Università degli Studi di L'Aquila

FACOLTA' DI SCIENZE MATEMATICHE, FISICHE E NATURALI

CORSO DI LAUREA IN INFORMATICA

Tesi di Laurea

Lo Spazio Pubblico Sensibile:
Un sistema di Computer Vision per installazioni artistiche

Relatore
Prof.ssa Paola Inverardi

Candidato
Luca Martellucci
Matricola 135045

Correlatore
Dott. Alessandro Marianantoni

Anno Accademico 2005-2006

Ringraziamenti

Inizio col ringraziare i miei genitori, Esilde e Giuseppe che in questi lunghi anni sono stati sempre al mio fianco non facendomi mai mancare il loro appoggio ed affetto. Un ringraziamento particolare anche ad Alessandro, con il quale ho condiviso parte della mia esperienza universitaria e che per un breve periodo mi ha accompagnato nell'avventura americana. Che dire poi di Erica, a lei rivolgo un ringraziamento speciale, per avermi sopportato, incoraggiato e sostenuto per tutto il tempo (senza di lei non ce l'avrei proprio fatta!).

Ringrazio inoltre Alessandro Marianantoni, che ha reso possibile, con l'aiuto della Provincia di Rieti e della Fondazione Varrone, la bellissima esperienza di Los Angeles e la realizzazione di Quartieri della Memoria.

Ringrazio Alessandro Bissacco per avermi introdotto e guidato nel mondo della Computer Vision, tutti i ragazzi del Remap Studio: Pablo, Vids, Javier, Paula, Ryan, Eitan e Vanessa che con la loro simpatia hanno reso più piacevoli le lunghe ore trascorse davanti ai monitor del buio e freddo Studio TV2 di Melnitz Hall.

Tutti gli amici incontrati in California: Chiara e la sua pasta, Valeria, Michelle ed i suoi splendidi cani, Clara, Alexandro, Jun, Emily, Caterina e John ma soprattutto Tommy e Sheila, con i quali ho trascorso sei mesi indimenticabili tra lavoro, feste e gite fuori porta in quel di Los Angeles.

Ultimi ma non meno importanti, ringrazio gli amici della Taverna di Vorly: CSC, Giasai, Ghemon, Pilone, Antò, il CapoSalame, Gnappo e MachoMan, per aver allietato le fredde serate aquilane con dell'ottimo Montepulciano d'Abruzzo e partite memorabili alla playstation (rigorosamente emulata sul mio PC scassato).

Indice

Premessa

Il presente documento intende descrivere un sistema di tracking per installazioni artistiche interattive, basato su computer vision, sviluppato nell'ambito della collaborazione fra la Provincia di Rieti e l'University of California Los Angeles (UCLA).

Tale collaborazione composta da 3 borse di studio della durata di sei mesi per un progetto di ricerca e sviluppo al centro Research in Engineering, Media and Performance (REMAP) a UCLA, ha avuto come obiettivo quello di sviluppare un primo prototipo dell'installazione interattiva "Quartieri della Memoria" che ha nei cittadini, negli spazi pubblici e nella loro interazione gli elementi fondanti. Porta D'Arce, il quartiere dei Pozzi e l'area del Ponte Romano, aree storiche della città di Rieti, sono i luoghi che formeranno il percorso, per la realizzazione dell'opera multimediale che coniugherà cultura e tecnologia e che quindi sarà il risultato di un lavoro multidisciplinare. Il presupposto del progetto è la rivalutazione dello spazio pubblico di Rieti, attraverso le memorie della cultura popolare fornite dagli stessi reatini che, non solo riscoprono e rivivono parte della loro città, ma diventano attori fornendo i contenuti stessi dell'installazione.

Previo concorso pubblico, svoltosi nel luglio 2005, le borse di studio sono state assegnate agli studenti Sheila Starace, Luca Martellucci e Tommy Gentile, rispettivamente studentessa della facoltà di architettura presso l'università degli studi di Roma La Sapienza e studenti di informatica presso l'Università degli studi dell'Aquila.

A Proposito della University of California, Los Angeles

L'University of California, Los Angeles (UCLA), con circa 100.000 studenti è una delle università pubbliche più importanti e all'avanguardia degli Stati Uniti. In una classifica stilata dalla Conferenza del Consiglio delle Ricerche, UCLA si colloca al quattordicesimo posto nella graduatoria nazionale delle università pubbliche e private. L'offerta formativa è suddivisa in un college e undici scuole professionali tra le quali ricordiamo la Facoltà di Lettere e Scienze, la Scuola di Arte ed Architettura, la Scuola di Ingegneria e Scienze Applicate Henry Samueli, e la Scuola di Teatro, Film e Televisione. Quest'ultima in particolare è una delle scuole più prestigiose del paese, essendo l'unica a combinare le tre discipline di teatro, cinema e televisione in un unico corso che, secondo la recensione di Princeton, è il migliore a livello nazionale.

UCLA è un'università all'avanguardia nel campo della ricerca, con circa 40.000 tra ricercatori, docenti e dottorandi che lavorano nelle sue strutture, ed investimenti annuali pari a 700 milioni di dollari.

Figura 1 – Veduta dell'auditorium, Royce Hall presso UCLA

Center of Research in Engineering, Media and Performance (REMAP)

Il centro per le ricerche in ingegneria, media e performance è uno sforzo collettivo della Scuola di Teatro, Film e Televisione e la Scuola di Ingegneria e Scienze Applicate Henry Samueli di UCLA.

Remap fonde il mondo dei docenti e studenti di HSSEAS e TFT per esplorare nuove forme culturali d'arricchimento e le situazioni sociali permesse dall'intreccio dell'ingegneria, delle arti e della comunità di sviluppatori. Remap è un ambiente creativo che abbraccia e promuove persone, progetti e ricerche che possono avere risonanza ed impatto a lungo termine nella relazione tra cultura e tecnologia. Remap è costruito sull'esperienza dell'HyperMedia studio di UCLA fondato nel 1997 per esplorare le espressioni creative uniche permesse dalla collaborazione tra mezzi di comunicazione, arti dello spettacolo ed ingegneria.

Hypermedia Studio

Fondato nel 1997 dal professore Fabian Wagmister con l'appoggio di Intel e Microsoft, l'Hypermedia Studio è una unità di ricerca unica all'interno della Scuola di Teatro, Film e Televisione dedicata alla collaborazione tra mezzi di comunicazione, arti dello spettacolo e tecnologia all'avanguardia, alla ricerca di nuovi generi di espressione creativa.

Le ricerche dello studio includono produzioni di lavori originali di professori e studenti, così come collaborazioni con altri dipartimenti del campus. Questi ambienti di espressioni artistiche sono un amalgama di teatro, mezzi di comunicazione interattivi e spazi sociali esistenti. Essi utilizzano tecnologie avanzate, inclusi sensori, database intelligenti e reti distribuite per modellare spazi dalla forma dinamica e riferiti al rapporto tra arte e tecnologia.

Figura 2 – Presentazione finale di “Quartieri della Memoria” presso Remap Studio

Capitolo 1

1.1 - Installazioni interattive per luoghi pubblici

Negli ultimi anni, le nuove tecnologie hanno avuto un'enorme influenza sulla produzione artistica, indirizzando tutte le avanguardie e proponendosi come potente mezzo espressivo.

Tutte le forme artistiche tradizionali vivono, secondo Costa, un'ibridazione reciproca ed anche una contaminazione da parte dei nuovi mezzi di cui dispongono. Tali contaminazioni sarebbero però solo le ultime strategie di sopravvivenza che la dimensione artistica tradizionale tenta di costruire, dato che le nuove tecnologie non riaprono assolutamente alcun dibattito nelle arti tradizionali, ma forse lo chiudono definitivamente.

Le nuove tecnologie permettono invece di accedere al concetto di "sublime tecnologico", ovvero la nozione del superamento dell'arte, un essere collocati al di là di quelle che erano le categorie specifiche dell'artistico, vale a dire: il soggetto, l'espressione, la creatività, lo stile.

Il sublime tecnologico è cioè una situazione di debolezza del soggetto, di sorpasso dell'espressione, di venir meno dello stile, del venir meno di tutte quelle che erano le caratteristiche fondamentali dell'arte tradizionalmente intesa.

In questo contesto, secondo Fred Forest, ci troviamo dinanzi alla fine del narrativo nell'arte, passando bruscamente dal dominio della rappresentazione a quello della presentazione. Ora non sono più l'immagine, l'oggetto, il gesto a dover essere fissati, ma il processo stesso di trasformazione nel quale questi elementi sono impegnati, solidali e interdipendenti. Nelle arti interattive abbiamo quindi una centralità dell'evento, un azione che viene generata dall'uomo, percepita attraverso l'impiego di sensori dalle macchine ed utilizzata da quest'ultime per la produzione di una "esperienza".

L'osservatore è così parte integrante di un processo, di un sistema, diventando uno degli agenti delle interazioni prodotte e trasformando in questo modo la sua posizione da osservatore neutro ad agente attivo nello svolgimento in corso. Nelle arti dell'interattività il destinatario potenziale non è più solo semplice spettatore dell'oggetto proposto, egli ne diviene co-autore ed inoltre in questo scenario, l'artista è ancora portatore di rappresentazioni, ma queste non sono più predeterminate. Le nuove tecnologie infatti, che

hanno basso grado di usabilità, pongono l'artista davanti all'accettazione della casualità e alla rinuncia al controllo nei suoi lavori.

Le installazioni artistiche interattive sono la diretta conseguenza di tali riflessioni, e ne seguono pienamente le linee guida: interazione uomo-macchina, centralità dell'evento, spettatore agente attivo e co-autore, aleatorietà del sistema. Molti artisti, sono stati impegnati negli ultimi anni nella produzione di installazioni interattive per luoghi pubblici ed i più famosi a livello internazionale sono senza dubbio Lozano Hemmer, Marie Sester, Myron Krueger ed altri. Nel paragrafo successivo diamo una breve descrizione dei lavori più famosi ed apprezzati.

1.1.1 - Alcuni Esempi

Il primo lavoro artistico interattivo ad incorporare la computer vision in maniera abbastanza interessante, fu anche uno dei primi lavori artistici interattivi. Il leggendario *Videoplace* di Myron Krueger sviluppato tra il 1969 e il 1975 fu motivato dalla credenza profondamente sentita che l'intero corpo umano dovrebbe avere un ruolo nella nostra interazione con i computer. Nell'installazione Videoplace, il partecipante sta in piedi di fronte ad un muro retro illuminato e guarda verso una video proiezione su uno schermo. La silhouette del partecipante è poi digitalizzata e la sua posizione, forma e movimento dei gesti vengono analizzati. In risposta, Videoplace sintetizza grafica come ad esempio, piccole creature che si arrampicano sulla silhouette proiettata del partecipante, o cappi colorati disegnati fra le dita, offrendo più di 50 differenti tipi di interazioni e composizioni.

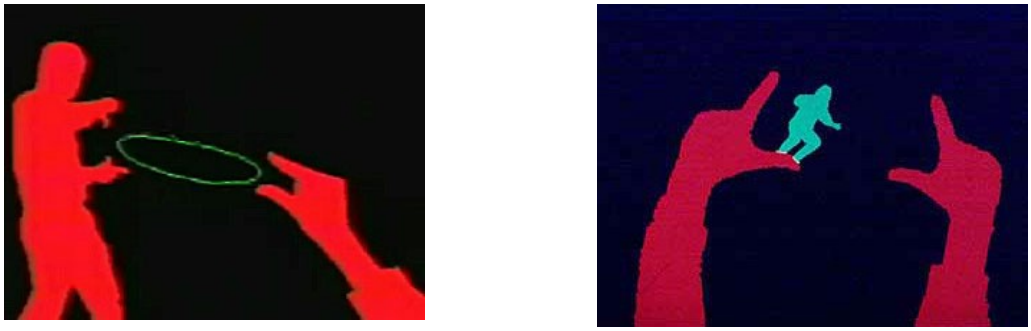

Figura 3, 4 – Videoplace

Videoplace fu rilevante per molti "firsts" nella storia dell'interazione tra uomo e computer. Ad esempio, alcuni dei suoi moduli di interazione permettevano a due utenti posti in locazioni remote, di partecipare allo stesso spazio video condiviso e connesso attraverso la rete. Si realizzò, in effetti, una implementazione della prima realtà virtuale multi persona o, come la chiamò Krueger, una realtà artificiale. Videoplace fu sviluppato, e dovrebbe essere notato, prima che il mouse di Douglas Englebart diventasse la onnipresente periferica desktop quale è oggi, e fu in parte creato per dimostrare interfacce alternative alla tastiera che dominò i calcolatori nei primi anni settanta. Abbastanza insolitamente, il sistema Videoplace originale è ancora in attività.

Messa di Voce, creato da Golan Levin in collaborazione con Zachary Lieberman utilizza interazioni basate sull'intera visione del corpo similmente al lavoro di Krueger, ma combina loro con l'analisi delle parole e le colloca entro una sorta di realtà arricchita basata sulle proiezioni. In questa performance audiovisiva, la parola urlata o cantata, prodotta da due vocalisti astratti è visualizzata e arricchita in tempo reale da grafica sintetizzata. Per realizzare ciò, un computer utilizza un insieme di algoritmi per tracciare la posizione delle teste dei partecipanti e ne analizza anche la voce che giunge da alcuni microfoni. In risposta il sistema mostra diversi tipi di visualizzazioni su uno schermo posizionato appena alle spalle dei partecipanti; queste visualizzazioni sono sintetizzate in modo da essere strettamente legate ai suoni parlati e cantati. Con l'aiuto del sistema di tracking delle teste, inoltre, queste visualizzazioni sono proiettate in modo tale da apparire emergenti direttamente dalle bocche dei partecipanti.

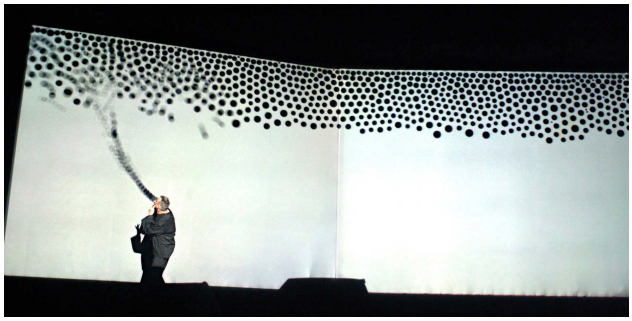

Figura 5 – Messa di Voce

Il tema della sorveglianza gioca un ruolo di primo piano nel *Sorting Daemon* (2003) di David Rokeby. Motivato dalle preoccupazioni dell'artista circa l'aumento dell'uso dei sistemi automatizzati per il tracciamento di "profili" dei cittadini, come componente della "guerra al terrorismo", questa installazione ha come obiettivo la costruzione automatica di un ritratto diagnostico del suo ambiente sociale e razziale. Rokeby scrive: "Il sistema guarda fuori nella strada, ruotando, inclinandosi e zoomando, cercando oggetti in movimento che potrebbero essere persone. Quando trova qualcosa che potrebbe essere una persona, rimuove l'immagine della persona dallo sfondo. La persona estratta è ripartita in accordo ad aree di colore simili. Il campione di colori risultante è poi organizzato, utilizzando tinta, saturazione e dimensione, entro il contesto arbitrario dell'immagine composta, proiettata nel luogo dove è accolta l'installazione."

Figura 6 – Sorting Daemon

Un altro progetto incentrato intorno alle tematiche della sorveglianza è *Suicide Box* di Natalie Jeremijenko e Kate Rich. Presentato come periferica per misurare l'ipotetico "indice di scoraggiamento" di una data località, Suicide box, è stato un sistema di video rilevamento del movimento posizionato nel raggio di vista del ponte Golden Gate a San Francisco. Suicide box monitorava costantemente il ponte e quando riconosceva del movimento verticale, lo catturava in una registrazione video. La risultante misurazione, mostrò, in un flusso video continuo, un gocciolamento di persone che saltavano giù dal ponte. Va

ricordato che il ponte Golden Gate è la prima destinazione di suicidi negli Stati Uniti, così durante i primi 100 giorni di utilizzo la suicide box registrò 17 casi, mentre durante lo stesso periodo di tempo l'autorità portuale ne contò solamente 13.

Altrove Jeremijenko spiegò che "l'idea era quella di tracciare un fenomeno sociale tragico che non era sotto l'attenzione dei media", ma la Suicide Box incontrò controversie considerevoli, variando da questioni etiche per quanto riguarda la registrazione dei suicidi, alla incredulità nella realtà delle registrazioni.

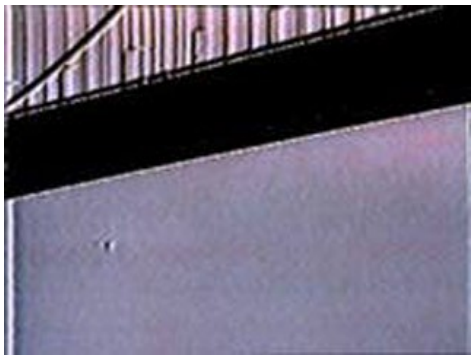

Figura 7 – Suicide Box

Access, di Marie Sester (2005), è una installazione artistica pubblica che utilizza il web e tecnologie di sorveglianza permettendo agli utenti del web di tracciare individui in spazi pubblici con un unico riflettore robotizzato ed un sistema acustico, senza che le persone indossino alcuna attrezzatura, esplorando le ambiguità tra la sorveglianza, controllo, visibilità e celebrità.

