Indice

Introduzione

Oggi più che mai sembra valere il detto "*Una figura vale mille parole*". Infatti, non solo l'afflusso di immagini attraverso televisione, Internet, e persino telefoni cellulari ci investe con un'intensità impensabile anche solo dieci anni fa, ma anche perché sono oggi disponibili tecniche più o meno sofisticate, ma sempre molto d'effetto, per utilizzare le immagini in un modo assai più ricco che non con la semplice esposizione.

Nel redigere questa tesi mi sono proposta di trovare e analizzare tecniche innovative per l'utilizzo di informazioni visive, e in particolare di sviscerarne gli aspetti matematici. La nostra disciplina, infatti, `Êe risultata sempre presente con maggiore o minore impatto: da spazi vettoriali di dimensione finita e funzionali di tipo energia, fino a trasformazioni geometriche.

Un primo capitolo è dedicato a due importanti campi "tradizionali" di interazione fra matematica e immagini: la visione e la grafica al calcolatore. Con il secondo capitolo, dedicato al ridimensionamento intelligente di immagini, si entra nel cuore della tesi. Segue un capitolo in cui si illustra brevemente l'animazione di fotografie. Il capitolo quattro tratta diffusamente la manipolazione "manuale" di sagome. L'ultimo capitolo utilizza il morphing su spazi vettoriali infinito dimensionali per il riconoscimento di volti.

Capitolo 1

Immagini e calcolatore: impieghi più usuali delle immagini sul calcolatore

1.1 Elaborazione di immagini: la visione artificiale

Sempre più spesso potenti calcolatori risolvono problemi un tempo riservati all'uomo, a questo è dovuta l'attualità di tale argomento.

All'interno di questa ricerca gli studi nell'uno e nell'altro campo risultano determinanti, infatti sono le nostre crescenti conoscenze sul cervello umano a fare da guida, ma allo stesso tempo i progressi scientifici nel campo dell'intelligenza artificiale ci stanno insegnando a capire il cervello. Le differenze dei sistemi calcolatore - cervello sono molteplici:

- a livello di struttura e meccanismi, cioè di hardware, i neuroni e le cellule nervose del cervello sono molto diverse dai conduttori e semiconduttori su cui sono basati i calcolatori di oggi;

- per l'organizzazione dello hardware, infatti i numerosi collegamenti tra i neuroni e la loro distribuzione tridimensionale sono assai diversi dai pochi conduttori distribuiti bidimensionalmente che collegano i componenti di un calcolatore;

- per la trasmissione dei segnali: il cervello trasmette informazioni sia attraverso segnali di *tutto o nulla* lungo le cellule nervose, sia tramite segnali graduati, sostanze chimiche, trasporto di ioni. Tale sistema è solo in parte paragonabile agli impulsi elettrici binari (on - off) del calcolatore;

- a livello di organizzazione temporale: i calcolatori elaborano l'informazione passo per passo, serialmente, ad altissima velocità, regolati da un sincronizzatore centrale; il cervello invece analizza l'informazione lentamente, su milioni di canali contemporaneamente, senza bisogno di un sincronizzatore generale.

In che senso allora possiamo paragonare un calcolatore al cervello? A questo proposito vengono di seguito riportati dei brani dall'articolo di Tomaso Poggio dal titolo "La visione nell'uomo e nella macchina" pubblicato sulla rivista "Le Scienze" [Poggio, 1984]:

> "C'è un livello al quale paragonare i due diversi meccanismi. Se consideriamo per esempio il calcolatore e il cervello come dei sistemi che elaborano informazione, possiamo paragonare e descrivere i compiti svolti da entrambi. A questo punto abbiamo un linguaggio comune: il linguaggio dell'elaborazione dell'informazione. È importante però notare che il linguaggio con cui si descrive lo svolgimento di determinati compiti è indipendente dalla struttura "fisica", dallo hardware, che svolge tali compiti. Questa separabilità è uno degli aspetti fondamentali della scienza dell'intelligenza artificiale, i cui obiettivi sono da un lato quello di rendere più utili i calcolatori, dotandoli di capacità *intelligenti*, e dall'altro quello di capire quali siano i principi che rendono possibile l'intelligenza. Lo studio di come vediamo rappresenta uno dei campi dell'intelligenza artificiale in cui si hanno le descrizioni più precise di problemi di elaborazione dell'informazione. Una delle ragioni è naturalmente che la visione rappresenta la modalità sensoriale dominante nell'uomo.
> [...]
> Dal punto di vista dell'elaborazione dell'informazione sia da parte del cervello sia da parte di un calcolatore si tratta di un problema a molti livelli: il livello del calcolo (quali compiti computazionali deve svolgere il sistema visivo); il livello dell'algoritmo (in quale successione di stadi elementari si completa un compito); il livello dello hardware (come potrebbero eseguire l'algoritmo i neuroni o i circuiti elettronici). Dobbiamo quindi basarci su dati di natura psicofisica (come vede l'uomo) e su dati di natura neurofisiologica (cosa fanno i neuroni) per poter affrontare il problema della visione. L'aspetto più critico riguarda gli algoritmi perché per essere realizzabili debbono soddisfare sia vincoli imposti dai calcoli sia vincoli imposti dallo hardware."

A questo punto l'autore dell'articolo pone l'accento sull'analisi di una serie di algoritmi utili ad estrarre informazione, in particolare ad evidenziare i contorni di immagini e quindi a calcolare, sulla base di questi contorni, la profondità degli oggetti nel mondo tridimensionale.

Un aspetto particolare di tale problema è rappresentato dalla visione stereoscopica o binoculare. Essa si presta ad un'indagine piuttosto dettagliata dei primissimi stadi dell'elaborazione dell'informazione visiva.

> "La visione stereoscopica è un processo ingannevolmente semplice, che il cervello svolge senza alcuno sforzo. Si vedrà però come lo sviluppo di un sistema dotato di *visione stereoscopica* si sia rivelato estremamente difficile.
> [...] La visione stereoscopica è dovuta al fatto che i nostri occhi vedono il mondo da angolazioni leggermente diverse; in altre parole gli occhi sono leggermente convergenti e i loro assi visivi si incontrano in un punto. Tale punto viene fissato dagli occhi e l'immagine del punto cade nel così detto centro della visione di ciascuna retina. Qualunque altro punto nel campo visivo sarà quindi proiettato su ciascuna retina a una certa distanza dal centro della visione. Generalmente questa distanza non è la stessa per ambedue gli occhi, ma la differenza di distanza varia con la profondità del punto nel campo visivo rispetto al punto fissato. La visione stereoscopica è il processo che codifica la tridimensionalità dalle disparità oculari. Si tratta di un processo assai complesso che consiste di almeno quattro stadi: in un'immagine si sceglie la proiezione di un punto nello spazio; si misurano le posizioni delle due immagini; dalla differenza tra le due misurazioni si calcola la distanza del punto.
>
> Gli ultimi due stadi sono un compito di trigonometria ma i primi due sono diversi. Essi richiedono l'identificazione della proiezione dello stesso punto del mondo fisico in ciascun occhio."

Il problema della corrispondenza fra punti delle due immagini, che corrispondono allo stesso punto nella scena, è stato successivamente semplificato da un accorgimento matematico, detto "vincolo epipolare", che riduce la ricerca della corrispondenza ad un problema monodimensionale, grazie ad una particolare proiettività fra due fasci di rette. È comunque interessante proseguire la lettura dell'articolo di Poggio, che illustra gli aspetti

percettivi e - direi - quasi filosofici della visione stereoscopica, attraverso un notevole esperimento ideato da un ricercatore dei Bell Laboratories.

> "Una scoperta di Bela Julesz mostra tutta la complessità del problema. Julesz ha prodotto quelli che ha chiamato *random dot stereograms* (stereogrammi a punti casuali). Si tratta di figure costituite da punti bianchi e neri distribuiti a caso e che contengono disparità binoculari. Ci sono due copie di ogni figura. In una copia una porzione della configurazione, per esempio un quadrato, viene leggermente spostata da una parte, mentre nell'altra un'egual porzione viene spostata in direzione opposta. La porzione rimasta vuota in ciascuna configurazione viene riempita con altri punti a caso. Viste una alla volta le due copie di una figura sembrano costituite da punti uniformemente casuali. Viste invece attraverso uno stereoscopio, in modo che ciascun occhio veda una delle due copie della figura e il cervello fonda le due immagini, il risultato è sorprendente: si ha la nettissima impressione che il quadrato si stacchi dallo sfondo della figura. Evidentemente la visione stereoscopica non richiede il riconoscimento degli oggetti visti precedentemente o di forme."

La ricerca di indizi per la corrispondenza fu uno degli stimoli che condussero Poggio e David Marr a studiare uno dei primi metodi per la determinazione dei contorni:

> "Nei punti dove vi sono variazioni fisiche in una superficie, l'immagine della superficie di solito presenta nette variazioni di intensità. Queste variazioni causate dalla tessitura della superficie e da variazioni della sua distanza, forniscono indicazioni molto più sicure ed efficaci per la corrispondenza che non i valori di intensità."

È importante sottolineare l'aspetto matematico della corrispondenza tra variazioni di intensità e valori di intensità adiacenti. La derivata prima rappresenta la rapidità con cui l'intensità varia lungo un percorso attraverso la matrice dei livelli di grigio. La posizione di un valore estremo, un massimo o un minimo, nella derivata prima identifica la posizione di un contorno dell'immagine. La rapidità di cambiamento della rapidità di cambiamento viene descritta dalla stima della derivata seconda, e si ottiene considerando le differenze tra i valori adiacenti della derivata prima. Nella derivata seconda un contorno di intensità nella matrice dei livelli di grigio corrisponde ad un "passaggio attraverso lo zero".

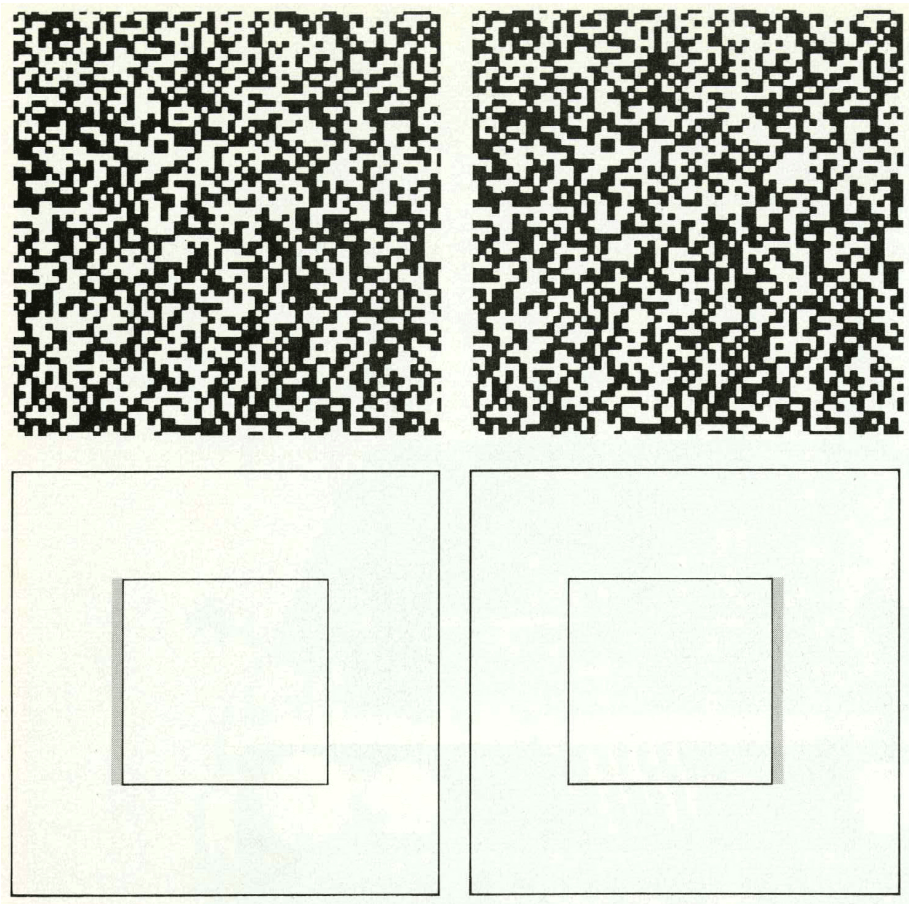


Figura 1.1: Gli stereogrammi a punti casuali inventati da Bela Julesz agli AT&T Bell Laboratories sono figure visive che non contengono altri indizi per la visione stereoscopica al di fuori delle disparità binoculari. Gli stereogrammi in sé sono una struttura casuale di punti bianchi e neri. In uno di essi un quadrato è spostato verso sinistra, nell'altro è spostato verso destra. La lacuna risultante in ciascuna immagine è riempita con altri punti casuali.

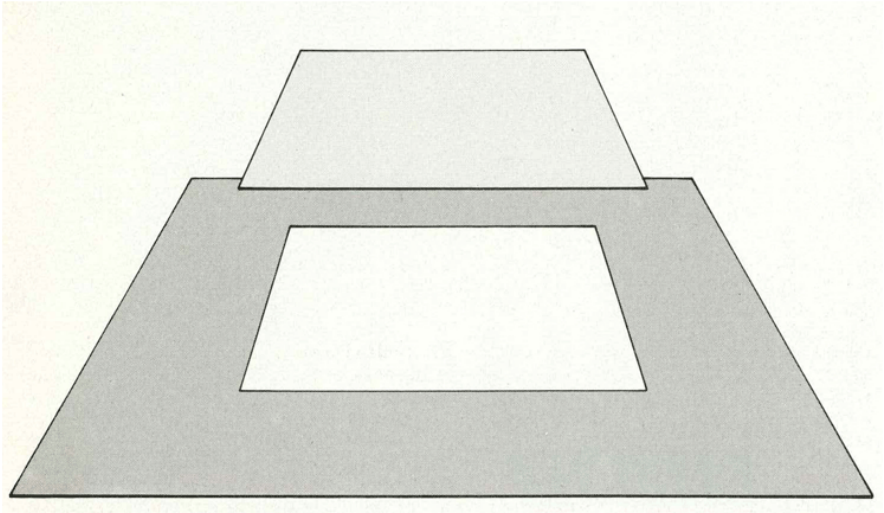


Figura 1.2: Quando gli stereogrammi a punti casuali vengono visti attraverso uno stereoscopio, in modo che ciascun occhio veda solo uno dei due stereogrammi e il cervello possa fondere le due immagini, si ha una vivida percezione di profondità. Questo fenomeno dell'immagine "fluttuante" dimostra che la visione stereoscopica non presuppone il riconoscimento di oggetti nel mondo visivo.

Nell'analisi delle variazioni di grigio ci troviamo di fronte a dei cambiamenti di intensità, ad esempio da elevata, più o meno costante, a bassa, sempre più o meno costante. Vorremmo porre un confine tra le due, ma è difficile stabilire dove sia. Procediamo per passi. Innanzi tutto andiamo a vedere come si comporta la derivata: sarà uguale a zero nei punti in cui l'intensità è costante, mentre avrà uno sbalzo nei punti in cui l'intensità cambia. Ci occorre quindi individuare bene i punti di minimo e di massimo, e per trovarli tiriamo in causa la derivata seconda: questa avrà un andamento prima fortemente negativo, in un secondo momento rapidamente positivo, poi di nuovo negativo fino a tornare a stabilizzarsi intorno allo zero.

Seguono due considerazioni: non ci interessa la presenza e la posizione degli zeri, perché per esempio dove abbiamo più o meno costante l'intensità, avremo più o meno costante a zero la derivata e più o meno costante a zero la derivata seconda. Gli zeri sono prima e dopo del cambiamento, ma non ci interessano. L'unico zero che ci interessa è quello che viene attraversato dalla derivata seconda tra il picco di massimo e quello di minimo, e lo rileviamo perché abbiamo un netto passaggio da negativo a positivo. Ovviamente dobbiamo fare i conti con il rumore, per cui i passaggi attraverso gli zeri

saranno continui, poiché la funzione non sarà mai realmente costante, ma in questo caso abbiamo un passaggio brusco, ciò significa che nel giro di pochi pixel si passa da qualcosa di molto negativo a qualcosa di molto positivo. In mezzo c'è quello che ci interessa.

