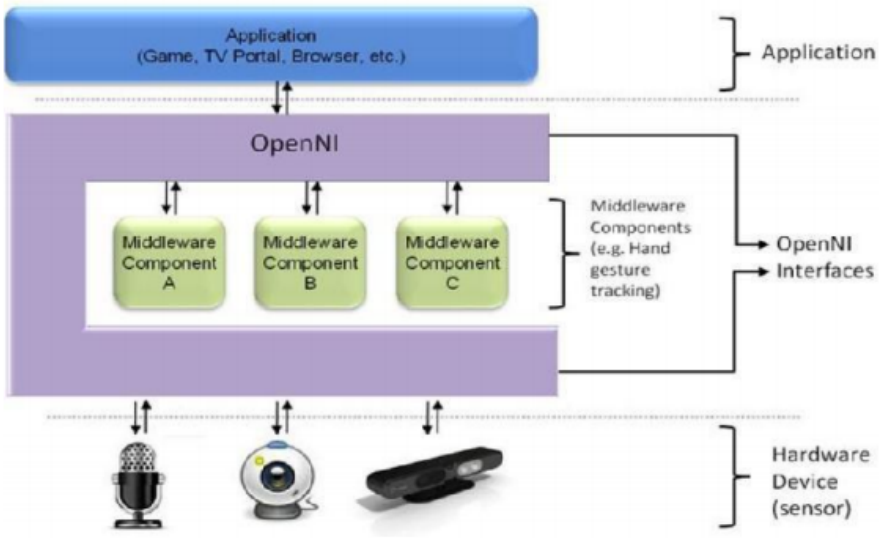

Figure 8.: Astrazione dei livelli di funzionamento di OpenNI.

Middelware componenti

- Analisi totale del corpo: è un componente software che elabora i dati sensoriali e genera informazioni relative al corpo come una struttura dati che descrive le articolazioni, il loro orientamento, il centro di massa del corpo e molto altro.
- Analisi della mano: è un componente software che elabora i dati sensoriali, individua la sagoma di una mano assegnando un punto alla posizione del palmo.
- Rilevamento gesto: è un componente software che identiica gesti predefiniti (*Push*, *Wave*, *Circle*) associandoli ad eventi.
- Analisi della scena: è un componente software che analizza l'immagine della scena al fine di produrre informazioni come individuare più persone nella scena, trovare le coordinate del piano, separare oggetti in primo piano da quelli sullo sfondo.

OpenNI ha rilasciato i driver e i propri codici sorgente per far sì che le sue librerie vengano implementate su più architetture possibili e su qualsiasi sistema operativo, in modo da accelerare l'introduzione di applicazioni di Natural Interaction sul mercato.

Con l'uscita del Kinect la popolarità di OpenNI è nettamente aumentata, grazie anche alla creatività dei numerosi sviluppatori che lavorano con queste librerie. Va sottolineato che OpenNI non è Kinect, ma la facilità del framework di comunicare con qualsiasi sensore ha solo facilitato l'uso del dispositivo Microsoft.

2.3.2 NITE

La libreria *OpenNI* restituisce dati a basso livello come mappe di profondità, mappe di colori, audio e quant'altro. *NITE* invece lavora ad un livello superiore: è un *toolbox* implementato sulle interfacce *OpenNI* chefornisce funzionalità aggiuntive e facilita la creazione di applicazioni di controllo basato sul movimento delle mani dell'utente e sullo scheletro.

Attualmente NITE è disponibile per i sistemi Windows, Linux e Mac.

NITE contiene implementazioni per tutti i moduli della libreria *OpenNI*.

Il funzionamento di *NITE* è basato sui livelli illustrati in figura 10.

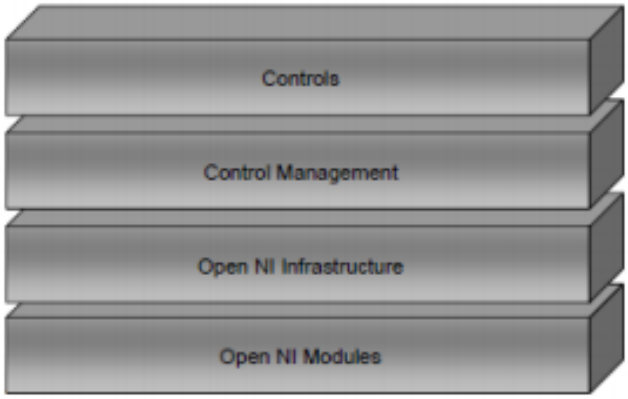

Figure 9.: Schema dei livelli NITE.

NITE è formato dai seguenti livelli:

OPENNI MODULES: i moduli di *OpenNI* supportati da *NITE* sono *Gesture Generator*, *Hands Generator* e *User Generator*. Inoltre supporta lo *Skeleton Tracking.*

OPENNI INFRASTRUCTURE: si veda il paragrafo 2.3.1.

CONTROL MANAGEMENT: riceve un insieme di punti e li indirizza verso il livello di controllo.

CONTROLS: ogni controllo riceve un insieme di punti e li traduce in un'azione specifica di tale controllo. Successivamente viene richiamata una funzione (*callback*) dall'applicazione che può cambiare il modulo di controllo attivo nel livello *Control*

Management. Questo definisce il flusso dell'applicazione.

Un esempio di controllo è il *controllo del punto*, che permette di registrare quando un punto viene creato, quando scompare e quando si muove. In realtà i controlli sono degli oggetti sempre in ascolto che attendono l'arrivo di nuovi dati ad ogni fotogramma. Le funzioni di *callback* saranno invocate solo quando un determinato evento si verifica.

2.4 Funzionamento del sensore

Per capire come funziona la periferica è possibile dividere il sistema in tre sotto blocchi: il monitoraggio dei movimenti, il riconoscimento vocale ed il motore.

La prima cosa che interessa ad un utente è farsi riconoscere. Questo compito è svolto dal sistema ottico, che permette di monitorare i movimenti in tempo reale. La struttura è molto complicata ma fornisce funzionalità che fino ad ora erano disponibili solo a fronte di spese notevoli.

Il sistema, come abbiamo visto, è composto principalmente da due parti: un proiettore IR e una fotocamera RGB.

La prima cosa che la periferica fa è creare una mappa di profondità della scena separando l'utente dagli oggetti inanimati.

A seconda della distanza dal sensore, le figure compariranno in diversi colori sullo schermo: gli oggetti in grigio scuro sono quelli più lontani, in grigio chiaro quelli più vicini. Le figure umane che vengono riconosciute possono essere blu, verde, rosso, e così via.

Per creare la mappa di profondità il proiettore IR del Kinect getta un fascio di raggi infrarossi (Microsoft ha assicurato che non sono pericolosi per il corpo e per la vista).

I raggi riflessi vengono catturati dalla telecamera ad infrarossi e con un algoritmo viene determinato quanto può essere lontano o vicino un punto. Sulla base di queste informazioni è possibile assegnare una tonalità di grigio ad oggetti più o meno distanti.

L'immagine acquisita dal sensore viene fatta passare in diversi filtri, in modo tale che il dispositivo possa capire cosa è una persona e cosa non lo è. L'intero

sistema segue delle linee guida, riguardanti la conformazione generale del corpo.

Questo permetterà in fase di calibrazione di non confondere gli oggetti con le persone.

Non tutte le persone hanno però la stessa conformazione fisica, inoltre spesso vengono utilizzati indumenti larghi o cappelli. Per questo vengono inseriti tra le linee guida degli algoritmi di riconoscimenti possibili cappelli o maglioni larghi.

Quando questa fase di calibrazione è terminata il dispositivo converte la parte dell'immagine relativa all'identificazione del corpo in uno scheletro che nella fase di tracking permette il movimento delle articolazioni, escluse per ora quelle delle dita.

L'intero sistema lavora a 30 fps ed ha 200 pose comuni per lo scheletro precaricate. Nel caso l'utente faccia un movimento che impedisca alla telecamera di riconoscere il gesto fatto, l'algoritmo userà una delle pose tra quelle presenti che più si adatta al caso per non perdere il tracciamento dell'utente.

La seconda funzionalità importante è il riconoscimento vocale. Abbiamo visto infatti che il Kinect ha un array di quattro microfoni pronti per essere usati a tale scopo.

Il sottosistema microfoni ha come obbiettivo di essere sensibile al riconoscimento delle voci fino a 10 metri di distanza cercando di ignorare rumori ambientali.

La larghezza del dispositivo Kinect è dovuta proprio al sistema di microfoni. Durante i suoi lavori Microsoft ha effettuato test in 250 abitazioni utilizzando 16 microfoni disposti in modo differente.

La soluzione ottima è stata l'array di quattro microfoni rivolti verso il basso, in modo da mantenere pulita la parte anteriore della periferica.

L'array funziona meglio nel raccogliere le voci a distanza, ma necessita di aiuto. C'è un unità di elaborazione a bordo del *Kinect* che toglie il rumore che si crea in prossimità dei sistemi surround 5.1, mentre un secondo sistema software *Beam Forming* agisce con la telecamera per capire dove si sta creando una possibile fonte di suoni intorno all'utente.

Questo permette di aiutare il *Kinect* a capire quando non è l'utente a parlare ma altre persone intorno a lui.

Il sistema di riconoscimento vocale, attivo solo su console, ha un modello acustico per ogni singolo paese che comprende anche diversi dialetti regionali. I microfoni sono in ascolto in ogni momento rendendo il sistema *Kinect open-mic*.

A questo punto rimane da capire come funziona il sotto blocco motore. L'idea di inserire un piccolo motore all'interno della base inferiore del Kinect è dovuta alle necessità di calibrazione nelle diverse abitazioni europee, asiatiche ed americane. Per Microsoft la telecamera doveva essere in grado di muoversi in su ed in giù per calibrare ogni singolo spazio, effettuando movimenti di circa 30 gradi.

Un'altra funzionalità importante del motore è quella dello zoom per la fotocamera, che permette di espandere lo spazio visivo.

Questa funzionalità è stata progettata per la video chat di *Kinect*, in modo che se più utenti sono nella scena ed uno viene tagliato il motore gestisce in automatico lo zoom della fotocamera per far entrare tutti i partecipanti della conversazione sullo schermo.

2.4.1 Calcolo della mappa di profondità

Le immagini di profondità semplificano molti problemi di *computervision* e di interazione come ad esempio:

- rimozione del background e segmentazione della scena;
- tracking di oggetti e persone;
- ricostruzione 3D degli ambienti;
- riconoscimento della posa del corpo;
- implementazione di interfacce basate su gesti.

La mappa di profondità della scena è un'immagine M di dimensione mxn, in cui ciascun pixel p(x, y) codifica la distanza nella scena 3D del punto (x,y) dal sensore.

In letteratura esistono molte tecniche per calcolarla e le più utilizzate sono: la *stereo triangolazione*, che calcola la profondità di un oggetto combinando le immagini catturate da due telecamere, la tecnica *time-of-flight*, che invece utilizza solo una telecamera calcolando la distorsione che subisce un segnale luminoso proiettato sugli oggetti ed infine la *proiezione di pattern*.

Questa ultima tecnica utilizza un sistema di visione stereo costituito da una coppia proiettore-telecamera. Nella scena viene proiettato un pattern luminoso (infrarosso) noto e la profondità degli oggetti è calcolata studiando la sua distorsione sugli oggetti. È possibile implementare questa tecnica con varie tecnologie:

- proiezione di linee e studio della loro curvatura sugli oggetti: non molto veloce e soggetta a disturbi quando gli oggetti sono in movimento;
- proiezione di pattern 2D periodici e studio del loro scostamento quando colpiscono gli oggetti: l'informazione 3D è ottenuta in real-time ma non è in grado di lavorare su lunghe distanze per via della distorsione del pattern;
- proiezione di pattern 2D pseudo-casuali: anche in questo caso i pattern sono 2D ma la loro casualità permette di ottenere accurate mappe 3D in real-time con un sistema molto semplice ed economico.