Il riflettore robotizzato segue automaticamente gli individui tracciati mentre il fascio acustico proietta dell'audio che solo essi possono udire. Gli individui tracciati non conoscono l'identità di chi li sta seguendo, o perché essi siano tracciati ne sono consapevoli di essere le uniche persone nel pubblico ad ascoltare il suono. Gli utenti del web non sono a conoscenza che le loro azioni provocano dei suoni verso l'obiettivo. In effetti entrambi, il soggetto tracciato e quello che traccia, sono in un ciclo comunicativo paradossale.

Access indirizza ed esplora l'impatto della scoperta e del controllo all'interno della società contemporanea. Access presenta strumenti di controllo che combinano tecnologie di sorveglianza con gli annunci pubblicitari e le industrie di Hollywood, creando

intenzionalmente situazioni ambigue, scoprendo l'ossessione-fascino per il controllo, vigilanza, visibilità e celebrità.

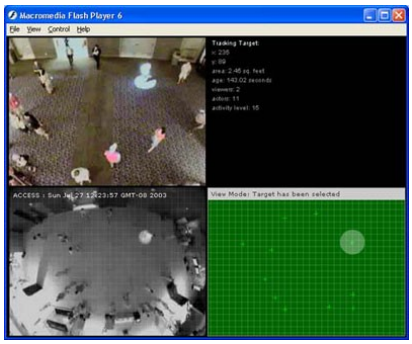

Figure 8, 9 - Access

Altro lavoro interessante è *Body Movies*, di Lozano Hemmer. Questa installazione trasforma lo spazio pubblico con proiezioni interattive di dimensioni variabili tra i 400 e gli 1800 metri quadrati. Centinaia di ritratti fotografici, presi nelle strade delle città dove la proiezione è esibita, sono mostrate utilizzando proiettori robotizzati. Tali ritratti appaiono solamente all'interno delle ombre proiettate dai passanti locali, le cui silhouette misurano in altezza dai 2 ai 25 metri, dipendentemente dalla distanza delle persone dalla potente sorgente di luce posizionata sul pavimento della piazza. Un sistema di computer vision aziona nuovi ritratti ogni volta che i vecchi sono mostrati.

Figura 10 – Body Movies di Lozano Hemmer

Sempre Lozano Hemmer è l'autore di *Vectorial Elevation*, un progetto artistico interattivo realizzato originariamente per celebrare l'arrivo dell'anno 2000 nella piazza Zócalo di Messico City. Tramite l'utilizzo di un sito web, si concedeva la possibilità a chiunque lo volesse, di progettare immense sculture di luce sopra il centro storico della città, utilizzando una interfaccia online tridimensionale. I progetti erano realizzati da 18 proiettori robotizzati posti intorno alla piazza i cui fasci di luce erano visibili fino a 15 km di distanza. Una pagina web personalizzata fu creata per ogni partecipante, con commenti, statistiche ed immagini reali e virtuali dei loro progetti. Il progetto fu poi mostrato durate vari eventi, come l'apertura del museo basco di arte contemporanea e Vitoria ed anche in occasione dell'espansione dell'Unione Europa a Dublino.

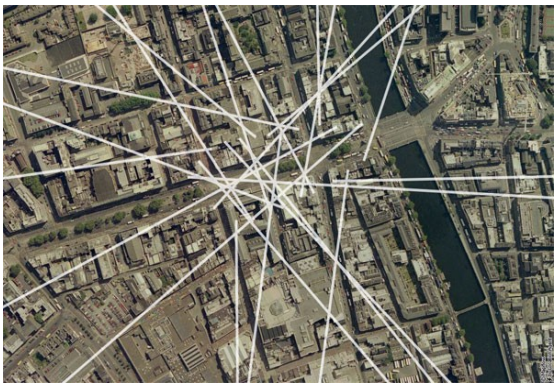

Figura 11 – Vectorial Elevation a Dublino

Come può essere notato dai precedenti esempi, che rappresentano solo una piccola selezione, i lavori artistici abbracciano tematiche molto differenti tra di loro, che variano dall'altamente formale ed astratto all'umorismo, a temi socio-politici utilizzando le attività di volenterosi partecipanti, di volontari pagati o di ignari sconosciuti.

1.2 - Quartieri della Memoria

1.2.1 - Parole chiave:

- rinnovo degli spazi pubblici storici
- sistemi distribuiti
- architettura di luce
- elaborazione degli eventi in tempo reale
- “quartieri dello spettacolo”
- interazione fra le persone e l’architettura
- tecnologia come interfaccia culturale
- mura medioevali come media

1.2.2 - Descrizione

Quartieri della Memoria è una installazione artistica interattiva basata su spazi pubblici, sui cittadini e sulla loro interazione. Le motivazioni che ci hanno spinto alla creazione di questa installazione, sono molteplici, così come gli elementi caratterizzanti l'installazione stessa.

Va detto che Quartieri della Memoria, nato dalla collaborazione con la Provincia di Rieti, è stato fin dalla sua nascita legato al territorio reatino, l'intero lavoro infatti trova fondamento nella città di Rieti, nella sua storia, nelle sue tradizioni e nei suoi cittadini.
Analizzando la situazione socio-economica e culturale della città, sono emersi innumerevoli elementi di interesse, che abbiamo cercato di analizzare e rielaborare attraverso la nostra installazione.

In particolare la scelta delle tre locazioni in cui posizionare Quartieri della Memoria, è stata effettuata considerando il fatto che tali aree, Porta d'Arce, i Pozzi e l'area circostante il Ponte Romano, sono alcune delle zone più antiche e ricche di cultura popolare dell'intera città. Ognuna di esse era caratterizzata in passato dalla presenza di un lavoro tradizionale, come ad esempio i “carrettieri” di Porta d'Arce, o gli “ortolani” dei Pozzi, lavori che oggi si stanno perdendo e che in un certo senso Quartieri della Memoria vorrebbe rivalutare, riportandoli all'attenzione dell'intera città. E' per questo motivo infatti che uno degli elementi

principali della nostra installazione è l'archivio delle memorie, una raccolta di foto e di testimonianze relative ai momenti di vita e di lavoro, degli ultimi 50 anni nelle tre aree in questione. Tale archivio è costruito con l'indispensabile apporto dei cittadini, che diventano così coautori dell'opera, e non più semplici fruitori. Durante le festività del Natale 2005, abbiamo raccolto le prime foto ed effettuato le prime interviste ad alcuni residenti dei tre quartieri. I dati ottenuti, tra cui soprattutto i contributi audio, sono stati molto interessanti in quanto ci hanno permesso di focalizzare la nostra attenzione sulla "Pianara", termine dialettale che sta ad indicare l'esondazione del fiume della città: il Velino. Già dall'analisi storica avevamo rilevato una notevole influenza di tali alluvioni sulla vita socio economica reatina, ma dalle interviste siamo rimasti impressionati dalla forza e dal radicamento di questi eventi nei ricordi dei cittadini più anziani. Questo ci ha spinto ad integrare la "Pianara" nel sistema, creando un apposito evento che nei nostri intenti farà rivivere in qualche modo le alluvioni del fiume Velino avvenute regolarmente fino a poche decine di anni fa nella città di Rieti.

Dal punto di vista architettonico inoltre, ogni locazione è caratterizzata dalla presenza di importanti monumenti che ne segnano l'identità, come ad esempio le mura medievali di Porta d'Arce, o le volte dei Pozzi. Questi monumenti con il passare del tempo sono stati un po' dimenticati ed in alcuni casi snaturati da interventi di utilità pubblica, Quartieri della memoria, vuole invece riqualificare tali monumenti, integrandoli al suo interno ed utilizzandoli come nuovo mezzo espressivo.

La scelta delle mura medioevali di Piazza del Suffragio come "new media", come nuovo strumento di comunicazione, è stato un importante intervento di riqualificazione e di rinnovo di spazi pubblici. Una interessante discussione sul rapporto che intercorre tra mura e città è stata condotta in questi mesi dalla collega Sheila Starace, mettendo in evidenza come le mura e tutte le fortificazioni in generale abbiano perso la loro funzione difensiva. Le mura infatti venivano percepite non più come elemento positivo di protezione, ma anzi come ostacolo alla comunicazione con il mondo esterno sempre più bisognoso di connessione. Va ricordato inoltre che le mura di Rieti, come moti altri casi in tutto il mondo, sono state oggetto di interventi utilitaristici, come l'apertura di varchi o demolizioni di interi brani murari, fino alla fine degli anni '80 che ne hanno snaturato l'identità.

Quartieri della Memoria vuole dare una nuova importanza alle mura dal punto di vista dell'immagine urbana ponendosi come strumento di differente percezione

dell'architettura e della città nel suo complesso. Differente percezione che significa passaggio dalla semplice visione all'osservazione e alla conseguente conoscenza di aspetti probabilmente prima non indagati. Di qui l'eventuale nuova immagine della città che potrebbe scaturire nel cittadino, osservatore e coautore, durante la partecipazione alla nostra installazione interattiva.

Un altro elemento molto interessante, emerso dalla ricerca, riguardante la nostra città, è dato da una particolare classifica nazionale che vede Rieti al terzo posto nel numero di telecamere di sorveglianza per abitante. La decisione di fare delle telecamere uno dei "pilastri" della nostra installazione vuole quindi essere un segnale forte verso questa nuova "forma di controllo", proponendo un utilizzo alternativo di tali tecnologie.

Rieti infine, è tra i primi posti per quanto riguarda l'età media dei suoi abitanti, è una città anziana, in cui la vita notturna non offre molti spunti per i giovani e poche persone raggiungono Rieti se non per visitare uno dei suoi centri commerciali; in quest'ottica Quartieri della Memoria è stato pensato per lavorare durante le ore notturne, è stato progettato come elemento riqualificante di monumenti e vuole essere un mezzo di aggregazione, proponendo un'inversione di tendenza e nuove prospettive alla città di Rieti.

Vediamo ora come tutti i sopracitati elementi sono stati integrati tra di loro, prendendo come esempio l'area di Porta d'Arce (Piazza del Suffragio), e ricordando che il sistema può essere replicato nella sua struttura nelle altre locazioni.

Al centro della piazza verrà posizionato il podio per il libro delle firme (guestbook), questa sarà l'area nevralgica dell'intera installazione, qui infatti avverrà l'interazione tra l'utente ed il sistema. Il podio è stato progettato per consentire l'utilizzo da parte di un solo utente alla volta, in modo da dare ai visitatori un esperienza unica e personale. Il podio inoltre è stato dotato di sensori che consentono l'identificazione degli utenti al momento della loro interazione. I cittadini che hanno contribuito alla creazione dell'archivio delle memorie tramite l'apporto di foto e frammenti audio, saranno infatti dotati di una speciale penna, che durante la firma del guestbook ci consentirà di recuperare la loro identità e quindi l'eventuale materiale da essi apportato. Tale materiale, sarà così riprodotto dall'installazione attraverso l'utilizzo di un sistema sonoro per i contributi audio, e tramite dei proiettori per le fotografie. In particolare, le foto saranno proiettate su degli schermi posti a copertura dei due fornici presenti nelle mura di Porta d'Arce.

La persistenza dio ogni immagine nella sequenza è regolata dai movimenti del

visitatore che ha appena interagito col sistema. Per ottenere informazioni su posizione, velocità e direzione degli spostamenti di quest'ultimo, è stato realizzato un sistema di tracking basato su computer vision, e quindi sull'utilizzo di telecamere di sorveglianza puntate nell'area centrale della piazza, dove è presente il guestbook. Il sistema di computer vision ci consente di individuare anche il numero di persone presenti al centro della piazza. Abbiamo sfruttato questa informazione per modellare lo speciale evento "Pianara" che verrà attivato quando l'area intorno al guestbook sarà molto affollata. Questo evento proporrà la riproduzione di foto storiche relative alle alluvioni e dei suoni delle acque attraverso il sistema audio.

Telecamere di sorveglianza saranno poi poste in prossimità del guestbook, che seguendo gli spostamenti delle persone presenti nella piazza, dovranno dare ai visitatori la sensazione di essere osservati. Le immagini provenienti da tali telecamere saranno visualizzate su uno schermo, insieme ad immagini dello stesso tipo provenienti dagli altri luoghi dell'installazione, come ad esempio i Pozzi o il Ponte Romano, in modo da dare la possibilità ai visitatori di "controllare" le altre aree e di essere a loro volta "controllati" da altri.

1.2.3 - Il sistema

Prendendo come riferimento l'area di Porta d'Arci, analizziamo ora Quartieri della Memoria in maniera più dettagliata, elencando le sue funzionalità e dando una breve descrizione dei moduli che lo compongono. Ricordiamo inoltre, per chiarezza, che tutte le foto e file audio riprodotti durante il funzionamento dell'installazione sono stati forniti, durante una fase di preproduzione, dai cittadini di Rieti e raccolti nel cosiddetto "archivio delle memorie".

Possiamo immaginare il sistema come una macchina a stati, in ogni momento l'installazione si troverà in un determinato stato e il passaggio da uno stato ad un altro verrà incoraggiato dal verificarsi o meno di determinate situazioni/eventi.

- Quando non c'è interazione con i visitatori dell'installazione, il sistema si troverà in uno stato "no action". Sullo schermo saranno proiettate immagini a colori, con ritagli di vita quotidiana Reatina mentre in sottofondo, per accrescere l'esperienza visiva, si

percepiranno echi di suoni registrati precedentemente nella città, riprodotti in surround dal sistema sonoro. I colori delle foto e i suoni ricercati vogliono evidenziare la contrapposizione tra i momenti del passato e la nostra situazione attuale.

- L'interazione si verifica quando un visitatore decide di avvicinarsi al centro dell'installazione e firmare il guestbook. Il sistema cercherà allora di riconoscere il visitatore ed eventualmente fornirgli una esperienza personale. Il visitatore che avrà contribuito con la raccolta delle memorie, durante l'atto della firma, vedrà lo schermo sfumare e mostrare il suo contributo fotografico. Il visitatore intanto udirà, in sottofondo, la sua storia eventualmente associata alla fotografia, oppure parte della sua intervista, completando la sua esperienza. Mentre il visitatore lascerà il guestbook, il sistema terrà traccia dei suoi movimenti influenzando la velocità di proiezione delle fotografie. Il visitatore in questo modo avrà la possibilità di vivere un esperienza unica, che proprio egli attraverso le sue azioni potrà determinare.

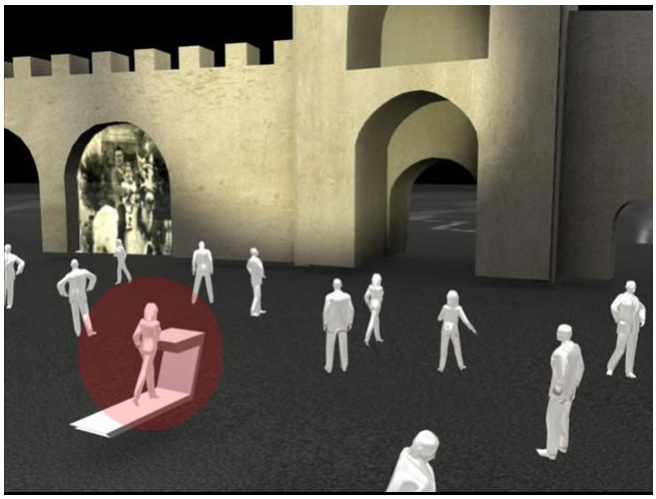

Figura 12 – Interazione con il sistema e visualizzazione immagini

- Dopo l'interazione di un utente, il sistema transita in uno stato di post interazione, e vi rimarrà fin quando un nuovo visitatore accederà al guestbook per lasciare una propria testimonianza.

- Alcuni degli avvenimenti, della storia di Rieti degli ultimi anni, che hanno segnato maggiormente la vita dei cittadini sono le alluvioni. Per ricordare ed evidenziare quanto drammatici furono quei momenti per la popolazione reatina di quei tempi, l'installazione genererà, in presenza di un elevato numero di persone, un evento “Pianara”. Il sistema sonoro e le fotografie sullo schermo rifletteranno i ricordi raccolti relativamente ai tempi delle alluvioni ed in sottofondo sfumeranno in maniera casuale i racconti degli intervistati relativi a quei tormentati momenti.