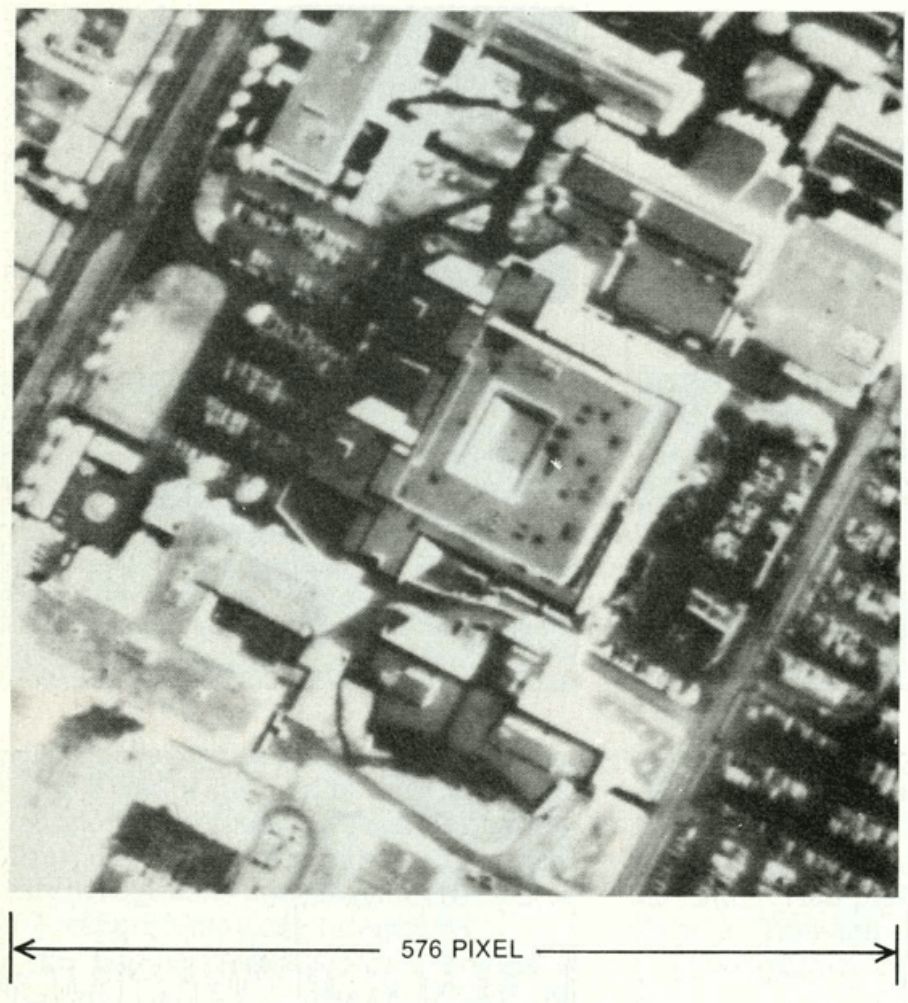
Oltretutto dobiamo tenere in considerazione la discretizzazione: può darsi benissimo che lo zero in realtà, tra i due picchi di massimo e minimo, non compaia, perché quando tutto viene discretizzato può capitare che in questo attraversamento si passi da valori negativi crescenti, direttamente a valori positivi, senza rilevare in particolare il valore zero. Quindi è necessario un esame locale e non puntuale. Quel che cerchiamo quindi è non la presenza di uno zero ma l'attraversamento dello zero perché ci interessa un cambiamento repentino da negativo a positivo o viceversa.

Nell'analisi delle immagini attraverso lo studio delle derivate riscontriamo però alcuni problemi: le variazioni di intensità in un'immagine reale raramente sono chiare e distinte da un valore di intensità ad un altro. Inoltre spesso cambiamenti diversi, lenti e veloci, si intrecciano a diverse scale spaziali. Infine le variazioni di intensità sono spesso mascherate da fluttuazioni casuali (l'analogo visivo del rumore) che si infiltrano a stadi diversi quando l'immagine formata dall'ottica dell'occhio viene trasformata in una matrice di misurazioni di intensità.

Per risolvere tali problemi l'immagine deve essere in primo luogo *regolarizzata* filtrando i valori di intensità contigui poi, solo in seguito, si possono ricavare le derivate prima e seconda. Uno dei metodi più semplici per la regolarizzazione e la differenziazione riunisce i due processi in un'unica operazione: l'immagine viene sottoposta a una convoluzione con filtro che incorpora una particolare funzione centro-periferia, il laplaciano di una gaussiana. La gaussiana specifica l'importanza che si deve assegnare alla periferia di ciascun pixel quando l'immagine viene regolarizzata. Con il crescere della distanza diminuisce l'importanza. Il laplaciano è una derivata seconda che assegna lo stesso peso a tutti i percorsi che si dipartono da un punto. Il laplaciano di una gaussiana trasforma la distribuzione a campana in una distribuzione simile ad un sombrero messicano. La campana si restringe e ai lati si sviluppa una falda circolare negativa. A questo punto l'articolo di Tomaso Poggio ci spiega, in termini non tecnici, cosa significa sottoporre un'indagine al laplaciano di una gaussiana:

> "Sottoporre un'immagine a una convoluzione con un filtro che incorpora il laplaciano di una gaussiana equivale a sostituire a ciascun pixel nell'immagine

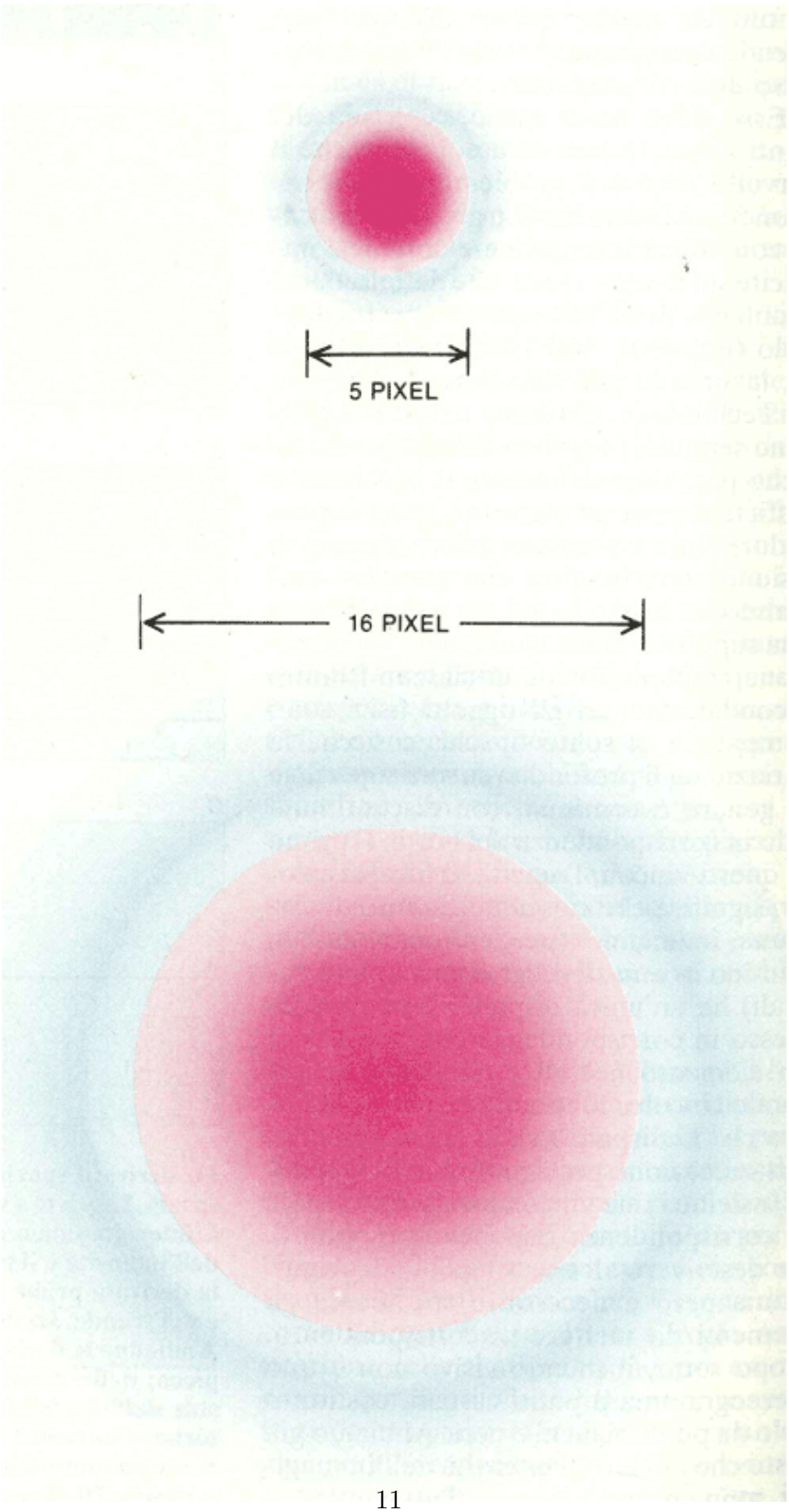
una media ponderata dei pixel vicini, dove i pesi sono forniti da laplaciano di una gaussiana. Il filtro viene quindi applicato a ciascun pixel e assegna il massimo peso positivo a quel pixel e pesi positivi decrescenti ai pixel vicini. Poi viene un anello ai cui pixel si attribuiscono pesi negativi. Il risultato del filtraggio complessivo è una matrice di numeri positivi e negativi: una sorta di derivata seconda dell'intensità dell'immagine alla scala del filtro. In questa matrice filtrata i passaggi attraverso lo zero corrispondono a punti dell'immagine originale in cui l'intensità varia più rapidamente.



Sorprendentemente sembra che nel cervello umano sia presente la maggior parte dello hardware necessario per eseguire un simile filtraggio. Negli anni Cinquanta e Sessanta si sono accumulati dati che fanno pensare che la retina

> compia qualcosa di simile al filtraggio centro-periferia. Il segnale di uscita di ogni retina è trasportato al resto del cervello da circa un milione di fibre nervose, ciascuna delle quali è l'assone di un neurone, chiamato cellula gangliare della retina. La cellula ottiene il suo ingresso (attraverso neuroni intermedi) da un gruppo di fotorecettori, che costituiscono un *campo recettivo*. Quello che i dati sperimentali suggeriscono è che per certe cellule gangliari il campo recettivo abbia una organizzazione centro-periferia che si avvicina molto al laplaciano di una gaussiana. La luminosità al centro del campo recettivo eccita la cellula gangliare; la luminosità di un anello periferico la inibisce. In breve il campo recettivo ha un centro "on" e una periferia "off", proprio come il sombrero.
>
> Altre cellule gangliari hanno proprietà opposte: un centro "off" e una periferia "on" . Se gli assoni potessero trasmettere come segnali numeri negativi,queste cellule sarebbero ridondanti: esse riferiscono semplicemente la negazione di quello che riferiscono le cellule del centro "on".
> I neuroni però non possono trasmettere un'attività negativa: quelli che trasmettono un'attività "tutto o nulla" sono o attivi o a riposo. Il cervello ha quindi bisogno di opposti neuronali. I valori positivi di un'immagine soggetta a filtraggio centro-periferia potrebbero essere rappresentati dall'attività di cellule a centro "on", i valori negativi potrebbero essere rappresentati dall'attività di cellule a centro "off". [...] Ecco quindi una congettura che collega una teoria computazionale della visione con lo hardware del cervello che entra in gioco nella visione biologica."

Abbiamo già accennato al problema della scala spaziale. Tale problema nasce dal fatto che in un'immagine vi sono variazioni di intensità sia fini sia grossolane; ciò che vogliamo è trovare un modo per rappresentarle tutte: la soluzione indicata dall'autore dell'articolo sta nell'uso di filtri centro-periferia di dimensioni diverse. I filtri *vedono* solo variazioni di intensità da pixel a pixel che non sono troppo rapide né troppo lente. Allora si può filtrare l'immagine con un filtro centro-periferia di una particolare dimensione, poi si cercano i passaggi attraverso lo zero nell'immagine filtrata. Così facendo i filtri grandi scoprono contorni morbidi o sfumati, nonché variazioni complessive di illuminazione; i filtri piccoli identificano particolari più dettagliati. I contorni netti verrebbero identificati da tutte le


5 PIXEL
16 PIXEL

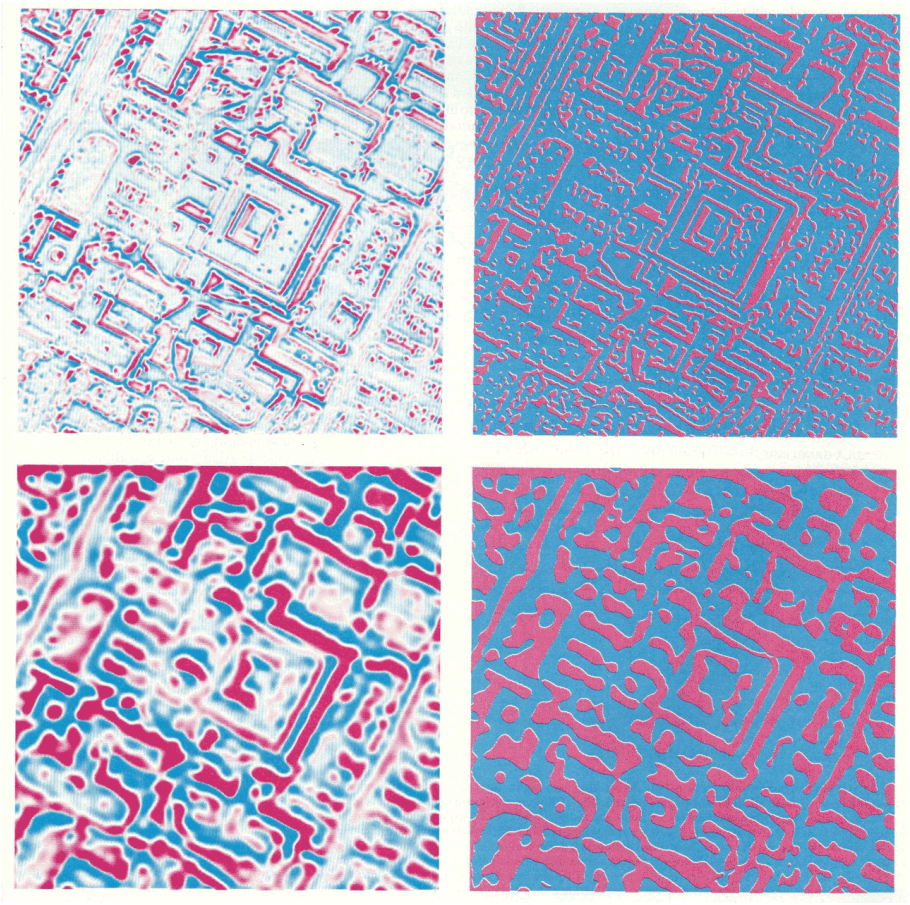


Figura 1.3: Il filtraggio centro-periferia di un'immagine serve sia a regolarizzarla, sia a calcolare la derivata seconda rispetto allo spazio. Viene fatto lo schema di due filtri della prima immagine.Ogni misurazione di intensità dell'immagine viene sostituita da una media ponderata delle misurazioni circostanti. Le misurazioni vicine contribuiscono alla media con pesi positivi; pertanto il centro del filtro é eccitatorio (in rosso). L'anello in cui le misurazioni contribuiscono con pesi negativi alla periferia del filtro é inibitorio (in blu). La terza parte dell'illustrazione mostra le mappe prodotte dai filtri. Non si tratta più di matrici di livelli di grigio: le mappe hanno sia valori positivi (in rosso) che negativi (in blu). Sono mappe della derivata seconda. Le transizioni da un colore all'altro sono passaggi attraverso lo zero, contrassegnano cioè le zone, nell'immagine di partenza, in cui l'intensità varia più rapidamente. Le quattro mappe finali mettono in evidenza i passaggi attraverso lo zero mostrando solo regioni positive (in rosso) e regioni negative (in blu).

scale. Analisi teoriche mostrano che caratteristiche simili a passaggi attraverso lo zero in una immagine filtrata possono essere ricche di informazioni: le mappe dei passaggi attraverso lo zero ottenute a scale diverse possono rappresentare l'immagine originale completamente, senza cioè perdita di informazione. L'indagine relativa a che cosa il modulo della visione stereoscopica metta in corrispondenza trova così una potenziale risposta: i passaggi attraverso lo zero rappresentano uno schema di codificazione che potrebbe rivelarsi ottimale ed avere una posizione di rilievo tra gli elementi che debbano essere messi in corrispondenza tra le due immagini retiniche. Riassumendo, il rilevamento e l'identificazione di variazioni di intensità in una immagine a diverse scale spaziali costituiscono un primo importante passo per la visione stereoscopica e altri processi visivi. Un modo per realizzare tale passo è filtrare l'immagine con il laplaciano di una gaussiana; i passaggi attraverso zero nella matrice filtrata corrispondono ai contorni di intensità nell'immagine. Informazioni simili sono implicite nell'attività delle cellule gangliari a centro *on* e di quelle a centro *off* della retina. Per rappresentare esplicitamente i passaggi attraverso lo zero, una classe di neuroni rilevatori di contorni nel cervello dovrebbe svolgere operazioni specifiche sull'uscita delle cellule a centro *on* e delle cellule a centro *off* che sono vicine alla retina. Nella parte conclusiva dell'articolo l'autore ci espone un algoritmo risolutivo per il problema della visione stereoscopica, algoritmo il quale ci permette di utilizzare le variazioni di intensità per la visione stereoscopica:

> "Consideriamo un algoritmo formulato da Marr e da me, che realizza l'unicità (un punto dato su una superficie fisica ha una sola collocazione, cosicché solo una corrispondenza binoculare è corretta) e la continuità (le variazioni di profondità in genere sono continue, cosicché le disparità binoculari tendono a variare con continuità). L'algoritmo ha successo nella risoluzione di stereogrammi a punti casuali di alcune immagini naturali. Viene eseguito da un calcolatore, pertanto la sua esecuzione effettiva non è altro che una successione di calcoli. Possiamo pensare tuttavia che l'algoritmo consista di una rete tridimensionale di nodi, in cui i nodi rappresentano tutte le possibili intersezioni di linee di vista che si dipartono dagli occhi nel mondo tridimensionale. Il vincolo dell'unicità si realizza imponendo che i nodi lungo una data linea di vista si inibiscano a vicenda. Al tempo stesso il vincolo della continuità si realizza imponendo che ogni nodo ecciti i suoi vicini. Nel

caso degli stereogrammi a punti casuali, la procedura è relativamente semplice: i corrispondenti dei pixel di ciascuna linea orizzontale di uno stereogramma vanno cercati solo nella corrispondente riga dell'altro stereogramma.