Figure 10.: Esempio di proiezione di pattern 2D.

In questo tipo di sensori la mappa di profondità della scena viene costruita utilizzando la tecnica della proiezione di pattern pseudo-casuali, mediante un sistema di visione stereo costituito da un proiettore IR e da una telecamera sensibile alla stessa banda. Questa tecnologia è stata brevettata nel 2005 da Zalevsky, Shpunt, Maizels e Garcia, sviluppata e integrata in un chip dalla compagnia israeliana *PrimeSense.*

Questa tecnica si basa su 3 elementi principali:

1. proiettore di pattern pseudo-casuali IR;
2. telecamera IR (in tecnologia CMOS);
3. unità di controllo (chip PS1080).

Il proiettore è molto semplice ed economico ed è costituito da un emettitore di raggi IR e da un generatore di pattern che devia tali raggi nella scena imprimendo ad essi angolazioni pseudo-casuali. Una volta proiettato il pattern, la telecamera acquisisce l'immagine IR della scena contenente il pattern distorto

e la invia all'unità di controllo che costruisce così la mappa di profondità della scena.

Con questo sistema è necessario acquisire una singola immagine e quindi utilizzare un singolo algoritmo di matching per determinare la profondità degli oggetti (dato che l'altra immagine è costituita dal pattern originale che è noto).

L'unità di controllo, conoscendo la struttura del pattern proiettato, calcola lo scostamento fra i punti proiettati e quelli ripresi dalla telecamera determinando in questo modo la mappa di profondità della scena.

La dimensione dei punti proiettati, la loro forma e orientazione non è costante ma dipende dalla distanza dal sensore. Il brevetto individua tre differenti tipologie di punti per tre differenti regioni dello spazio come mostrato in figura 11: una prima regione R1 (0.8 – 1.2 m) in cui si ha la massima risoluzione, una seconda regione R2 (1.2 – 2 m) con una buona accuratezza e una terza regione R3 (2 – 3.5 m) dove l'accuratezza è scarsa.

PrimeSense non ha solo sviluppato un nuovo sistema di acquisizione della mappa 3D della scena, ma soprattutto una tecnologia capace di elaborare questi dati realizzando molti task di processing 3D. Infatti il chip PS1080 ha al suo interno molte funzionalità di processing per il tracciamento, per la ricostruzione della scena e per il riconoscimento di gesti.

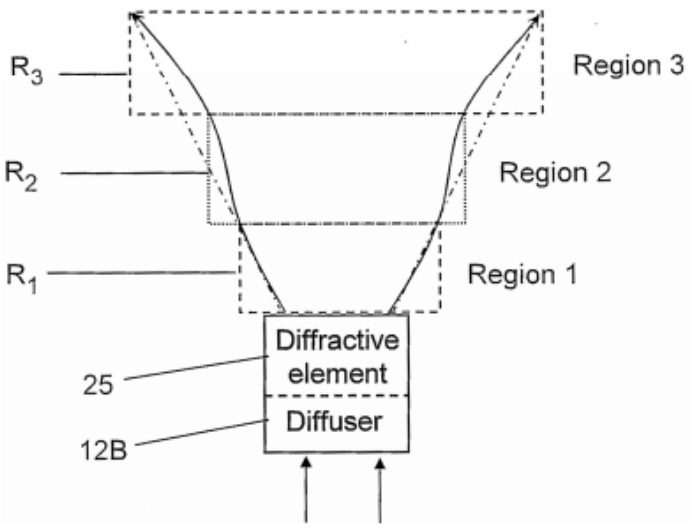

Figure 11.: Regioni dello spazio individuate dal brevetto *PrimeSense*.

3 STRUTTURA DEL PROGETTO REALIZZATO

Osservando la soluzione proposta per realizzare lo schermo pervasivo, si può notare, con un alto livello di astrazione, che è caratterizzata dall'integrazione di tre macro blocchi principali:

1. riconoscimento dell'attenzione delle persone davanti al display;
2. identificazione dell'osservatore;
3. interazione naturale.

In questo capitolo ci si concentrerà il più possibile sul comportamento del singolo componente, astraendo da tutti gli argomenti che riguardano più da vicino la loro integrazione e interazione.

3.1 Riconoscimento dell'attenzione delle persone davanti al display

3.1.1 L'attenzione

L'attenzione è la capacità dell'individuo di lavorare delle informazioni con la possibilità di selezionare gli stimoli ambientali ignorandone altri. Si assume che questa capacità sia limitata per ogni individuo e che nell'eseguire qualsiasi compito venga richiesta una certa quantità della capacità totale.

La misura dell'attenzione umana è un campo di ricerca attivo che comprende molte discipline (ad esempio le scienze cognitive, psicologia, intelligenza artificiale e data mining); un metodo generale per misurarla o classificarla ancora non esiste [3].

3.1.2 Approcci usati in letteratura

3.1.2.1 Stima dell'attenzione in base all'analisi del comportamento

Nel lavoro di [3] si cerca di stimare l'attenzione delle persone che transitano davanti a uno schermo pubblico in base alle variazioni del comportamento degli utenti e del contenuto informativo del display. Per catturare i dati necessari viene utilizzato il Kinect

essendo in grado di catturare immagini di profondità e immagini RGB. Viene utilizzato un modello di attenzione molto simile a quello introdotto da Wickens [4] in cui gli stimoli esterni possono essere filtrati in due modi: attivamente (concentrando direttamente l'attenzione su qualcosa, top-down) o passivamente (percezione degli stimoli in input, bottom-up).
Inoltre si assume che un uomo abbia una quantità limitata di attenzione e che possa essere distribuita a diversi processi contemporaneamente, in base a differenti priorità. Si tiene conto solamente degli stimoli esterni, in quanto quelli interni e lo stato emotivo (top-down) non possono essere analizzati dal sistema.

La seguente tabella mostra le varie tipologie di passanti che sono state osservate, i numeri da zero a 10 in ordine crescente rappresentano un indice per misurare l'attenzione.

Tipologia di passante	**Valutazione**
Indifferente: non ci sono reazioni visibili, nessun impatto sul comportamento, niente di osservabile	**0:** nessuna reazione, il soggetto non è nel campo di vista, nessun contatto con gli occhi
	1: nessuna reazione, il soggetto potrebbe essere nel campo di vista ma è lontano, il contatto con gli occhi non è percepibile
Glimpser ("scrutatori"): minimo impatto sul movimento della testa, durata dello sguardo breve, nessun impatto sul movimento (direzione o velocità)	**2:** occhiata molto corta, minima torsione della testa in direzione dello schermo, nessun impatto sul movimento, il soggetto è a media distanza dallo schermo
	3: occhiata di media durata, la testa si gira verso lo schermo, il soggetto ha notato il display ed è vicino
	4: occhiata di lunga durata, la testa è girata verso lo schermo, il soggetto ha percepito il contenuto
Osservatori: impatto sul movimento e sul comportamento, il soggetto guarda consapevolmente lo schermo	**5:** sguardo breve e consapevole, nessuno impatto sul movimento
	6: sguardo di media durata e consapevole, movimento definito della testa, diminuzione della velocità, il soggetto è a media

	distanza dallo schermo, lieve torsione della testa
	7: sguardo lungo e consapevole, impatto sul movimento, rallentamento, cambio nella direzione, media torsione della testa, il soggetto è in avvicinamento
Stopper: il soggetto è concentrato sullo schermo, impatto visibile sul movimento e sul comportamento, il soggetto annulla le precedenti attività, sosta molto lunga	**8:** sguardo intenso, sosta breve, torsione della testa ma il corpo non si gira verso lo schermo, cambio distinto della velocità
	9: sguardo intenso, sosta di media durata, orientamento del corpo verso lo schermo
	10: sguardo intenso, sosta di lunga durata, corpo orientato verso schermo, pieno impegno

Table 2.: Varie tipologie di passanti davanti ad un display

Le caratteristiche che sono state estratte per analizzare il comportamento sono:

- velocità
- direzione
- accelerazione
- distanza del soggetto
- posizione
- linearità del percorso del soggetto

Per l'analisi dei movimenti della testa e degli occhi è stato utilizzato un *face tracker* sviluppato da terze parti (SHORE face tracker [5])che permette di misurare l'angolo di torsione della testa rispetto allo schermo.

In [3] è stato rilevato che una percezione cosciente del contenuto del display è possibile solamente per un angolo di torsione della testa uguale a 0 C°, inoltre una deviazione dello sguardo di +-45° permette solo una percezione periferica del contenuto.

3.1.2.2 Relazione tra il controllo della postura e l'attenzione

In [6] vengono presi in considerazione diversi studi che mettono in relazione l'attenzione, la postura e l'andatura in giovani adulti. Il controllo della postura è stato definito come il controllo della posizione del corpo nello spazio con lo scopo di bilanciamento e orientamento.

Per lo studio dell'attenzione e della postura è stato utilizzato il paradigma del "dual task" in cui il controllo della postura (il task principale) viene eseguito concorrentemente al task secondario (diverso a seconda degli esperimenti); la misura in cui le performance di uno e di entrambi i task diminuiscono indica un'interferenza tra i processi che controllano i due task e quindi la misura in cui i due task condividono risorse attenzionali.

Il primo studio che viene analizzato è quello di in cui si è verificato che task di bilanciamento difficili (le misurazioni sono state effettuate attraverso delle piattaforme di forza per calcolare il centro di pressione) interferiscono sulla memoria spaziale (visiva) ma non su quella verbale, visto che il controllo posturale richiede un'elaborazione visiva/ spaziale. Si è inoltre dedotto che negli adulti non tutti i task cognitivi interferiscono nello stesso modo nel controllo posturale.

Un altro studio è quello di in cui si è considerato l'effetto di task concorrenti sul controllo dell'andatura. I parametri (tempo per percorrere un passo e *double support time*[3]) sono stati misurati sia eseguendo un singolo task (solamente camminare) che eseguendo contemporaneamente altri task come: memorizzazione di alcune cifre, un piccolo task motorio (aprire e chiudere un bottone di un cappotto), tamburellare con le dita a 5 o più Hz. Si è visto che l'andatura subisce dei cambiamenti quando si tamburellano le dita (il numero di passi diminuisce) e quando viene eseguito il task motorio in sincronia al task mnemonico (cambia il *double support time*).

Un ulteriore studio che viene esaminato è quello di in cui ai soggetti viene chiesto di reagire ad uno stimolo uditivo nel minor tempo possibile mentre stanno

[3] Il double support time è l'intervallo in cui il peso del corpo è supportato da entrambe le gambe

seduti, in piedi (sia con una base di sostegno [4] normale che ridotta) e mentre camminano (single support phase [5] e double support phase). I risultati di questo studio indicano che il tempo di reazione è più veloce mentre i soggetti sono seduti e più lento quando sono in piedi e camminano. I tempi di reazioni sono maggiori quando i soggetti sono in piedi con una base di sostegno piccola rispetto a una base di sostegno normale. Inoltre i tempi di reazioni sono più lenti durante la single support phase in confronto a quelli della double support phase.

Dagli esempi precedenti è possibile osservare che è richiesta meno attenzione per compiti posturali facili come stare seduti o eretti (con i piedi alla larghezza delle spalle) rispetto a posizioni più impegnative (ad esempio il test di Romberg: in piedi a talloni uniti e braccia distese in avanti) o "strette", per riprendersi da perturbazioni esterne o rispondere a cambiamenti ambientali. Quindi con determinate posture la quantità di attenzione richiesta è minore e si possono dedicare più risorse attenzionali ad altri scopi.