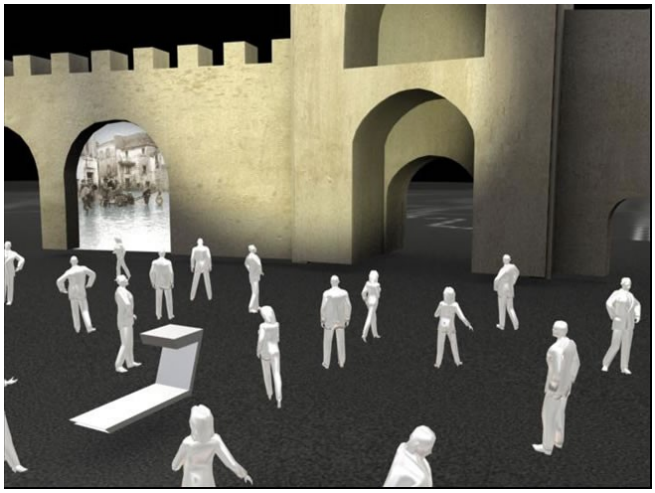

Figura 13 – Immagini e suoni della “Pianara”

Notiamo che una peculiarità del sistema è l'identificazione del visitatore. Infatti al momento dell'interazione, sarebbe impossibile riproporre attraverso gli schermi e il sistema audio il contributo apportato da quel particolare utente senza conoscerne l'identità. Questo problema è risolto brillantemente tramite l'utilizzo della tecnologia RFID di cui parleremo nei prossimi paragrafi. Ad ogni cittadino che fornisce i suoi dati, foto e racconti, verrà infatti assegnata una speciale penna contenente un tag RFID associato ad un codice che individuerà univocamente un visitatore, permettendoci di recuperare tutti i suoi dati al momento dell'interazione con la nostra installazione. L'identificazione avviene durante la firma del guestbook, tramite l'utilizzo di antenne poste in prossimità del podio.

1.2.4 - I componenti del sistema

Quartieri della Memoria è costituito da una serie di componenti la cui collaborazione è indispensabile per il funzionamento del sistema. Possiamo identificare in esso diverse parti: un Lettore RFID, un Database contenente le memorie popolari, una componente che si occupa della Visualizzazione, il sistema sonoro e ultimo ma non nella sua importanza il componente della Computer Vision, descritto più dettagliatamente nei capitoli successivi di questo documento.

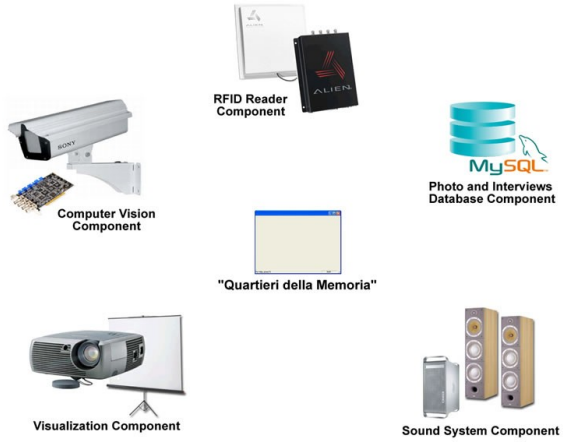

Figura 14 – Componenti sistema “Quartieri della Memoria”

1.2.4.1 - Il componente RFID

Radio Frequency Identification (RFID) è una tecnologia che usa onde radio per identificare fisicamente oggetti etichettati con un tag. Quando tali oggetti, sono nel raggio del lettore RFID, il tag trasmette il suo “codice univoco” al lettore che lo distribuisce in rete recapitandolo al componente centrale del sistema. Un tag passivo assomiglia ad un codice a barre, ma è un più sofisticato di quest'ultimo, infatti al suo interno integra un antenna utilizzata per trasmettere passivamente le informazioni al lettore e un chip dove è immagazzinato un codice numerico. Questa tecnologia è molto economica ed è abbastanza

affidabile nell'identificazione di oggetti sulle brevi distanze. La sua utilizzazione all'interno della nostra installazione è stata una scelta naturale, considerando anche l'esperienza maturata negli ultimi anni dal Remap Studio con questo tipo di sensori.

Figure 15, 16 – Tag e lettore RFID

1.2.4.2 - L'archivio

L'identificazione di un visitatore, come visto in precedenza, coincide con la lettura di un codice numerico da parte del sistema RFID. Tale codice è impiegato, dall'applicazione principale, per cercare nel database delle memorie tutte le informazioni relative al visitatore che sta interagendo con l'installazione. Il database contiene tutte le fotografie di Rieti, sia storiche che attuali, foto relative alla “Pianara” e i vari contributi audio. Alle memorie sono ovviamente associate delle informazioni su chi ha fornito il materiale ed inoltre, a scopo di catalogazione, richiediamo al fornitore di tali memorie di definire un soggetto, un riferimento temporale ed la provenienza geografica del materiale. Il database utilizza il DBMS MySQL per una maggiore compatibilità con il web server APACHE e il linguaggio di scripting per il web, PHP e HTML, con cui è stato realizzato il sito di Quartieri della Memoria.

1.2.4.3 - Il sistema sonoro

Lo scopo del sistema sonoro è di immergere i vari luoghi dell'installazione in ritagli di suoni e rumori registrati in particolari aree di Rieti, descrivendo in questo modo i “suoni della città”. Al momento dell'identificazione di un visitatore che sta interagendo con l'installazione, il sistema sonoro verrà fornito delle eventuali testimonianze audio di quel particolare visitatore immagazzinate nel database e provvederà alla creazione di uno strato

sonoro attraverso la riproduzione della voce della persona intervistata. Il sistema audio conoscerà in ogni momento lo stato dell'installazione, saprà ad esempio se è in corso un interazione oppure se è questa terminata da poco, compreso il momento della generazione della Pianara. La Pianara è gestita attraverso un particolare schema audio, composto da suoni che evocano e ricordano i tempi delle alluvioni. Il sistema sonoro verrà sviluppato interamente da David Beaudry del centro REMAP tramite l'utilizzo dell'ambiente di sviluppo audio Max/MSP. Il collegamento con Max/MSP è stato realizzato programmando una connessione UDP in Java con cui vengono trasmesse in rete le varie informazioni richieste da Max/MSP per modulare e creare il suono surround da emettere nell'installazione.

1.2.4.4 - Il componente della visualizzazione

Il componente della visualizzazione può essere pensato come la composizione di due diverse applicazioni:

Schermo Principale

Lo schermo principale, posto nel fornice di sinistra, è il mezzo con cui si manifesta l'interazione. In esso infatti verranno proiettate le immagini eventualmente forniteci dal visitatore che sta interagendo col sistema, ed è sempre qui che la “Pianara” verrà rievocata, attraverso la visualizzazione di foto storiche.

Dal punto di vista tecnico, la gestione dello schermo consiste in un applicazione autonoma realizzata in Flash ActionScript, che mostra delle fotografie ricordando un vecchio proiettore a diapositive. Il proiettore dinamicamente attende l'arrivo di nuove immagini da parte del sistema occupandosi semplicemente di sfumare e dare un ritmo alla proiezione. Il ritmo, o meglio il tempo di visualizzazione di ogni fotografia, viene suggerito dalla velocità di movimento del visitatore che ha appena firmato il guestbook, tracciato dal modulo di Computer Vision. Utilizzare Flash per questo lavoro significa poter aggiungere in futuro diverse transizioni alle immagini in maniera veloce, ma soprattuto avere la possibilità di creare maschere per proiettare correttamente in un arco e riuscire ad estendere la proiezione a 2 schermi allo stesso tempo creando un effetto ancora più interessante a livello artistico.

Figura 17 – I due schermi della visualizzazione

Lo Schermo Secondario

Lo schermo secondario è stato ideato per collegare fisicamente le tre locazioni, Porta d'Arce, i Pozzi ed il Ponte Romano, dove si è scelto di posizionare Quartieri della Memoria. Lo schermo mostrerà una composizione di immagini registrate direttamente dalle telecamere di sicurezza presenti in ogni locazione dell'installazione. Il sistema di Computer Vision potrebbe essere in grado di pilotare il movimento delle telecamere di sicurezza cercando di comprendere il movimento delle persone nel luogo delle riprese, dando la sensazione di essere osservati.

1.2.4.5 - Il modulo della Computer Vision

Lo scopo principale del sistema di tracking basato su computer vision è quello di tracciare i movimenti delle persone che stanno interagendo con il sistema, fornendo la loro posizione, velocità di movimento e direzione. Come è facile intuire quindi, tale sistema rivolgerà la propria attenzione al podio contenente il guestbook, dato che proprio lì avviene l'interazione tra utenti ed installazione.

La figura 18 mostra un possibile scenario di utilizzo del sistema di tracking nella zona di Porta d'Arce. In giallo è evidenziata l’area di copertura della telecamera che

corrisponde alla copertura del sistema di tracking, mentre intorno al guestbook abbiamo definito in rosso un area sensibile immaginaria. Il sistema di tracking seguirà gli spostamenti del solo visitatore che occuperà quest'area e cioè del visitatore che sta interagendo con l'installazione, firmando il guestbook. Il partecipante verrà tracciato fin quando non uscirà dall'area di copertura della telecamera (in giallo) o fin quando un nuovo partecipante non interagirà con l'installazione firmando il guestbook, entrando cioè nell'area sensibile (in rosso).

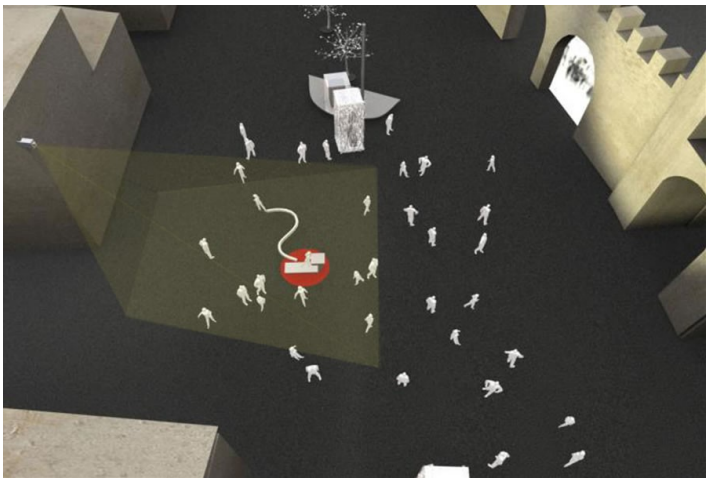

Figura 18 – Possibile scenario dell'utilizzo del sistema di Computer vision

Come già accennato in precedenza, i dati raccolti dal sistema di tracking, andranno ad influenzare il comportamento dell'installazione. In particolare la velocità degli spostamenti del visitatore modificherà il ritmo di proiezione delle immagini sugli schermi posti nei fornici, mentre il numero di persone presenti nell'area di copertura della telecamera potrà generare l'evento speciale "Pianara".

1.2.4.5 - Kolo e Sensor Fusion

Il collegamento delle varie componenti del sistema è effettuato tramite l'utilizzo di Kolo, un framework per il controllo distribuito dei media nelle arti, sviluppato da Eitan Mendelowitz e Jeff Burke del centro REMAP di Ucla.

Kolo consente il raggruppamento, attraverso una vista gerarchica, di tutti i dati forniti in tempo reale dai vari componenti della nostra installazione. Nel nostro caso, il componente Computer Vision e quello relativo alla tecnologia RFID gestiscono dati provenienti da sensori che Kolo provvede ad unificare in una stessa gerarchia, creando così una situazione di Sensor Fusion che ci da la possibilità di conoscere l'identità del visitatore che sta interagendo con il sistema e i suoi spostamenti nell'area circostante il guestbook.

1.2.5 - Alcune considerazioni

Quartieri della Memoria è stato progettato per essere un sistema distribuito composto da tre sottosistemi posti in altrettanti luoghi, che comunicano tra di loro inviando streaming video relativo agli avvenimenti di ciascuna postazione. Il tempo di sviluppo ha permesso di portare a termine il lavoro di progettazione dell'intero sistema e la realizzazione del solo sottosistema di Porta d'Arce, che comunque potrà essere replicato con facilità nelle altre postazioni. Il sistema informatico è infatti identico per ogni locazione, escludendo le maschere in Flash realizzate per proiettare le immagini esattamente negli schermi posti a copertura dei fornici.

La componente mancante è sono dunque quella relativa alla connessione in rete dei tre luoghi, che è stata ampiamente discussa ma mai avviata a livello implementativo, e la sua realizzazione è il primo passo verso l'estensione ed il completamento del sistema.

Da notare, inoltre, che lo studio tecnico dell'installazione prende in considerazione anche dei requisiti di riuso, dando la possibilità a gruppi di studenti, artisti e ricercatori di riutilizzare le componenti del sistema, ampliandole o servendosi di esse per nuovi scopi.
Per quanto riguarda la costruzione dell'archivio delle memorie, va notato, che raccogliere una cosi ingente quantità di informazioni richiederebbe una numerosa squadra di persone e costerebbe, tra l'altro, molto tempo e risorse. Per questo motivo, si è discussa la possibilità di coinvolgere gli istituti scolastici nella raccolta dei dati, studenti infatti sono in grado di garantire una raccolta capillare e organizzata delle memorie senza dover necessariamente procedere con una vasta operazione “porta a porta”.

Per facilitare la raccolta dei dati, si è deciso di allestire un sito internet, per dare la possibilità agli studenti di inviarci materiale in maniera semplice, veloce e stimolante.

Inoltre, chiunque desideri contribuire con delle memorie all'installazione potrà collegarsi direttamente al sito e inviare le sue fotografie ed interviste indicando anche se il materiale è riferito al fenomeno delle Pianare.

Ogni utente che ha partecipato alla costruzione dell'archivio, riceverà una speciale penna RFID caratterizzata da un codice univoco ed attraverso una apposita funzione fornita dal sito, avrà la possibilità di associare tale penna al suo account o a quello dei suoi intervistati, in modo da creare un collegamento univoco penna-visitatore.

Capitolo 2

2.1 - Introduzione alla Computer Vision

Prima di tutto la percezione visiva, punto di partenza e presupposto dell'intero lavoro:

"…Al centro della percezione visiva e' l'inferenza della struttura del mondo reale, derivata dalla struttura dell'immagine. La teoria della visione e' esattamente la teoria di come ciò può essere fatto, e il suo interesse principale e' nei confronti dei limiti fisici e delle assunzioni che rendono possibile tale inferenza"

La percezione visiva, insomma, produce un modello di informazioni su ciò che esiste nel mondo circostante, individua gli oggetti, la loro localizzazione, i relativi cambiamenti nel tempo e la loro rappresentazione. Grazie a tale modello un sistema biologico, l'uomo, o automatico, la macchina, possono esperire il mondo esterno ed interagire con esso.

La Computer Vision studia le tecniche e le tecnologie necessarie all'analisi automatica di immagini, finalizzate ad acquisire informazioni sul mondo esterno. Le tecniche studiate modellano diversi livelli cognitivi caratteristici della visione animale, da quello più basso (acquisizione dell'immagine) a quello più alto (interpretazione della scena).

La visione e' qualcosa più complesso della semplice capacita' sensoriale, infatti i processi mentali che occorrono dal rilevamento del pattern di luce sulla retina fino alla formazione di una rappresentazione del mondo, sono difficilmente scindibili dai più elevati processi cognitivi.

La visione artificiale si struttura su tre livelli di astrazione: inizia dal livello più basso, qui si produce una nuova immagine e poi, attraverso il medio livello si estraggono informazioni di tipo strutturale, fino a giungere al livello più alto, detto semantico, dove viene prodotta una interpretazione della scena. La visione a basso livello comprende un gran numero di operazioni atte a modificare l'immagine per evidenziare determinate caratteristiche. Tali elaborazioni non producono nuove informazioni, sfruttano le informazioni ricavabili dall'immagine stessa. In questa fase si generano una o più immagini

a partire da quella in ingresso. La visione a medio livello opera su immagini per estrarne informazioni di tipo strutturale, ovvero la composizione dell'immagine, il numero e le relazioni spaziali fra gli oggetti in essa presenti. La visione ad alto livello opera sulle informazioni provenienti dalla visione a medio livello per comporre un modello semantico della scena. Tale modello comprende una interpretazione della scena, per esempio il riconoscimento e la classificazione degli oggetti presenti nella scena. In genere questo livello fa uso di "conoscenza a priori" che può essere rappresentata da modelli della scena o degli oggetti che in essa possono presentarsi.

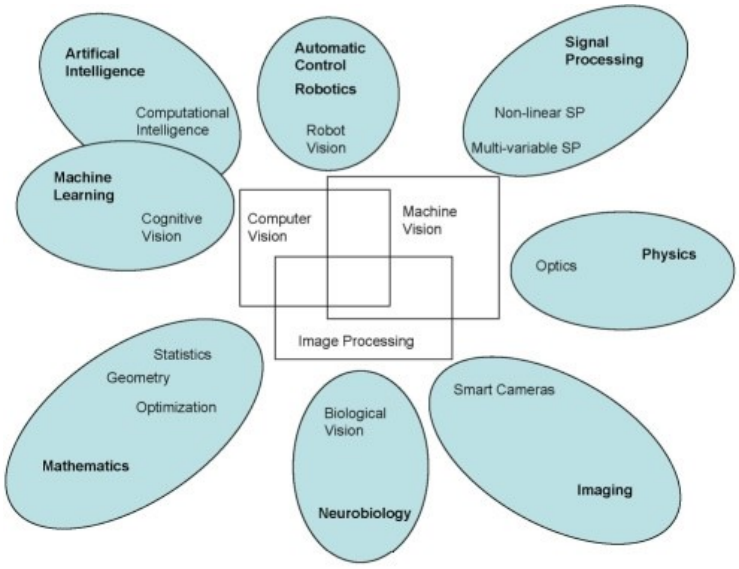

Figura 19– Computer vision, veduta generale.

