Progetto di laboratorio di Elaborazione delle Immagini e Visione

Corso di Laurea in Informatica
Falcoltà di Scienze MM.FF.NN. dell'Università di Verona

Rappresentazione Stratificata di una Sequenza Video basata su Mosaici

Aprile Michele
Marzotto Roberto

Prof. Vittorio Murino
Dott. Andrea Fusiello

1 INTRODUZIONE

Molti nuovi algoritmi e standard sono stati sviluppati negli ultimi anni per la codifica video. Tuttavia molte di queste tecniche, ad esempio MPEG, riducono la loro efficacia quando il moto della telecamera diviene rilevante. Un approccio alla compressione video basato su mosaici, inizialmente introdotto da Irani [1], costituisce in questi casi, una possibile alternativa.

L'obiettivo di questo lavoro è la realizzazione di una applicazione in grado di fornire una rappresentazione efficiente di un video shot come mosaico statico del background più la sequenza degli oggetti in movimento in foreground. Un *video shot* è definito come una sequenza di immagini acquisite con una singola operazione della telecamera con continuità di azione nello spazio e nel tempo. In questo caso il vincolo forte è quello di essere in presenza di una scena planare o di moto puramente rotazionale della telecamera.

Il primo passo è quello di costruire il mosaico dello sfondo stimando il moto dominante grazie a feature-tracking lungo la sequenza con una tecnica robusta che scarta le feature attaccate agli oggetti in movimento. Tali oggetti risulteranno rimossi nel mosaico finale utilizzando la mediana come operatore di blending.

Successivamente avviene la segmentazione operata calcolando la differenza pixel a pixel tra i frame della sequenza originale e il mosaico statico e sogliando il risultato in modo automatico. Sono stati proposti altri approcci a questa fase: in [2] il moto era calcolato ad ogni pixel con una tecnica robusta e gli outliers corrispondevano ad oggetti in movimento; lo stesso obiettivo era raggiunto in [3] dall'analisi temporale dei livelli di grigio, basata su modelli probabilistici ed informazione a priori; [4] usava un'analisi del disallineamento locale basato sul flusso normale tra un mosaico dinamico e la sequenza originale. Nel nostro caso, dal momento che il mosaico è privo di oggetti in movimento, una semplice differenza pixel a pixel tra il mosaico e i frame della sequenza produce un buon risultato.

Allo scopo di filtrarle dal rumore e ottenere regioni più compatte, le mappe binarie ottenute verranno ulteriormente processate e verrà tenuto conto della coerenza temporale effettuando il tracking degli oggetti in movimento lungo la sequenza.

Il metodo descritto è stato proposto a completamento del lavoro illustrato in [5] permettendo di gestire casi con oggetti multipli entranti e uscenti, occlusioni tra oggetti e con elementi statici della scena ed è perfettamente adattabile allo standard MPEG-4 per il video coding.

2 NOTE INTRODUTTIVE

Mosaico

Un *mosaico* è il risultato dell'allineamento automatico di più immagini rispetto ad un sistema di riferimento comune. Tipicamente in questo particolare tipo di applicazione si utilizza il mosaico panoramico, ottenuto da viste prese da una telecamera che ruota attorno al suo centro ottico (*panning*). L'immagine risultante è una vista globale virtuale equivalente ad una ottenuta con una inquadratura della scena da una telecamera reale lasciando invariata la posizione del centro ottico, diminuendo la distanza focale, ma senza perdita di risoluzione.

La creazione del mosaico è possibile solamente in situazioni particolari che consentono una semplificazione del modello geometrico utilizzato ovvero quando due viste della stessa scena sono legate da una omografia.