3.1.2.3 Stima dell'attenzione di un utente attraverso il movimento degli occhi

In [7] per stimare l'attenzione di un utente davanti al monitor di un computer sono stati analizzati il movimento (rotatorio e traslatorio) della testa e la direzione dello sguardo. Viene inoltre misurata la distanza dal monitor e la distanza tra gli occhi in frame consecutivi. Effettuando test su adulti e bambini è stato notato che una persona che è attenta al contenuto dello schermo ha le seguenti caratteristiche:

- i movimenti della testa sono molto limitati
- anche i movimenti degli occhi sono limitati e rimangono concentrati nell'area dello schermo
- ci devono essere piccoli cambiamenti nella distanza tra gli occhi tra frame successivi

Un altro modello proposto da [8] fornisce un resoconto su come il processo cognitivo che regola l'attenzione e l'elaborazione lessicale di un testo determini quando e dove gli occhi si muovono durante la lettura. In [8]

[4] La base di sostegno è l'area all'interno del contorno formato da tutti i punti del corpo che sono a contatto con la terra

[5] La single support phase è la fase della camminata in cui il peso del corpo è supportato da una sola gamba

viene dimostrato che l'attenzione viene allocata serialmente, ad una parola per volta, quindi l'attenzione è intrinsecamente collegata al processamento lessicale. In altre parole, l'elaborazione visiva necessaria per il processamento lessicale non è sufficiente all'identificazione delle parole, l'identificazione richiede la focalizzazione dell'attenzione sulla parola che si sta processando. Pertanto l'attenzione è necessaria per unire le features degli "oggetti" visivi in modo che possano essere codificati in una rappresentazione singola e unificata.

Dalle precedenti affermazioni viene dedotto che durante la lettura il movimento degli occhi avviene in una finestra relativamente stretta, in cui avviene l'dentificazione delle lettere o parole correnti. Il movimento più ampio si ha quando si deve iniziare a leggere una nuova riga di testo, in cui si cerca, attraverso la visione periferica, di identificare il margine sinistro. Il processamento lessicale inizia non appena l'attenzione viene focalizzata sulle feature visive della parola. Di conseguenza bisogna fare una distinzione tra un primo stage in cui si riserva l'attenzione al processo visivo ed in seguito uno stage in cui l'attenzione è concentrata all'elaborazione lessicale. La motivazione di questo fatto è che le informazioni sulla retina non vengono trasmesse immediatamente al cervello, ma, come indicano recenti esperimenti, c'è un ritardo approssimativo di 50 ms. A questo bisogna aggiungere il tempo necessario per spostare gli occhi da una posizione all'altra che è all'incirca di 125 ms, quindi il tempo complessivo per capire e spostarsi alla parola successiva è di 175 ms circa.

3.1.2.4 L'attenzione e il tempo

In viene studiata la relazione tra l'attenzione e il tempo. Infatti il processamento delle informazioni richiede tempo e la natura delle rappresentazioni mentali costruite dall'integrazione degli stimoli e dei segnali esterni cambia continuamente con esso.

Ad esempio, se il task è quello di determinare le caratteristiche più importanti in un'immagine, le prime occhiate individueranno le caratteristiche salienti, mentre le occhiate successive saranno pressochè a caso. Il fatto che le performance diminuiscano con il passare del tempo suggerisce che le informazioni sulle caratteristiche principali degradino in funzione del

tempo, anche se sono lo scopo principale del task. L'idea è che le informazioni salienti percepite dominino la prima forma della rappresentazione, quindi quando l'osservatore risponde velocemente allo stimolo l'importanza di quello che si è percepito ha più priorita rispetto alla rilevanza del task. Ciò nonostante, allo scorrere del tempo, la rappresentazione cambia e diventa più sofisticata visto che vengono integrate altre informazioni, come la conoscenza preliminare e i goal dell'osservatore. Di conseguenza quando gli osservatori sono più lenti e viene speso più tempo per il processamento visivo, l'attenzione è guidata da una conoscenza di più alto livello.

3.1.3 In riferimento al nostro sistema

Come descritto nelle sezioni precedenti, considerando di trovarsi nelle vicinanze di un display, l'attenzione può essere misurata utilizzando principalmente i seguenti parametri:

- velocità
- direzione
- accelerazione
- distanza del soggetto
- posizione
- linearità del percorso del soggetto

Quindi un soggetto viene considerato attento al contenuto del display se la sua velocità, direzione, accelerazione e distanza diminuiscono, se è orientato con la testa e il corpo verso il display o se effettua un percorso lineare per avvicinarsi allo schermo.

Ulteriori discriminanti potrebbero essere:

- se il soggetto ha qualcosa in mano (ad esempio un cellulare o un giornale) che potrebbe interferire con l'attenzione da dedicare al display pubblico (diminuzione dell'attenzione, Figura 12 e 13);
- se il soggetto sta parlando con un'altra persona, che potrebbe distrarlo dal contenuto del display (diminuzione dell'attenzione)
- se è sbilanciato con il peso su una gamba e non guarda in direzione del display (diminuzione dell'attenzione, Figura 14).

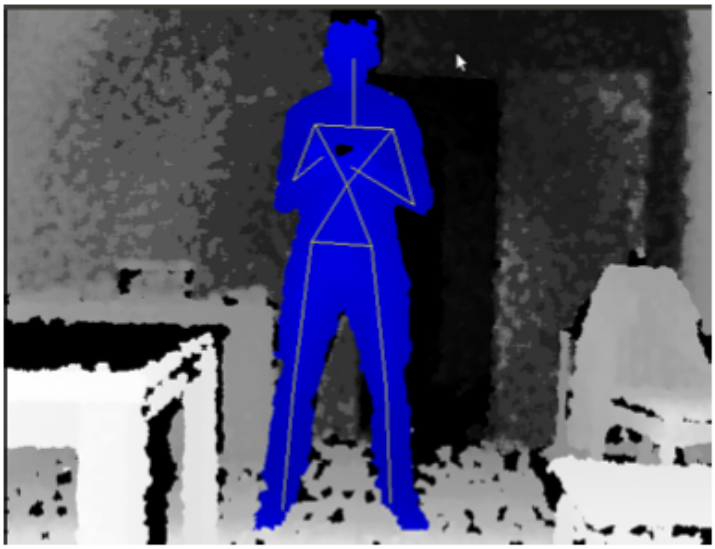

Figura 12.: Soggetto che usa il cellulare.

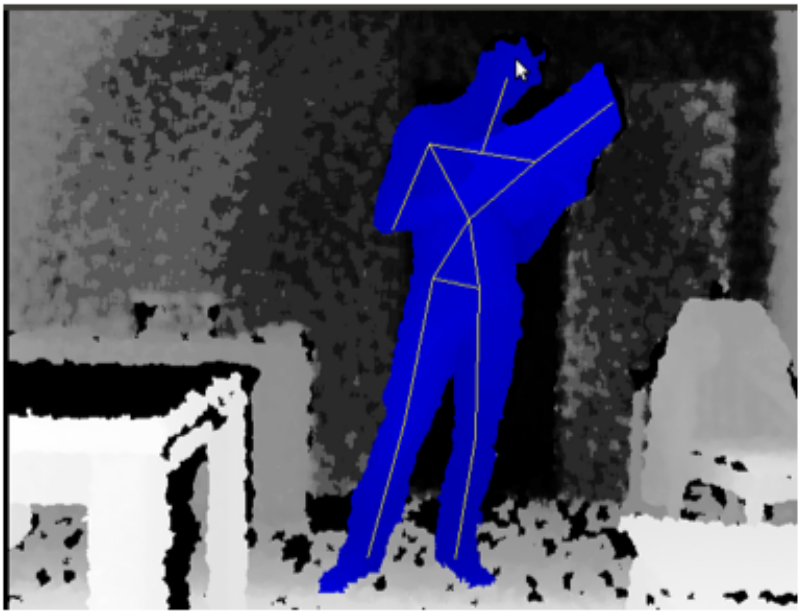

Figure 13.: Soggetto che legge il giornale.

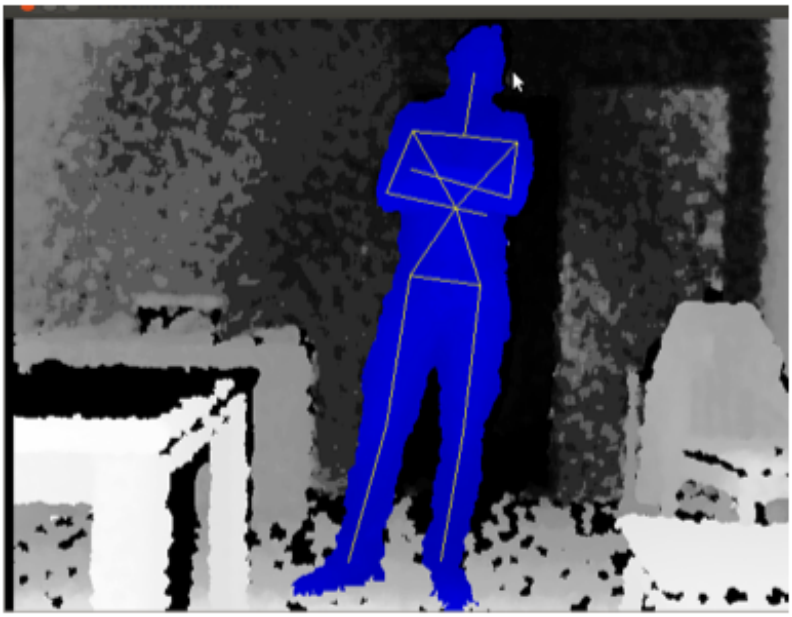

Figure 14.: Soggetto in piedi con il peso sbilanciato su una gamba e braccia conserte.

3.1.4 Progetto

Questo prototipo è ispirato al lavoro Alois Ferscha nel progetto europeo SAPERE.

E' stato scelto di sviluppare un prototipo per il riconoscimento dell'attenzione degli utenti individuati dal sensore Kinect. E' stata sfruttata la capacità di OpenNI di ottenere la *depth image (*paragrafo 2.4.1*)* della scena e di calcolare la posizione di vari punti del corpo nello spazio. Il progetto è suddiviso nei seguenti package:

- *control*: contiene le classi che si occupano di gestire i dati provenienti dal sensore Kinect e di analizzare i parametri calcolati.

  -*Kinect Manager*: crea e gestisce tutti gli strumenti che interpretano i dati provenienti dal sensore: il *Depth Generator* (per il recupero delle informazioni sulla profondità), l'*UserGenerator* (per identificare gli utenti nella scena) con la capacità di riconoscere lo scheletro (*SkeletonCapability*) in modo da avere la posizione delle parti del corpo che interessano. Per ogni frame vengono aggiornate le posizioni di tutti i giunti dello scheletro presi in considerazione (testa, collo, spalle, gomiti, mani, torso, anche, ginocchia, piedi; alcuni servono solamente per la visualizzazione dello scheletro). Tutte le posizioni dei giunti sono espresse in *coordinate mondo* (il centro del sistema di riferimento è il sensore con l'asse z uscente dal sensore) e misurate in millimetri.

  -*UserAnalizer*: thread creato dal *KinectManager* al rilevamento di un nuovo utente in scena. A partire dai dati sulla posizione dello scheletro stima il livello di attenzione di un utente specifico, calcolando velocità, direzione delle spalle, parametri relativi alla postura e tempo di permanenza davanti allo schermo. Aggiorna infine l'interfaccia grafica.

- *featureExtractor*: contiene le classi che si occupano di estrarre i parametri necessari per ottenere il livello di attenzione di ogni singolo utente.

  -*SpeedCalculator*: calcola la velocità di un utente in base ai movimenti compiuti dal torso. La velocità viene calcolata facendo la

differenza tra due posizioni successive e poi dividendo tale distanza per il tempo impiegato a percorrerla.

-*ShoulderDirectionCalculator*: l'orientamento della parte superiore del corpo dell'utente viene calcolato usando la direzione della normale al corpo rispetto all'asse z del sensore. La normale alla parte superiore del corpo è la normale al piano formato dai seguenti punti: spalla destra, spalla sinistra e torso.