L'algoritmo inizia con l'assegnazione del valore 1 a tutti i nodi che rappresentano una corrispondenza binoculare fra due pixel bianchi e due pixel neri nella coppia di stereogrammi, mentre agli altri nodi viene assegnato il valore zero. Gli 1 pertanto contrassegnano tutte le corrispondenze, false e vere. Poi l'algoritmo esegue una somma algebrica per ciascun nodo. In questo calcolo i nodi vicini che hanno valore 1 contribuiscono con pesi positivi mentre i nodi che hanno valore 1 lungo le linee di vista contribuiscono con pesi negativi. Se il risultato supera un certo valore di soglia, al nodo si assegna valore 1 altrimenti il nodo è zero. Questo rappresenta una iterazione della procedura. Dopo qualche iterazione la rete raggiunge la stabilità. Il problema della visione stereoscopica è risolto."

1.2 Ricerca di immagini digitali

Trattiamo ora il problema della ricerca di immagini inserite in grandi raccolte. Un'analisi automatizzata delle immagini sembra la soluzione più auspicabile per affrontare ricerche di questo genere. Si tratterebbe di utilizzare un programma che identifichi l'oggetto indipendentemente da variazioni di colore, dimensioni, aspetto o punto di vista. Purtroppo però, le nostre conoscenze nel processo di riconoscimento delle immagini sono ancora molto lontane da questo obiettivo. I programmi per calcolatore possono analizzare il contenuto di immagini da selezionare in vari modi: trovare immagini simili a un esemplare noto; analizzare le immagini in base all'omogeneità del sostrato (regioni di colore o trama pressoché costanti); identificare "cose" proprio come facciamo noi. In qualsiasi caso, sembra che vi sia un rapporto inverso tra semplicità di attuazione e utilità.

Per quanto riguarda il primo caso il confronto tra due immagini è piuttosto immediato, ma serve solo ad individuare immagini correlate in modo superficiale. Non è molto utile per trovare oggetti, perché modifiche nella loro posizione, composizione o configurazione fanno fallire la maggior parte dei confronti con un campione.

La ricerca di immagini basata sul sostrato ha maggiori potenzialità, tale è infatti l'orientamento della maggior parte degli attuali sistemi per la ricerca di immagini. Si

veda ad esempio il *Query by Image Content* [QIBC], il sistema di database per l'immagine sviluppato da un gruppo di ricerca dell'IBM, il quale permette all'operatore di specificare le immagini richieste in termini di proprietà come il colore, la distribuzione spaziale, la trama. Sebbene tali sistemi siano interessanti, richieste circoscritte al sostrato sono fruttuose solo entro certi limiti e poiché in linea di massima gli utenti sono interessati a oggetti, né il sostrato né la somiglianza iconica potranno da soli fornire un fondamento sufficiente per evadere richieste basate sul contenuto delle immagini.

David Forsyth, Jitendra Malik e Robert Wilensky hanno costruito un sistema per la ricerca di immagini nell'ambito del *Digital Library Project* in corso presso l'Università della California a Berkeley [Digital Library Project]. Tale sistema permette la ricerca sia di un oggetto sia di un sostrato; ma trovare oggetti è difficile, perciò spesso è meglio richiedere immagini con un sostrato adatto a comporre gli oggetti desiderati. Le rappresentazioni di oggetti si ottengono attraverso composizioni del sostrato: quindi le richieste più utili relativamente al sostrato sono quelle che potrebbero servire a costruire rappresentazioni di oggetti.

Purtroppo nella ricerca di informazioni, una maggiore accuratezza comporta una minore capacità di recupero e viceversa. Nel sistema preso in considerazione è possibile avanzare richieste solo per un numero limitato di oggetti; nello specificare il sostrato, si possono fornire percentuali dei colori, nonché il numero e le dimensioni delle "macchie di colore" che dovrebbero essere presenti; è inoltre possibile specificare altre caratteristiche ricavate dal contenuto dell'immagine ed esaminare i metadati.

Il calcolatore potrebbe anche arrivare a dedurre dal sostrato quali siano gli oggetti presenti in un'immagine, ed entriamo così nell'ambito del "riconoscimento di oggetti", tema centrale della visione tramite calcolatore. Le tecniche attuali funzionano solo quando le immagini contengono pochi oggetti, con forme note nei particolari, osservabili da un numero limitato di punti di vista e separati dallo sfondo. In altre parole, il programma deve essere capace di organizzare il sostrato in oggetti. Tale problema prende il nome di "raggruppamento percettivo". Sono stati individuati numerosi fattori che potrebbero essere utilizzati per determinare le parti di un'immagine che quasi certamente traggono origine da uno stesso oggetto della scena. A livello più complesso le regioni di un'immagine bidimensionale simmetriche rispetto ad un'asse possono essere raggruppate come proiezioni di un oggetto simmetrico tridimensionale. Ciò che si vuole raggiungere

è la conversione di queste regole di buon senso in algoritmi funzionanti. Serge Belongie, Chad Carson, Hayit Greenspan e Malik hanno sviluppato un sistema che fornisce un'utile scomposizione dell'immagine in un piccolo insieme di regioni coerenti rispetto al colore e alla trama. Ogni "macchia" (o *blob*) viene descritta con attributi che ne rappresentano la posizione, la forma, il colore, la trama.Ogni blob cioè riassume le proprietà di base per la composizione dell'immagine. È significativo che una tale rappresentazione di oggetti, in termini di configurazioni spaziali di regioni di colore e trama particolari, si presti bene a tecniche di apprendimento automatico. I classificatori esistenti sono attualmente in grado di categorizzare le scene esaminando solo informazioni su trama e colore; provvisti di informazioni appropriate sulla forma delle regioni, dovrebbero arrivare a distinguere i gruppi di regioni che formano oggetti da quelli che non hanno questa proprietà. L'apprendimento fornisce una cornice naturale per affrontare il problema delle variazioni irrilevanti in una classe di oggetti, perché un classificatore può adattarsi secondo modalità che permettono di trascurare deviazioni non significative.

Riportiamo di seguito un brano tratto da un articolo di David Forsyth, Jetendra Malik e Robert Wilensky dal titolo "In cerca di immagini digitali" pubblicato su "Le Scienze" [Forsyth et al., 1997]:

> "Il modulo software per il riconoscimento di volti sviluppato da Takeo Kanade e collaboratori alla Carnegie Mellon University è un buon esempio di applicazione delle tecniche di apprendimento. Altri ricercatori hanno incontrato difficoltà nell'identificare accuratamente esempi di bocca, occhi e naso, se presi separatamente l'uno dall'altro; Kanade e colleghi hanno invece addestrato una rete neurale a segnalare la presenza simultanea, in configurazione appropriata, di tutte queste caratteristiche. Essi stanno anche studiando l'integrazione di dati visivi e acustici per analizzare spezzoni di film e videoregistrazioni. Per esempio, un calcolatore potrebbe effettuare un riconoscimento del parlato sulla traccia audio di un servizio trasmesso nel telegiornale e associare così il nome di una celebrità al volto che appare sullo schermo.
>
> Colore e trama possono facilitare l'individuazione di regioni dell'immagine.

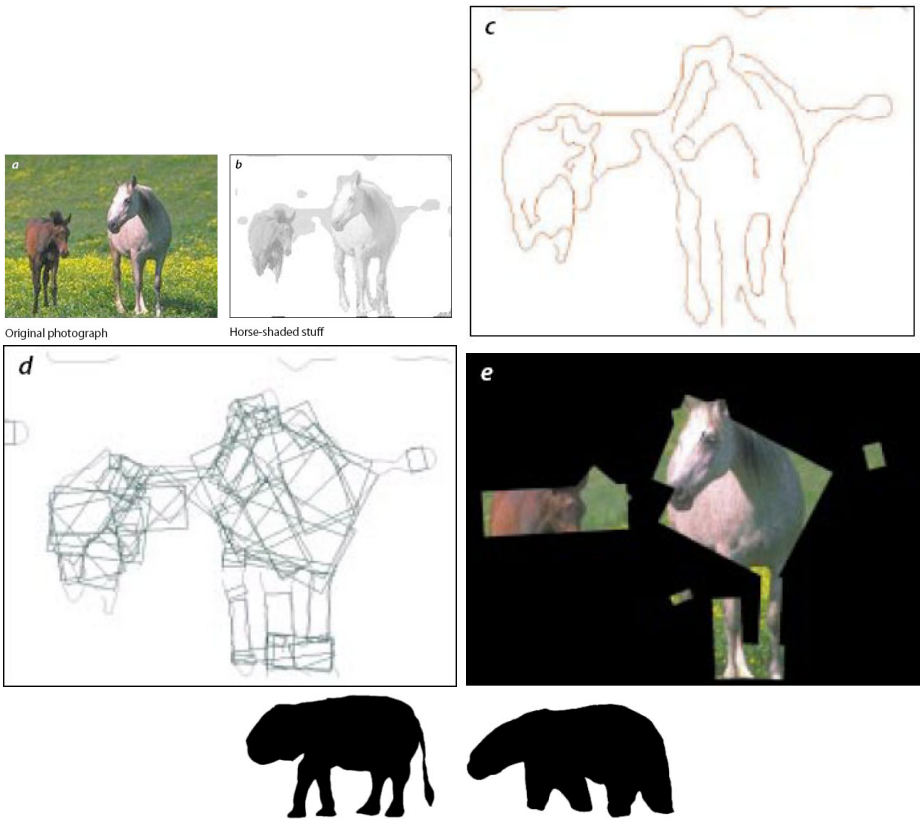


Figura 1.4: Gli algoritmi per il riconoscimento di oggetti operano raggruppando elementi dell'immagine in regioni via via più estese e complesse, per arrivare poi a generare ipotesi su ciò che tali regioni potrebbero rappresentare. Un riconoscitore di animali a quattro zampe a partire da un attento esame dell'immagine (a), cercherebbe zone con colori tipici del mantello equino (b). Quindi, passerebbe a definire i contorni di quelle regioni (c) e a cercare regioni della forma giusta (il corpo e le zampe di un cavallo sono approssimativamente dei cilindri) (d). A questo punto il programma cercherebbe di scartare regioni con relazioni spaziali impossibili (il corpo e le zampe devono essere perpendicolari) (e). Classificazioni definitive richiedono talvolta una conoscenza particolareggiata del colore e della trama di superficie degli oggetti. Anche un essere umano potrebbe trovarsi in difficoltà dovendo distinguere, solo sulla base della forma, tra la *silhouette* di un elefante con la proboscide ripiegata (ultima foto, sinistra) e quella di un orso (ultima foto, destra).

> Altri indizi contribuiscono a risolvere le difficoltà che si incontrano nel combinare le regioni di un'immagine che corrispondono a oggetti. Innanzitutto, molti oggetti hanno parti con forme tridimensionali semplici, e spesso anche le relazioni tra queste sono piuttosto semplici. Inoltre, le forme tridimensionali di base si presentano come regioni dell'immagine di forma generalmente semplice - per esempio, un cilindro appare quasi sempre come una regione delimitata da segmenti di retta più o meno paralleli. Di conseguenza, un programma può identificare queste parti con relativa facilità."

Questi metodi sono comunque ancora lontani da prestazioni ottimali o anche semplicemente utili, come riconoscono gli stessi autori:

> "Poiché non vi sono processi di ricerca perfetti, la prestazione di un sistema di ricerca automatico non deve essere paragonata agli ipotetici risultati di una ricerca perfetta. Perfino le migliori ricerche automatiche su materiale attentamente catalogato arrivano raramente ad individuare più del 50 per cento degli elementi rilevanti senza contemporaneamente segnalare un gran numero di elementi irrilevanti. Per quanto riguarda le ricerche manuali effettuate da esseri umani su archivi di immagini, abbiamo riscontrato - trascurandone i costi proibitivi - l'omissione di una porzione di immagini irrilevanti."

1.3 Grafica al calcolatore: costruzione di immagini

Affrontando lo studio delle basi della computer grafica ho potuto io stessa verificare quanto programmare computer per produrre immagini possa essere un buon modo per capire l'utilità, la bellezza, la praticità e, a volte, la semplicità della matematica: moltiplicare diviene sinonimo di ridurre in scala, dividere crea prospettive, seni e coseni ruotano gli oggetti, le tangenti creano dei tagli e la geometria e la trigonometria forniscono gli strumenti analitici per risolvere tutti gli altri tipi di problemi.

In una breve intervista al dottor Gian Marco Todesco, direttore della software company Digital Video che si occupa di strumenti per la produzione di animazione [Todesco], alla domanda *"Lei usa della geometria euclidea?"*, il dottor Todesco risponde:
"Si, così tanti concetti da rendere difficile l'enumerazione. Certamente sistemi di riferimento, coordinate, rappresentazione matriciale delle trasformazioni geometriche,

algebra vettoriale, coordinate baricentriche..."
È bastato infatti consultare il testo di matematica per computer grafica "Mathematics for computer graphics" di John Vince [Vince et al., 2006], per capire quanto il campo sia vasto: i grafici di funzione sono usati nella computer animation per controllare il movimento di oggetti, le luci e la camera virtuale, in sistemi di riferimento che mettono in relazione il movimento, la rotazione, la luminosità, il colore, ecc., con il tempo.

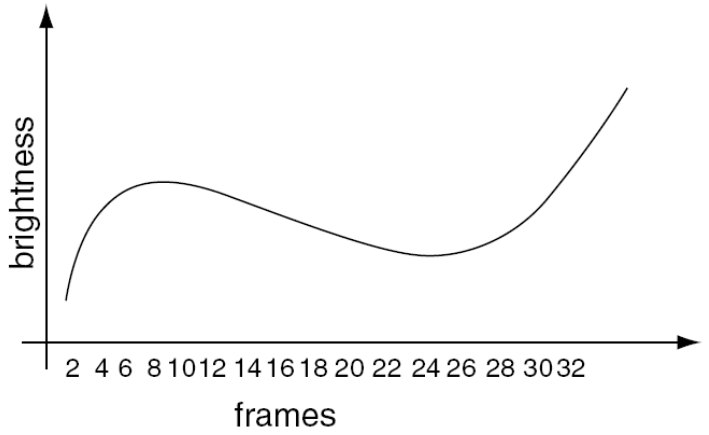


Figura 1.5: Esempio nel quale troviamo la sequenza degli istanti sull'asse x e la luminosità sull'asse y.