Omografia

Un'*omografia* è una trasformazione lineare non singolare del piano proiettivo in se stesso. In generale è rappresentata da una matrice H 3x3 non singolare:

$$\lambda \begin{bmatrix} x'_1 \\ x'_2 \\ 1 \end{bmatrix} = \begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,1} & H_{3,2} & H_{3,3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$$

I punti sono espressi in coordinate omogenee; il punto 2-D nel piano immagine denotato con $(x_1 \ x_2 \ x_3)$ corrisponde in coordinate cartesiane a $(x_1/x_3 \ x_2/x_3)$. La matrice H è definita a meno di un fattore di scala (8 gradi di libertà). Si noti che la trasformazione è lineare in coordinate proiettive, ma non in coordinate cartesiane:

$$\begin{cases} x'_1 = \frac{H_{1,1}x_1 + H_{1,2}x_2 + H_{1,3}}{H_{3,1}x_1 + H_{3,2}x_2 + H_{3,3}} \\ \\ x'_2 = \frac{H_{2,1}x_1 + H_{2,2}x_2 + H_{2,3}}{H_{3,1}x_1 + H_{3,2}x_2 + H_{3,3}} \end{cases}$$

Due immagini prese da una telecamera in movimento sono legate da una omografia se:

- il punto di vista non cambia, ovvero la telecamera ruota attorno al suo centro ottico;
- la scena è approssimabile con un piano (prospettiva debole).

Si può vedere che due punti $\tilde{m}$ e $\tilde{m}'$, proiezioni del punto 3-D w rispettivamente sulla prima e sulla seconda telecamera, sono in relazione come segue:

$$k'\tilde{m}' = kA'RA^{-1}\tilde{m} + A't$$

Nel primo caso $t = 0$ e si avrà:

$$k'\tilde{m}' = kA'RA^{-1}\tilde{m}$$

$H_\infty = A'RA^{-1}$ è un'omografia e il campo di moto non dipende dalla struttura 3-D della scena.

Nel secondo caso i punti w giacciono tutti sul piano Π definito da $n^T w = d$; l'equazione generale si trasforma in:

$$\frac{k'}{k}\tilde{m}' = \left(H_\infty + \frac{A'tn^TA^{-1}}{d}\right)\tilde{m}$$

Esiste una trasformazione del piano proiettivo tra le due viste indotta dal piano Π data da $H_\Pi = H_\infty + A't\frac{n^T}{d}A^{-1}$. Nel particolare caso in cui $d \to \infty$ H_∞ è l'omografia indotta dal piano infinito.

Nel caso generale (moto arbitrario della telecamera con scena 3-D), la relazione tra due viste può essere modellata con un'omografia più un termine di parallasse dipendente dalla struttura della scena. In due immagini quattro corrispondenze di punti, a tre a tre non collineari, determinano un'unica omografia tra le due immagini. Se sono disponibili più di quattro corrispondenze si può impostare un sistema lineare sovradeterminato risolvibile con il metodo ad esempio dei minimi quadrati.

3 COSTRUZIONE DEL MOSAICO

La costruzione del mosaico avviene attraverso tre fasi: la stima del moto lungo la sequenza, la registrazione rispetto ad un riferimento comune e il rendering finale del mosaico.

3.1 Stima del moto dominante

Il *moto dominante* è il moto della maggior parte dei punti di una sequenza. Assumendo che i punti appartenenti ad oggetti in movimento siano una minoranza, esso coincide con il moto dei punti realmente statici nella scena, ma apparentemente mobili nella sequenza a causa del movimento della telecamera. La conoscenza del moto dominante permetterà quindi di compensare tale movimento al fine di allineare i frame rispetto ad un riferimento fisso. La rimanente minoranza dei punti sarà identificabile come appartenente ad oggetti in movimento sulla scena.

La stima del moto dominante si realizza attraverso il calcolo di un campo di moto 2D sparso utilizzando il *features-tracker* realizzato da Kanade, Lucas e Tomasi (di cui si possono trovare informazioni all'indirizzo *http://vision.stanford.edu/~birch/klt/*) che, per ogni coppia di frame consecutivi, produce un insieme di corrispondenze cui si applica un metodo di regressione per determinare l'omografia che descrive il moto frame-to-frame

dei pixel. L'insieme di tutte le omografie costituisce il modello di moto 2D globale della sequenza.