Le applicazioni della Computer vision sono fra le più varie, per quanto riguarda le automazioni dei processi industriali ci sono apparati per ispezioni visuali e controlli di qualità; applicazioni spaziali e militari; molto usata nei sistemi di sorveglianza e tracking per sicurezza di aree ed edifici, per il traffico stradale; in generale nelle interazioni uomo-computer, per il riconoscimento di volti e per "gesture recognition"; nella realtà virtuale per la ricostruzione di scene e modelli.

Le discipline correlate alla computer vision sono l'image processing, che studia le proprietà delle immagini e le trasformazioni cui esse possono essere sottoposte, ad esempio i filtraggi, la compressione, la registrazione 3D, proiezioni; e la pattern recognition legata al

riconoscimento e classificazione degli oggetti, non solo visuali anche voice recognition per esempio.

2.2 - Stato dell'arte

Molte tecniche sono state utilizzate in lavori precedenti per il tracking di pedoni in spazi pubblici. La maggior parte di esse utilizzano i metodi di *frame differencing* e *background subtraction*.

Tramite il *frame differencing* si procede al calcolo della differenza tra due fotogrammi consecutivi in modo da evidenziare gli spostamenti di oggetti e persone, occorsi tra i due frame. Questa tecnica è generalmente accompagnata dall'estrazione di *features point* ed il conseguente calcolo del *flusso ottico* [16,17,18] applicato a questi punti, tramite il quale è possibile rilevare l'entità dello spostamento. Da notare che questo tipo di approccio consente l'individuazione dei soli individui in movimento ed è quindi poco affidabile nel caso in cui si sia interessati a conoscere costantemente la presenza di oggetti e individui, anche immobili, nella scena. Inoltre questa tecnica è in generale molto dispendiosa, dato che il calcolo del flusso ottico richiede l'utilizzo intensivo della CPU ed infatti, in alcuni lavori [17,18], si è tentato di ovviare a questo inconveniente spostando il carico computazionale dall'unità centrale al chip grafico, molto più performante nell'esecuzione di questo tipo di operazioni.

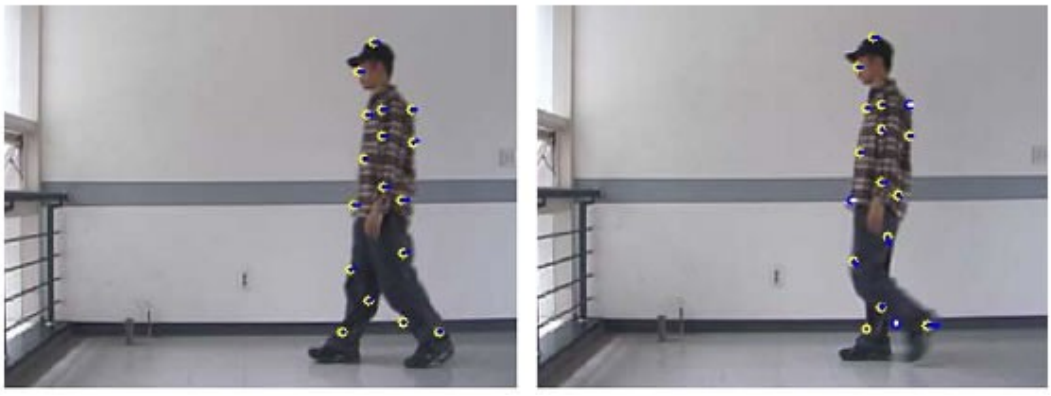

Figure 21, 21 – Frame differencing e flusso ottico

La tecnica del *background subtraction* consiste invece nel sottrarre da un modello del background calcolato a priori, ogni fotogramma, in modo da evidenziare tutti gli individui statici o in movimento presenti nella scena. Ora i dati estrapolati dall'eliminazione dello sfondo possono essere elaborati utilizzando diverse soluzioni: è possibile applicare nuovamente il *flusso ottico*, previa estrazione dei *features point*, oppure raggruppare i pixel che individuano gli individui in *cluster* ed analizzarne gli spostamenti da frame a frame servendosi, ad esempio, di algoritmi che si basano sulle caratteristiche del colore dell'individuo da tracciare come *Camshift* [4,5,]. Altro metodo che ha trovato svariate applicazioni è quello relativo all'utilizzo di classificatori che riconoscono le sagome dei pedoni in movimento [6,7,8,9,10,11]. Un esempio famoso di questa classe di metodi è il lavoro di Viola e Jones, in cui si utilizza il classificatore *AdaBoost* [6,7], istruito tramite svariati training set contenenti sagome di individui a passeggio, in corsa, immobili ed altre ancora. Questi metodi dipendono in un certo senso dal *background subtraction* che è molto sensibile ai repentini cambiamenti di luminosità e che potrebbe fallire anche nel caso in cui gli individui abbiano indumenti molto simili alle tonalità della scena.

Altri tecniche più sofisticate prevedono inoltre, la creazione di modelli cinematici basati sul *particle filtering* [13,14,15], o l'utilizzo di modelli statistici della scena e dell'individuo presente in essa, come ad esempio l'ottimo *Pfinder* [12]. Queste tecniche sono molto raffinate e consentono, sotto determinate condizioni oltre al tracking dell'individuo all'interno della scena, anche il riconoscimento del movimento degli arti: la cosiddetta "*gesture recognition*".

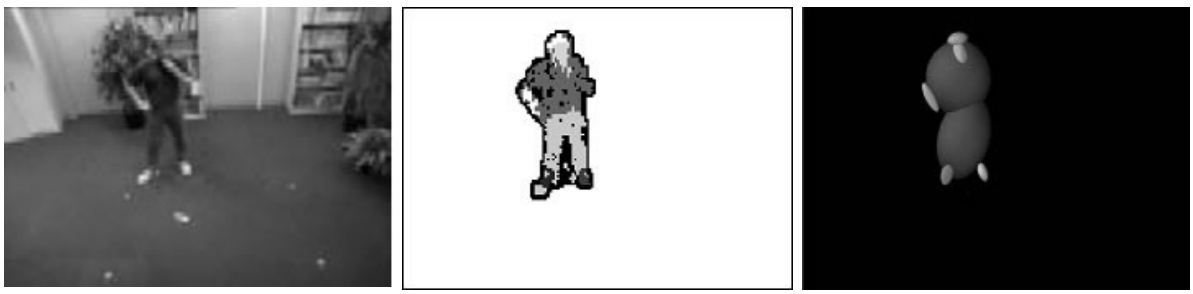

Figura 22 – Pfinder, modello statistico a blob

2.3 - Come, introduzione al metodo

Il presente lavoro ha privilegiato un approccio basato sulla tecnica del background subtraction, per avere la possibilità di rilevare la presenza di qualsiasi individuo od oggetto presente nella scena. Questa scelta è stata dettata dalla relativa facilità di realizzazione, e dalla possibilità di controllare l'illuminazione dell'ambiente in modo da garantire una certa affidabilità del metodo. Il risultato ottenuto da questa prima fase è una maschera di pixel in primo piano (attivi) che rappresentano le persone presenti nel campo visivo della telecamera. Il passo successivo consiste nel raggruppamento in cluster dei pixel, in modo da poter individuare gli individui ed estrapolare due caratteristiche fondamentali per ciascuno di essi: il centro di massa, con il quale approssimiamo la posizione all'interno dello spazio bidimensionale dell'immagine, ed il bounding box, ovvero il rettangolo che racchiude tutti i pixel di un individuo.

Quando una persona accede all'area sensibile definita all'interno della scena, il centro di massa ed il bounding box dell'individuo vengono trattate dal modulo che implementa l'algoritmo CamShift. Quest'ultimo provvederà a tracciare la posizione dell'individuo all'interno del campo visivo della telecamera basandosi sulle caratteristiche di colore, nel nostro caso tonalità di grigio, della persona. In questo modo saremo in grado di fornire per ogni istante la posizione e la velocità dell'individuo tracciato, dati che saranno resi disponibili per le altre componenti dell'installazione sfruttando le funzionalità fornite da Kolo, un sistema di gestione di oggetti distribuiti in rete, sviluppato internamente all'Hypermedia Studio.

2.3.1 - Rimozione dello sfondo

Si consideri la scena inquadrata dalla telecamera, tale scena può essere scomposta in due parti osservabili. Gli individui di cui si deve analizzare il movimento e tutto il resto che non siano individui. Questo secondo elemento, lo sfondo, non è interessato da alcun calcolo numerico, e per l'obiettivo preposto costituisce informazione trascurabile.

Il seguente paragrafo descrive le funzioni principali che rendono possibile la costruzione di un modello statistico relativo allo sfondo per la sua successiva eliminazione. Il termine sfondo (background) si riferisce ad una serie di pixel statici, non in movimento, appartenenti all'immagine, cioè a quei pixel che non appartengono a nessun oggetto in movimento di fronte alla camera. Questa definizione può variare se considerata in altra tecniche di eliminazione dell'oggetto. Per esempio, ottenuta una mappa di profondità della scena, lo sfondo può essere determinato come quella parte della scena stessa che è localizzata sufficientemente lontano dalla telecamera.

Il più semplice modello statistico di sfondo assume che la luminosità di ogni pixel dello sfondo stesso vari in modo indipendente, secondo la distribuzione normale. A questo punto si possono calcolare le caratteristiche dello sfondo attraverso l'accumulazione di diverse dozzine di fotogrammi (immagini), e i loro quadrati numerici. Ciò vuol dire che bisogna calcolare la somma dei valori di luminosità dei pixel nella posizione $S_{(x,y)}$ e la somma dei quadrati dei valori $Sq_{(x,y)}$ per la posizione di ogni pixel.
La media è calcolata come:

$$m_{(x,y)} = \frac{S_{(x,y)}}{N}$$

dove N è il numero dei fotogrammi collezionati nell'accumulatore. La media $m_{(x,y)}$ rappresenta quindi il modello statistico del background.
La deviazione standard è:

$$\sigma_{(x,y)} = \sqrt{\frac{Sq_{(x,y)}}{N} - \left(\frac{S_{(x,y)}}{N}\right)^2}$$

dopo questo non rimane che individuare quei pixel, che in una certa posizione e in un certo fotogramma, sono considerati parte di un oggetto in movimento se si verifica la condizione:

$$\left|m_{(x,y)} - P_{(x,y)}\right| > C \cdot \sigma_{(x,y)}$$

in cui $P_{(x,y)}$ è il fotogramma corrente e C è una costante. Se C è uguale a 3, si tratta della regola delle "tre-sigma".

Per ottenere il modello dello sfondo nella fase iniziale i due accumulatori $S_{(x,y)}$ e $Sq_{(x,y)}$ devono essere caricati alcuni fotogrammi privi di individui ed oggetti; tali fotogrammi devono riguardare solamente lo sfondo. Nella fattispecie N e' uguale a 300 e l'inizializzazione del modello impiega circa una decina di secondi utilizzando una telecamera NTSC avente frame-rate pari a 29.97.

La rimozione dello sfondo è eseguita per ogni fotogramma, ed il risultato di tale operazione è una maschera di pixel accesi che identificano gli individui ed oggetti presenti nella scena. Ad ogni fotogramma quindi è associata una maschera che mette in evidenza sia le parti in movimento che quelle statiche non appartenenti al modello dello sfondo.

La tecnica suddetta può essere migliorata. Si può cominciare col rendere il modello adattabile alle graduali variazioni di luminosità, per esempio agli spostamenti del sole, delle nuvole e cosi' via. Per questo è stata utilizzata una funzione di aggiornamento che va a modificare il modello statistico del background e che risponde alla seguente legge:

$$m_{(x,y)} = (1-\alpha) \cdot m_{(x,y)} + \alpha \cdot P_{(x,y)} \quad \text{se} \quad mask_{(x,y)} = 0$$

dove $mask_{(x,y)}$ è la maschera associata al fotogramma corrente $P_{(x,y)}$ e α è un parametro che regola la velocità di aggiornamento, indicando quanto velocemente il modello dimentica i contributi dei fotogrammi precedenti. Da notare che la condizione $mask_{(x,y)} = 0$ permette di aggiornare solamente i pixel che compongono lo sfondo, scartando il pixel relativi agli individui statici o in movimento presenti nella scena.

2.3.2 - Erosione

La rimozione dello sfondo ci da in output una maschera di pixel attivi corrispondenti agli oggetti e/o persone presenti nella scena. Tale maschera può presentare del rumore, ovvero dei pixel attivi sparsi in maniera casuale sulla maschera che non contengono informazioni significative. Tale rumore può essere eliminato utilizzando l'operatore morfologico Erode.

Erode è, insieme a Dilate, uno dei due operatori basilari nel campo della morfologia matematica. È tipicamente applicato ad immagini binarie (bianco-nero), ma in alcune versioni può lavorare anche con immagini a toni di grigio. L'effetto basilare dell'operatore su una immagine binaria è quello di erodere i bordi delle regioni dei pixel attivi (bianchi). In questo modo le aree composte da pixel attivi si restringono, ed i buchi all'interno di esse diventano più grandi. L'operatore di erosione prende in input una coppia composta dall'immagine che deve essere erosa e da un piccolo insieme di coordinate di punti conosciuto come elemento strutturato o kernel. È quest'ultimo che determina l'effetto dell'erosione sull'immagine in input. L'elemento strutturato consiste in un modello specificato dalle coordinate di un numero di punti discreti relativi a qualche origine. Generalmente sono utilizzate le coordinate cartesiane, così una conveniente rappresentazione è data da una piccola immagine in una griglia rettangolare. La figura 22 mostra alcuni elementi strutturati di diverse dimensioni, aventi l'origine evidenziata da un anello.

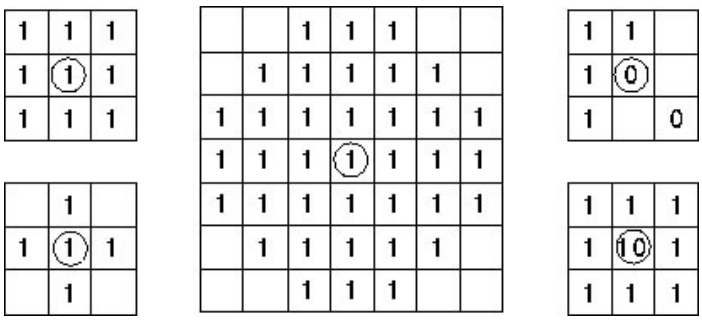

Figura 23 – Alcuni elementi strutturati

La definizione matematica dell'operatore *Erode* per immagini binarie è data da:

- supponiamo che *X* sia l'insieme di coordinate euclidee, corrispondenti all'immagine binaria in input, e *K* sia l'insieme di coordinate dell'insieme strutturato.
- denotiamo con *Kx* la traslazione di *K* in modo tale da avere la sua origine nel punto *x*.
- l'erosione di *X* utilizzando *K* è semplicemente l'insieme di tutti i punti *x* tali che *Kx* è un sottoinsieme di *X*.

Per calcolare l'erosione di una immagine binaria in input utilizzando ad esempio un elemento strutturato 3x3, consideriamo, a turno, ogni pixel attivo dell'immagine. Per ogni pixel attivo sovrapponiamo l'elemento strutturato sull'immagine in input in modo che l'origine dell'elemento strutturato coincida con le coordinate del pixel in input. Se per ogni pixel dell'elemento strutturato, il corrispondente pixel nell'immagine sottostante è un pixel attivo, allora il pixel in input è lasciato come è. Se qualcuno dei pixel corrispondenti nell'immagine è disattivato, allora anche il pixel in input è a sua volta disattivato.

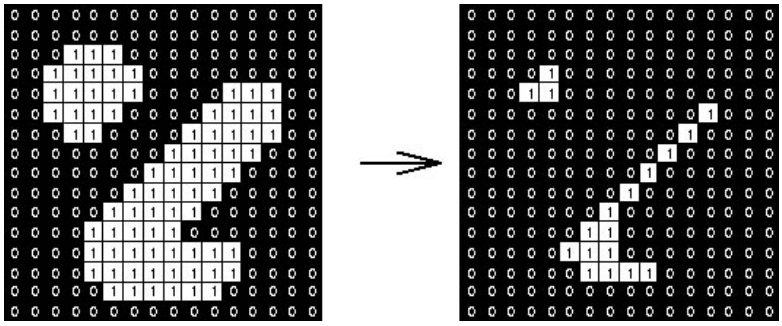

Figura 24 – Effetti operatore Erode

2.3.3 - Clustering

Una volta ottenuta la maschera degli oggetti presenti nella scena, tramite la tecnica di rimozione del background, siamo in presenza di una serie di pixel attivi (bianchi) su uno sfondo completamente nero. Tali pixel hanno poco significato se considerati singolarmente, le informazione che ne possiamo ricavare sono solamente le loro coordinate all'interno dello spazio bidimensionale dell'immagine. Ogni pixel però contribuisce alla composizione di un oggetto o un individuo a cui siamo interessati, quindi l'idea principale è quella di raggruppare tutti i pixel che compongono un oggetto tra di loro per considerarli in maniera unitaria. Tale raggruppamento è ottenuto mediante tecniche di clustering, di cui diamo una breve descrizione nel paragrafo seguente.