La figura (1.5) è un esempio in cui sull'asse x troviamo la progressione degli istanti di tempo della scena mentre l'asse y corrisponde alla luminosità di una fonte di luce virtuale; i poligoni sono strumenti importanti per la computer grafica in particolare per la loro bidimensionalità: tale proprietà ha un particolare rilievo poiché gli algoritmi di rendering assumono che i poligoni siano bidimensionali. Ad esempio è abbastanza semplice definire un quadrilatero in 3D i cui vertici non giacciano sullo stesso piano. Quando un poligono del genere però viene sottoposto a rendering ed animazione, possono risultare dei falsi riflessi, e questo perché le tecniche geometriche (che assumono la bidimensionalità del poligono) danno luogo ad errori. I vettori ci forniscono un mezzo per calcolare il comportamento di oggetti dinamici nella computer animation e nei modelli di illuminazione di rendering, inoltre vengono utilizzati ad esempio per specificare l'orientamento di una superficie, la direzione delle fonti di luce e della camera virtuale. Gli scalari comprendono invece colori, profondità, luminosità, numero di immagini, ecc.
Nella computer grafica vengono utilizzati quattro sistemi di riferimento: *object space*, è

un dominio in cui gli oggetti vengono modellati ed assemblati; *world space*, che è il luogo in cui gli oggetti vengono posizionati ed animati attraverso appropriate trasformazioni quali riduzione in scala, traslazione, rotazione, riflessione e taglio di forme e immagini; *camera space* è una trasformazione del *world space* dal punto di vista della camera; *image space* è una proiezione solitamente prospettica del camera space in un'immagine piana. Il processo di conversione dal sistema di riferimento *world space* a quello *camera space* risulta essere molto interessante, la procedura usata in genere dipende dal metodo applicato per definire la struttura di riferimento della camera all'interno del *world space*, tale metodo può coinvolgere l'uso del coseno, degli angoli di Eulero o delle "quaterne" (cioè di vettori in 4D, i quali possono essere utilizzati per ruotare punti intorno ad un asse immaginario e quindi anche l'orientamento degli oggetti e la camera virtuale). Inoltre, la proiezione che fa uso della prospettiva è molto utilizzata. Per arrivare alle coordinate del piano proiettivo si attraversano due fasi: la prima riguarda il passaggio dalle coordinate *world space* al sistema di riferimento della camera, la seconda trasforma le coordinate della camera nelle coordinate del piano prospettico. Anche l'interpolazione risulta essere un

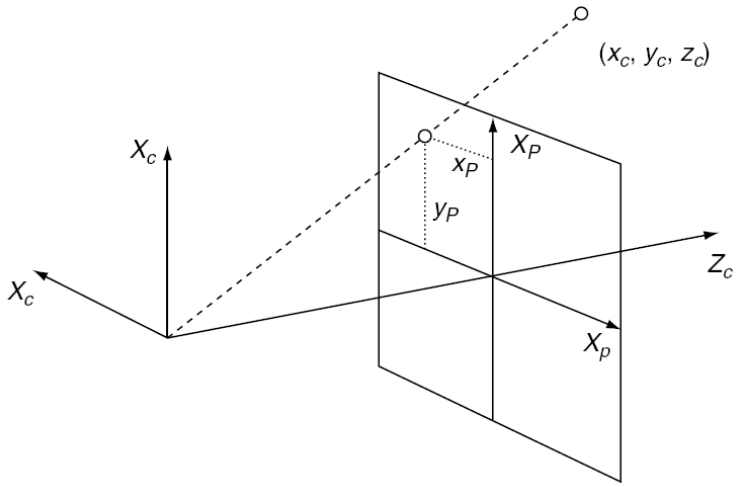


Figura 1.6: Traduzione delle coordinate del piano prospettico.

mezzo utile. Un'interpolazione è, banalmente, un modo di convertire un numero in un altro attraverso un numero finito n di operazioni. Questo procedimento, che può essere anche molto complesso, ci serve nella computer grafica quando ad esempio dobbiamo muovere un oggetto molto lentamente e gradualmente aumentare la sua velocità, oppure se si vuole portare un oggetto in movimento ad arrestarsi diminuendo la sua velocità a poco a poco.

Per quanto riguarda invece l'uso della topologia, spiega il dottor Todesco, certamente nella costruzione dei modelli 3D le relazioni fra le griglie di punti di controllo sono determinanti. Inoltre a volte si utilizzano delle nozioni di topologia nell'analisi delle sequenze per fare pattern matching, ma è un ambito vastissimo e piuttosto tecnico. Riportiamo infine parte dell'intervista in cui il dottor Todesco pone l'accento sui processi di traduzione della descrizione geometrica delle scene in immagine:

Può fare una panoramica sulle tecniche per la costruzione di immagini 2 dimensionali e 3 dimensionali?
"Dire "immagine tridimensionale" è un non senso, posso invece dire costruisco un modello tridimensionale da cui ricavo un'immagine bidimensionale. È bene distinguere perch`Êe sono due processi totalmente diversi che utilizzano programmi in genere diversi: nel primo caso ho bisogno di un modellatore , un oggetto che costruisce l'immagine tridimensionale, nel secondo caso ho invece bisogno di un render, cioè un oggetto che a partire da un modello tridimensionale mi costruisce un'immagine bidimensionale.

Costruzione di immagini bidimensionali di nuovo è un termine molto vasto, perché abbiamo immagini che vengono acquisite con scanner e quindi partono da un disegno fatto a mano sulla carta con la penna, e poi diventano essenzialmente una serie di punti digitali, oppure immagini formate da vettori che possono essere disegnate per esempio con una tavoletta grafica."

Come avviene il processo di traduzione della descrizione geometrica delle scene in immagini?
"Anche in questo caso il 2D e il 3D sono due cose totalmente diverse tra loro. Supponiamo di fissare il 3D. Ho una scena 3D e la devo descrivere in immagine. Tipicamente il programma che si utilizza per fare questa traduzione si chiama render, vale a dire un programma che prende una descrizione geometrica della scena e la traduce in un'immagine prospettica della scena medesima. Ci sono tantissimi render sia nel senso che ci sono tanti programmi che fanno questo, sia nel senso che ci sono tanti algoritmi che vengono utilizzati per fare questo; un algoritmo non molto noto si chiama *ray tracing* cioè tracciamento dei raggi: l'idea è che viene rovesciato il concetto della prospettiva tradizionale, quello inventato da Paolo Uccello. Per ogni pixel dello schermo si definisce

una linea che va dall'occhio dell'osservatore a questo pixel e questa retta viene intersecata con tutti gli oggetti che stanno dentro la scena. Per ogni oggetto ci possono essere delle intersezioni, quelle intersezioni vengono ordinate dalla più distante alla più vicina e alla fine è la più vicina quella che conta per dare il colore di quel pixel. Se la sostanza di cui è costituito l'oggetto intersecato in quel punto è una sostanza trasparente oppure riflettente tipicamente viene generato un altro raggio rispettivamente rifratto o riflesso e di nuovo ricorsivamente si richiama la stessa procedura per ottenere un colore associato a questo raggio e questo colore viene sommato al colore del primo oggetto incontrato con un'opportuna funzione di *blending* (mescolamento) per cui alla fine il colore di quel pixel sarà in parte il colore dell'oggetto, in parte il colore dell'oggetto riflesso dal primo oggetto in quel punto.

Analogamente, per calcolare le ombre quel che si fa è: dal punto di contatto del raggio "di vista", per così dire dall'occhio dell'osservatore all'oggetto che stiamo considerando, si mandano dei raggi a tutte le sorgenti di luce presenti nella scena, ogni raggio viene intersecato con tutti gli altri raggi presenti nella scena per vedere se c'è un'intersezione, e quindi se c'è un oscuramento. Nel caso in cui non ci sia alcun oscuramento allora quella luce dà un particolare contributo al colore in quel punto. Il modello con cui la luce contribuisce al colore in quel punto può essere un modello anche molto complicato, in genere si utilizza uno schema semplificato in cui si va a controllare il coseno dell'angolo compreso fra la normale e la direzione della luce in quel punto, e questo dà un contributo alla luminosità, poi c'è un altro contributo che rende conto delle riflessioni. Per questo bisogna in realtà guardare anche l'angolo che da un punto va verso l'occhio dell'osservatore. In realtà il modello arbitrario è un modello molto più complicato in cui entrano in gioco la direzione del raggio di vista, quindi la direzione dell'osservatore, la direzione dei vari soggetti di luce su quel punto non schermati da oggetti che fanno ombra, e la normale alla superficie in quel punto. Questi vettori vengono mescolati insieme dal modello che come ho detto può essere anche molto complicato, per dare origine al particolare colore che va verso l'osservatore e che quindi andrà ad interessare il pixel da cui siamo partiti. Questi modelli molto complicati permettono di realizzare per esempio delle superfici che hanno delle riflessioni anisotrope.

Un'altra tecnica completamente diversa si chiama *radiosity* ed è una tecnica molto più sofisticata ma molto adatta ad utilizzare le immagini in penombra (per esempio l'interno

di abitazioni, di chiese, eccetera). Quel che fa è imporre il bilanciamento termico fra tutte le superfici presenti nella scena: ogni superficie presente sulla scena viene considerata una superficie che emette luce e ovviamente la radiazione emessa, se l'oggetto non è una lampada, è una funzione della radiazione ricevuta, e quello che il programma fa è risolvere un grosso sistema di equazioni lineari in n incognite: queste incognite sono la luce emessa da ogni singola superficie, una volta risolto questo sistema si capisce quanta luce effettivamente emette una superficie magari perché è stata illuminata da un'altra superficie. Quando si guarda ad esempio sotto il tavolo si vedono gli oggetti che sono sotto al tavolo, la luce che arriva su questi oggetti non è diretta ma si è riflessa sul fondo del tavolo, sul pavimento, sulle pareti, con tutta una serie di riflessioni multiple che con questo sistema vengono gestite efficacemente. La quantità di algoritmi per generare l'immagine tridimensionale, cioè l'immagine prospettica che corrisponde alla scena è veramente enorme. Un altro algoritmo da considerare è *reyes* sviluppato dalla Pixar: è un algoritmo ancora diverso, il render più noto che usa questo algoritmo si chiama *rendermann* é un algoritmo molto efficiente poiché è molto rapido nel generare le immagini e nello stesso tempo permette un controllo molto raffinato delle caratteristiche della superficie degli oggetti. In particolare è possibile fare fibre come tessuti, o il particolare effetto traslucido della pelle umana, è possibile realizzare i capelli, il pelo, e tutto questo grazie a una tecnica che si chiama *shading procedurale*: è infatti possibile scegliere in maniera procedurale il tipo i interazione tra la luce e gli oggetti punto per punto.

Tutto questo serve per produrre i pixel, i quali vengono poi ulteriormente gestiti in vari modi per eliminare alcuni difetti, un difetto molto tipico è la *lising* cioè, siccome stiamo facendo un campionamento dell'immagine in punti discreti che sono punti pixel, se nell'immagine ci sono delle frequenze più alte, il campionamento può essere insufficiente e questo secondo la teoria dei segnali che gestisce in maniera molto efficace questo tipo di soggetto secondo le frequenze spurie, che appaiono per esempio come scalettature nelle diagonali; per eliminare queste scalettature, molto fastidiose agli occhi, si utilizzano delle tecniche di filtraggio che possono essere anch'esse molto sofisticate."

Capitolo 2

Ridimensionamento di immagini

Il ridimensionamento di immagini è uno strumento adoperato in molti processi di rielaborazione di immagini. Soprattutto ultimamente, c'è un interesse crescente per questo aspetto dell'elaborazione di immagini poiché si stanno cercando delle nuove tecnologie per cambiare le dimensioni di una figura mantenendo intatti i punti importanti; l'importanza viene assegnata attraverso opportuni metodi di rilevazione che costruiscono una mappa dei punti salienti dell'immagine, come i detector facciali. In questo capitolo proponiamo un semplice operatore per l'immagine, chiamato *seam carving*, che può cambiare la dimensione di un'immagine estraendo o inserendo pixel in diverse parti dell'immagine, a seconda che l'operazione desiderata sia di allargamento o di ridimensionamento. Generalmente nei metodi di ridimensionamento risulta difficile evitare le deformazioni e tenere in considerazione il contenuto dell'immagine. Proprio questo è il merito del *seam carving*: è possibile cambiare le dimensioni dell'immagine presa in considerazione senza deformarla e, soprattutto, tenendo in considerazione il contenuto.

Data un'immagine si tracciano delle *seam*, delle giunture, tecnicamente dei cammini connessi di pixel, che attraversano l'immagine da sinistra verso destra e dall'alto verso il basso. Attraverso una funzione energia si attribuisce un ordine alle varie linee a seconda dei pixel di cui sono costituite: risulteranno quindi cammini costituiti da pixel aventi bassa energia e cammini aventi alta energia. Per la riduzione, la selezione delle giunture ci permette di preservare la struttura dell'immagine rimuovendo molti pixel a bassa energia e pochi pixel ad alta energia. Analogamente, per l'allargamento l'ordine dei cammini inseriti assicura un bilanciamento tra il contenuto dell'immagine originale e i pixel inseriti artificialmente, in modo tale che l'effetto finale non risulti distante dall'immagine di partenza.

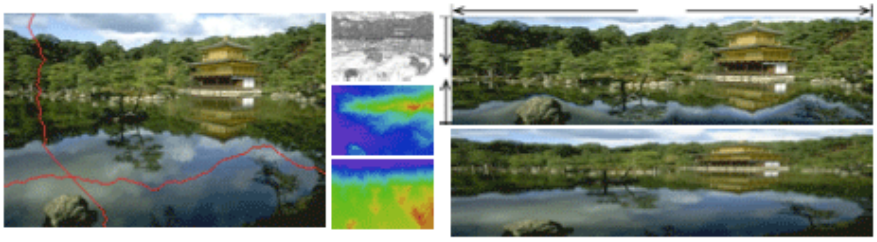


Figura 2.1: Una giuntura è un cammino connesso costituito da pixel a bassa energia in un'immagine. A sinistra c'è l'immagine originale con una giuntura verticale e una orizzontale. Nel mezzo, la funzione energia utilizzata in questo esempio insieme a mappe di cammini verticali ed orizzontali usate per calcolare le giunture. A destra vengono paragonati i risultati del ridimensionamento ottenuto attraverso il *seam carving* (in alto) e attraverso la riduzione in scala standard (in basso).