Si è adottata la tecnica di regressione robusta *Least Median of Squares* (LMedS) poiché la presenza di oggetti in movimento sulla scena richiede che i pixel appartenenti ad essi vengano trattati come *outlier* ovvero come punti da ignorare nella stima del modello perché si discostano da esso in maniera troppo marcata. Si tratta di un procedimento a raffinamenti succesivi che ad ogni passo estrae casualmente dall'insieme di corrispondenze una quadrupla che permette di determinare univocamente una omografia. Applicando poi questa omografia a tutte le feature della prima immagine si potrà valutare la bontà del fit calcolando la mediana dei quadrati dei residui definiti come la distanza tra i punti trasformati e quelli corrispondenti nella seconda immagine. Ripetendo il procedimento con altre quattro corrispondenze si valuterà se il nuovo fit è migliore del precedente e in tal caso quest'ultimo verrà sostituito. Il procedimento di regressione avrà termine dopo un certo numero di iterazioni.

3.2 Registrazione

Una volta ottenute le omografie tra coppie di frame adiacenti $H_{i,i+1}$ sarà possibile allineare (*warping*) tutte la immagini rispetto ad un unico sistema di riferimento scelto arbitrariamente. Si è scelto di adottare la modalità *frame-to-frame* nella quale la trasformazione $H_{i,ref}$, con i frames i e ref in generale non contigui, sarà calcolata come prodotto di tutte le trasformazioni intermedie da i fino a ref:

$$H_{i,j} = \prod_{k=i}^{j-1} H_{k,k+1}$$

Scegliendo diversi frame di riferimento si ottengono mosaici differenti, ma questo non influisce sul risultato della segmentazione.

3.3 Rendering del mosaico

L'allineamento di tutti i frame su un comune sistema di riferimento precede il *rendering*, la fase di assegnazione di un preciso colore a ciascun pixel del mosaico. In generale uno stesso punto della scena compare su diverse immagini della sequenza. Il colore di quel punto nel mosaico sarà quindi determinato combinando opportunamente (*blending*) i diversi colori con cui esso appare in ciascun frame. A tal proposito esistono diverse politiche con cui effettuare l'assegnamento: *use-first*, il colore del pixel nel primo frame in cui il punto compare; *use-last*, il colore del pixel nell'ultimo frame in cui il punto compare; *average*, la media dei colori con cui quel pixel appare nei frame della sequenza; *median*, la mediana dei colori con cui quel pixel appare nei frame della sequenza.

Il primi due metodi sono applicabili anche se si attua una costruzione incrementale del mosaico (modalità *on line*). La media ha il vantaggio di rimuovere il rumore non impulsivo, ma essendo sensibile agli outlier non farebbe scomparire gli eventuali oggetti in movimento, che apparirebbero come figure trasparenti sul mosaico.

Figura 1: Esempio di mosaico.

La mediana, oltre a rimuovere il rumore impulsivo, consente di eliminare tutti gli oggetti che stazionano per meno della metà dei frame su una specifica posizione. La scelta dell'operatore mediano impone di calcolare il mosaico in modalità *batch* ovvero è necessario avere a disposizione prima tutti i frame della sequenza.

4 SEGMENTAZIONE

In questa sezione verranno descritte le tecniche utilizzate per estrarre dal video-shot la sequenza di foreground. Il procedimento si basa su *residual motion analysis* ovvero sull'analisi delle immagini-differenza tra i frame originali e quelli sintetici di *background* ricavati dal mosaico una volta compensato il movimento della telecamera.

4.1 Mappe dei residui

Grazie all'omografia $H_{i,ref}$, ottenibile invertendo $H_{ref,i}$, è possibile allineare il mosaico, originalmente espresso nel frame di riferimento ref, rispetto ad uno qualsiasi dei frame della sequenza. Questo permette la ricostruzione del background sintetico di ciascun frame. Calcolando ora, per ogni frame, la differenza pixel a pixel tra i livelli di grigio del frame originale e del frame di background si otterrà una sequenza di matrici che verranno chiamate mappe dei residui (fig.2).