-*PoseParameterCalculator*: i parametri usati per caratterizzare la postura sono: base di sostegno, sbilanciamento dell'utente rispetto ad una posizione perfettamente centrata, single o double support phase, utente in piedi o seduto.

-*TimeCalculator*: il tempo totale di permanenza di un utente viene aggiornato durante il tracciamento prendendo come istante iniziale il momento in cui l'utente viene rilevato dal sensore. Il tempo viene azzerato se l'utente esegue una torsione con le spalle entro una certa soglia.

-AttentionPoseDetector: rileva una posizione "pensante" (Figura 15)

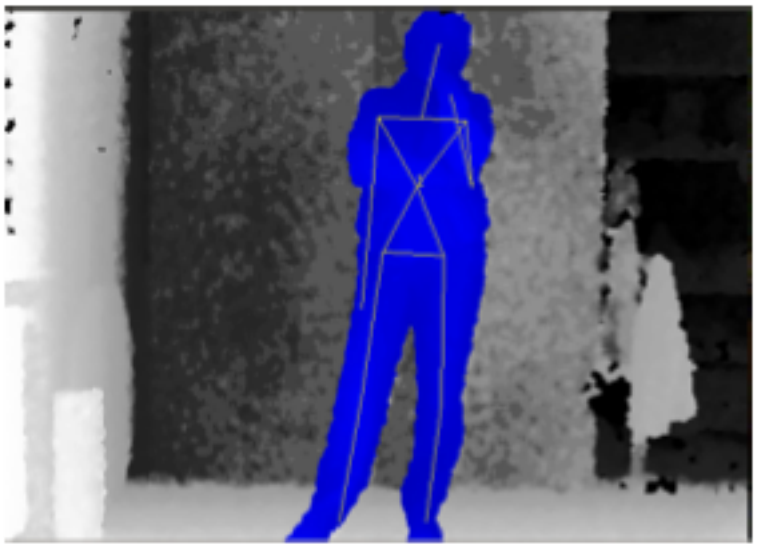

Figure 15.: Soggetto in una posizione che indica possibile attenzione.

- *Observer:* classe di supporto per la notifica della calibrazione di un utente.
  -*CalibrationCompleteObs*: comunica al KinectManager che la calibrazione è completata. La calibrazione permette l'acquisizione dei dati relativi allo scheletro senza la necessità che l'utente assuma una posa specifica [9].

- *util:* classi di supporto.
  -*PosAndTime:* rappresenta la posizione e il momento dell'acquisizione di un giunto dello scheletro.

  -*UserData*: contiene tutte le posizioni dei giunti dello scheletro e il centro di massa relativi ad uno specifico utente in un determinato frame.

  -*Vector3D*: rappresenta un vettore e le possibili operazioni che è possibile applicare: prodotto vettoriale, prodotto scalare, norma, modulo, angolo tra 2 vettori.

  -*LoadParam:* permette di leggere i parametri contenuti nel file di configurazione dell'applicazione.

- *View:* classi relativi all'interfaccia grafica e alla visualizzazione dei dati provenienti dal sensore Kinect.

  -*SceneDrawer:* interfaccia grafica per la visualizzazione della scena corrente, con l'evidenziazione degli utenti rilevati e il disegno dello scheletro di ciascuno.

  -*SkeletonDrawer*: si occupa di disegnare i segmenti che formano lo scheletro.

  -*AttentionIndexFrame*: interfaccia grafica per la visualizzazione dei risultati, in cui ad ogni utente è associato il relativo indice di attenzione stimato.

3.1.4.1 Indice dell'attenzione

Per la stima dell'indice dell'attenzione ad ogni feature misurata viene associato un valore che indica quanto un determinato vincolo è stato rispettato. Prendendo spunto dagli studi sull'attenzione già presenti

in letteratura si può dire che l'attenzione di un utente aumenta se:

- l'utente è seduto;
- non è sbilanciato ma ha una posizione centrata rispetto all'asse verticale del corpo;
- la base di sostegno è circa pari alla larghezza delle spalle;
- è in double support phase piuttosto che in single support phase;
- il corpo dell'utente è rivolto verso il sensore;
- l'utente è fermo o si muove molto lentamente;
- resta un certo tempo davanti al display per leggerne e capirne il contenuto.

Nella seguente tabella sono indicati i valori assegnati di default a ciascun aspetto precedentemente elencato.

Parametri utente	**Punteggio/Penalizzazione**
Postura	**+28:** seduto
	+14: in piedi
	+14: double support phase
	+7: single support phase
	+7: sbilanciato verso destra o sinistra
	+14: nessun sbilanciamento
	+14: base di sostegno normale
	+7: base di sostegno stretta o larga
Direzione delle spalle	**+14:** angolo minore di 15°
	+2: altrimenti
Velocità	**+14:** fermo
	+3: lento
	+1: veloce
Tempo	**+t*14/tempoNecessario**
Posizione "pensante"	**+2**, altrimenti **0**

Table 3.: valori per l'indice di attenzione

Dove t è il tempo totale che l'utente rimane davanti allo schermo con un angolo di torsione delle spalle ridotto e temp0Necessari0 deve essere deciso in base al numero di parole da leggere: ad esempio se il testo ha 50 parole, il tempo minimo necessario per leggerlo Ë 50*175:875° millisecondi. 175ms è il tempo minimo per capire una parola e spostarsi a quella successiva. In questo modo l'utente che rispetta tutti i vincoli ha un indice di attenzione vicino a 100, mentre decresce se i vincoli non vengono rispettati. Un utente si può

considerare attento se il suo indice di attenzione è maggiore di 80, mentre per valori inferiori a 60 l'utente si può valutare come non attento al contenuto dello schermo.

3.1.4.2 Punti critici e assunzioni

Dalle specifiche di OpenNI, la segnalazione che un utente è uscito di scena avviene circa 10 secondi dopo che l'utente è effettivamente fuori dal campo di vista e per ora non c'è modo di modificare questo valore.

Questa bassa reattività per l' applicazione considerata non è ideale: l'obiettivo infatti è che lo schermo reagisca in tempo reale all'uscita di un osservatore. Nell' ipotesi di uno scenario in cui un utente se ne va dallo schermo dopo avere interagito con esso, utilizzando questa specifica funzionalità dalle librerie OpenNI lo schermo continua a offrire un contenuto informativo relativo alla persona che se ne è andata ancora per 10 secondi circa.

Invece, sempre utilizzando le specifiche OpenNI, la segnalazione di un nuovo utente nella scena risulta immediata; tuttavia non si è utilizzato neanche quest'ultima: consideriamo lo scenario che l'utente contraddistinto da OpenNI con ID 2 se ne vada dalla scena, però per OpenNI questo è ancora in scena per circa ulteriori 10 secondi di conseguenza se questo utente volesse tornare subito a usufruire dello schermo OpenNI non lo rileverebbe.

Per evitare questi scenari si è preferito implementare un modo alternativo all'uso delle primitive di OpenNI. Dalla libreria OpenNI è possibile ottenere una struttura dati contenente degli interi ognuno dei quali si riferisce ad un determinato pixel dell'immagine. Se l'intero fa riferimento ad un pixel che è costituente di una determinata persona assume il valore del suo identificativo altrimenti assume valore zero. Questa struttura dati viene aggiornata con una frequenza di circa 30 HZ. Attuando un controllo quindi ogni qualvolta avviene l'aggiornamento su questa struttura riusciamo ad avere l'informazione aggiornata senza latenze di chi entra in scena e di chi esce. Questo ovviamente a discapito di un costo computazionale.

In riferimento sempre alle due librerie utilizzate (OpenNI, NITE), per una corretta segmentazione dell'utente, vengono fatte una serie di assunzioni riguardo alla sua posizione e a come si muove nella scena. Si possono verificare delle imprecisioni nei seguenti scenari: l'utente è troppo vicino ad altre

persone o oggetti nella scena (muri, sedie, ecc.), due utenti si toccano mentre si muovono, se ci sono occlusioni o l'utente esce di scena il suo ID può essere perso oppure scambiato erroneamente con un altro. Per un tracciamento corretto dello scheletro vengono fatte le seguenti assunzioni: la parte superiore del corpo dell'utente è all'interno della visuale e la distanza ideale dal sensore è di 2.5 metri; inoltre, vestiti troppo larghi e capelli lunghi potrebbero degradare la qualità del tracciamento.

Per quanto riguarda la calibrazione automatica (attiva di default), affinchè questa possa funzionare correttamente, la maggior parte del corpo dell'utente dovrebbe essere visibile ed essere distante almeno 1 metro dal sensore, in quanto funziona per utenti già in piedi e non per quelli seduti. Per completare la calibrazione più velocemente ed avere risultati migliori sarebbe bene seguire ulteriori raccomandazioni: l'utente dovrebbe essere rivolto verso il sensore e stare in piedi, le mani non dovrebbero nascondere l'area del torso, l'utente dovrebbe evitare movimenti veloci. Per il tracciamento dello scheletro vengono usate delle euristiche che forniscono dei dati più plausibili quando il valore di confidenza è zero (i giunti sono stati persi o sono ostruiti). Ad esempio se il valore di confidenza di un braccio è zero la sua posizione viene aggiustata in modo da essere a lato del corpo e puntare verso il basso. Le transizioni tra le posizioni tracciate e quelle calcolate tramite le euristiche avvengono in modo graduale su un certo numero di frame in modo da evitare salti.

3.2 Identificazione dell'osservatore

3.2.1 Concetto di trasparenza

L'identificazione dell'osservatore è di fondamentale importanza per definire a pieno il sistema studiato un *sistema pervasivo, in quanto* riusciamo in questo modo a garantire il concetto di *trasparenza* (approfondito nel primo capitolo).

Infatti, dando allo schermo la potenzialità di identificare almeno sotto alcuni aspetti l'osservatore, i contenuti mostrati possono automaticamente adattarsi a quella determinata persona.

Si ha così una esemplificazione dello sforzo nell'accedere alle informazioni di interesse portando un

numero sempre maggiore di aspetti computazionali in background.

3.2.2 Una possibile soluzione

Per raggiungere questo obiettivo, non appena viene rilevato un osservatore grazie alle informazioni fornite dal primo macro blocco, deve avvenire un accoppiamento fra la persona virtuale rilevata dal Kinect e la persona fisica.

Un possibile modo per effettuare questo accoppiamento può essere sfruttare il concetto di localizzazione:

- grazie al Kinect possiamo risalire alla distanza tra l'utente e il sistema Kinect-schermo (con il prerequisito che siano allineati). In questo modo si può stabilire un valore limite massimo del raggio entro il quale un utente può divenire effettivamente un osservatore, così quando un osservatore viene rilevato sicuramente è nel raggio prestabilito;
- secondo punto è l'utilizzo della periferica bluetooth. Il nostro sistema potrebbe cioè effettuare il calcolo della potenza del segnale bluetooth fra la propria periferica e quella di ciascun smartphone presente nel raggio di azione del segnale bluetooth. Dall'informazione della potenza è possibile ottenere una stima della distanza, in base alla quale il sistema riesce ad identificare l'osservatore fra tante altre persone, perché è colui che si trova entro quel determinato raggio prestabilito.
- Infine tramite una comunicazione, sempre via bluetooth ad esempio, fra le due entità ovvero lo schermo pervasivo e l'osservatore, quest'ultimo può fornire le sue informazioni allo schermo.

Purtroppo però la potenza del segnale bluetooth subisce delle grosse variazioni anche rispetto a piccole variazioni dell'ambiente circostante, come il passaggio di persone o superfici che riflettono particolari tipi di onde per cui è difficile ottenere delle informazioni accurate per il nostro caso.

3.2.3 Soluzione adottata

Dato le problematiche appena esposte si è percorso per raggiungere questo obiettivo una strada alternativa basata sull'interazione con un QR Code.