2.1 L'operatore

Nel nostro approccio il lavoro principale è rimuovere pixel mantenendo armonia nel contenuto dell'immagine. Per fare ciò dovremmo togliere pixel che si fondono con l'area circostante. Utilizziamo quindi una semplice funzione energia e_1 definita in questo modo:

$$e_1(\mathbf{I}) = \mid \frac{\delta}{\delta x}\mathbf{I} \mid + \mid \frac{\delta}{\delta y}\mathbf{I} \mid \tag{2.1}$$

Sia $\mathbf{I}$ un'immagine $n \times m$, allora, $\forall i : \mid x(i) - x(i-1) \mid \leq 1, i = 1, \ldots, n$, definiamo una giuntura verticale come:

$$\mathbf{s^x} = \{s_i^x\} = \{(x(i), i)\} \tag{2.2}$$

dove x é una funzione $x : \{1, \ldots, n\} \longrightarrow \{1, \ldots, m\}$. Cioè, una giuntura verticale é un cammino connesso di pixel dell'immagine che va dall'alto verso il basso, e che contiene un solo pixel per riga. Analogamente, sia $y : \{1, \ldots, m\} \longrightarrow \{1, \ldots, n\}$ una funzione, allora, $\forall j : \mid y(j) - y(j-1) \mid \leq 1, j = 1, \ldots, m$, una giuntura orizzontale si definisce così:

$$\mathbf{s^y} = \{s_j^y\} = \{(j, y(j))\} \tag{2.3}$$

I pixel, ad esempio in un cammino verticale $\mathbf{s}$, sono allora $\mathbf{I_s} = \{\mathbf{I}(s_i)\} = \{\mathbf{I}(x(i), i)\}$ con $i = 1, \ldots, n$. Rimuovere i pixel di una linea dall'immagine crea solo un effetto locale: tutti i pixel dell'immagine vengono spostati a sinistra o in alto per compensare la parte di cammino che è stata rimossa. L'impatto visivo si ha solo lungo la giuntura mentre il resto dell'immagine risulta intatta. Data una funzione energia e, definiamo il costo di una

giuntura come $E(\mathbf{s}) = E(\mathbf{I_s}) = \sum_{i=1}^{n} e(\mathbf{I}(s_i))$. Cerchiamo una giuntura ottimale s^* che minimizzi il costo di giuntura scritto sopra:

$$s^* = min_s E(\mathbf{s}) = min_s \sum_{i=1}^{n} e(\mathbf{I}(s_i)). \tag{2.4}$$

La giuntura ottimale si trova utilizzando un programma dinamico. Si attraversa dapprima l'immagine dalla seconda riga all'ultima e si calcola l'energia minima cumulativa M per tutti i possibili cammini connessi:

$$M(i,j) = e(i,j) + min_s(M(i-1,j-1), M(i-1,j), M(i-1,j+1)) \tag{2.5}$$

Al termine di questo processo, il valore minimo dell'ultima riga in M indicherà la fine del cammino verticale connesso. A questo punto, torniamo indietro da questo minimo fino a M per trovare il cammino ottimale. Analogamente troviamo M per i cammini orizzontali. Esistono varie strategie per il ridimensionamento di immagini attento al contenuto e per valutarne l'efficacia si può esaminare l'energia media dei pixel di un'immagine durante il ridimensionamento: $\frac{1}{|\mathbf{I}|}\sum_{p\in I} e(p)$. Dal momento che il ridimensionamento che tiene conto del contenuto tende a rimuovere i pixel a bassa energia, nel nostro caso la media non si manterrà la stessa ma tenderà ad aumentare.

2.2 Ridimensionamento discreto dell'immagine

2.2.1 Ordine di rimozione

Se volessimo modificare l'immagine in una delle sue dimensioni da $n \times m$ a $n \times m'$, dove $m - m' = c$, sarebbe sufficiente rimuovere c linee verticali da $\mathbf{I}$; a differenza della riduzione in scala, questa operazione non altera parti importanti dell'immagine, come abbiamo già descritto in precedenza, e crea un ridimensionamento dell'immagine *non-uniforme* ma attento al contenuto. Lo stesso tipo di correzione si può ottenere anche aumentando il numero di righe di un fattore m / m'. È anche possibile cambiare tutte e due le dimensioni dell'immagine, passare cioè da $n \times m$ a $n' \times m'$, assumendo, per ora, che $m' < m$ e $n' < n$. La domanda che ci si pone è: qual é il corretto ordine in cui togliere le giunture?
L'ordine ottimale è dato dall'ottimizzazione della funzione:

$$min_{s^x, s^y, \alpha} \sum_{i=1}^{k} E(\alpha_i \mathbf{s}_i^x + (1-\alpha_i)\mathbf{s}_i^y) \tag{2.6}$$

in cui k=r+c, r=m-$m^{'}$, c=n-$n^{'}$ e α_i viene utilizzato come parametro che determina se al passo i rimuoviamo una giuntura verticale o orizzontale: $\alpha_i \in \{0,1\}, \sum_{i=1}^{k} \alpha_i = r, \quad \sum_{i=1}^{k}(1-\alpha_i) = c$.

Troviamo l'ordine ottimale di rimozione usando una mappa di trasporto $\mathbf{T}$ che specifica, per ogni dimensione $n^{'} \times m^{'}$, il costo della sequenza ottimale delle operazioni di rimozione verticale e orizzontale. T(r,c) rappresenta il costo minimo necessario per ottenere un'immagine di dimensioni n-$r \times m$-c. Calcoliamo $\mathbf{T}$ attraverso un programma dinamico: iniziando da $\mathbf{T}$(0,0)=0, riempiamo ogni ingresso (r,c) scegliendo la migliore delle due opzioni: rimuovere una giuntura orizzontale da un'immagine di dimensioni n-$r \times m$-c+1 o rimuovere una giuntura verticale da un'immagine di dimensioni n-r+1$\times m$-c:

$$\mathbf{T}(r,c) = min(T(r-1,c) + E(\mathbf{s}^x(\mathbf{I}_{n-r-1\times m-c})), T(r,c-1) + E(\mathbf{s}^y(\mathbf{I}_{n-r\times m-c-1}))) \quad (2.7)$$

in cui $\mathbf{I}_{n-r\times m-c}$ denota un'immagine di dimensioni n-$r \times m$-c, $E(\mathbf{s}^x(\mathbf{I}))$ e $E(\mathbf{s}^y(\mathbf{I}))$ sono i costi delle rispettive operazioni di rimozione delle giunture.

Figura 2.2: In alto l'immagine originale. In basso, da sinistra a destra, l'immagine ridimensionata usando *seam carving*, la riduzione in scala e il metodo di ritaglio.

Grazie ad una semplice mappa $n \times m$ sappiamo quale delle due opzioni viene scelta di volta in volta durante il programma: scegliendo un intorno sinistro si rimuove una giuntura verticale, un intorno in alto corrisponde a rimuovere una giuntura orizzontale. Date le dimensioni $n' \times m'$, dove $n' = n - r$ e $m' = m - c$, torniamo indietro da $\mathbf{T}$(r,c) a $\mathbf{T}$(0,0) e applichiamo le relative operazioni di rimozione.

Figura 2.3: Ridimensionamento di una sola dimensione dei lavori del giapponese Utagawa Hiroshige

2.2.2 Allargamento

Il processo di rimozione di giunture orizzontali e verticali può essere visto come un processo di evoluzione temporale. Denotiamo con $\mathbf{I}^{(t)}$ la più piccola immagine creata

dopo che t giunture siano state tolte dall'immagine originale. Per allargare l'immagine approssimiamo un'inversione di questa evoluzione temporale ed inseriamo nuove giunture artificiali all'immagine. Quindi, per allargare le dimensioni di un'immagine $\mathbf{I}$ calcoliamo le relative giunture ottimali verticali e orizzontali $\mathbf{s}$ su $\mathbf{I}$ e duplichiamo i pixel di $\mathbf{s}$ facendo una media dei loro intorni destro e sinistro nel caso verticale e superiore e inferiore nel caso orizzontale.

Usando la notazione evolutiva, denotiamo l'immagine che otteniamo come $\mathbf{I}^{(-1)}$. Purtroppo ripentendo questo processo andiamo incontro a degli artefatti. Per ottenere un effettivo allargamento è importante bilanciare il contenuto dell'immagine originale e le parti artificiali inserite in un secondo momento. Quindi, per allargare un'immagine di un valore k, troviamo le prime k giunture da rimuovere e le duplichiamo in modo tale da arrivare a $\mathbf{I}^{(-k)}$. Quest'operazione può essere interpretata come il processo di attraversamento indietro nel tempo per mantenere pixel che sarebbero stati rimossi.

Duplicando tutte le giunture in un'immagine si ottiene lo stesso risultato della riduzione in scala standard, per ingrandire una figura di più del 50% spezziamo il processo in diversi passi in ognuno dei quali ingrandiamo l'immagine di una frazione di se stessa rispetto al passo precedente, cercando di non allargare anche il contenuto. Un allargamento eccessivo produce notevoli artefatti.

Combinando *seam carving* con la riduzione in scala è possibile ampliare il contenuto dell'immagine preservandone le dimensioni. Per preservare il più possibile il contenuto dell'immagine usiamo dapprima la riduzione standard per allargare l'immagine e solo in un secondo momento applichiamo il *seam carving* sull'immagine allargata per riportare l'immagine alla sua dimensione originale.

Figura 2.4: Ampliamento del contenuto: a sinistra l'immagine originale, a destra una combinazione di *seam carving* e riduzione in scala.

A volte è utile combinare *seam carving* con la ricostruzione di Poisson [Perez et al., 2003]: calcoliamo l'immagine della funzione energia come abbiamo fatto prima, ma invece di rimuovere le giunture dall'immagine originale, lavoriamo sul dominio del gradiente e rimuoviamo le giunture dalle derivate calcolate rispetto ad x,y dell'immagine originale. Alla fine di tale processo usiamo Poisson per ricostruire l'immagine.

2.2.3 Rimozione di oggetti

Attraverso una semplice interfaccia è possibile utilizzare il *seam carving* per la rimozione di oggetti. Si seleziona l'oggetto da rimuovere e vengono man mano eliminate dall'immagine le relative giunture finché tutti i pixel selezionati risultino rimossi. Il sistema calcola automaticamente il più piccolo dei diametri orizzontali e verticali della regione selezionata ed avvia la rimozione. Questo sistema altera l'intera immagine, comprese le sue dimensioni ed il suo contenuto.

Figura 2.5: Rimozione di oggetti: si seleziona la parte da rimuovere (in verde) e la parte da proteggere (in rosso). Nella figura a sinistra sono state rimosse giunture verticali finch é tutti i pixel verdi non sono stati rimossi.

2.3 Conclusioni

Questo metodo non lavora automaticamente su tutte le immagini ed a volte, come in figura, il metodo fallisce. Nel caso dell'immagine (2.6) si è raggiunto un risultato ottimale solo combinando il *seam carving* con un detector facciale. Ci sono due fattori in particolare che limitano il sistema studiato: il primo riguarda immagini con un alto numero di contenuti, vale a dire immagini nelle quali non ci sono aree non importanti. In questo caso nessun metodo di *seam carving* è efficiente. Il secondo riguarda la disposizione dei contenuti nell'immagine: in alcuni tipi di immagini il contenuto è disposto in un modo che non permette alle giunture di oltrepassare parti importanti.

Ci sono numerose estensioni possibili per questa applicazione; in futuro si cercherà di estenderlo in particolare al campo del ridimensionamento di video.

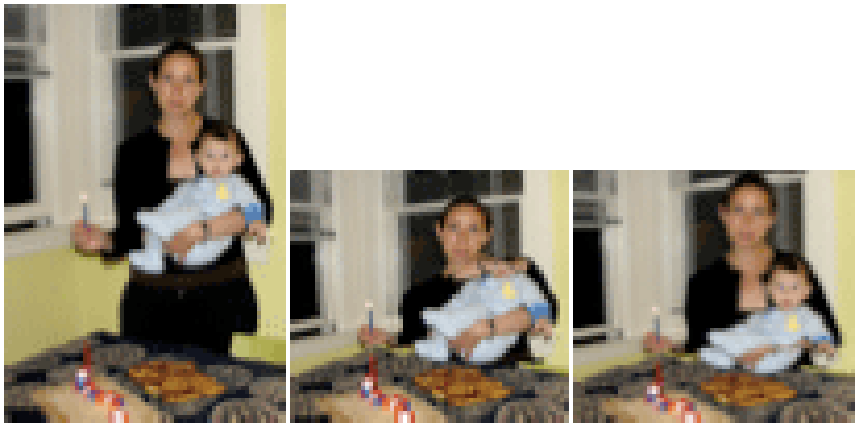


Figura 2.6: Immagine originale, immagine ridimensionata utilizzando prima solo e_1 poi e_1 combinato con il detector facciale.

Capitolo 3

Animazione di foto

Il modello che descriviamo in questo capitolo è interessante non tanto per l'aspetto matematico, quanto per i risultati che attraverso di esso è possibile raggiungere. È possibile infatti, partendo da una comune foto che ritrae il volto di una persona, animare quel volto.

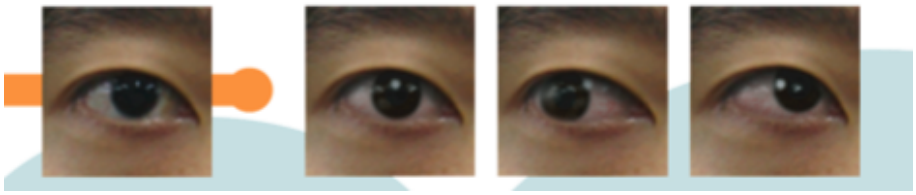
Motion Portrait è una semplice tecnica elaborata dalla Silicon Studio Corporation di Tokyo per la creazione di animazioni facciali realistiche [Motion Portrait]. Attraverso un processo molto semplice, usando una sola immagine, viene creato un modello verosimile. È possibile applicare questo metodo per varie finalità: innanzi tutto per **generare modelli facciali 3D partendo da una singola immagine di un volto in posizione frontale**: il nostro sistema riconosce automaticamente i punti caratteristici del volto e, partendo da questi, crea un modello 3D, adattando i punti caratteristici precedentemente trovati ad un modello standard. Entrambi i lati del viso vengono estesi anche alla parte posteriore per generare la zona mancante del viso, quella cioè che non è visibile nella foto. Per fare ciò viene calcolata un'adeguata profondità. Anche i capelli possono essere ricostruiti automaticamente.

Andando avanti con la scoperta delle applicazioni di questo strumento, scopriamo che il motore ci consente di **controllare varie espressioni facciali** attraverso una semplice interfaccia. Anche con modelli ad alta risoluzione, sono necessarie tecniche avanzate per creare e controllare le espressioni facciali naturali. *Motion Portrait* ha un

motore di espressione facciale che realizza espressioni facciali di base, come se la persona rappresentata stia sorridendo, o parlando, o girando gli occhi. L'espressione facciale naturale viene controllata attraverso un'interfaccia astratta.

Con *Motion Portrait* partendo da una singola immagine frontale, il processo di animazione facciale stile video risulta immediato poiché è completamente automatico. Un'**animazione realistica stile video** si raggiunge elaborando zone del volto dettagliate: il sistema, infatti, riconosce accuratamente i bulbi oculari, ne desume la tessitura al fine di riprodurne una il più possibile analoga; a questo punto, i bulbi oculari originali vengono sostituiti con la nuova tessitura oculare.

Anche il bordo che separa il bulbo oculare e le palpebre viene riconosciuto con facilità. Tale riconoscimento permette di generare modelli 3D foto-realistici ed animazioni facciali video-realistiche. Un'interessante applicazione di questo strumento permette l'utilizzo anche ai "non addetti ai lavori": supponiamo di avere una foto frontale di un viso umano o del musetto di un cane. Questa tecnologia ricostruisce automaticamente la persona seguendo le sembianze del volto che appare nella foto, poi ci permette di muovere lo sguardo di quel nuovo volto semplicemente muovendo a nostro piacere il mouse. Anche le palpebre, come è ben visibile nella figura (3) seguiranno lo spostamento della pupilla e di conseguenza si ottiene una deformazione della pelle a seconda del movimento degli occhi.



Infine, è curioso vedere come le applicazioni fin qui descritte possono essere utilizzate per rendere espressivi dei disegni 2D, come nel caso dei **cartoni animati**.

Soprattutto per quanto riguarda lo sviluppo dei videogiochi, i processi di creazione di animazioni sono un campo in continua evoluzione.

Avviando la parte automatica del processo, che include l'estrazione dei punti caratteristici del volto e la generazione di un nuovo modello facciale, si può manualmente ottenere un'animazione facciale di alta qualità. Applicando questo sistema a sagome 2D, come quelle dei cartoni animati, è semplice raggiungere delle animazioni facciali lisce e sottili che non perdono la loro originale atmosfera.