Il *Clustering* o *analisi dei cluster* o *analisi di raggruppamento* è una tecnica di analisi multi variata dei dati volta alla selezione e raggruppamento di elementi omogenei in un insieme di dati. Tutte le tecniche di clustering si basano sul concetto di distanza tra due elementi. Infatti la bontà delle analisi ottenute dagli algoritmi di clustering dipende essenzialmente da quanto è significativa la metrica e quindi da come è stata definita la distanza. La distanza è un concetto fondamentale dato che gli algoritmi di clustering raggruppano gli elementi a seconda della distanza e quindi l'appartenenza o meno ad un insieme dipende da quanto l'elemento preso in esame è distante dall'insieme. Le tecniche di clustering si possono basare principalmente su due filosofie.

Dal basso verso l'alto

Questa filosofia prevede che inizialmente tutti gli elementi siano considerati cluster a sé e poi l'algoritmo provvede ad unire i cluster più vicini. L'algoritmo continua ad unire elementi al cluster fino ad ottenere un numero prefissato di cluster oppure fino a che la distanza minima tra i cluster non supera un certo valore.

Dall'alto verso il basso

All'inizio tutti gli elementi sono un unico cluster e poi l'algoritmo inizia a dividere il cluster in tanti cluster di dimensioni inferiori. Il criterio che guida la divisione è sempre quello di cercare di ottenere elementi omogenei. L'algoritmo procede fino a che non ha raggiunto un numero prefissato di cluster. Questo approccio è anche detto gerarchico

La tecnica da noi utilizzata è di tipo dal basso verso l'alto ed è nota col nome di *k-mean clustering*.

2.3.3.1 - K-mean clustering

Creato nel 1967 da MacQueen, semplicemente parlando, *K-mean clustering* è un algoritmo per classificare o raggruppare oggetti in base ad attributi/caratteristiche in *K* gruppi (cluster), dove *K* è un numero intero positivo. Il raggruppamento è eseguito minimizzando la somma dei quadrati delle distanze tra i dati e i corrispondenti centroidi del cluster. Lo scopo di *K-mean clustering* è quello di classificare le informazioni.

L'algoritmo mira a minimizzare una funzione obbiettivo che in questo caso è una funzione quadratica di errore:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|$$

dove $\left\| x_i^{(j)} - c_j \right\|$ è una data misura di distanza scelta tra un punto $x_i^{(j)}$ e il centroide del cluster c_j ed è un indicatore della distanza tra gli *n* punti dai loro rispettivi centri dei cluster. Nella nostra applicazione i punti $x_i^{(j)}$ sono i pixel attivi nella maschera e la misura di distanza utilizzata è quella euclidea. L'iterazione dell'algoritmo è descritta dal seguente diagramma a blocchi:

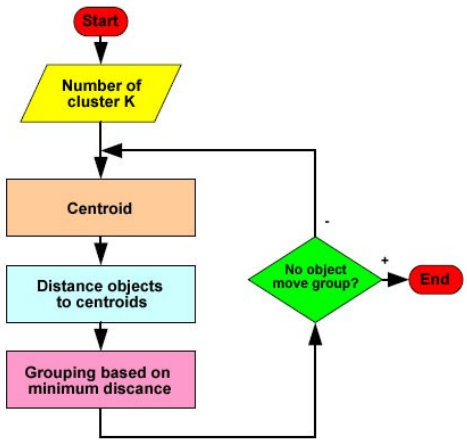

Figura 25 – Diagramma a blocchi algoritmo K-mean

Se il numero di oggetti è minore del numero di cluster allora consideriamo ogni oggetto come centroide di un cluster in modo che ogni oggetto sarà un cluster. Altrimenti per ogni oggetto, calcoliamo la distanza da tutti i centroidi e lo assegnamo al cluster avente il centroide più vicino.

Poiché non siamo sicuri sulla localizzazione dei centroidi, abbiamo bisogno di correggere la loro posizione basandoci sui dati correntemente aggiornati. Assegnamo poi tutti gli oggetti ai questo nuovo centroidi, basandoci sempre sulle distanze. Questo procedimento è ripetuto finché non ci sono più movimenti di oggetti da un cluster all'altro. Questa iterazione è matematicamente convergente.

2.3.3.2 - Esempio

Proponiamo adesso un piccolo esempio numerico per comprendere meglio il funzionamento dell'algoritmo *K-mean clustering*. Supponiamo di avere una serie di oggetti (quattro tipi di medicine) aventi ciascuno due attributi o caratteristiche, come mostrato nella tabella sottostante. Il nostro obiettivo è quello di raggruppare gli oggetti in $K=2$ gruppi di medicine basandoci sulle due caratteristiche (*pH* e indice di peso).

Oggetto	***Caratteristica 1 (X): peso***	***Caratteristica 2 (Y): pH***
Medicina A	1	1
Medicina B	2	1
Medicina C	4	3
Medicina D	5	4

Tabella 1 – Caratteristiche oggetti

Ogni medicina rappresenta un punto con due caratteristiche *(X,Y)* che rappresentano le coordinate in uno spazio caratteristico come mostrato nella seguente figura:

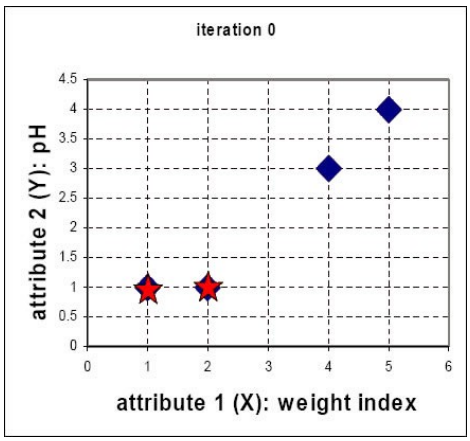

Figura 26 – K-mean – Iterazione 0

Passo 1

Iterazione 0, valore iniziale dei centroidi. Supponiamo di utilizzare le medicine *A* e *B* come primi centroidi e denotiamo con c_1 e c_2 le coordinate dei centroidi, quindi $c_1=(1,1)$ e $c_2=(2,1)$.

Passo 2

Iterazione 0, distanza oggetti-centroidi. Calcoliamo la distanza tra il centroide del cluster ed ogni oggetto, utilizzando la distanza euclidea. La matrice delle distanze all'iterazione *0* è la seguente:

$$D^0=\begin{bmatrix}0 & 1 & 3.61 & 5\\ 1 & 0 & 2.83 & 4.24\end{bmatrix} \quad \begin{matrix}c_1=(1,1) & gruppo-1\\ c_2=(2,1) & gruppo-2\end{matrix}$$

$$\begin{matrix}A & B & C & D\end{matrix}$$

$$\begin{bmatrix}1 & 2 & 4 & 5\\ 1 & 1 & 3 & 4\end{bmatrix} \quad \begin{bmatrix}X\\ Y\end{bmatrix}$$

Ogni colonna nella matrice della distanze rappresenta l'oggetto. La prima riga della matrice delle distanze corrisponde alla distanza di ogni oggetto dal primo centroide e la seconda colonna è la distanza di ciascun oggetto dal secondo centroide. Per esempio la distanza della

medicina $C=(4,3)$ dal primo centroide $c_1=(1,1)$ è $\sqrt{((4-1)^2+(3-1)^2)}=3.61$ e la sua distanza dal secondo centroide $c_2=(2,1)$ è $\sqrt{((4-2)^2+(3-1)^2)}=2.83$ e così via.

Passo 3

Iterazione 0, raggruppamento degli oggetti. Assegnamo gli oggetti ai gruppi basandoci sulla distanza minima, così la medicina *A* è assegnata al gruppo *1*, e le medicine *B, C* e *D* al gruppo *2*. Gli elementi della matrice dei gruppi sono posti ad *1* se e solo se l'oggetto è assegnato al quel gruppo.

$$G^0=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 1 & 1\end{bmatrix}\quad \begin{matrix}gruppo-1\\ gruppo-2\end{matrix}$$

$$A \quad B \quad C \quad D$$

Passo 4

Iterazione 1, determinazione dei centroidi. Conoscendo i membri di ogni gruppo, adesso possiamo calcolare i nuovi centroidi di ogni gruppo basandoci su queste nuove appartenenze. Il gruppo 1 ha solamente un elemento così il centroide rimarrà $c_1=(1,1)$ mentre il gruppo *2* ha tre membri che andranno a modificare le posizione del centroide. Questo infatti è dato dalla media delle coordinate dei tre membri: $c_2=\left(\frac{11}{3},\frac{8}{3}\right)$.

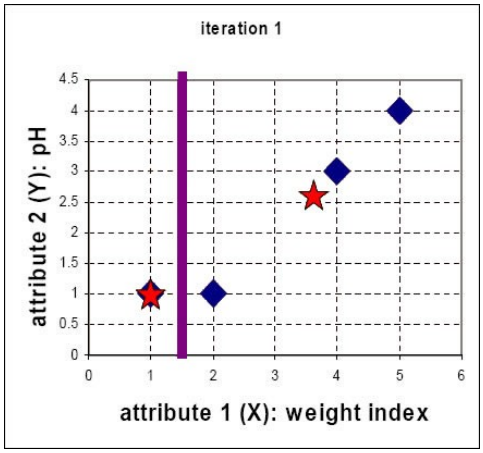

Figura 27 – K-means – Iterazione 1

Passo 5

Iterazione 1, distanza oggetti-centroidi. Il passo successivo è quello di calcolare la distanza di tutti gli oggetti dai nuovi centroidi. In maniera simile al passo due la matrice delle distanze all'iterazione uno è:

$$\boldsymbol{D}^1=\begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} \boldsymbol{c}_1=(1,1) & gruppo-1 \\ \boldsymbol{c}_2=(\frac{11}{3},\frac{8}{3}) & gruppo-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{bmatrix} X \\ Y \end{bmatrix}$$

Passo 6

Iterazione 1, raggruppamento degli oggetti: in modo simile al passo tre assegnamo ogni oggetto, in base alla distanza minima, ai gruppi. Basandoci sulla nuova matrice delle distanze muoviamo la medicina *B* nel gruppo 1 mentre tutti gli altri oggetti rimangono immutati. La nuova matrice dei gruppi è:

$$\boldsymbol{G}^1=\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} gruppo-1 \\ gruppo-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Passo 7

Iterazione 2, calcolo dei centroidi: adesso ripetiamo il passo 4 per calcolare le coordinate dei nuovi centroidi basandoci sul raggruppamento ottenuto dalla precedente iterazione. Il gruppo *1* e *2* hanno entrambi due elementi, così i nuovi centroidi sono:

$$\boldsymbol{c}_1=\left(\frac{3}{2},1\right) \quad \text{e} \quad \boldsymbol{c}_2=\left(\frac{9}{2},\frac{7}{2}\right)$$

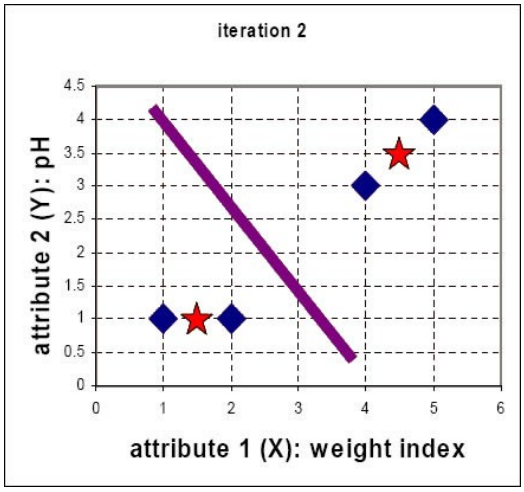

Figura 28 – K-means – Iterazione 2

Passo 8

Iterazione 2, distanza oggetti-centroidi. Ripetiamo il passo 2 di nuovo ed otteniamo la nuova matrice delle distanze:

$$\boldsymbol{D}^1=\begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} \boldsymbol{c}_1=(\frac{3}{2},1) & gruppo-1 \\ \boldsymbol{c}_2=(\frac{9}{2},\frac{7}{2}) & gruppo-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{bmatrix} X \\ Y \end{bmatrix}$$

Passo 9

Iterazione 2, raggruppamento degli oggetti. Di nuovo, assegnamo ogni oggetto basandoci sulla distanza minima:

$$\boldsymbol{G}^2=\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} gruppo-1 \\ gruppo-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Otteniamo il risultato $\boldsymbol{G^1 = G^2}$ comparando i raggruppamenti delle ultime due iterazioni, otteniamo come risultato che nessun oggetto si è spostato da un gruppo all'altro. Così la computazione del K-mean clustering ha raggiunto la sua stabilità e non occorrono ulteriori iterazioni.

Similmente ad altri algoritmi, k-mean ha alcuni punti deboli:

- Quando il numero dei dati non è così elevato, il raggruppamento iniziale determinerà significativamente i cluster.
- Il numero dei cluster deve essere determinato a priori.
- Non abbiamo mai una clusterizzazione reale, utilizzando gli stessi dati, potremmo ottenere raggruppamenti differenti.
- Non siamo a conoscenza di quale attributo contribuisce in maniera più determinante al processo di clustering poiché assumiamo che ogni attributo abbia lo stesso peso.

Una possibile soluzione a questi inconvenienti è quello di utilizzare k-mean solo in presenza di un numero elevato di dati.

Per generalizzare l'algoritmo k-mean clustering in *n* attributi, definiamo i centroidi come vettori dove ogni componente è il valore medio di quella componente. Ogni componente rappresenta un attributo, così ogni punto *j* ha *n* componenti ed è denotato da: $p_j(x_{j1}, x_{j2}, x_{j3}, \ldots, x_{jm}, \ldots, x_{jn})$. Se abbiamo *N* punti di addestramento, le *m* componenti del centroide possono essere calcolate come:

$$\bar{x_m} = \frac{1}{N} \cdot \sum_j x_{jm}$$

2.3.4 - Tracking

Una volta individuati gli oggetti all'interno dell'immagine ed averne calcolato il centro di massa e la distribuzione, è possibile seguire i loro spostamenti da fotogramma a fotogramma, utilizzando tecniche di tracking. Il metodo utilizzato nel sistema è l'algoritmo *CamShift* rivelatosi flessibile ed adattabile al nostro problema ed anche computazionalmente efficiente.

2.3.4.1 - CamShift

CamShift (Continuously Adaptive Mean Shift), sviluppato da *Bradski* nel 1998, è principalmente progettato per realizzare il tracking efficiente della testa e del viso in un'interfaccia utente percettiva. E' basato su un adattamento dell'algoritmo *Mean Shift* che data una densità di probabilità di una immagine, trova la media (moda) della distribuzione effettuando delle iterazioni nella direzione della massima crescita (gradiente) della densità di probabilità.

I due algoritmi si differenziano sostanzialmente nella gestione delle distribuzioni di probabilità *CamShift* infatti utilizza una distribuzione di probabilità continuamente adattata (la distribuzione è calcolata per ogni frame) mentre *Mean Shift* è basato su una distribuzione statica che non è aggiornata fin quando l'obiettivo non modifica in maniera significativa la sua forma, colore e dimensione. Inoltre dato che *CamShift* non mantiene distribuzioni statiche, i momenti spaziali sono utilizzati per iterare verso la moda della distribuzione. Questo è in contrasto con l'implementazione convenzionale dell'algoritmo *Mean Shift* dove l'obiettivo e le distribuzioni candidate sono utilizzate per iterare verso il massimo incremento di densità utilizzando il rapporto tra la corrente distribuzione (candidata) e l'obiettivo.

Vediamo ora in dettaglio i due algoritmi. L'algoritmo *Mean Shift* può essere descritto brevemente dai seguenti passi:

1. Scegliere la dimensione della finestra di ricerca.
2. Scegliere la locazione iniziale della finestra di ricerca.
3. Calcolare la locazione media all'interno della finestra.
4. Centrare la finestra di ricerca nella locazione media calcolata al passo 3.

5. Ripetere i passi 3 e 4 fino alla convergenza. (o finché la locazione media si muove meno di una soglia prefissata).

La prova di convergenza di tale algoritmo riflette i passi indicati qui sopra, infatti, considerando uno spazio di distribuzione Euclidea contenente la distribuzione *f*, abbiamo:

1. Scegliere la dimensione *s* della finestra di ricerca *W*.
2. Posizionare la finestra di ricerca iniziale nel punto p_k .
3. Calcolare la locazione media all'interno della finestra:

$$\hat{\bar{p}}(W)\frac{1}{|W|}\sum_{j\in W} p_j$$

4. qui *Mean Shift* risale il gradiente di *f(p)*:

$$\hat{\bar{p}}(W)-p_k\approx\frac{f'(p_k)}{f(p_k)}$$

5. Centrare la finestra nel punto $\hat{\bar{p}}(W)$.
6. Ripetere i passi 3 e 4 fino alla convergenza.

Vicino alla moda, $f'(p)\approx 0$ otteniamo così la convergenza dell'algoritmo.