3.2.3.1 Struttura

Questo sistema prevede due entità principali:

- Un'applicazione per smartphone (Android), che costituisce la parte client;
- Un'entità che svolga il ruolo di server lato schermo pervasivo.

3.2.3.2 Interazione e comportamento

Il funzionamento di questo sistema può essere caratterizzato dai seguenti punti:

- L'applicazione per device deve dare la possibilità all'utente di poter configurare e memorizzare i suoi dati personali;
- Il server deve mostrare a video il QR Code con al suo interno l'encodifica di una stringa del tipo "Ipaddress:port?sessionID", e successivamente mettersi in attesa di ricevere e gestire connessioni TCP da parte di device client;
- L'applicazione per smartphone deve quindi fornire all'utente un'interfaccia per interagire con un QR Code, in modo da decodificare la stringa contenuta nel codice, ottenendo così le informazioni necessarie che permettono al device di "raggiungere" tramite TCP il server;
- Stabilita la connessione TCP fra lo schermo e il device quest'ultimo può procedere all'invio dei dati personali precedentemente memorizzati verso il server, previo riconoscimento tramite il sessionID che può conoscere solamente chi analizza il QRCode, in quanto generato casualmente.

3.2.3.3 Progetto

L'applicazione per device non ha una struttura complicata in quanto è formata da 3 sole Activity:

- *MainActivity.java*, costituisce la schermata principale, in cui l'utente potrà far partire l'identificazione;
- *SaveActivity.java*, consente all'utente di modificare e salvare le proprie informazioni personali;
- *ConnectActivity.java,* fa da 'background' alle operazioni di connessione e comunicazione, esplicitando all'utente che si è in fase di scambio dei dati, chiusa non appena le operazioni di riconoscimento finiscono, sia positivamente che non.

Le operazioni che coinvolgono scambio di dati in rete vengono eseguite in un worker Thread diverso da quello principale dell'applicazione, in modo da non bloccarlo in caso qualcosa vada storto nella comunicazione in rete.

I punti salienti dell'applicazione non sono molti; di seguito ne viene proposta una lista, senza però soffermarsi sul piano implementativo.

- Interfacciamento con l'applicazione per lo scanning di QR Code (zxing Barcode Scanner): avvio dell'applicazione e trattamento dei risultati dello scanning. Quindi l'applicazione sviluppata non deve occuparsi della decodifica di QRCode, già fornita dalle funzionalità di Barcode Scanner, ma si concentra sul trattamento di queste informazioni.
- Comunicazione tramite TCP come Client: utilizzo di un thread separato, semplice protocollo richiesta-risposta tra client e server, invio e ricezione dati come stringhe.
- Salvataggio dei dati in un file XML: nome, cognome, matricola, e-mail, anno accademico, corso di laurea.
- Layout dell'applicazione.

3.3 Interazione naturale

Spesso capita di avere a che fare con dispositivi di interazione dall'utilizzo non intuitivo o poco efficace, oppure con sistemi che necessitano di un tempo di apprendimento lungo. Nasce così la necessità dell'attuazione di un paradigma evoluto che renda l'interazione fra uomo e macchina equiparabile in

spontaneità alla quotidiana interazione fra uomo ed oggetti fisici.

Proprio il paradigma dell'*interazione naturale* mira ad offrire la possibilità di interagire attraverso *interfacce* che entrino il più possibile in simbiosi con le nostre azioni quotidiane (come ad esempio i gesti, le parole, le espressioni del viso, i movimenti) e che si confondano con l'ambiente circostante (ubiquitous computing, primo capitolo).

Ognuno dei cinque sensi umani deve poter rappresentare un possibile canale di comunicazione fra uomo e macchina.

L'interazione naturale riduce il divario fra i mezzi informatici ed il mondo fisico della vita di tutti i giorni e richiede che le interfacce siano diverse dalle tradizionali interfacce uomo-macchina basate su menù, icone, mouse e tastiera. In sintesi un'interfaccia per l'interazione naturale presenta le seguenti caratteristiche:

- gli schemi di interazione sono basati sui modelli di interazione persona-persona e persona-oggetto fisico;
- è in grado tramite apposito hardware e software di comprendere azioni ed intenzioni degli utenti;
- consente una elevata mobilità all'utente per una piena naturalezza nei movimenti;
- scompare il più possibile nell'ambiente e negli oggetti creandone un potenziamento nelle loro funzionalità;
- presta particolare attenzione ad aspetti estetici e cognitivi.

3.3.1 Interazione naturale dello schermo pervasivo

La tipologia di interazione messa a disposizione dal sistema studiato per comunicare con esso non prevede l'uso di nessun strumento aggiuntivo oltre che al nostro corpo, si interagisce completamente grazie allo spostamento di una mano. Ci avviciniamo quindi all'*interazione naturale* appena descritta.

In particolare possiamo individuare:

- spostamento della mano da destra a sinistra o viceversa, utilizzato per scorrere raccolte di foto;
- movimenti della mano caratterizzati da un certo numero di cambiamenti di direzione entro un certo tempo, ad esempio come se dovessimo cancellare qualcosa, questa *gesture* viene utilizzata per oltrepassare la fase di autenticazione qualora non disponessimo dell'applicazione per device.
- quest'ultima *gesture* porta invece un maggior contributo all' applicazione sviluppata in quanto consente di mappare la nostra mano con un oggetto a schermo più precisamente una *"mano-icona"*, per rendere questo approccio ancora più intuitivo (Figura 19). Questa immagine segue la nostra mano, permettendo in questo modo di premere diverse tipologie di pulsanti quasi come se fossero pulsanti fisici.

3.3.2 NITE

Prima di affrontare lo sviluppo di questa parte diamo qualche approfondimento sulle funzionalità della libreria utilizzata (NITE).

Come già accennato nel primo capitolo NITE è un *toolbox* che opera sulla libreria OpenNI.

Sebbene OpenNI sia in grado di monitorare i movimenti della mano, NITE mette a disposizione degli algoritmi per facilitare i programmatori non solo ad individuare un punto che identifichi il palmo della mano, ma anche ad elaborare i dati provenienti dal sensore in maniera tale da trarre delle informazioni più accurate sugli spostamenti effettuati dalla mano.

Questa parte relativa alle *gesture* è stata realizzata avvalendosi proprio di questa libreria.

La libreria è caratterizzata da particolari oggetti, *controlli,* che elaborano ed analizzano dati inerenti alla mano e al suo spostamento al fine di individuare specifici movimenti.

I controlli supportati da NITE sono i seguenti:

- **Push Detector**
  Questo controllo gestisce l'evento Push che consiste nel muovere la mano verso il sensore e

tirarla indietro. Il gesto Push può essere usato, per esempio, per selezionare un oggetto o aprire una cartella.

- **Swipe Detector**
  Rileva il gesto Swipe sia verso l'alto, il basso, a sinistra o a destra. Lo Swipe è un movimento breve in una specifica direzione dopo il quale la mano si ferma. Ad esempio, tale gesto potrebbe essere usato per sfogliare le pagine di un libro.
- **Steady Detector**
  Il controllo cerca di riconoscere quando la mano è ferma per un determinato lasso di tempo. Il gesto Steady avviene appunto quando la mano è completamente ferma o quasi e la sua varianza delta è circa zero. Questo gesto è utile per gli altri controlli, per far sì che il successivo evento parta da mano ferma.

- **Wave Detector**
  Identifica il movimento ondulatorio della mano. Il gesto Wave consiste in un un certo numero di cambiamenti di direzione della mano entro un certo timeout. Di solito sono necessari quattro cambiamenti di direzione per rilevare un Wave.

- **Circle Detector**
  Questo controllo rileva movimenti circolari della mano. Affinché gesto venga riconosciuto è necessario che la mano compia un giro completo sia in una direzione che nell'altro. La direzione positiva è considerata quella in senso orario, mentre quella in senso antiorario è considerata negativa.

- **SelectableSlider1D**
  Il controllo riconosce uno scorrimento della mano in una delle tre direzioni degli assi X,Y,Z, cioè sinistra-destra, sopra-sotto, vicino-lontano. Lo scorrimento è diviso in parti uguali in un certo numero di aree e ogni area definisce un singolo oggetto. Può essere utilizzato per creare menu in cui ogni oggetto corrisponde ad una opzione del menu.

- **SelectableSlider2D**
  Il controllo rileva uno scorrimento della mano in due direzioni sul piano X-Y.4444

Tutti questi oggetti estendono dall'oggetto Point Control (Figura 16) il quale riceve gli *hand point* attivi ad ogni frame, tenendo in questo modo traccia di tutti i punti (rappresentati in coordinate rapportate allo schermo) dello spostamento della mano.

Ogni sottoclasse poi analizza questi punti notificando qualora si verificasse quel determinato spostamento per cui la classe è specializzata.

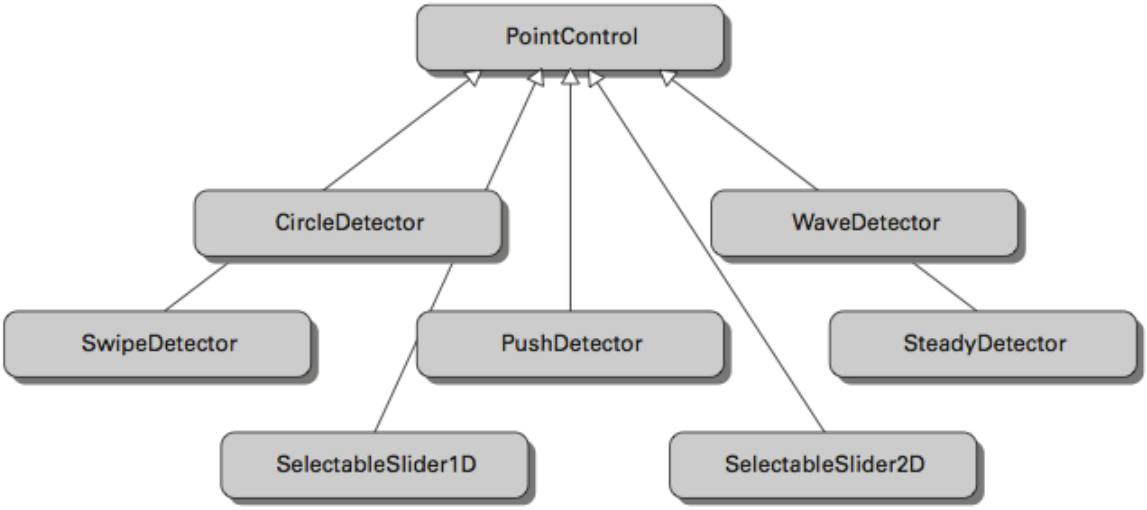

Figure 16.: Controlli di NITE

Anche l'oggetto Point Control può scatenare delle notifiche in particolare con la creazione di nuovi punti, con il movimento dei punti e la scomparsa dei punti.

Le notifiche vengono gestite con un meccanismo ad eventi.

3.3.3 Sviluppo

In base a quanto appena esposto è quindi possibile affermare senza problemi che si è utilizzato un oggetto *SwipeDetector* per poter sfogliare le foto, un oggetto *WaweDetector* per oltrepassare l'autenticazione e un oggetto Point Control per mappare il movimento della mano sullo schermo e poter interagire con i pulsanti.

Mentre i primi due meccanismi di *gesture* sono molto semplici e non necessitano di ulteriori approfondimenti, l'interazione naturale con i pulsanti richiede una maggior attenzione.

A livello progettuale questo terzo punto è incentrato sulla tassonomia di seguito riportata (Figura 24), dalla quale emergono gli oggetti Gesture GUI e GGUIsManager.

3.3.3.1 Gesture GUI

Sono stati previsti tre tipi di pulsanti (*Gesture GUI*): un pulsante vero e proprio (ButtonPanel), un cursore (SliderPanel), e una manopola (DialPanel).