Capitolo 4

Manipolazione rigida di forme bidimensionali

4.1 Descrizione

In questo capitolo descriviamo un sistema interattivo ideato da Takeo Igarashi dell'Università di Tokyo [Takeo Igarashi], che permette di manipolare (muovere, deformare, ruotare, allungare, curvare, ecc) immagini 2D o disegni fatti a mano. La figura è rappresentata da una mesh di triangoli, ovvero un complesso simpliciale bidimensionale, ed in essa si possono muovere diversi vertici come se si muovessero delle manopole fisse. Il sistema calcola poi le posizioni dei restanti vertici liberi minimizzando la distorsione di ogni triangolo. Ciò che descriviamo è un algoritmo che serve per trovare la forma che minimizzi la distorsione; tale algoritmo è composto da due fasi: nella prima si cerca la rotazione appropriata di ogni triangolo, nella seconda se ne aggiusta la grandezza. L'idea di base è usare una metrica dell'errore quadratico in modo tale da rendere ogni problema di minimizzazione un sistema di equazioni lineari. Dopo aver risolto le equazioni simultanee all'inizio dell'interazione possiamo trovare velocemente le posizioni dei vertici liberi durante l'interazione. Lo scopo è mantenere una certa rigidità all'interno del modello durante la manipolazione della sagoma usando un semplice approccio geometrico. Quindi: si deve importare una forma 2D nel sistema, con la sola limitazione che la sagoma deve poter essere descrivibile come un semplice poligono chiuso; il sistema allora genera una mesh di triangoli interna al contorno dell'immagine. Per ottenere un buon risultato è preferibile ricoprire la regione interessata con triangoli equilateri o quasi, approssimativamente uguali tra loro. Si inizia con una triangolazione standard di Delaunay [De Berg et al., 1997], il

sistema provvede poi ad aggiustare le posizioni dei vertici e la connettività generale del tessuto. Il sistema corrente è in grado di generare in pochi secondi tessuti con vertici che variano dai 100 ai 300. Il tessuto triangolato che ne risulta viene chiamato forma a riposo. Si procede quindi cliccando sulla sagoma per posizionare le manopole che permetteranno la manipolazione: trascinandole a piacimento si ottiene un movimento di tutta l'immagine. Per ora è possibile posizionare tali manopole solo in vertici predefiniti del tessuto ma l'obiettivo per il futuro è quello di posizionarle secondo le proprie necessità. La manipolazione dell'immagine viene applicata prima alla mesh di triangoli, quindi il sistema trasforma la mesh originale in quella deformata. Chiamiamo questa fase *compilazione* perché il processo prepara una funzione che dà come input la configurazione delle manopole e come output la forma che ne risulta.

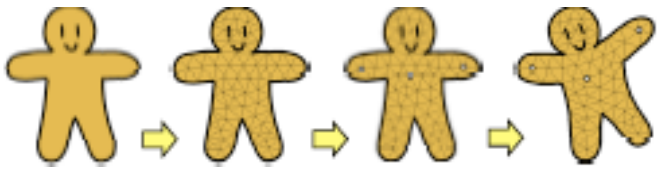


Figura 4.1: Dall'immagine originale si passa all'immagine triangolata (immagine a riposo), poi si applicano le manopole (compilazione), infine si muove la sagoma (manipolazione).

4.2 Algoritmo

Ciò che vogliamo è controllare la distorsione di ogni singolo triangolo dell'immagine triangolata. Poiché è impossibile trovare una singola funzione quadratica dell'errore che rappresenti la distorsione totale, ci troviamo costretti a scindere il problema in due parti: quella della rotazione e quella della riduzione in scala, in modo tale che ciascuna delle due parti abbia una propria funzione quadratica dell'errore.

4.2.1 Prima fase

L'input che viene dato al sistema sono le coordinate *x,y* dei vertici vincolati e l'output sono le coordinate *x,y* dei rimanenti vertici liberi. Assegniamo una funzione quadratica dell'errore ad ogni singolo triangolo.
Abbiamo il triangolo deformato, di vertici $\{v_0', v_1', v_2'\}$ e il triangolo corrispondente della forma a riposo, di vertici $\{v_0, v_1, v_2\}$.
Il sistema innanzi tutto calcola le coordinate di v_2 nel sistema di riferimento definito da

v_0, v_1:

$$v_2 = v_0 + x_{01}\mathbf{v}_0\mathbf{v}_1 + y_{01}R_{90}\mathbf{v}_0\mathbf{v}_1 \tag{4.1}$$

in cui R_{90} denota la rotazione di 90 gradi in senso antiorario. Dati $v_0^{'}, v_1^{'}, v_2^{'}, x_{01}, y_{01}$, il sistema può calcolare la posizione desiderata per $v_2^{'}$:

$$v_2^{desid} = v_0^{'} + x_{01}\mathbf{v}_0^{'}\mathbf{v}_1^{'} + y_{01}R_{90}\mathbf{v}_0^{'}\mathbf{v}_1^{'} \tag{4.2}$$

in cui

$$\mathbf{R_{90}} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

L'errore associato a $v_2^{'}$ corrisponde allora alla quantità:

$$E_{\{v_2\}} = \|v_2^{desid} - v_2^{'}\|^2 \tag{4.3}$$

Allo stesso modo possiamo definire v_0^{desid} e v_1^{desid}, e definiamo l'errore associato all'intero triangolo come:

$$E_{\{v_0,v_1,v_2\}} = \sum_{i=1,2,3} \|v_i^{desid} - v_i^{'}\|^2 \tag{4.4}$$

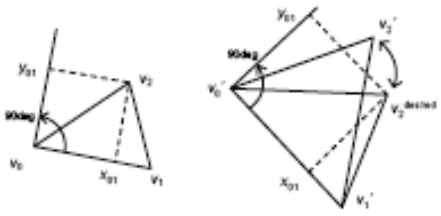


Figura 4.2: Metrica dell'errore utilizzata nella fase 1. v_2^{desid} si ottiene sistemando il triangolo originale su quello che si vuol ottenere attraverso la rotazione, traslazione, riduzione in scala, in modo tale che $v_0^{'}$ e $v_1^{'}$ incontrino v_0 e v_1

L'errore per l'intera mesh della sagoma è dato dalla somma di tutti gli errori dei triangoli della mesh. Dal momento che la metrica dell'errore è una forma quadratica in $\mathbf{v}^{'} = (v_{0x}^{'}, v_{0y}^{'}, \ldots, v_{nx}^{'}, v_{ny}^{'})$, possiamo esprimerla in forma matriciale:

$$\mathbf{E_{1\{v'\}}} = \mathbf{v}^{'T}\mathbf{G}\mathbf{v}^{'} \tag{4.5}$$

Il problema della minimizzazione si risolve ponendo le derivate parziali della funzione $E_{1\{\mathbf{v}'\}}$ rispetto alle variabili libere $\mathbf{u} = (u_{0x}, u_{0y}, \ldots, u_{mx}, v_{my})^T$ in $\mathbf{v}^{'}$ pari a zero. Per riscrivere $\mathbf{v}^{'}$ poniamo $\mathbf{v}^{'T} = (\mathbf{u}^T\mathbf{q}^T)$, dove $\mathbf{q}$ rappresenta i vertici vincolati. Quindi:

$$\mathbf{E_1} = \mathbf{v}^{'T}\mathbf{G}\mathbf{v}^{'} = \begin{pmatrix} \mathbf{u} \\ \mathbf{q} \end{pmatrix}^T \begin{pmatrix} \mathbf{G}_{00} & \mathbf{G}_{01} \\ \mathbf{G}_{10} & \mathbf{G}_{11} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{q} \end{pmatrix} \tag{4.6}$$

$$\frac{\delta E_1}{\delta \mathbf{u}} = (\mathbf{G}_{00} + \mathbf{G}_{00}^T)\mathbf{u} + (\mathbf{G}_{01} + \mathbf{G}_{10}^T)\mathbf{q} = 0 \tag{4.7}$$

Riscriviamo la (4.7) così:

$$\mathbf{G}^{'}\mathbf{u} + \mathbf{B}\mathbf{q} = 0 \tag{4.8}$$

Osserviamo che $\mathbf{G}^{'}$ e $\mathbf{B}$ sono fissati e solo $\mathbf{q}$ cambia durante la manipolazione. Il calcolo affrontato in questa prima fase risulta molto semplice e si possono traslare e ruotare delle immagini con successo usando solo questa parte di calcolo. Tuttavia, come già precisato all'inizio, poiché la funzione errore non calcola i cambi di scala, la forma si allarga quando le manopole si allontanano tra loro e si riduce quando si avvicinano. Per questo motivo abbiamo bisogno della fase due che ci accingiamo a descrivere.

4.2.2 Seconda fase

Ora abbiamo questa situazione: i triangoli della forma a riposo, i triangoli del risultato intermedio ottenuto dalla fase uno, quelli in cui cioè abbiamo ruotato e traslato le figure ma non c'è stata alcuna riduzione in scala, e i nuovi triangoli adattati in modo tale da essere congruenti ai triangoli della forma a riposo e che allo stesso tempo minimizzino la quantità:

$$E_{ad\{v_0^{adatt}, v_1^{adatt}, v_2^{adatt}\}} = \sum_{i=1,2,3} \|v_i^{adatt} - v_i^{'}\|^2 \tag{4.9}$$

Può essere molto difficile ottenere direttamente questo risultato, per questo prima minimizziamo l'errore, poi aggiustiamo la scala.

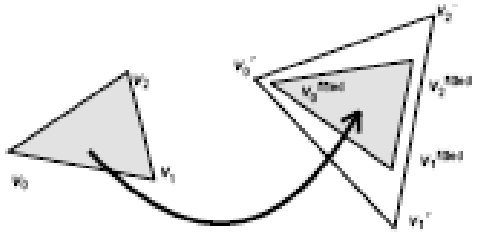


Figura 4.3: Adattamento del triangolo originale al triangolo intermedio attraverso rotazione e traslazione.

Usando le coordinate x_{01} e y_{01} definite nella prima fase dell'algoritmo, possiamo esprimere v_2^{adatt} usando v_0^{adatt} e v_1^{adatt}:

$$v_2^{adatt} = v_0^{adatt} + x_{01}\mathbf{v}_0^{adatt}\mathbf{v}_1^{adatt} + y_{01}R_{90}\mathbf{v}_0^{adatt}\mathbf{v}_1^{adatt} \tag{4.10}$$

così la funzione di adattamento dipende solo dalle coordinate di v_0^{adatt} e v_1^{adatt}, una forma quadratica nelle quattro variabili libere $\mathbf{w}$=$(v_{0x}^{adatt}, v_{0y}^{adatt}, v_{1x}^{adatt}, v_{1y}^{adatt})^T$. Minimizziamo la (4.9) ponendo le derivate parziali di E_{ad} uguali a zero. Il risultato è un semplice sistema lineare 4×4 che si può esprimere in forma matriciale come:

$$\frac{\delta E_{ad}}{\delta \mathbf{w}} = (\mathbf{F}\mathbf{w} + \mathbf{C})\mathbf{u} = 0 \tag{4.11}$$

F è fissato e **C** è definito dal risultato della prima fase. Risolvendo questa equazione otteniamo un nuovo triangolo adattato $\{v_0^{adatt}, v_1^{adatt}, v_2^{adatt}\}$. Lo rendiamo congruente semplicemente riducendolo in scala secondo il fattore: $\|v_0^{adatt} - v_1^{adatt}\|/\|v_0 - v_1\|$. Applichiamo questa operazione di adattamento su tutti i triangoli della mesh. Rimane un unico problema: ogni vertice della mesh originale appare in diversi triangoli e quindi corrisponde a vertici multipli nei triangoli adattati. Il sistema calcola quindi le coordinate *x,y* finali dei vertici liberi, partendo dalle coordinate dei vertici vincolati, minimizzando la differenza tra il triangolo che risulta nella mesh e il triangolo adattato. Definiamo la funzione dell'errore quadratico tra un triangolo originale e lo stesso triangolo adattato come:

$$E_{2\{v_0'', v_1'', v_2''\}} = \sum_{i,j \in \{(0,1),(1,2),(2,0)\}} \|\mathbf{v}_i'' \mathbf{v}_j'' - \mathbf{v}_i^{adatt} \mathbf{v}_j^{adatt}\|^2 \tag{4.12}$$

È importante notare che associamo gli errori ai lati e non ai vertici, quindi utilizziamo la rotazione del triangolo adattato e non ne consideriamo la posizione. L'errore viene minimizzato quando i triangoli $\{v_0'', v_1'', v_2''\}$ e quelli $\{v_0^{adatt}, v_1^{adatt}, v_2^{adatt}\}$ sono identici. Ma finché ad esempio il vertice v_0'' giace su più triangoli, la posizione ottimale per v_0'' sarà una media delle posizioni desiderate per ogni triangolo in cui appare.

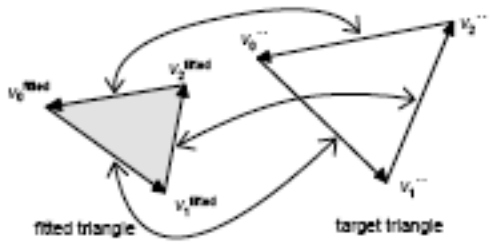


Figura 4.4: Metrica dell'errore usata nella fase due: misura la differenza tra i vettori dei lati del triangolo adattato e quelli del triangolo che si vuole ottenere.

L'errore complessivo della mesh in forma matriciale risulta pari a:

$$E_{2\{v''\}} = \mathbf{v}''^T \mathbf{H} \mathbf{v}'' + \mathbf{f}\mathbf{v}'' + c \tag{4.13}$$

Osserviamo che $\mathbf{H}$ è definito attraverso la connettività della mesh originale ed è indipendente dai triangoli adattati, mentre $\mathbf{f}$ e c sono determinati dai triangoli adattati e per questo cambiano durante l'interazione.

Minimizziamo E_2 ponendo le derivate parziali di E_2 sui vertici liberi $\mathbf{u}$ pari a zero. Per riscrivere $\mathbf{v}''$, poniamo $\mathbf{v}''^T = (\mathbf{u}^T \mathbf{q}^T)$, e quindi:

$$\mathbf{E_2} = \mathbf{v}''^T \mathbf{H} \mathbf{v}'' + \mathbf{f} \mathbf{v}'' + c = \begin{pmatrix} \mathbf{u} \\ \mathbf{q} \end{pmatrix}^T \begin{pmatrix} \mathbf{T}_{00} & \mathbf{T}_{01} \\ \mathbf{T}_{10} & \mathbf{T}_{11} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{q} \end{pmatrix} + (\mathbf{f}_0 \mathbf{f}_1) \begin{pmatrix} \mathbf{u} \\ \mathbf{q} \end{pmatrix} + c \tag{4.14}$$

$$\frac{\delta E_2}{\delta \mathbf{u}} = (\mathbf{H}_{00} + \mathbf{H}_{00}^T)\mathbf{u} + (\mathbf{H}_{01} + \mathbf{H}_{10}^T)\mathbf{q} + \mathbf{f}_0 = 0 \tag{4.15}$$

Riscriviamo la (4.15) così:

$$\mathbf{H}' \mathbf{u} + \mathbf{D}\mathbf{q} + \mathbf{f}_0 = 0 \tag{4.16}$$

$\mathbf{H}'$ e $\mathbf{D}$ sono fissati ma $\mathbf{q}$ e $\mathbf{f}_0$ cambiano durante la manipolazione. Calcoliamo $\mathbf{H}'$ attraverso la fattorizzazione LU all'inizio, poi risolviamo l'equazione durante l'interazione utilizzando il risultato ottenuto.

4.3 Estensioni

Sono necessari alcuni aggiustamenti per far sì che il sistema dia i risultati migliori. Un problema ad esempio si presenta con la sovrapposizione fra più parti dell'immagine, come è ben visibile nella figura (4.5)

Figura 4.5: Figura a riposo e figura manipolata prima senza attribuire profondità, poi attribuendo profondità.

Per evitare ciò è necessario assegnare appropriate profondità alle varie parti che si sovrappongono, tuttavia non è possibile assegnare valori di profondità statica consistenti che vadano bene per tutte le possibili deformazioni. Ciò che possiamo fare è aggiustare dinamicamente le profondità durante l'interazione, monitorando continuamente la mesh e assegnando valori di profondità adeguati alle parti che si sovrappongono. Individuiamo

innanzi tutto le intersezioni dei bordi, poi cerchiamo le regioni che si sovrappongono partendo dalle intersezioni sui bordi che abbiamo trovato. Rimane ancora difficile determinare l'ordine di profondità di regioni che si sovrappongono, il metodo che si utilizza al momento assegna un ordine statistico predefinito.

La figura 4.6 è un chiaro esempio della limitazione di assegnazioni di profondità statiche: supponiamo di avere dei valori di profondità continui sull'immagine; dati tre punti *a,b,c* assumiamo che le loro profondità seguano l'ordine $prof(a) \leq prof(b) \leq prof(c)$. Se portiamo il vertice b tra a e c con una deformazione, deve esistere un limite, attraverso il quale un vertice è più profondo di b ed un altro meno. Questo produce la spiacevole conseguenza visibile nella figura. Per controllare la rigidità delle sagome è importante

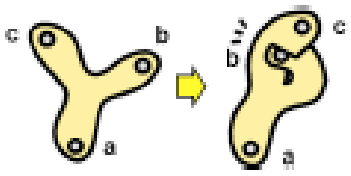


Figura 4.6: Limitazioni dell'attribuzione statica di profondità.

assegnare pesi diversi alle diverse parti della mesh. In una interfaccia di disegno si possono rendere alcune zone più rigide di altre. È possibile ottenere una perfetta rigidità anche riducendo il numero di variabili libere nella minimizzazione, ma il metodo dei pesi produce risultati più soddisfacenti in casi di distorsioni estreme.