L'algoritmo *CamShift* può essere invece riassunto nei seguenti passi, da notare come *CamShift* estenda ed utilizzi al suo interno l'algoritmo *Mean Shif*t:

1. Impostare la regione di interesse (ROI) dell'immagine di distribuzione di probabilità all'intera immagine.
2. Selezionare una locazione iniziale della finestra di ricerca di *Mean Shift*. La locazione selezionata è la distribuzione obiettivo da tracciare.
3. Calcolare una distribuzione di probabilità di colore della regione centrata nella finestra di ricerca di *Mean Shift*.

4. Iterare l'algoritmo *Mean Shift* per trovare il centroide dell'immagine di probabilità. Memorizzare il momento zeresimo e la posizione del centroide.
5. Per il frame successivo, centrare la finestra di ricerca nella posizione media trovata al passo 4 e impostare la dimensione della finestra in funzione del momento zeresimo. Tornare al passo 3.

La struttura dell'algoritmo è mostrata dal seguente diagramma a blocchi, dove la parte evidenziata in grigio identifica l'algoritmo *Mean Shift*:

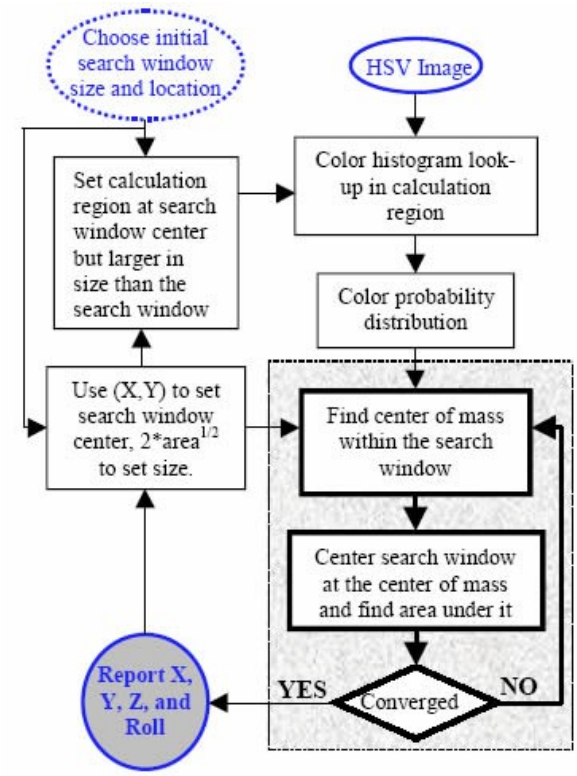

Figura 29 – Diagramma a blocchi dell'algoritmo CamShift

Per il calcolo dell'immagine della distribuzione di probabilità è possibile utilizzare qualsiasi metodo che associ il valore di un pixel alla probabilità che il pixel in questione appartenga all'obiettivo. Un metodo comune, utilizzato da *Camshift* è conosciuto come *Histogram Back-Projection*. Allo scopo di generare tale immagine un istogramma iniziale è

calcolato al primo passo dell'algoritmo utilizzando la regione di interesse iniziale.

L'istogramma utilizzato da *Bradski* si basa sul canale colore (*hue*) dello spazio colore HSV tuttavia possono essere utilizzati istogrammi multidimensionali di ogni spazio colore, nel nostro caso infatti lavoreremo utilizzando fotogrammi a 256 toni di grigio. L'istogramma è quantizzato in "contenitori" (*bins*) che riducono la complessità spaziale e computazionale e permettono a colori simili di essere raggruppati insieme. Tali "contenitori", sono poi scalati tra l'intensità massima e minima dell'immagine di probabilità.

L'*Histogram Back-Projection* è una operazione primitiva che associa i valori dei pixel dell'immagine con il valore del corrispondente *bin* dell'istogramma. La retro proiezione dell'istogramma obiettivo con qualsiasi fotogramma consecutivo genera un'immagine di probabilità dove il valore di ogni pixel caratterizza la probabilità che il pixel in input appartenga all'istogramma utilizzato. Dati gli *m-bin* dell'istogramma in uso, definiamo le *n* locazioni dei pixel dell'immagine come $\{x_i\}_{i=1...n}$ e l'istogramma come $\{\hat{q}\}_{u=1...m}$.

Definiamo anche una funzione $c:\mathbb{R}^2 \rightarrow \{1...m\}$ che associa al pixel posto nella locazione x_i^* il contenitore (*bin*) di indice $c(x_i^*)$ dell'istogramma. L'istogramma non pesato è calcolato come:

$$\hat{q}_u = \sum_{i=1}^{n} \delta\left[c(x_i^*) - u\right]$$

In ogni caso, i valori dei contenitori dell'istogramma sono scalati entro un range di pixel discreto dell'immagine di distribuzione di probabilità bidimensionale, usando l'equazione:

$$\left\{ \hat{p}_u = min\left(\frac{255}{max(\hat{q})} \hat{q}_u \ , \ 255 \right) \right\}_{u=1...m}$$

cioè, i valori dei *bin* dell'istogramma sono scalati nuovamente da $[0 \ , \ max(q)]$ al nuovo range $[0 \ , \ 255]$, dove i pixel con la più alta probabilità di essere nel semplice istogramma saranno mappati come entità visibili nell'immagine bidimensionale della *Histogram Back-Projection.*

Da notare inoltre come la locazione media (centroide) all'interno della finestra di

ricerca dell'immagine di probabilità discreta calcolata al terzo passo dell'algoritmo *Mean Shift*, è adesso trovata utilizzando i momenti (*Horn*, 1986; *Freeman* ed altri, 1996; *Bradski*, 1998). Definendo ***I**(x,y)* l'intensità dell'immagine di probabilità discreta in *(x,y)* all'interno della finestra di ricerca, il centro di massa è calcolato in base ai seguenti passi:

a) Calcolare il momento zeresimo:

$$M_{00} = \sum_x \sum_y \boldsymbol{I}(x, y)$$

b) Trovare il momento primo per x ed y:

$$M_{10} = \sum_x \sum_y x\, \boldsymbol{I}(x, y)$$

$$M_{01} = \sum_x \sum_y y\, \boldsymbol{I}(x, y)$$

c) Calcolare la finestra media di ricerca:

$$x_c = \frac{M_{10}}{M_{00}} \;\; ; \;\; y_c = \frac{M_{01}}{M_{00}}$$

La componente *Mean Shift* dell'algoritmo è quindi implementata ricalcolando continuamente nuovi valori di (x_c, y_c) per la posizione della finestra calcolata nel frame precedente finché non ci sono significativi spostamenti della posizione. Il massimo numero di iterazioni di *Mean Shift* è generalmente impostato a 10-20 iterazioni. Dato che l'accuratezza sub-pixel non può essere osservata a occhio nudo, un minimo spostamento di un pixel in una delle due direzioni *x* od *y* è selezionato come criterio di convergenza. Inoltre l'algoritmo deve terminare nel caso in cui M_{00} è zero, corrispondente ad una finestra il cui contenuto è interamente posto ad intensità zero.

Capitolo 3

3.1 - Sistema di Tracking

Dopo la breve descrizione teorica dei paragrafi precedenti, esaminiamo ora i dettagli implementativi del sistema, analizzando le singole componenti e l'architettura software creata, soffermandoci in particolare sulla descrizione degli input e degli output gestiti dal sistema.

Gli elementi principali sono dunque:

- Telecamera;
- Frame Grabber (scheda di acquisizione video);
- Librerie di Computer Vision (openCV e Integrated Performance Primitive della Intel)
- Modulo di tracking
- Kolo/Spread (per l'integrazione con le altre componenti dell'installazione QdM)

3.1.1 - Telecamera

Questo componente è senza dubbio uno degli elementi critici del sistema e sapendo che Quartieri della Memoria è stato progettato per essere utilizzato durante le ore notturne, abbiamo preferito acquistare una telecamera Night & Day, in grado di operare indifferentemente a qualsiasi ora della giornata.

Il modello prescelto è stato *Sony Dyna View SSC-DC593 color camera*. Tale modello è molto duttile e si presta bene al lavoro sia durante le ore diurne, con immagini di alta qualità a colori, che durante la notte, con immagini in bianco e nero estremamente nitide. La telecamera incorpora un sensore 1/3 di tipo interline transfer CCD da 380,000 pixel ed è in grado di funzionare con una illuminazione minima di 0.07 lx in modalità bianco e nero. Con la tecnologia CCD IRIS inoltre la telecamera è in grado di settare la corretta esposizione in relazione alla luminosità dell'ambiente. Per i nostri scopi, abbiamo impostato la camera in modo tale da ottenere immagini in bianco e nero, di dimensioni pari a 640*480 pixel e con velocità di aggiornamento pari a 29.97 frame/sec, (standard video NTSC).

Figure 30, 31 – Installazione telecamera MacGowan Hall (UCLA)

La telecamera, durante la fase di sviluppo, è stata posizionata sul tetto di un edificio (MacGowan Hall) adiacente il nostro studio, e puntata verso un cortile molto frequentato dagli studenti della scuola di cinema di UCLA.

Per l'installazione della telecamera, è stata richiesta la costruzione di una piccola base di legno, su cui ancorare la telecamera stessa. Per dare solidità alla struttura, al suo interno sono stati posti dei sand bag (sacchi di sabbia) utilizzati per il teatro. E' stato inoltre installato un cavo BNC video che trasporta il segnale video direttamente all'interno del nostro studio, ed un cavo elettrico per alimentare il tutto. Ad una altezza di circa 10 metri dal suolo e con una inclinazione di circa 60 gradi, la copertura della telecamera di forma trapezoidale e le sue dimensioni sono pari a circa 16 metri in larghezza e 14 in lunghezza. Con questa configurazione, ogni individuo all'interno della scena è composto mediamente da circa 1000 pixel. La scelta della posizione della telecamera è stata fatta in modo da poter effettuare dei test in maniera coerente alla condizioni presenti a Rieti nell'area per cui è stata progettata l'installazione.

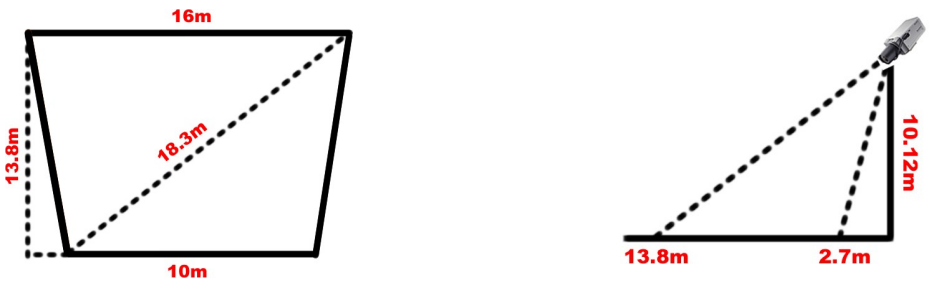

Figure 32, 33 – Area di copertura della telecamera

3.1.2 - Frame Grabber

Le immagini fornite dalla telecamera, sono trasportate direttamente al computer tramite un cavo BNC video, tale cavo è collegato ad una scheda di acquisizione video, *Matrox Morphis* dotato di 4 ingressi video, ed accompagnato dalle librerie *MIL Lite 8.0* che danno supporto nelle operazioni di manipolazione dei fotogrammi e di input/output quali acquisizione da camera, lettura/scrittura su disco, compressione/ decompressione ed altre.

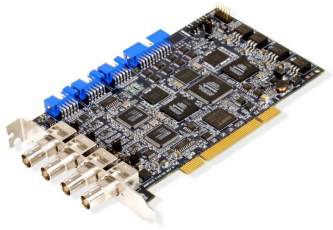

Figure 34, 35 – Componenti: Telecamera Sony SSC-DC593 e Frame grabber Matrox Morphis

3.1.3 - Librerie per Computer Vision

Le librerie utilizzate durante la fase di sviluppo del sistema di tracking sono le librerie open source *OpenCV* versione 5a di Intel. Queste librerie sono principalmente rivolte al supporto della computer vision in tempo reale e forniscono una ampia varietà di strumenti per l'interpretazione delle immagini. *OpenCV* è compatibile con l' *Image Processing Library* (IPL) di Intel che implementa operazioni a basso livello su immagini digitali, e ne condivide parte delle strutture dati. Nonostante primitive come il filtraggio o la raccolta di statistiche per le immagini, *OpenCV* è una libreria ad alto livello che implementa algoritmi per tecniche di calibrazione della telecamera, riconoscimento di caratteristiche, tracking (flusso ottico), analisi del moto, analisi delle forme ed altri. *OpenCV* è ottimizzata per le architetture Intel, ed in questo senso trae beneficio dall'utilizzo della libreria *Integrated Performance Primitives* (IPP) di Intel che fornisce funzioni a basso livello altamente ottimizzate.
OpenCV è una libreria C/C++ che gode di un ottimo supporto da parte della comunità open source di sviluppatori di computer vision ed è provvista di una buona documentazione.

3.1.4 - Modulo di Tracking

Questa componente è il cuore del sistema di tracking, qui vengono gestiti ed elaborati i fotogrammi forniti dalla telecamera e prodotti gli output del sistema, come la posizione e velocità dell'individuo tracciato, il numero di persone presenti nella scena, ed altri. L'architettura del sistema, evidenziata dal seguente schema, è molto semplice, privilegiando una separazione delle varie funzionalità in classi come ad esempio bgModel, dataOut, kmeans, tracker ed altre, ognuna delle quali fornisce i propri servizi alla classe quartieriMemoria che gestisce al suo interno il loop principale.

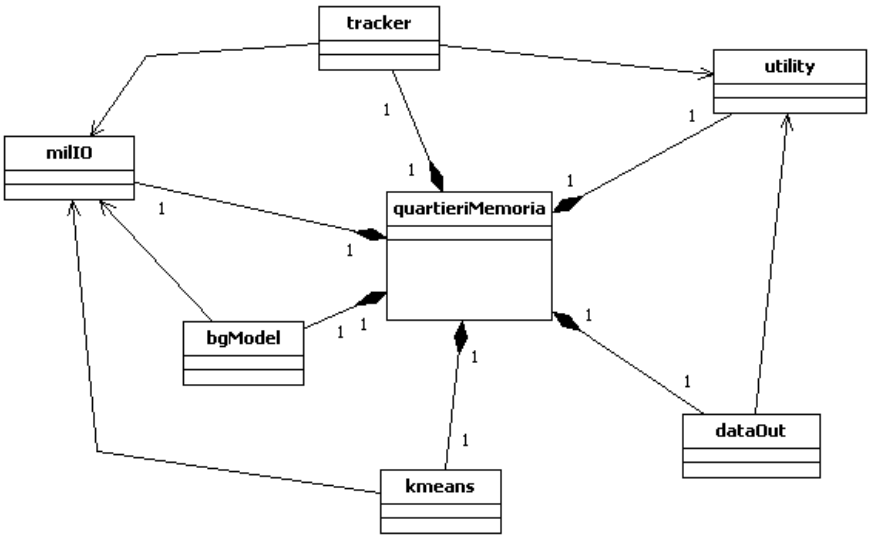

Figura 36 – Class Diagram semplificato del modulo di tracking

Vediamo ora una breve descrizione delle caratteristiche di ogni classe.

3.1.4.1 - quartieriMemoria

Questa è la classe principale del modulo, al suo interno vengono aggregate tutte le altre classi, ed è qui che viene gestito il loop principale composto dai seguenti passi:

- setup dell'area sensibile intorno al guestbook
- acquisizione dei frame
- sottrazione dello sfondo
- clusterizzazione
- controllo dell'area sensibile
- tracking
- output dei valori

Da notare che ciascuna delle precedenti funzionalità è fornita dalle classi secondarie come ad esempio tracking o kmeans, dai cui nomi deduciamo facilmente la tipologia di servizi implementati al loro interno. Particolarmente interessante è inoltre, la gestione del tracking che rispecchia il seguente diagramma:

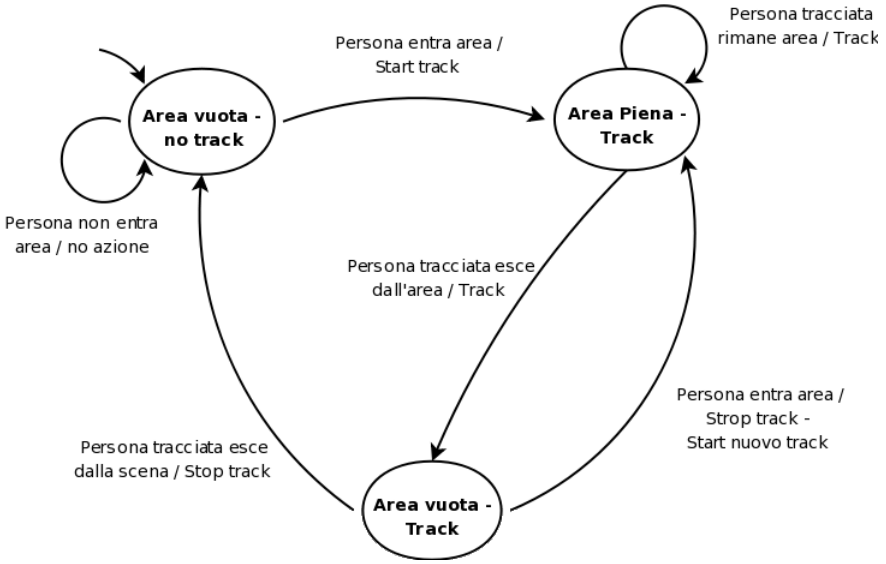

Figura 37 – Diagramma gestione del tracking

Lo stato iniziale, in cui è posto il sistema al suo avvio, è uno stato di attesa/inattività dove l'area sensibile intorno al guestbook è vuota e non si sta tracciando alcun individuo. La transizione nello stato AREA PIENA-TRACK avviene non appena un visitatore entra nell'area sensibile intento a lasciare un messaggio e l'azione che ne scaturisce è l'inizio del

tracking.