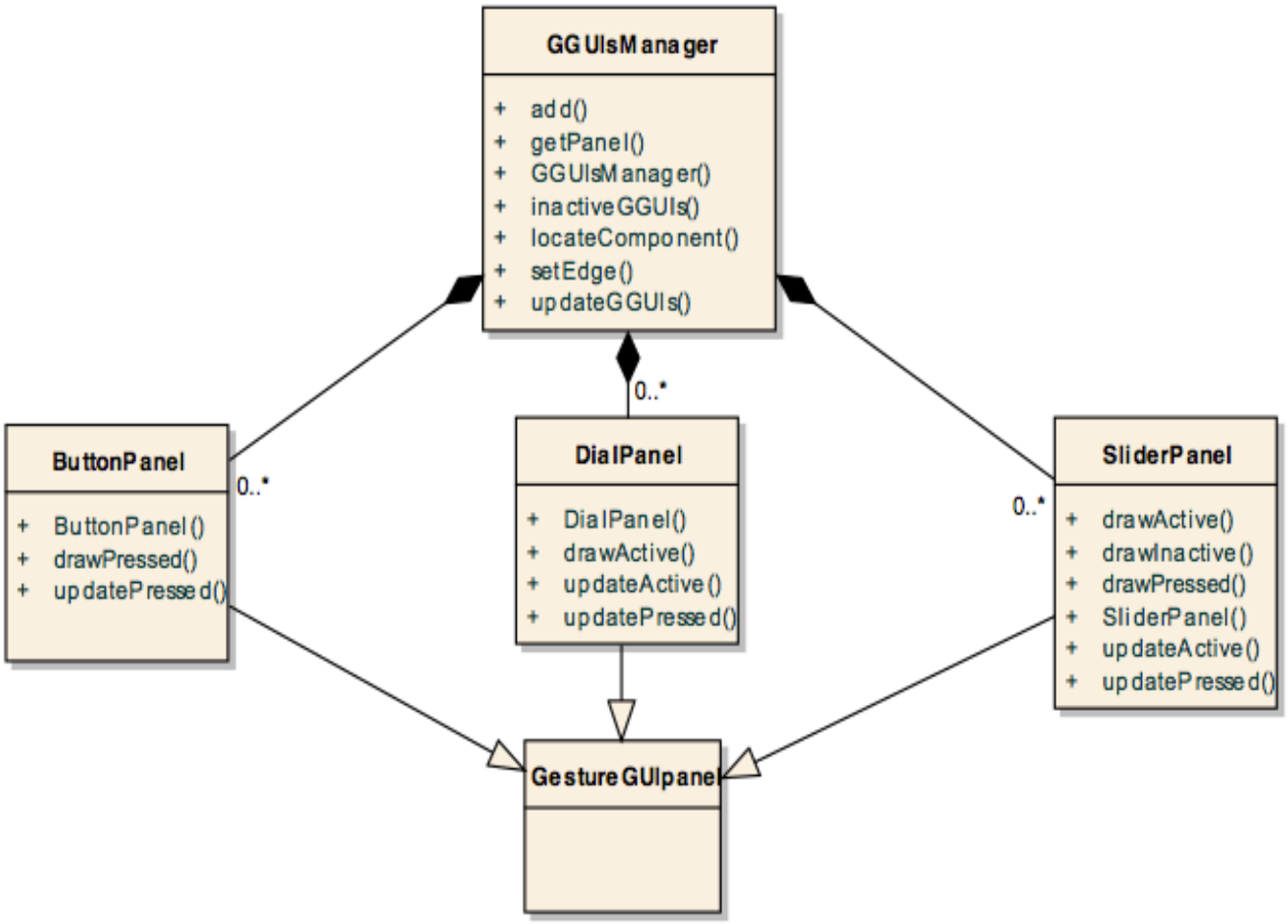

Figure24.: Tassonomia delle classi relativa all'interazione coi pulsanti

Come si può notare dalla figura 24 questi componenti ereditato dalla classe GestureGUIpanel. In particolare condividono lo stesso comportamento generale che consente ad un componente di muoversi fra tre stati: *inattivo, attivo, pressato.* Viene mostrato qui di seguito il diagramma degli stati per rappresentare queste transizioni (Figura 17).

Un componente diventa attivo quando la mano dell'utente entra nella sua zona sullo schermo; questa variazione è caratterizzata da un cambiamento dell'immagine del componente e può eventualmente reagire al movimento della mano all'interno della sua porzione di schermo; ad esempio il cursore si sposta da sinistra a destra e la manopola ruota. Ora se l'utente non muove la mano per un certo tempo prestabilito (nella nostra applicazione 1 secondo) si passa nello stato pressato. Anche questa transizione è caratterizzata da una variazione dell'immagine del componente.

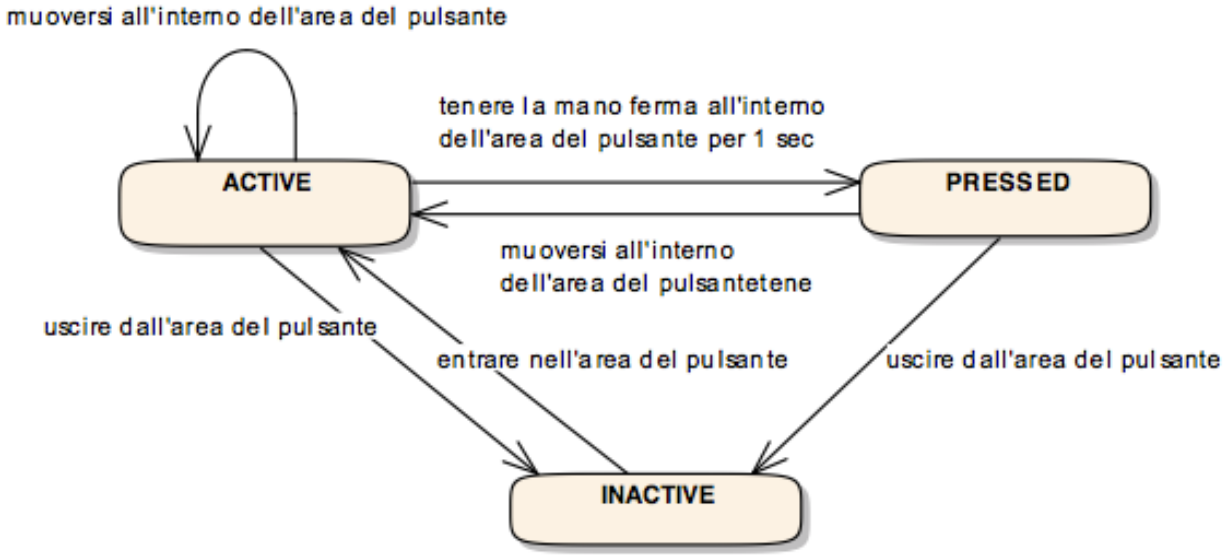

Figure 17.: Diagramma degli stati dei componenti Gesture GUI

3.3.3.2 Gesture GUI Manager

Questo oggetto è atto a contenere e gestire i componenti Gesture GUI. In particolare si occupa del loro posizionamento e della memorizzazione di quest'ultimo.

Memorizzando la posizione che occupano nello schermo e attraverso le notifiche generate dal controllo Point Control, ovvero tramite la notifica a ogni frame da parte di questo controllo della posizione della mano in coordinate rapportate allo schermo, il Gesture GUI Manager risale in quale Gesture GUI è posizionata la mano dell'utente.

3.3.3.3 Meccanismo di notifica

Si è creato inoltre un meccanismo per notificare la transizione dallo stato attivo allo stato premuto. Si dà

cosi la possibilità di impostare determinate reazioni alla pressione di determinati pulsanti. Questo meccanismo di notifica è basato sulla tassonomia riportata in figura 25. E' caratterizzato da oggetti che contengono tutte le informazioni del pulsante premuto, nel caso del cursore ad esempio il valore in cui è posizionata la tacca.

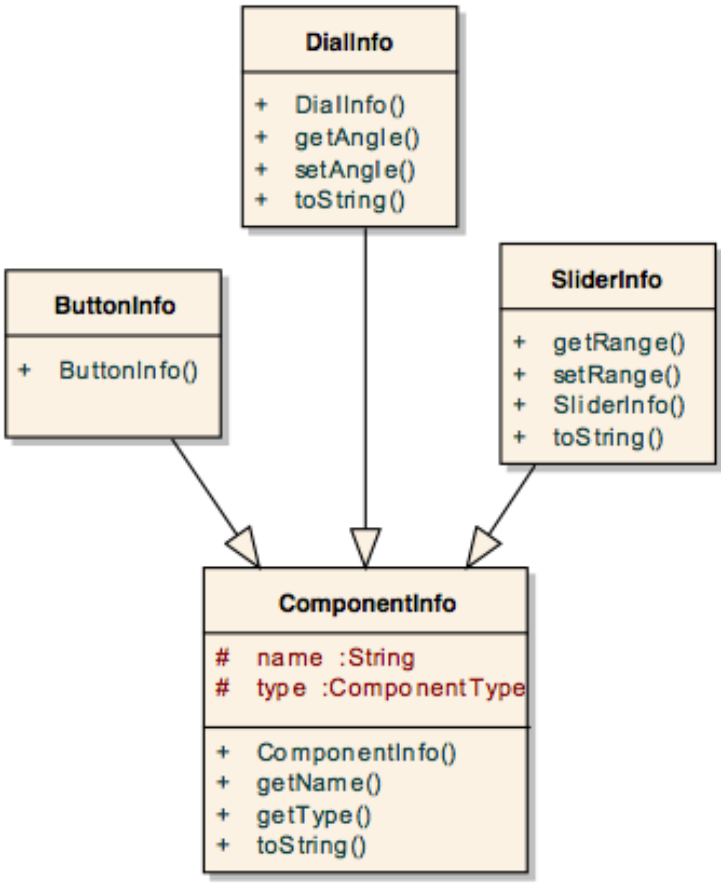

Figure25.: Tassonomia delle classi relativa al meccanismo di notifica dei pulsanti

4 SCHERMO PERVASIVO ADATTATIVO

4.1 Diagramma a stati

Dall'integrazione dei tre componenti presentati nel terzo capitolo otteniamo un sistema il cui comportamento può essere rappresentato dal seguente diagramma a stati (Figura 18).