Figura 4.7: Immagine a riposo, immagine senza attribuzione di peso, immagine con attribuzione di peso. Aggiungendo un peso extra ad alcune regioni, si possono prevenire distorsioni indesiderate.

Un'importante estensione di questa applicazione è quella che si ottiene utilizzando uno SmartSkin touchpad [Rekimoto, 2002] a punti multipli come nel caso delle figure sottostanti:

Un'analoga applicazione di questa particolare funzione è stata esposta nel febbraio 2006 da Jeef Han, consulente del Dipartimento di Computer Science dell'Università di New York, a Monterey, in California, durante l'annuale conferenza TED, tecnology,

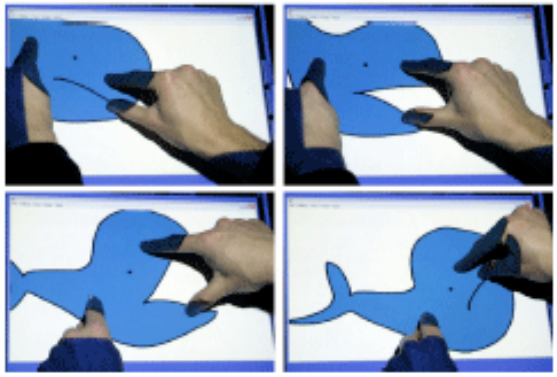


Figura 4.8: SmartSkin touchpad a punti multipli: si può deformare l'intera figura come se si stesse manipolando un oggetto reale.

Figura 4.9: Possono essere usate diverse dita per deformare la figura e creare piccole animazioni.

entertainment, design [TED]. L'intervento di Han inizia con un'immagine simile ad una colata di lava virtuale su uno schermo multitouch ad alta risoluzione. Più che sulla tecnologia del sistema, Han si sofferma su ciò che con questo sistema si può fare: le bolle di lava possono essere fuse o separate solo guidando con le dita le loro immagini sullo schermo fino a farle incontrare o scindere, a seconda dei casi, in modo totalmente intuitivo. Lo stesso vale per un tavolo virtuale in cui vengono sfogliate, spostate, ruotate ed ingrandite delle foto come fossero materiale cartaceo, cosa che un anno più tardi farà anche la Apple con l'Apple iPhone, ma con la possibilità di utilizzare solo due dita [Apple iPhone]. Sullo schermo viene poi visualizzata una tastiera che può essere ingrandita o ristretta secondo le nostre esigenze. In un'immagine del globo terrestre visto dal satellite Han procede con lo stesso metodo: trascinando le dita sullo schermo raggiunge il livello della terra ed in essa naviga e senza alcuna interfaccia cambia punto di vista, come muovendo un oggetto 3D. Nell'ultima applicazione mostrata da Han viene disegnata con due dita un'immagine sullo schermo, vengono poste delle manopole in modo analogo a quello descritto in questo capitolo, e viene manipolata l'immagine a piacere.

4.4 Conclusioni

L'algoritmo introdotto in questo capitolo funziona piuttosto bene nella maggior parte dei casi, ma in altri vengono a galla le sue limitazioni. Ad esempio i vertici liberi delle figure si muovono solo parallelamente sulla linea che collega i vertici vincolati, e questo crea non pochi problemi, come si vede bene nell'immagine

Figura 4.10: Una limitazione dell'algoritmo: se si allunga la figura (immagine a sinistra) l'algoritmo ci dà il risultato dell'immagine centrale. L'immagine a destra é il risultato che vorremmo ottenere.

Una mappatura da un triangolo originale ad un triangolo deformato in cui si ignori la traslazione può essere rappresentata da una matrice di trasformazione affine 2×2 [Sernesi, 2000]. Utilizzando un valore singolare di decomposizione, la matrice A può

essere rappresentata come una combinazione di una parte di rotazione R_γ, una parte di sezionamento s_h e una parte di riduzione in scala s_x, s_y [Shoemake et al., 1992]:

$$A = R_\alpha D R_\beta = (R_\alpha R_\beta)(R_\beta^T D R_\beta) = R_\gamma \begin{pmatrix} s_x & s_h \\ s_h & s_y \end{pmatrix} \tag{4.17}$$

Data tale formula, si può ottenere una mappatura il più rigida possibile minimizzando $\|s_h\|$, $\|s_x - 1\|$ e $\|s_y - 1\|$. Trovare un modo per minimizzare questo errore direttamente rappresenta uno degli obiettivi futuri di questi studi.

Altro obiettivo riguarda il mantenimento del volume: un oggetto si allunga verticalmente quando viene ristretto orizzontalmente in modo tale da mantenere costante il suo volume.

Si vorrebbe infine estendere l'applicazione al 3D ma la realizzazione risulta ancora difficoltosa.

Figura 4.11: Manipolazione di un'immagine in cui si controllano i punti finali.

Figura 4.12: Manipolazione di un'immagine in cui si controllano i punti interni.

Figura 4.13: Restrizione e allargamento di una forma senza una struttura articolata.

Capitolo 5

Riconoscimento di volti basato sull'adattamento di un morphable model 3D

5.1 Descrizione dell'applicazione

In questo capitolo ciò che vogliamo fare è creare un algoritmo capace di riconoscere un volto rappresentato in un'immagine bidimensionale. Non tutte le immagini facciali sono uguali ma differiscono per parametri intrinseci ed estrinseci: con il termine "intrinseci" intendiamo le caratteristiche proprie del volto come forma e tessitura della superficie facciale, per "estrinseci" invece intendiamo, ad esempio, la geometria della scena e dell'immagine, la direzione e l'intensità dell'illuminazione. Adattare un *morphable model* 3D ad un'immagine è un modo per ottenere la completa separazione tra forma e orientamento: il sitema che studiamo combina un morphable model 3D con una simulazione di proiezione e illuminazione fatta attraverso la computer grafica [Blanz et al., 2003].

Questo rende forma e tessitura completamente indipendenti dai parametri estrinseci, per cui la rotazione del volto ed i cambi di illuminazione non rappresentano più un problema. Il processo è lungo ed articolato: abbiamo un database di scansioni facciali 3D, una galleria di immagini note al sistema e un'immagine prova.

Quel che l'algoritmo fa è:

1. costruire un volto medio

   si analizzano le scansioni facciali 3D del database in modo tale da catturare le

caratteristiche dei visi ed attraverso queste costruire un morphable model che rappresenti forme e tessiture dei volti come vettori in uno spazio facciale alto dimensionale. Si coinvolge l'uso della funzione di densità di probabilità delle facce naturali nello spazio delle facce;

2. comparare

   si adatta il modello costruito alle immagini della galleria e si compara ogni singola immagine con il volto medio per determinare i coefficienti che rappresentano la forma intrinseca e la tessitura delle facce della galleria;

3. identificare

   data un'immagine prova, l'algoritmo di adattamento ne calcola i coefficienti che vengono quindi comparati con tutti i dati relativi ai volti della galleria per trovare il dato che fra questi più si avvicina a quello dell'immagine prova.

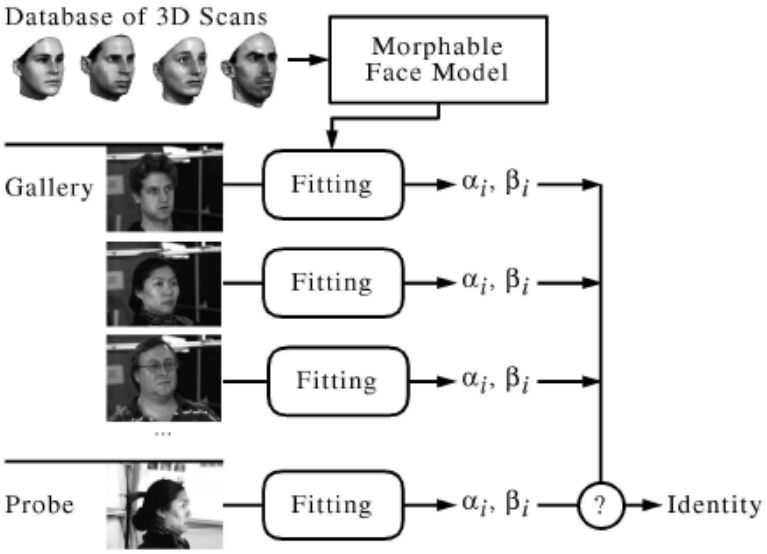


Figura 5.1: Dedotto da scansioni laser, il morphable model 3D viene utilizzato per decodificare immagini della galleria o immagini prova.

5.2 Costruzione del morphable model

5.2.1 Costruzione dei vettori di riferimento

Il morphable model del volto è basato su una rappresentazione facciale su uno spazio vettoriale. Tale spazio vettoriale viene costruito in modo che ogni combinazione convessa di vettori di forma e tessitura $\mathbf{S}_i$ e $\mathbf{T}_i$ di un insieme di esempi descriva una realistica faccia umana:

$$\mathbf{S} = \sum_{i=1}^{m} a_i \mathbf{S}_i \qquad , \qquad \mathbf{T} = \sum_{i=1}^{m} b_i \mathbf{T}_i. \tag{5.1}$$

Continui cambiamenti del valore del parametro a_i generano una transizione liscia tale che ogni punto della superficie iniziale si muova verso un punto della superficie finale. Le corrispondenze dense punto per punto sono quindi cruciali per definire i vettori di forma e tessitura.

Il morphable model viene dedotto da scansioni 3D di 100 uomini e 100 donne, i cui volti vengono rappresentati in coordinate cilindriche relative ad un asse verticale centrato rispetto alla testa. La forma viene descritta attraverso l'angolo ϕ che copre i 360° in 12 passi angolari, l'altezza h è costruita con 512 passi verticali ad uno spazio 0,615 mm l'uno dall'altro, il congegno misura raggio r. I dati della tessitura sono R, G, B, red, green, blue.

$$\mathbf{I}(h,\phi) = (r(h,\phi), R(h,\phi), G(h,\phi), B(h,\phi))^T, h, \phi \in \{0,\dots,511\} \tag{5.2}$$

in (5.2) combiniamo i dati di raggio e tessitura.

Analizziamo ora la corrispondenza punto per punto tra ogni faccia e la faccia di riferimento. La corrispondenza viene data da un campo vettoriale denso $\mathbf{v}(h,\phi) = (\Delta h(h,\phi), \Delta\phi(h,\phi))^T$ in modo tale che ogni punto $\mathbf{I}_1(h,\phi)$ nella prima scansione corrisponda al punto $\mathbf{I}_2(h+\Delta h, \phi+\Delta\phi)$ nella seconda scansione. Applichiamo un algoritmo modificato di flusso ottico per determinare questo campo vettoriale. Consideriamo quindi nel volto di riferimento I_0 i vertici $k \in \{1,\dots,n\}$ dedotti dalla scansione laser, collocati nei rispettivi punti $(h_k, \phi_k, r(h_k,\phi_k))$ in coordinate cilindriche e in (x_k, y_k, z_k) in coordinate cartesiane, di colori (R_k, G_k, B_k). I vettori di riferimento di forma e tessitura sono quindi definiti da:

$$\mathbf{S}_0 = (x_1, y_1, z_1, x_2, y_2, z_2, \dots, x_n, y_n, z_n)^T, \mathbf{T}_0 = (R_1, G_1, B_1, R_2, G_2, B_2, \dots, R_n, G_n, B_n)^T, \tag{5.3}$$

Allora per decodificare una nuova scansione $\mathbf{I}$ calcoliamo il campo di flusso da $\mathbf{I}_0$ a $\mathbf{I}$ e convertiamo $\mathbf{I}(h', \phi')$ nelle coordinate cartesiane x(h', ϕ'), y(h', ϕ'), z(h', ϕ'). Le coordinate (x_k, y_k, z_k) ed i valori dei colori (R_k, G_k, B_k) per la forma e la tessitura $\mathbf{S}$ e $\mathbf{T}$ sono allora presi come campione nella formula
$h'_k = h_k + \Delta h(h_k, \phi_k), \phi'_k = \phi_k + v_\phi(h_k, \phi_k)$.

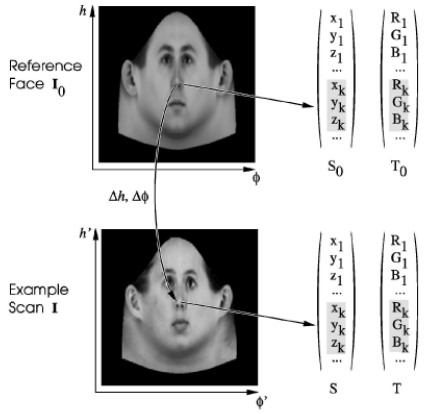


Figura 5.2: Per le scansioni 3D parametrizzate con coordinate cilindriche (h, ϕ), il flusso di campo, che mette in relazione ogni punto del volto di riferimento (in alto) con il corrispondente punto del volto di esempio (in basso), viene usato per formare forma e tessitura dei vettori $\mathbf{S}$ e $\mathbf{T}$.

5.2.2 Analisi della componente principale

Eseguiamo l'analisi della componente principale sui vettori di forma e tessitura $\mathbf{S}_i$ e $\mathbf{T}_i$ dell'insieme di esempi di volti i=1,...,m. Analizziamo forma e tessitura separatamente. Per la forma sottraiamo la media $\overline{\mathbf{s}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{S}_i$ da ogni vettore di forma, $\mathbf{a}_i = \mathbf{S}_i - \overline{\mathbf{s}}$, e definiamo una matrice $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m)$. Il passo più importante che tale analisi ci permette di fare è calcolare gli autovettori $\mathbf{s}_1, \mathbf{s}_2, \ldots$ della matrice di covarianza $\mathbf{C} = \frac{1}{m}\mathbf{A}\mathbf{A}^T$ che si può ottenere da una scomposizione di valori singolari di $\mathbf{A}$ [Press et al., 1992].Gli autovalori di $\mathbf{C}$, $\sigma^2_{S,1} \geq \sigma^2_{S,2} \geq \ldots$ sono le varianze dei dati rispetto ad ogni autovettore.
Con la stessa procedura otteniamo gli autovettori della tessitura t_i e le varianze $\sigma^2_{T,i}$. Gli autovettori formano una base ortogonale

$$\mathbf{S} = \overline{s} + \sum_{i=1}^{m-1} \alpha_i \cdot \mathbf{s}_i, \qquad \mathbf{T} = \overline{t} + \sum_{i=1}^{m-1} \beta_i \cdot \mathbf{t}_i \tag{5.4}$$

e l'analisi della componente principale [PCA] fornisce una stima della densità di probabilità all'interno dello spazio delle facce:

$$p_S(\mathbf{S}) \sim \exp^{-\frac{1}{2}\sum \frac{\alpha_i^2}{i\sigma_{S,i}^2}} \tag{5.5}$$

$$p_T(\mathbf{T}) \sim \exp^{-\frac{1}{2}\sum \frac{\beta_i^2}{i\sigma_{T,i}^2}} \tag{5.6}$$

5.3 Adattare il modello ad un'immagine

5.3.1 Analisi dell'immagine basata sul modello

Il punto centrale dell'analisi dell'immagine basata sul modello è quello di rappresentare un volto nuovo attraverso i coefficienti α_i e β_i che compaiono nella formula (5.4) e fornire una ricostruzione della forma 3D. Inoltre, tale analisi calcola tutti i parametri rilevanti della scena 3D, come la posa, la focale della lente della camera, l'intensità di luce, il colore, la direzione. L'algoritmo deve arrivare a produrre un'immagine il più simile possibile all'immagine in input $\mathbf{I}_{input}$. L'ottimizzazione iterata inizia dal "volto medio" e dalle condizioni standard di rendering. All'inizio il sistema richiede coordinate di circa sette punti caratteristici facciali, come il contorno degli occhi o la punta del naso. Attraverso uno strumento interattivo si possono definire tali punti $j = 1, \ldots, 7$ cliccando alternativamente su un punto della testa di riferimento per selezionare il vertice k_j e sul corrispondente punto $q_{x,j}, q_{y,j}$ dell'immagine. Durante la procedura di adattamento, l'algoritmo determina potenziali punti di contorno basati sull'angolo formato tra la superficie normale e la direzione dello sguardo e segna come k_j il punto di contorno più vicino in ogni iterazione.