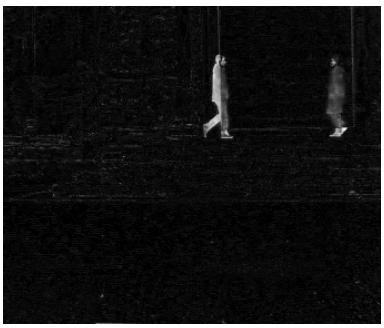

Figura 2: Mappa dei residui.

Tali mappe dovrebbero risultare quasi nulle ovunque eccetto che nelle zone in corrispondenza di oggetti in movimento. In realtà esse possono presentare dei picchi anche altrove a causa di tre principali ragioni:

- disallineamenti nella registrazione del mosaico dovuti ad errori accumulati nel calcolo delle omografie;
- variazione della luminosità lungo la sequenza dovuta ad effetti fotometrici della telecamera conseguenti al cambiamento dell'inclinazione con cui i raggi luminosi incidono sull'obiettivo;
- rumore di acquisizione.

Mentre alla terza causa di disturbo non si può porre rimedio se non dotandosi di strumenti di acquisizione più sofisticati, alle prime due è possibile

contrapporre opportune strategie per limitarne gli effetti. Infatti, utilizzare una tecnica di *registrazione globale* che vada a calcolare le omografie non solo tra frame consecutivi, ma tra ogni coppia di frame dove sia possibile una sovrapposizione di vista sufficiente, permetterebbe di compensare gli errori così da ridurre i disallineamenti [6]. Per quanto riguarda la variazione della luminosità si può intervenire operando una *normalizzazione fotometrica* su tutte le immagini che compongono la sequenza prima della costruzione del mosaico. L'applicazione realizzata prevede questa possibilità anche se va detto che se gli effetti fotometrici sono molto accentuati il problema non può essere eliminato.

4.2 Sogliatura automatica

Una volta ottenute le mappe dei residui viene applicata su di esse una sogliatura allo scopo di segmentare le immagini in regioni che distinguano gli oggetti in movimento dallo sfondo statico. È critica la scelta della soglia poiché da essa dipende fortemente la qualità delle mappe binarie che si otterranno. Si necessita di una soglia che non danneggi la forma dei blob corrispondenti ad oggetti in movimento minimizzando nel contempo la probabilità che vengano prodotti blob indesiderati (spuri) dovuti ai fattori di disturbo prima citati.

Il *Median Absolute Deviation* (MAD) è un operatore che fornisce una misura robusta della dispersione di una distribuzione attorno al suo valore mediano.

$$MAD = median(|X_i - X_{med}|)$$

È possibile applicare l'operatore MAD alla sequenza di frame allo stesso modo con cui veniva impiegato l'operatore di mediana nella fase di rendering del mosaico. Questo ha come risultato una mappa che, per ogni pixel, esprime quanto i livelli di grigio dei frame della sequenza si discostano da quello del mosaico senza dare peso agli outlier. Dall'immagine in fig.3 è evidente che, come avviene per il mosaico, nel MAD non c'è traccia del passaggio di oggetti in movimento. Le zone più intense corrispondono a quelle in cui c'è una forte variabilità dei livelli di grigio lungo gran parte della sequenza ovvero dove le oscillazioni dei livelli di grigio sono "fisiologiche" (con distribuzione gaussiana) e quindi accettabili. È il caso ad esempio dei contorni, delle superfici riflettenti, della vegetazione fitta, etc.