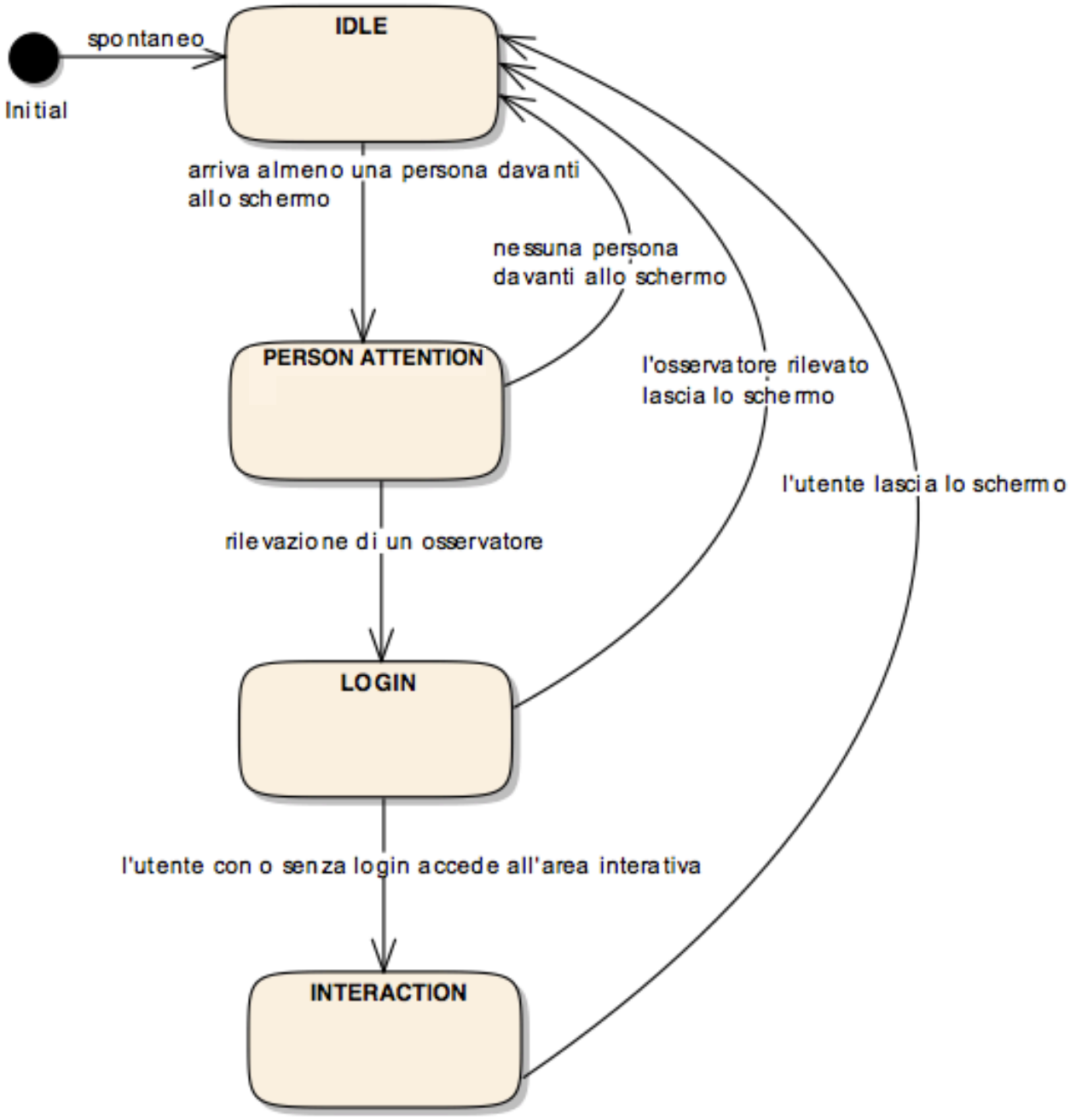

Figure 18.: Diagramma a stati del sistema

Questi stati sono ben visibili anche a livello implementativo, in quanto ognuno di loro è mappato in una sua specifica funzione.

L'utilizzo di questo *pattern* della macchina a stati ha facilitato molto lo sviluppo dell'applicazione dando un contributo sia a livello d'interfaccia grafica che funzionale.

4.2 Descrizione del diagramma a stati

Questo prototipo è stato realizzato calandolo nel contesto accademico dell'Alma Mater Studiorum-Università di Bologna in particolare per la seconda facoltà di Ingegneria di Cesena. Si premette che, in quanto prototipo atto a dimostrare le potenzialità di un sistema pervasivo di questo genere, è stato, a livello di contenuto informativo che può offrire, completato solo parzialmente.

Si suppone che questo schermo sia installato in una particolare sede della facoltà.

Appena viene avviata l'applicazione, si va in automatico nello stato di IDLE. In questo stato viene mostrato uno schermo completamente nero, dando così l'impressione di uno schermo spento.

Non appena perviene un utente davanti allo schermo si passa allo stato PERSON ATTENTION.

In questo stato tutti i dati dell'aggiornamento periodico provenienti dal sensore vengono utilizzati per il primo componente ovvero per misurare l'attenzione di ogni persona che si trova nella scena.

Una persona davanti allo schermo diventa un osservatore che vuole interagire con esso quando vengono soddisfatti i seguenti requisiti:

- Raggiunge un determinato indice di attenzione prestabilito e lo mantiene per un certo tempo prestabilito;
- Si trova ad una determinata distanza prestabilita dallo schermo;

Quest'ultimo punto non serve per un concetto di localizzazione come nell'esempio trattato nel paragrafo 3.2.2 ma, dato il metodo di autenticazione con scanning di QR Code che richiede di essere a una distanza ravvicinata dallo schermo è inutile ottenere il controllo dello schermo anche essendo molto lontani.

Ad ogni modo, garantire la possibilità di interagire con lo schermo anche ad elevate distanze porterebbe via

molto spazio nella zona in quanto nessuna persona potrebbe poi passare tra l'utente e lo schermo perché interferirebbe con l'interazione.

Appena riconosciuto l'osservatore si passa nello stato LOGIN (Figura 23), dove si ha la possibilità di autenticarsi tramite scanning del QR Code oppure oltrepassare questa fase tramite la *gesture "cancella".*

Figura 23.: Schermata di Login

In entrambe i casi transitiamo nello stato di INTERAZIONE.

Se viene effettuata l'autenticazione, il sistema salva tutti i dati dell'utente, grazie ai quali:

- Offre il benvenuto alla persona specificando nome e cognome;
- fornisce gli orari delle lezioni del giorno direttamente del suo anno accademico;
- caratterizza questa sessione con questo utente con la scritta "Utente loggato: Nome Cognome"(Figura 19 e 20).

In caso contrario si arriva nella prima schermata del contenuto informativo messo a disposizione in questa demo.

Questo stato di INTERAZIONE è suddiviso a livello implementativo in altri stati, ma si è preferito in questa descrizione raggrupparli in uno unico, in quanto a livello concettuale riguardano sempre aspetti di interazione. Questo per dare una visione complessiva dell'applicazione in modo non dispersivo.

All'inizio l'utente ha la possibilità di scegliere se visionare la localizzazione delle varie sedi della facoltà o accedere all'area degli orari accademici.

Optando per la prima opzione, "mappa", viene mostrata una schermata con una mappa geografica con tutte le sedi della facoltà, molto utile per avere una visione generale di come le varie sedi sono distribuite sul territorio. Inoltre vi è un pulsante per ogni sede (Figura 19).

Quest'ultimo ci permette di accedere a delle informazioni più specifiche sulla relativa sede:

- indirizzo;
- aule;
- mappa geografica con itinerario stradale sul come raggiungere la sede dalla posizione attuale;
- immagini relative alla planimetria della sede ed una raccolta di foto delle varie aule presenti in essa. Mentre le altre informazioni sono visibili tutte nella schermata principale della specifica sede in questione, queste ultime informazioni sono accessibili interagendo tramite pulsanti analoghi a quelli in Figura 19.

Figure 19.: Mappa generale

Optando invece per l'opzione "Orario delle lezione", lo schermo ci da la possibilità di scegliere l'anno di interesse, quindi possiamo visionare immediatamente l'orario delle lezioni del giorno.

Dato l'obiettivo di questa demo che esula dalla completezza delle informazioni reperibili si è trascurato il Corso di Laurea.

E' stato previsto anche un collegamento fra le varie aule e la loro sede (Figura 20), in modo tale da garantire al nuovo studente un abbinamento immediato fra l'aula e la relativa sede, e sul raggiungimento di quest'ultima.

A supporto dell'interazione con le immagini sono stati specializzati i pulsanti descritti nel paragrafo 3.3.3.1 al fine di realizzare i seguenti strumenti:

- uno strumento di "zoom" tramite il pulsante cursore, per aumentare e diminuire le dimensioni delle immagini (Figura 21);
- uno strumento che permette di ruotare le immagini tramite il pulsante manopola. Nel caso che l'immagine sia una mappa in questa manopola è presente anche l'indicazione del punto cardinale Nord (Figura 22).

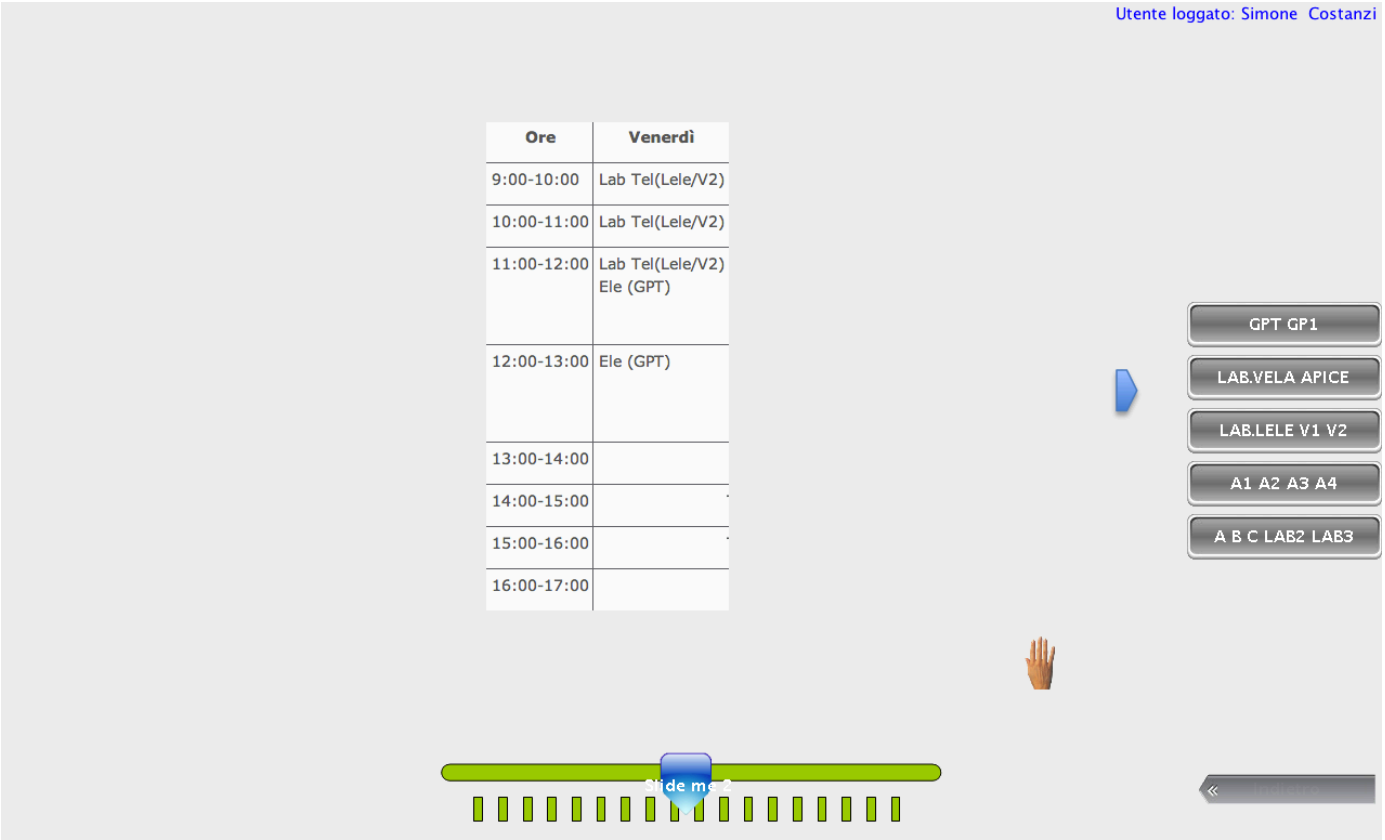

Figure 20.: Orario delle lezioni del giorno con relativo collegamento alle aule

Figure 21.: Cursore per zoom sull'immagine

Figure 22.: Manopola per ruotare l'immagine, col punto cardinale Nord

Sia nello stato di LOGIN che in qualsiasi punto dello stato di INTERAZIONE, se l'utente abbandona lo schermo si passa nello stato di IDLE perdendo traccia di tutte le informazioni inerenti all'utente. Si rimane in questo stato fino a quando non perviene un'altra persona davanti allo schermo.