In particolare, la posizione di un visitatore all'interno dell'area sensibile è rilevata considerando la posizione di ogni cluster presente nella scena. Infatti, se il centro di massa di un cluster, che individua un visitatore, si trova in prossimità del centro dell'area sensibile, ovvero entro una certa distanza *d* da esso, allora possiamo affermare di essere in presenza di area occupata.

Quando la persona tracciata si allontana dal guestbook, il sistema si pone nello stato AREA VUOTA-TRACK, il tracking prosegue ed il guestbook si rende disponibile per una nuova interazione. A questo punto, due eventi differenti possono condizionare il comportamento del sistema. Se la persona di cui stiamo facendo il tracking esce dalla scena, il sistema si riporta nello stato iniziale AREA VUOTA-NO TRACK, mentre se un nuovo visitatore occupa l'area sensibile lo stato assunto dal sistema è nuovamente AREA PIENA-TRACK, e le conseguenze della transizione in questo stato, sono lo stop il tracking precedente, e il lancio di un nuovo tracking sul visitatore che adesso si trova all'interno dell'area sensibile.

L'assunzione fondamentale su cui si basa tale schema logico è che un solo visitatore alla volta può accedere all'area sensibile. Possiamo avere una buona certezza che ciò avvenga, considerando il fatto che il podio su cui è posizionato il guestbook è stato progettato appositamente per rispondere a questa esigenza.

Tornando alla classe quartieriMemoria, va detto che oltre al ciclo principale, abbiamo al suo interno l'implementazione di innumerevoli metodi predisposti al settaggio del sistema, come ad esempio metodi per impostare la dimensione dei cluster, la velocità max di spostamento di un individuo, la soglia massima per la rimozione dello sfondo, vari attributi necessari per l'erode e molti altri.

3.1.4.2 - milIO

La classe milIO, dal cui nome possiamo facilmente dedurne le caratteristiche, fornisce tutte le funzionalità legate all'input/output del sistema. Risiedono qui infatti i metodi per la gestione della telecamera, come ad esempio quelli relativi all'acquisizione dei frame o quelli necessari ad ottenere informazioni sul settaggio della telecamera stessa, come

dimensioni, profondità di colore e numero di canali delle immagini, frame rate ed altri ancora. milIO racchiude inoltre anche metodi per la lettura/scrittura su file, tutti gestiti tramite l'utilizzo delle librerie MIL 8.0 lite fornite con il frame grubber Matrox Morphis (come descritto in precedenza).

3.1.4.3 - bgModel

La classe bgModel è predisposta alla creazione e gestione del modello dello sfondo, è qui infatti che ci si occupa dell'acquisizione di *N* frame, immagazzinati in un accumulatore, e del calcolo della loro media. Il modello del background così costruito, è mantenuto costantemente aggiornato tramite il metodo *updateModel()* che provvederà alla somma dei nuovi frame al modello stesso, in accordo alla funzione:

$$acc(x,y)=(1-\alpha)\cdot acc(x,y)+\alpha\cdot image(x,y) \qquad if\ mask(x,y)=0$$

dove α regola la velocità di aggiornamento, ovvero quanto velocemente l'accumulatore dimentica i fotogrammi precedenti e dove la condizione $if\ mask(x,y)=0$ consente l'aggiornamento dei soli pixel appartenenti allo sfondo e non ad oggetti o visitatori presenti nella scena in quell'istante.

I frame utilizzati all'interno della classe, sono ottenuti utilizzando i metodi forniti da milIO, mentre il modello dello sfondo, è utilizzato per il calcolo della rimozione del background nella classe principale quartieriMemoria.

Figure 38, 39, 40 – Rimozione del background

3.1.4.4 - kmeans

In questa classe risiede l'implementazione dell'algoritmo k-mean e di tutti i metodi relativi alle problematiche di gestione della clusterizzazione. Vengono qui infatti individuati i cluster presenti nella scena grazie ad una chiamata al metodo *applyKmeans()* e forniti, per ognuno di essi, il centro di massa ed il bounding box medio, approssimato dall'ellisse media, in accordo alle seguenti formule:

- centro di massa

$$C_i(x,y)=\left(\frac{1}{n_i}\cdot\sum_{k=1}^{n_i} p_{i,k}(x)\ ,\ \frac{1}{n_i}\cdot\sum_{k=1}^{n_i} p_{i,k}(y)\right)$$

  dove $C_i(x,y)$ è il centro del cluster i-esimo, n_i è il numero di pixel che compongono l'i-esimo cluster e $p_{i,k}(x)$, $p_{i,k}(y)$ indicano rispettivamente l'ascissa e l'ordinata del k-esimo pixel appartenente all'i-esimo cluster.

- bounding box

$$a_i=2\cdot\sqrt{\frac{1}{n_i}\cdot\sum_{k=1}^{n_i}\left(p_{i,k}(x)-C_i(x)\right)^2}\quad\text{e}\quad A_i=2\cdot\sqrt{\frac{1}{n_i}\cdot\sum_{k=1}^{n_i}\left(p_{i,k}(y)-C_i(y)\right)^2}$$

  dove a_i e A_i sono rispettivamente l'asse minore e maggiore dell'ellisse dell'i-esimo cluster ottenuti raddoppiando la deviazione standard dei pixel appartenenti all'i-esimo cluster.

Tramite l'utilizzo dei bounding box e delle loro dimensioni massime e minime settate inizialmente in base alle caratteristiche della telecamera, vengono calcolati, per ogni frame, il numero di cluster in cui suddividere i pixel in primo piano ottenuti dalle rimozione dello sfondo.

In particolare vengono controllate le intersezioni tra i box: se abbiamo una sovrapposizione, allora probabilmente stiamo partizionando i pixel in un numero troppo

grande di cluster, viceversa, non avendo intersezioni, ma avendo dimensioni dei box troppo grandi, ci troviamo allora in presenza di una partizione composta da troppo pochi cluster e quindi avremo bisogno di incrementare la loro quantità. L'intersezione è calcolata approssimando i box con delle circonferenze di raggio r_1 ed r_2 pari alle dimensioni maggiori dei due box e secondo la regola $d < r_1 + r_2$ dove $d = \sqrt{|c_1 - c_2^2|}$ è la distanza euclidea tra i due centri c_1 e c_2 .

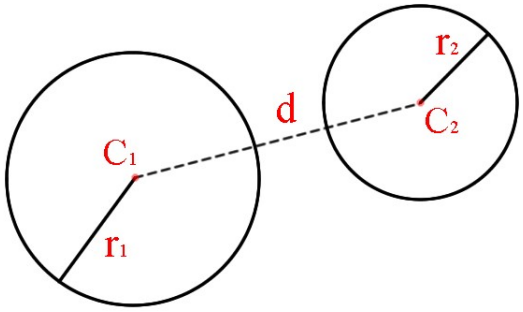

Figura 41 – Intersezione di due circonferenze approssimanti i bounding box

La clusterizzazione è così utilizzata per individuare i visitatori che entrano nell'area sensibile posta intorno al guestbook.

3.1.4.5 - tracker

Questa è la classe predisposta alla gestione del tracking e necessita di dati relativi al cluster da tracciare, ovvero centro di massa, bounding box ed informazioni sulle tonalità dei pixel che compongono il cluster stesso, ottenute dall'analisi dei fotogrammi.

Interessante è la fase di inizializzazione dell'algoritmo *Camshift* che avviene calcolando l'istogramma dei pixel che compongono l'individuo da tracciare, ottenuti tramite la clusterizzazione, e la sua *back-projection* nell'intera immagine, in modo da ottenere una maschera di pixel che mostra le zone della scena aventi la più alta probabilità di contenere i pixel che compongono l'istogramma. La back-projection ed il bounding box, vengono utilizzati dall'algoritmo *Camshift* che provvederà, per ogni frame, ad individuare all'interno

della scena la nuova posizione del centro di massa del cluster tracciato, aggiornando automaticamente il bounding box intorno ad esso. Per motivi di robustezza, ogni *n* frame, con *n* fissato dall'utente e nel nostro caso posto a 100, si provvede nuovamente al calcolo dell'istogramma e della back-projection, dato che i pixel che compongono l'individuo potrebbero essere soggetti a cambiamenti di luminosità.

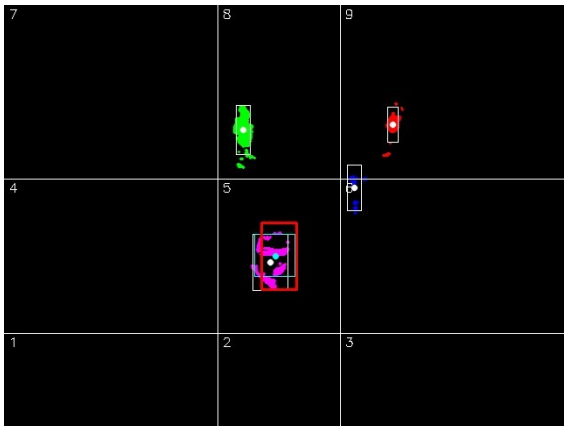

Figure 42, 43 – Clustering e Tracking

3.1.4.6 - dataOut

Questa classe è stata sviluppata per trattare il problema della pubblicazione dei dati in modo da renderli disponibili alle altre componenti del sistema. Abbiamo qui dei metodi per il calcolo della velocità di movimento, della direzione e della posizione assunta dall'individuo tracciato all'interno della scena. Altri metodi provvedono all'effettiva pubblicazione dei dati su Kolo, utilizzando le API di *Kolo JNI* (*Java Native Interface*) ma anche la pubblicazione tramite lo *Spread Toolkit*.

3.1.4.7 - utility

La classe utility è essenzialmente una collezione di metodi di utilità, come ad esempio quelli relativi al calcolo della distanza tra due punti, il test di intersezione di due box, il calcolo della prossimità di due cluster ed altri.

3.1.5 - Input/output del modulo di tracking

Consideriamo ora i dati in ingresso/uscita del modulo di tracking, cercando, dalla loro analisi, di evidenziare gli scopi per i quali tale sistema è stato sviluppato. I dati di input output sono riassunti nella seguente tabella:

	Input	Output
Init	Fotogrammi per modello BG Area sensibile Parametri vari	—
No Track	Fotogrammi	Numero persone in scena Settore più popolato
Track	Fotogrammi	Numero persone in scena Settore più popolato Posizione Velocità Direzione

Tabella 2 – Input/Output modulo di tracking

3.1.5.1 - Input

L'input principale del sistema, è costituito senza alcun dubbio, dai fotogrammi provenienti dalla telecamera. Qualsiasi immagine (fotogramma) è trattata come *IplImage*, in maniera omogenea alle librerie OpenCV utilizzate. Tramite questo tipo di rappresentazione l'immagine è vista come un array monodimensionale di pixel e le righe che compongono l'immagine sono individuabili considerando il cosiddetto “passo”, ovvero la lunghezza in pixel di ogni riga.

In generale le caratteristiche più interessanti di un fotogramma sono le sue dimensioni, la profondità o *depth* che indica la quantità di informazioni presenti in ogni pixel, ed il numero di canali. Quest'ultima informazione ad esempio è determinante nel differenziare i fotogrammi a colori da quelli in tonalità di grigio, infatti per i primi avremo 3 canali, uno per ogni canale-colore (RGB), mentre per i secondi ne sarà sufficiente solamente uno.

I fotogrammi ottenuti dalla nostra telecamera, come già detto in precedenza hanno dimensioni pari a 640 pixel in larghezza e 480 in altezza, 8 bit di informazione per ogni pixel ed un solo canale "colore". Le nostre immagini saranno quindi in scala di grigio a 256 tonalità.

Da notare inoltre che il pixel (0,0) indica l'angolo in alto a sinistra, mentre il pixel (639, 479) indica quello in basso a destra, come evidenziato nella figura 44.

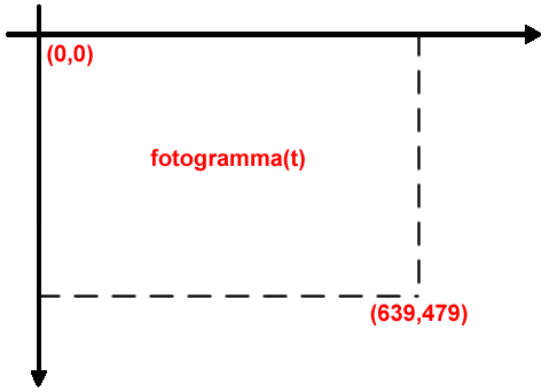

Figura 44 – Struttura immagine IplImage

Altro elemento in input essenziale per il corretto funzionamento del sistema è quello relativo all'area sensibile posta intorno al guestbook presente al centro della scena, che se attraversata da un visitatore intento a lasciare una firma, scatenerà l'evento tracking. La posizione di tale area, varia in accordo a quella del guestbook ed è definita in fase di setup dell'installazione utilizzando una piccola utility visuale realizzata ad hoc. Il modulo di computer vision, necessita di questa informazione per conoscere quando e dove far partire il tracking di un individuo, e per costruire una griglia che partiziona la scena in 9 settori. Questa griglia, sarà sfruttata per ottenere alcune informazioni, come la direzione assunta dall'individuo tracciato e ed il settore della scena contenente il maggior numero di persone.

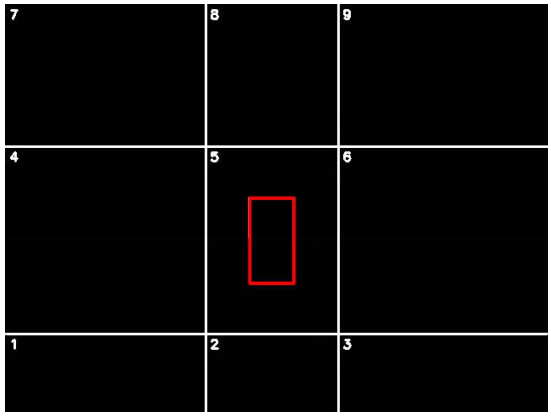

Figure 45, 46 – Definizione area sensibile e griglia

Ricordiamo che altri input sono quelli relativi al settaggio generale del sistema, come ad esempio alcuni attributi per l'operatore erode, o velocità massima di un visitatore ed altri. Tra questi abbiamo una serie di informazioni cruciali per la clusterizzazione, come la dimensione massima e minima dei cluster che considerando tutte le informazioni a nostra disposizione sullo stato della telecamera, quali l'altezza dal suolo (10m), l'angolo di inclinazione (60°), la focale della lente e le dimensioni dell'area di copertura (16m*14m) abbiamo posto a 80-90 pixel in altezza e 55-65 pixel in larghezza, stimando anche che, in media, ogni persona sarà composta almeno da 1000 pixel.

3.1.5.2 - Output

I dati prodotti in output dal sistema sono:

- numero di persone presenti sulla scena
- settore più popolato
- posizione visitatore tracciato
- velocità di spostamento del visitatore tracciato
- direzione del visitatore tracciato

I primi due risultati sono, come si può intuire facilmente, disponibili ad ogni frame, e

sono diretta conseguenza del processo di clusterizzazione. Il numero di persone presenti nella scena, infatti coinciderà con il numero di cluster individuati dall'algoritmo k-mean. La seconda informazione inoltre, è facilmente ottenibile calcolando il centro di massa di ogni cluster ed individuando la sua posizione all'interno di uno dei nove settori in cui abbiamo suddiviso l'immagine. E' ovvio che il settore contenente il maggior numero di di cluster è l'informazione che stiamo cercando. Gli altri dati sono invece dipendenti dal tracking di un individuo e sono quindi disponibili solamente durante questa fase.