5.3.2 Posizione dei vertici e illuminazione

Le posizioni in 3D dei vertici del modello ed il loro colore sono dati dai coefficienti α_i e β_i e dall'equazione (5.4). Una trasformazione rigida traccia le coordinate centrate di ogni vertice k, $\mathbf{x}_k = (x_k, y_k, z_k)^T$, in una posizione relativa alla camera:
$(w_{x,k}, w_{y,k}, w_{z,k})^T = \mathbf{R}_\gamma \mathbf{R}_\theta \mathbf{R}_\phi \mathbf{x}_k + \mathbf{t}_\omega$. Gli angoli ϕ e θ controllano le rotazioni in profondità intorno agli assi verticale ed orizzontale, e γ definisce una rotazione lungo l'asse della camera. $\mathbf{t}_\omega$ rappresenta un cambio spaziale. Le coordinate $p_{x,k}, p_{y,k}$ risultano

quindi:

$$p_{x,k} = P_x + \frac{w_{x,k}}{w_{z,k}} \ , \ p_{y,k} = P_y + \frac{w_{y,k}}{w_{z,k}} \tag{5.7}$$

f è la lunghezza focale della camera collocata all'origine, e P_x, P_y definiscono la posizione dell'asse ottico dell'immagine piana.

Per quanto riguarda l'illuminazione, le ombre sulla superficie dipendono dalla direzione della superficie normale $\mathbf{n}$. Il vettore normale al triangolo $k_1k_2k_3$ del viso si ottiene attraverso un prodotto vettoriale tra gli spigoli, $(\mathbf{x}_{k_1} - \mathbf{x}_{k_2})\mathrm{x}(\mathbf{x}_{k_1} - \mathbf{x}_{k_3})$, che viene normalizzato alla lunghezza unitaria e ruotato con la testa. Per adattare il modello all'immagine è sufficiente considerare solo il centro dei triangoli. Sono molte le componenti che contribuiscono all'illuminazione finale del volto, la luce dell'ambiente L_{amb} di intensità rossa, verde e blu, una luce diretta e parallela L_{dir} anch'essa di intensità R, G, B, proveniente da una direzione $\mathbf{l}$ definita da due angoli θ_l e ϕ_l. In ogni vertice k, ciascun canale di intensità (R, G, B) è definito dalla relativa componente di riflessione diffusa situata sul vettore di tessitura $\mathbf{T}$, una riflessione speculare k_s, una distribuzione angolare ν della riflessione speculare e da aggiunte, bilanciamenti e contrasti di colore tali per cui l'illuminazione finale L di un punto colorato risulti

$$L = 0.3L_r + 0.59L_g + 0.11L_b \tag{5.8}$$

Il contrasto di colore interpola tra il valore del colore originale e questa illuminazione, quindi, ad esempio per il canale del rosso, abbiamo

$$I_r = g_r(cL_r + (1-c)L) + o_r \qquad . \tag{5.9}$$

I colori I_r, I_g, I_b vengono situati nella posizione (p_x, p_y) nell'immagine finale $\mathbf{I}_{model}$.

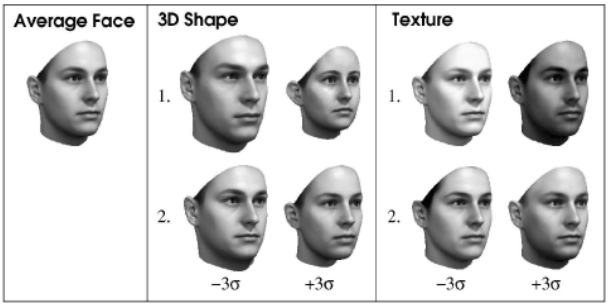


Figura 5.3: Il volto medio e le prime due componenti principali di un insieme di 200 scansioni 3D, visualizzate aggiungendo e togliendo il valore 3σ al volto medio.

5.3.3 Adattamento

L'algoritmo di adattamento ottimizza i coefficienti di forma e tessitura

$$\alpha = (\alpha_1, \alpha_2, \ldots)^T \tag{5.10}$$

$$\beta = (\beta_1, \beta_2, \ldots)^T \tag{5.11}$$

insieme a 22 parametri di rendering, concatenati in un vettore ρ : angoli di assetto ϕ, θ, γ, traslazioni 3D $\mathbf{T}_w$, lente focale f, intensità delle luci dell'ambiente $L_{r,amb}, L_{g,amb}, L_{b,amb}$, intensità della luce diretta $L_{r,dir}, L_{g,dir}, L_{b,dir}$, gli angoli della luce diretta θ_l, ϕ_l, contrasto di colore c, aggiunte e bilanciamenti $g_r, g_g, g_b, o_r, o_g, o_b$.
Data un'immagine input $\mathbf{I}_{input}(x, y) = (I_r(x, y), I_g(x, y), I_b(x, y))^T$ il primo passo è minimizzare la quantità:

$$E_I = \sum_{x,y} \|\mathbf{I}_{input}(x, y) - \mathbf{I}_{model}(x, y)\|^2 \tag{5.12}$$

Le prime iterazioni utilizzano i punti caratteristici definiti manualmente $(q_{x,j}, q_{y,j})$ e le posizioni (p_{x,k_j}, p_{y,k_j}) dei corrispondenti vertici k_j nella funzione

$$E_F = \sum_j \| \begin{pmatrix} q_{x,j} \\ q_{y,j} \end{pmatrix} - \begin{pmatrix} p_{x,k_j} \\ p_{y,k_j} \end{pmatrix} \|^2 . \tag{5.13}$$

Poiché durante la minimizzazione di questa funzione potremmo avere degli effetti di overfitting, ci serviamo di un approssimatore maximum a posteriori: data un'immagine input $\mathbf{I}_{input}$ ed i punti caratteristici F, il lavoro sta nel trovare parametri di modello con una probabilità maximum a posteriori:

$$p(\alpha, \beta, \rho \mid \mathbf{I}_{input}, F) \sim \; p(\mathbf{I}_{input}, F \mid \alpha, \beta, \rho) P(\alpha, \beta, \rho). \tag{5.14}$$

Se sorvoliamo su alcune relazioni tra le variabili, il lato destro diventa:

$$p(\mathbf{I}_{input} \mid \alpha, \beta, \rho) \sim p(F \mid \alpha, \beta, \rho) \cdot P(\alpha) \cdot P(\beta) \cdot P(\rho). \tag{5.15}$$

Le probabilità $P(\alpha)$, $P(\beta)$ vengono stimate attraverso l'analisi della componente principale. Assumiamo che $P(\rho)$ sia una distribuzione normale ed usiamo i valori iniziali per $\overline{\rho_i}$ e valori appropriati per $\sigma_{R,i}$. Per il rumore della gaussiana dei pixel con una deviazione standard σ_I, la probabilità, dati α, β, ρ, di osservare l'immagine $\mathbf{I}_{input}$ è il

prodotto di distribuzioni normali unidimensionali, con una distribuzione per ogni pixel ed ogni canale di colore. Questo si può esprimere come:

$$p(\mathbf{I}_{input} \mid \alpha, \beta, \rho) \sim \exp(\frac{-1}{2\sigma_2^F} \cdot E_I). \tag{5.16}$$

Allo stesso modo le coordinate dei punti caratteristici possono essere soggette a rumore, quindi risulta:

$$p(\mathbf{I}_F \mid \alpha, \beta, \rho) \sim \exp(\frac{-1}{2\sigma_2^F} \cdot E_F). \tag{5.17}$$

Concludiamo quindi dicendo che la probabilità a posteriori è così massimizzata minimizzando la quantità

$$E = -2 \cdot logp(\alpha, \beta, \rho \mid \mathbf{I}_{input}, F) \tag{5.18}$$

$$E = \frac{1}{\sigma_I^2} E_I + \frac{1}{\sigma_F^2} E_F + \sum_i \frac{\alpha_i^2}{\sigma_{S,i}^2} + \sum_i \frac{\beta_i^2}{\sigma_{T,i}^2} + \sum_i \frac{(\rho_i - \overline{\rho_i})}{\sigma_{R,i}^2} \tag{5.19}$$

Il punto principale della procedura di modellazione è minimizzare la funzione (5.19). L'ottimizzazione evita i minimi locali cercando una porzione più ampia di spazio di parametri e riduce i tempi di calcolo. L'algoritmo seleziona un insieme K di 40 triangoli a scelta in ogni iterazione e valuta E_1 ed il suo gradiente solo nel centro

$$E_{I,approx} = \sum_{k \in K} \|\mathbf{I}_{input}(p_{x,k}, p_{y,k}) - \mathbf{I}_{model,k}\|^2 \tag{5.20}$$

Per far si che il valore di aspettazione $E_{I,approx}$ sia uguale a E_I calcoliamo la probabilità di selezionare un particolare triangolo proporzionale a tale area nell'immagine. I coefficienti di tessitura β_i e i parametri di illuminazione influenzano solo i valori di colore di un vertice. I coefficienti di forma α_i influenzano sia le coordinate dell'immagine $(p_{x,k}, p_{y,k})$ sia i valori di colore $\mathbf{I}_{model,k}$. La prima iterazione ottimizza solo i primi parametri α_i, β_i, i $\in 1, \ldots, 10$ e tutti i parametri ρ_i. Le iterazioni successive considerano ancora altri coefficienti. Dalle componenti principali in un database di 200 facce, usiamo solo i 99 coefficienti rilevanti α_i, β_i. Dopo aver adattato l'intero modello facciale all'immagine, il naso, gli occhi, la bocca e le regioni circostanti vengono ottimizzate separatamente. Il processo di adattamento richiede 4.5 minuti in una postazione con un processore Pentium di 2GHz.

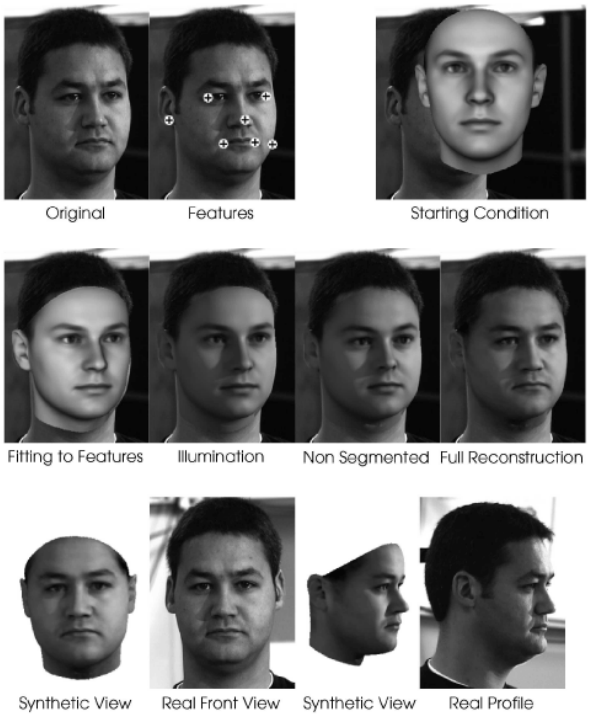


Figura 5.4: Ricostruzione di un volto partendo da una singola immagine (in alto a sinistra) e da un insieme di punti caratteristici (in alto, al centro). Partendo da posizione e illuminazione standard (in alto, a destra), l'algoritmo calcola una trasformazione rigida. Poi viene stimata l'illuminazione. Forma, tessitura, trasformazione e illuminazione vengono quindi ottimizzati nell'intero volto e rifiniti in ogni segmento (seconda fila). Partendo dal volto ricostruito, si possono creare nuove facce.

5.4 Conclusioni

L'algoritmo, testato su due database di immagini che coprono un'ampia gamma di variazioni di posa e illuminazione, raggiunge promettenti risultati: rispettivamente 95.0 e 95.9 percento di identificazioni corrette. Questo indica che il morphable model 3D è un'ampia e versatile rappresentazione per i volti umani. È immediato estendere tale modello a differenti età, gruppi etnici, espressioni facciali, includendo vettori facciali ottenuti da diverse scansioni 3D. Tale sistema al momento ignora occhiali, barba o ciocche di capelli che coprono una parte del viso. Ottimizzare questi effetti e ridurre i tempi di adattamento del modello all'immagine è il prossimo traguardo nello studio dell'algoritmo di processi di costruzione e identificazione 3D.

Conclusione

In questa tesi si è potuto vedere come la matematica sia un prezioso supporto per la fantasia di informatici geniali che hanno dotato il concetto di immagini di valenze fino ad ora impensabili. Modelli artificiali di volti permettono sia l'animazione di fotografia, sia gli enormi progressi nel riconoscimento di persone; un funzionale energia permette l'eliminazione di spazi vuoti da fotografie; i classici gruppi di trasformazioni del piano permettono manipolazioni che fino a pochi anni fa comparivano solo in film di fantascienza. Un'ultima osservazione: senz'altro potente ed efficace, ma la matematica vista qui è "vecchia" di parecchi decenni. Che miracoli tecnici potremmo aspettarci dall'uso della matematica "moderna"?

Bibliografia

[Apple iPhone] Informazioni relative al nuovo sistema iPhone della Apple sono disponibili all'indirizzo Internet http://www.apple.com/iphone/

[Avidan et al., 2003] Avidan S., Shamir A., "Seam Carving for Content Aware Image Resizing", informazioni relative agli studi di Shai Avidan e Ariel Shamir sono reperibili al sito: http://blogs.ugidotnet.org/PuntoRete/archive/2007/09/03/interessante-tecnologia-per-image-resizing.aspx
.

[Blanz et al., 2003] Blanz V.,Vetter T., "Face Recognition Based on Fitting a 3D Morphable Model", 2003, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no.9.

[De Berg et al., 1997] M. De Berg, M. van Kreveld, M. Overmars, O.Schwarzkopf, "ComputationalGeometry - Algorithms and Applications", 1997, Springer Verlag.

[PCA] Duda R. O., Hart P. E., Stork D. G., "Pattern Classification" seconda edizione, 2001, John Wiley & Sons.

[Digital Library Project] Il sito del Digital Library Project dell'Università della California a Berkeley si trova all'indirizzo Internet http://elib.cs.berkeley.edu

[QIBC] Flikner M. e altri, "Query by Image and Video Content-The QBIC System", 1995, Computer, 28, no. 9.

[Forsyth et al., 1997] Forsyth D., Malik J., Wilensky R., "In cerca di immagini digitali", 1997, Le Scienze, 348, 62–67.

[Motion Portrait] Informazioni relative all'applicazione "Motion Portrait" sono disponibili al sito: http://www.motionportrait.com/about/TIdog.swf

[Perez et al., 2003] Perez P., Gangnet M., Blake A., "Poisson Image Editing", 2003, ACM Trans. Graph. 22, 3, 313–318.

[Poggio, 1984] Poggio T.,"La visione nell'uomo e nella macchina", 1984, Le Scienze, 190, 68–80.

[Press et al., 1992] Press W. H., Teukolsky S. A.,Vetterling W. T., Flannery B. P., "Numerical Recipes in C", 1992, Cambridge University.

[Rekimoto, 2002] Rekimoto J., "SmartSkin: An infrastucture for freehand manipulation on Interactive Surfaces", 2002, Proceedings of CHI '02, 113–120.

[Sernesi, 2000] Sernesi E., "Geometria I", 2000, Bollati Boringhieri.

[Shoemake et al., 1992] Shoemake K. e Duff T., "Matrix Animation and Polar Decomposition", 1992, Proceeding of Graphics Interfaces '92, 258–254.

[Takeo Igarashi] http://www-ui.is.s.u-tokyo.ac.jp/ takeo/research/rigid/index.html

[TED] Informazioni relative alla conferenza presentata da Jeff Han si trovano presso il sito Internet http://it.youtube.com/watch?v=PLhMVNdplJ

[Todesco] In formazioni relative al dottor Gian Marco Todesco si trovano presso la Sua homepage all'indirizzo Internet http://www.toonz.com/personal/todesco

[Vince et al., 2006] Vince J.,"Mathematics For Computer Graphics-second edition", 2006, Springer Verlag.