4.3 Flusso di esecuzione

Il flusso di esecuzione della nostra applicazione è un ciclo caratterizzato da:

- Attesa dei dati aggiornati provenienti dal sensore Kinect, aggiornamento che avviene con una frequenza di circa 30 HZ.
- Analisi della specifica struttura dati per ottenere informazioni aggiornate sugli utenti presenti nella scena, in particolare la notifica di quando perviene un nuovo utente e quando uno lascia la scena.
- A seconda dello stato in cui viene a trovarsi l'applicazione si attiva o il componente addetto alla misurazione dell'attenzione o quello per l'interazione naturale, fornendogli i dati pervenuti dall'aggiornamento.

Per quanto riguarda il componente relativo all'autenticazione, esso è stato integrato con un flusso di esecuzione a se stante. Questa scelta è stata dettata da una serie di considerazioni. Ma principalmente dallo scenario di seguito descritto: appena viene riconosciuto un osservatore gli viene mostrata l'interfaccia per effettuare il login, più nel dettaglio questo significa che l'applicazione è bloccata in attesa su una server socket di connessioni TCP, quindi se quel osservatore lascia la scena senza effettuare lo scanning del QR Code, l'applicazione non effettuando il controllo continuo sul numero degli utenti non se ne accorge. Rimarremmo cosi bloccati sull'interfaccia di login. Questo scenario non si sarebbe presentato qualora si fossero utilizzate le primitive di OpenNI, in quanto forniscono le informazioni sui nuovi utenti e quelli che lasciano la scena con un meccanismo ad eventi.

Il mancato utilizzo di queste primitive a causa delle problematiche esposte in dettaglio nel paragrafo 3.1.4.2, inerenti alla loro eccessiva latenza di funzionamento, l'analisi di quella struttura dati per trarre queste informazioni in maniera più aggiornata è eseguita ripetutamente a prescindere dallo stato in cui ci si trova. Questo lo si può notare anche dal flusso principale di esecuzione esposto. Quindi anche nello stato di IDLE vi è un dispendio computazionale, anche se minimo.

4.4 Implementazione

E' possibile visionare il codice sviluppato accedendo al seguente repository: https://bitbucket.org/scostanzi/realizzazione-di-un-prototipo-di-schermo-pervasivo-adattativo.

5 CONCLUSIONI E SVILUPPI FUTURI

Con la presente tesi è stata analizzata una possibile soluzione per realizzare uno schermo pervasivo.

Il prototipo realizzato ha solamente scopo dimostrativo e risulta pertanto incompleto di contenuto informativo; colmando anche quest'ultimo aspetto, è comunque possibile immaginare l'importanza e l'utilità che susciterebbe in questo campo accademico.

Inoltre questo sistema potrebbe risultare utile in tantissimi altri ambiti, trai quali pubblicità, arte, sociologia e intrattenimento.

È importante evidenziare il forte carattere *sperimentale* della tesi; il prototipo realizzato deve essere visto come un punto di partenza per soluzioni più avanzate e affidabili, modellate rispetto a particolari esigenze e magari realizzate utilizzando in futuro sensori ancora più potenti e precisi.

Tramite questo lavoro è stato anche possibile mettere in evidenza le potenzialità (e i limiti) del sensore Kinect e delle librerie OpenNI e NITE.

Di seguito vengono elencati possibili miglioramenti e sviluppi futuri di ognuna delle tre parti costituenti il progetto.

5.1 Riconoscimento dell'attenzione

Nella prima parte è possibile stimare il livello di attenzione di una persona in base alle caratteristiche posturali, alla velocità di movimento e al tempo trascorso. Si possono notare però alcuni punti critici quando l'utente assume delle posizioni che OpenNI non riesce a tracciare (le mani non sono visibili, la persona è troppo vicina al sensore, alcune parti del corpo sono coperte, due persone sono troppo vicine, ci sono oggetti in scena che possono essere riconosciuti erroneamente come persone, ecc.).

Per ovviare a questi problemi si potrebbe ad esempio pensare di utilizzare due o più sensori connessi tra loro e posizionati in modo da coprire tutta la scena.

Bisognerebbe però riuscire a coordinare i vari dispositivi, ad esempio combinando i dati tra di loro o scegliendo man mano i dati più affidabili seguendo una gerarchia tra i dispositivi.

In questo modo si eviterebbero le cosiddette zone d'ombra e si avrebbe una percentuale di area visibile molto più ampia. Ad esempio in una stanza si potrebbe pensare di posizionare due sensori in due angoli opposti del soffitto, o addirittura posizionarne quattro, uno in ogni angolo, in modo da coprire quasi tutta la stanza.

Ad ogni modo, essendo la prima versione di questi sensori e visto il grande successo riscosso, si può ragionevolmente aspettare presto l'uscita di sensori molto più potenti sia a livello hardware che software, che ridurrebbero gli errori nello Skeleton Tracking e permetterebbero applicazioni sempre più prestanti e affidabili.

Sempre in riferimento a questa prima parte, possibili sviluppi futuri potrebbero consistere nell'integrazione di tecniche di elaborazione delle immagini per estrarre ulteriori feature, quali la posizione degli occhi e il riconoscimento delle espressioni facciali.

5.2 Identificazione dell'osservatore

Soffermandosi sulla seconda parte, autenticazione dell'osservatore, è possibile affermare che l'interazione è molto veloce, in quanto, una volta configurati i propri dati personali, per eseguire l'identificazione, l'utente non deve fare altro che avviare l'applicazione e premere il pulsante che permette di effettuare lo scanning del QR Code. Anche quest'ultima analisi, una volta centrato il QR Code con la fotocamera è molto rapida.

Tuttavia questo metodo che fa uso di QR Code risulta all'utente meno trasparente rispetto a quello precedentemente esposto (paragrafo 3.2.2) con localizzazione tramite segnale bluetooth.

L'utente, con lo scenario del bluetooth, per iniziare l'interazione contestualizzata con lo schermo non avrebbe dovuto fare nient'altro che avere la determinata applicazione attiva, diversamente dallo scenario del QR Code.

Quindi, un possibile miglioramento di questa parte potrebbe essere l'utilizzo della periferica bluetooth per autenticarsi. Occorre pertanto ricercare un modo che consenta di ottenere delle informazioni sulla potenza del segnale bluetooth di maggior precisione, in modo da avere una stima della distanza fra periferiche più accurata, ad esempio sfruttando dati provenienti da più sensori bluetooth (tecniche di triangolazione di sensori).

Un'altra possibile soluzione che garantisce sempre un alto livello di trasparenza potrebbe essere costruire un sistema basato su etichette RFID (*Radio Frequency Identification*).

Quest'ultima è una tecnologia utilizzata per l'identificazione e/o memorizzazione dati automatica di oggetti, animali o persone (AIDC Automatic Identifying and Data Capture) basata sulla capacità di memorizzazione di dati da parte di particolari dispositivi elettronici (detti *tag* o *transponder*) e sulla capacità di questi di rispondere all'"interrogazione" a distanza da parte di appositi apparati fissi o portatili chiamati per semplicità "lettori" (in realtà sono anche "scrittori") a radiofrequenza comunicando (o aggiornando) le informazioni in essi contenute. Data questa caratteristica e considerando il fatto che determinati modelli di questi *tag* hanno un segnale che ha una portata massima di circa due metri (inferiore a quella del bluetooth che in media si aggira intorno ai 10 metri), è possibile pensare di sostituire l'identificazione tramite QR Code con questa tecnologia.

Questa soluzione si basa sempre sul concetto di localizzazione, sfruttato questa volta in un altro modo rispetto allo scenario bluetooth.

Per l'applicazione considerata, è necessaria una misura della distanza dotata di una precisione di circa due metri; data la portata del segnale bluetooth molto maggiore, nel caso in cui si voglia utilizzare quest'ultimo come metodo di identificazione, occorre cercare di quantificare la potenza del segnale almeno in modo da avere una stima della distanza con la precisione di due metri.

Invece la portata del segnale della tecnologia RFID è in media attorno ai due metri, la distanza ricercata, ed è possibile pertanto ragionare in termini di "chi è presente nel suo raggio di azione", senza dover risalire alla distanza dalla potenza.

Inoltre, non sono state toccate questioni riguardanti la sicurezza e la privacy delle informazioni scambiate, aspetti che assieme alla trasparenza sono di fondamentale importanza in contesti di pervasive computing.

5.3 Interazione naturale

La parte realizzata relativa alle gesture sfrutta molto le potenzialità offerte dal sensore Kinect e dalle librerie, avvicendandosi notevolmente ai principi dell'*interfacciamento naturale*.

Tuttavia, si potrebbe portare un maggior contributo a quest'ultimo sfruttando anche il canale audio oppure interfacce grafiche tridimensionali.

Con questo sistema si vuole fare un passo in avanti verso l'utilizzo d'interfacce orientate ai naturali modi di comunicare dell'uomo a discapito di dispositivi tradizionali ormai entrati nell'uso quotidiano che però a volte rappresentano una barriera.

Si pensi ad esempio al semplice utilizzo di un televisore; anziché utilizzare un normale telecomando si potrebbe pensare di interagire con il televisore tramite semplici gesti.

Rapportando il meccanismo delle gesture di questo prototipo si potrebbe ad esempio cambiare canali come se sfogliassimo delle foto, controllare il volume con la mano utilizzando il pulsante cursore, cambiare la luminosità con gesti circolari attraverso la manopola, accendere il televisore con un gesto *Push* e spegnerlo con un gesto di *Wave*.

BIBLIOGRAFIA

[1] Weiser M., "The Computer for the 21st Century", Scientific Am., Settembre 1991; riedito dall'IEEE Pervasive Computing, GennaioMarzo 2002.

[2] Mascolo C., Capra L., Emmerich W., "Mobile Computing Middleware", Dept. of Computer Science, University College London, 2002.

[3] B. Gollan, B. Walley, and A. Ferscha, Automatic attention estimantion in an interactive system based on behaviour analysis, in Proceedings of the 15th Portuguese Conference on Artificial Intelligence (EPIA2011), 2011.

[4] C.D. Wickens and J.S. McCarley, Applied attention theory, CRC Press, Boca Raton, 2008.

[5] C. Kublbeck, A. Ernst, Face detection and tracking tracking in video squences using the modified census transformation Image and Vision Computing, 2006, Face Processing in Video Sequences, http://www.iis.fraunhofer.de/EN/bf/bv/kognitiv/biom/dd.jsp

[6] M. Woollacott, A.Shumway-Cook, Attention and the control of posture and gait: a review of an emerging area of research, Elsevier Science, 2001.

[7] S. Asteriadis, K. Karpouzis, S. Kollias, A Neuro-fuzzy approach to user attention recognition, ICANN, 2008.

[8] E.D. Reichle, A. Pollatsek, K. Rayner, E-Z Reader: A cognitive-control, serial-attention model of eye-mevement behavior during reading, Elsevier, 2005.

[9] Documentazione OpenNI.

[10] Documentazione NITE.

[11] http://www.wikipedia.org/

[12] Greg Borenstein, My Making Things See.

[13] Jeff Kramer, Nicolas Burrus, Florian Echtler, Daniel Herrera C., and Matt Parker, Hacking the Kinect.

Ringraziamenti

Un sincero ringraziamento va a tutti coloro che mi hanno aiutato in vario modo a raggiungere questo importante traguardo.

Desidero ringraziare in primo luogo il prof. Mirko Viroli per la disponibilità e la cortesia con cui mi ha aiutato durante l'attività sperimentale e la stesura di questa tesi.

Un sentito ringraziamento alla mia famiglia che con il suo sostegno morale ed economico mi ha permesso di raggiungere questo importante obiettivo e che mi ha sopportato nei momenti di difficoltà.

Grazie a tutti gli amici con cui ho condiviso questi piacevoli anni di studio, ed in particolare Simone, compagno di innumerevoli progetti.

Sinceri ringraziamenti a tutti i miei amici, in particolar modo a Francesco, per il sostegno dato e la fiducia mai mancata.