La posizione del visitatore tracciato è fornita direttamente dall'algoritmo Camshift. Assimiliamo tale posizione al centro di massa dell'individuo, ottenuto, come già detto, calcolando la media delle coordinate di tutti i pixel che compongono il relativo cluster.
Per semplicità abbiamo assunto di non avere alcuna corrispondenza tra mondo reale e campo visivo della telecamera, non abbiamo cioè provveduto alla cosiddetta "fase di calibrazione", quindi la posizione da noi ottenuta è quella relativa allo spazio immagine ed oscillerà quindi nel range (0,639) per le ascisse e (0,479) per le ordinate.

Un dato molto interessante è senza dubbio quello relativo alla velocità degli spostamenti dell'individuo tracciato. Confrontando la distanza tra due posizioni successive, p_1 e p_2 percorsa in un lasso di tempo *t*, otteniamo una misura di velocità espressa in pixel/sec. Per avere un dato più leggibile, tale velocità è quantizzata in una scala di 4 valori:

- 0 = nessun movimento
- 1 = movimento lento
- 2 = movimento medio
- 3 = corsa

La quantizzazione avviene tramite una funzione:

$$q(x): \Re \rightarrow \{0,1,2,3\} \quad \text{con} \quad q(x) = int\left(\frac{4 \cdot x}{VMAX}\right)$$

dove *VMAX* è la velocità massima raggiungibile dal visitatore. Sperimentalmente abbiamo impostato il parametro *VMAX* a 100 pixel/sec.

L'ultima informazione prodotta dal modulo di computer vision, è come già accennato in precedenza, la direzione di movimento del visitatore. Avremo potuto ottenere questa informazione calcolando il vettore degli spostamenti, ma per semplicità abbiamo preferito sfruttare il partizionamento in nove settori della scena ed indicare in quale di questi settori si trovi l'utente. In generale abbiamo effettuato questa scelta, sapendo che la griglia della partizione è molto stretta intorno all'area sensibile, così non appena l'utente lascerà la zona del guestbook, avremo subito una misura macroscopica della direzione del suo movimento: destra, sinistra e così via.

Ricordiamo che tutti i dati sono prelevati ogni 30 video frame, ovvero circa ogni secondo e costantemente resi disponibili agli altri moduli del sistema, utilizzando i servizi forniti da *Kolo* e *Spread*.

3.1.6 - Kolo

Kolo è un framework basato su Java, progettato per il controllo di collezioni di sensori e di periferiche distribuite. E' stato sviluppato dall'Hypermedia Studio di UCLA con lo scopo di supportare artisti nella creazione di intrattenimento a tema, esibizioni dal vivo e media art utilizzando elementi multimediali distribuiti. Kolo permette agli sviluppatori ed agli autori di applicazioni di creare una struttura di controllo semplice e consistente per le loro applicazioni distribuite e consente di accedere a periferiche di input/output tramite semplici API senza curarsi della loro effettiva collocazione.

Kolo provvede al trasporto dati affidabile e distribuito su UDP/IP utilizzando i servizi offerti dallo Spread Toolkit ed è inoltre costruito su Java per sfruttarne le caratteristiche di portabilità.

Essendo progettato per incorporare driver di periferiche, Kolo dispone anche di una interfaccia nativa per C/C++ chiamata *Java Native Interface* (JNI) attraverso la quale è possibile incorporare molte entità tra cui anche software come *Macromedia Director* e *Cycling '74 Max/MSP*.

Kolo è costruito su poche astrazioni di base, gli elementi principali sono i nodi (*knob*), i valori (*value*), le sottoscrizioni (*subscription*), i gruppi (*group*), le relazioni (*relation*) e gli arbitri (*arbitrator*), di cui daremo una breve descrizione nei paragrafi sottostanti.

3.1.6.1 - Nodi (*Knob*)

Il *kolo network object* (*knob*) è il blocco di base su cui è costruito kolo e può essere instanziato utilizzando le API in Java. I knob sono organizzati in alberi: ogni knob può avere al massimo un genitore ed un numero arbitrario di figli. Ciascun knob ha un nome che non può essere condiviso con suoi fratelli, così una lista ordinata di nomi di ascendenti, un percorso, identifica univocamente un knob. Sebbene non richiesto dal framework, generalmente i knob vengono organizzati in un unico albero con un nodo radice chiamato “root”. I knob sono accessibili da qualsiasi processo nella rete di kolo come se fossero locali e possono essere anche creati su un processo remoto di kolo.

I knob possono avere valori che possono rappresentare letture di sensori fisici, stati

interni o qualsiasi quantità astratta. I knob possono essere scritti e tali cambiamenti possono, a volte, provocare effetti collaterali, come ad esempio il controllo di periferiche o attuatori fisici.

3.1.6.2 - Valori (*Value*)

Una difficoltà nell'interconnessione di dati eterogenei in una rete, è data dall'accordo del tipo di dati. Questo ha portato allo sviluppo di un singolo tipo di valore polimorfico per kolo. Internamente il valore è un long, un double, una stringa, un booleano, una lista o è indefinito. Quando un valore è letto dalle API java, lo sviluppatore deve richiedere tale valore in un tipo primitivo ed il framework emette valori del tipo richiesto in maniera trasparente.

3.1.6.3 - Sottoscrizioni (*Subscription*)

Le sottoscrizioni sono l'unico modo con cui è possibile settare i valori dei knob in kolo. I knob ricevono sempre i loro valori da una o più sottoscrizioni ognuna delle quali può essere un altro knob (e così variabile nel tempo), oppure una costante. Un knob può sottoscrivere valori di altri knob senza curarsi della loro posizione fisica nella rete. Le sottoscrizioni possono essere basate sul tempo (periodiche), sui cambiamenti oppure su entrambi. Una sottoscrizione ibrida è caratterizzata da un periodo minimo T_{min} e da un periodo massimo T_{max} ed un massimo valore di cambiamento *d*. Se t' è il tempo trascorso da quando è occorso l'ultimo aggiornamento dei dati e d' è il valore del nodo dopo quest'ultimo aggiornamento, allora un nuovo valore è settato quando $T_{min} < t' < T_{max}$ e $d' \geqslant d$ o quando $t' = T_{max}$.

Queste sottoscrizioni ibride, sono ottime per i sensori che in generale hanno episodi di rapidi cambiamenti, seguiti da lunghi periodo di inattività. Notiamo che sia le sottoscrizioni periodiche che quelle basate sul cambiamento, sono un caso particolare di quelle ibride, le prime hanno infatti $T_{min} = T_{max}$ e le seconde invece $T_{min} = 0$, $T_{max} = \infty$ e *d=0* .Il comportamento della classe base knob (*simpleKnob*) nelle API di kolo, è quella di determinare il suo valore dalla sua ultima sottoscrizione ricevuta. Gli altri knob, potrebbero

settare il loro valore aggregando più di una sottoscrizione in arrivo, o arbitrando tra sottoscrizioni in competizione utilizzando un *arbitrator*.

3.1.6.4 - Gruppi (*Group*)

Gli sviluppatori di applicazioni aventi il controllo di elementi ambientali distribuiti, desiderano spesso indirizzare collezioni di periferiche all'unisono (ad esempio banchi di luce in locazioni fisiche differenti ma focalizzate su di una stessa area). Un gruppo di kolo è un knob che mantiene una lista che aggrega i valori di tutti i suoi membri. Qualsiasi operazione che può essere svolta su di un knob, può essere eseguita su un gruppo, ed il gruppo applica l'operazione a tutti i suoi membri.

3.1.6.5 - Relazioni (*Relationship*)

Una relazione è un knob, il cui valore è definito come una funzione dei suoi sottoscrittori. Ad esempio l'attributo di intensità di una luce può essere reso uguale al minimo delle distanze di due attori dal pubblico, creando una relazione che ritorni il minimo delle sue sottoscrizioni e che sottoscriva la luce a quella relazione. Le relazioni forniscono un mezzo molto potente agli artisti per le specifiche di controlli dinamici sui media, basandosi sull'input in tempo reale di sensori.

3.1.6.6 - Arbitri (*Arbitrator*)

In ambienti complessi relazioni multiple potrebbero competere nel controllo di un singolo attributo. Gli “arbitri” sono relazioni privilegiate che esaminano sottoscrizioni in competizione ed arrivano al singolo valore per un knob. Ad esempio, la relazione *Min* dell'esempio precedente, può essere utilizzata come arbitro. Nelle relazioni sottoscriviamo la luce di scena la knob *Min*, se si sottoscrive poi la luce di scena alla posizione di un terzo attore la luce non sarà più influenzata da *Min*. Se vogliamo assicurare che l'intensità è sempre determinata dal minimo delle sue sottoscrizioni, dovremo assegnare *Min* come arbitro. L'utilizzo di arbitri, permette in maniera consistente e ben definita di risolvere conflitti tra relazioni. Il loro utilizzo libera l'autore da compiti, spesso complicati, di

risoluzione dei conflitti.

3.1.6.7 - Implementazione

Una rete punto-punto di *Knob Manager* gestisce le funzionalità della rete di Kolo. Un knob manager agisce come un demone per i processi java e supporta la creazione, il referenziamento, e la distruzione di knob, relazioni e gruppi, la creazione e terminazione di sottoscrizioni e l'assegnamento di arbitri. Ogni processo kolo deve avere al massimo un knob manager e all'interno di una rete kolo, ogni knob manager deve avere un nome univoco.

Quando un knob è creato, la sua presenza è resa nota a tutti i knob manager, che sono collettivamente responsabili del mantenimento di uno spazio dei nomi consistente. I knob manager comunicano tra di loro attraverso un bus, supportando sia messaggi *unicast* che *broadcast*, controllando il traffico di rete aggregando le sottoscrizioni di dati quando possibile e permettendo agli sviluppatori di lavorare in remoto gestendo i knob come se fossero oggetti locali.

Grazie alla sua bassa latenza ed alle proprietà multicast, il bus utilizza UDP per tutte le comunicazioni di rete. Il bus di kolo attualmente utilizza Spread come trasportatore per i messaggi unicast e multicast e trasporto UDP affidabile. Spread è una API di rete altamente ottimizzata e open source.

3.1.7 - Spread Toolkit

Spread è una raccolta di strumenti open source che fornisce un servizio di invio di messaggi ad alte prestazioni attraverso reti locali e geografiche. *Spread* è utilizzato come canale di messaggi unificato per applicazioni distribuite, e fornisce comunicazione multi cast, di gruppo e supporto punto-punto. I servizi forniti da *Spread* vanno dal semplice invio di messaggi, all'invio con garanzia di recapito. *Spread* può essere utilizzato in molte applicazioni che richiedono alta affidabilità, alte prestazioni ed una robusta comunicazione attraverso i vari nodi. *Spread* è stato progettato per incapsulare i difficili aspetti delle reti asincrone ed abilitare la costruzione di applicazioni distribuite sicure e scalabili. Il toolkit è composto da una libreria alla quale sono collegate le applicazioni utente, ed un demone eseguibile che è in esecuzione su ogni computer su cui risiede l'applicazione utente distribuita.

Conclusioni

Durante la prova generale dell'installazione “Quartieri della Memoria”, svoltasi a Los Angeles presso la scuola di Teatro Film e Televisione (TFT) di UCLA, abbiamo avuto la possibilità di testare accuratamente il sistema in tutte le sue componenti.
Per quanto riguarda il modulo di computer vision, abbiamo riscontrato una buona affidabilità nel tracking in presenza di un numero limitato di persone all'interno della scena. I visitatori che interagivano con l'installazione, firmando il guestbook, venivano infatti seguiti dal sistema di tracking senza alcun problema entro tutto il campo visivo della telecamera calcolando correttamente, per ognuno di essi, la posizione, la velocità e la direzione.

Il sistema di tracking sviluppato non è comunque esente da problemi, in particolare, la fase di clusterizzazione non da risultati apprezzabili quando la scena è occupata da molte persone che si trovano molto vicine tra loro, in questo caso infatti k-mean tende ad accorpare in un solo cluster due o più persone molto vicine tra loro. A volte inoltre, sempre k-mean, non è troppo stabile da frame a frame, infatti può accadere che due persone molto vicine siano considerate in un frame come un unico oggetto, e nel frame successivo come 2 persone per poi essere considerate come blocco unico nel frame seguente e così via.

Altre volte inoltre, durante la fase di tracking, quando due individui aventi abbigliamento molto simile, e quindi facilmente confondibile tra di loro, entravano in “contatto” camminando per un breve periodo molto vicini, abbiamo riscontrato degli scambi di persona. Un possibile rimedio a tale inconveniente è dato dall'implementazione di un filtro predittivo, come il *Kalman Filter,* che ci da una stima della posizione dell'individuo tracciato nei frame successivi in modo da poter intuire la direzione del visitatore ed agganciarsi ad essa senza commettere errori.
Un ultimo problema riscontrato durante la prova generale è stato quello relativo alle ombre, dovuto soprattutto alla mancata possibilità di controllare l'illuminazione della scena. In alcuni casi ad esempio, ombre molto lunghe di visitatori nella scena, ma anche ombre di persone poste appena al di fuori di essa, sono state trattate come dei visitatori veri e propri.

Considerando che l'installazione è stata progettata per lavorare durante le ore notturne, intervenendo sui livelli della luce all'interno della scena, eliminando cioè fonti luminose troppo intense, possiamo avere buone probabilità di eliminare, o almeno attutire notevolmente tale problema.

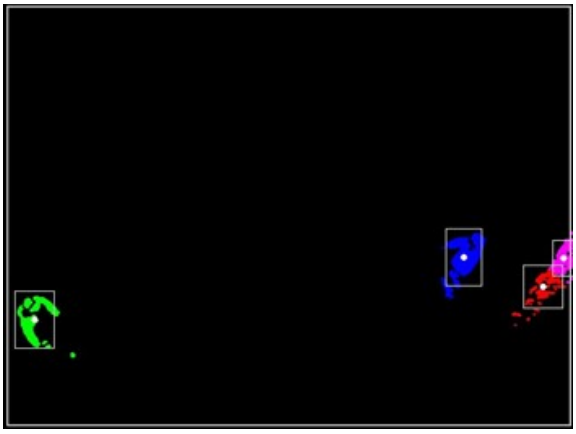

Figure 47, 48 – Alcuni problemi del modulo di Computer vision

Analizzando le prestazioni fornite dal modulo di computer vision, dobbiamo ricordare che in generale, questo tipo di applicazioni, è molto esoso dal punto di vista computazionale. Il computer utilizzato per il test era dotato di una cpu Xeon Intel dual core 3 Ghz con 1 GB di ram installata: con questa configurazione, il modulo di CV richiedeva un utilizzo non troppo elevato delle risorse di sistema, attestandosi sempre al di sotto del 40% di carico di cpu, garantendo una buona fluidità (circa 25 frame al secondo) e tracking, con relativa pubblicazione di dati da esso estrapolati, in tempo reale.

Bibliografia

[1] Perceptual human-computer interface head pose estimation from single camera - Alessandro Marianantoni

[2] Open Source Computer Vision Library (Intel OpenCV) - Reference Manual

[3] K-Means Clustering Tutorial - Kardi Teknomo

[4] Computer Vision Face Tracking For Use in a Perceptual User Interface - Gary R. Bradski

[5] Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces - John G. Allen, Richard Y. D. Xu, Jesse S. Jin

[6] Detecting Pedestrians Using Patterns of Motion and Appearance – P. Viola, M. Jones, Snow.

[7] Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade - P. Viola, M. Jones

[8] Shape-based pedestrian detection - A. Broggi et al.

[9] Shape-based pedestrian detection and tracking - D. M. Gavrila and J. Geibel

[10] Learning flexible models from image sequences - A. Baumberg and D. Hogg

[11] An Efficient Method for Contour Tracking Using Active Shape Models - A. Baumberg and D. Hogg

[12] Pfinder: Real-Time Tracking of the Human Body - Christopher Richard Wren, Ali Azarbayejani,Trevor Darrell, and Alex Paul Pentland

[13] Articulated body motion capture by annealed particle filtering - J. Deutscher et al.

[14] Interactive marker-less tracking of human limbs – Srinivasa G. Rao, larry F. Hodges

[15] Condensation, conditional density propagation for visual tracking – M. Iard and A.

Blake

[16] Optical flow-based real-time object tracking using non-prior training active feature model - Jeongho Shina, Sangjin Kima, Sangkyu Kangb, Seong-Won Leec, Joonki Paika, Besma Abidid, Mongi Abidid

[17] Real-Time Motion Estimation and Visualization on Graphics Cards - Robert Strzodka and Christoph Garbe

[18] Image Registration by a Regularized Gradient Flow A Streaming Implementation in DX9 Graphics Hardware - R. Strzodka, Bonn, M. Droske and M. Rumpf

[19] Kolo and Nebesko: A Distributed Media Control Framework for the Arts - Eitan Mendelowitz, Jeff Burke

[20] A user guide to Spread version 0.11 - Johnathan R. Stanton

[21] Il sublime tecnologico - Mediamente: Intervista a Mario Costa

[22] Ancora sull'estetica della comunicazione - Mario Costa

[23] Tecno-poetiche: la creazione con le nuove tecnologie - Julio Plaza

[24] La fine del lineare e l'arte interattiva - Fred Forest