L'idea che si è perseguita è quella di utilizzare il MAD, opportunamente allineato e moltiplicato per un coefficiente di amplificazione, come soglia da applicare alle mappe dei residui in modo da mascherare gli scarti trascurabili e far risaltare gli outlier. Il MAD ha il vantaggio di essere una soglia automatica che si adatta alle varie zone dell'immagine. La mappa binaria

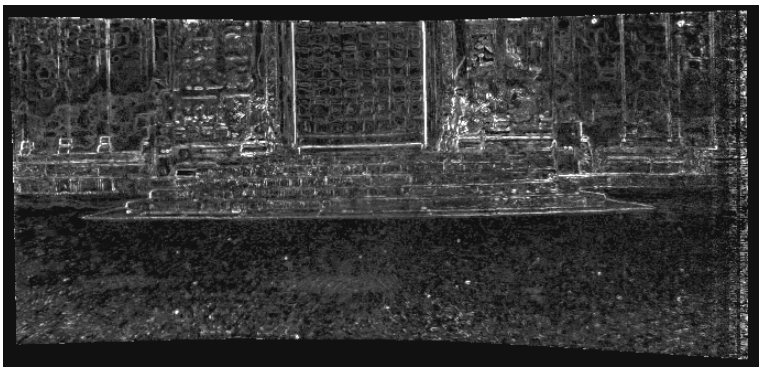

Figura 3: MAD.

i-esima BM_i, sarà calcolata dalla seguente formula:

$$BM_i(x,y) = \begin{cases} 1 & \text{se } \|(F_i(x,y) - BG_i(x,y))\| > T_i(x,y) \\ 0 & \text{altrimenti} \end{cases}$$

dove F_i è il frame i-esimo, BG_i è il background sintetico del frame i-esimo ottenuto dallo warping del mosaico e T_i è la soglia variabile del frame i-esimo ottenuta dallo warping del MAD.

Le mappe binarie così ottenute conterranno i blob dovuti alla presenza di oggetti in movimento sulla scena, ma non è comunque garantita l'assenza di blob spuri. Occorrono pertanto successive fasi di elaborazione per migliorare il risultato della segmentazione.

4.3 Filtraggio

Allo scopo di individuare nelle mappe binarie quali siano i blob relativi ad oggetti in movimento, occorre innanzitutto constatare, sulla base dei test effettuati, alcune proprietà morfologiche ricorrenti, in modo da realizzare un filtraggio mirato alla tipologia di rumore da eliminare:

- i blob riconducibili ad oggetti in movimento sono solitamente quelli di area maggiore. Talvolta un unico oggetto può apparire frammentato in più regioni, avere i contorni frastagliati e buchi al suo interno. Questi effetti indesiderati si verificano in modo particolare quando i livelli di grigio dell'oggetto sono assimilabili a quelli dello sfondo;
- i blob dovuti al disallineamento hanno spesso la forma di piccoli segmenti;
- il rumore di acquisizione ha carattere puntiforme ed è distribuito perlopiù uniformemente su tutta la mappa.

L'utilizzo dell'operatore di *chiusura* renderebbe più compatte le regioni, ma potrebbe provocare l'aggregazione del rumore in alcune zone dell'immagine. Un *size-filter* con una soglia alta, pur prestandosi a rimuovere il rumore, non è appropriato al nostro scopo perché tende ad eliminare anche piccoli dettagli di oggetti in movimento che spesso risultano partizionati dopo la sogliatura. Si è quindi scelto di partire applicando l'operatore morfologico *spur*[1] che serve a ridurre la lunghezza dei segmenti. Data una immagine binaria esso agisce rimuovendo i punti estremi delle linee senza eliminare completamente i piccoli oggetti. L'applicazione ripetuta più volte di questo operatore ha proprio l'effetto di accorciare i blob a forma di segmento fino ad ottenere, al limite, regioni di un pixel. Lo scopo è di eliminare i blob dovuti al disallineamento senza andare ad intaccare gli oggetti in movimento. Conseguentemente viene applicato un size-filter (con parametro di area molto piccolo) che va a rimuovere il rumore di acquisizione e ciò che resta dei blob dovuti al disallineamento. Infine vengono riempiti eventuali buchi all'interno delle regioni con l'operatore *holes*[2]. Nelle figure che seguono vengono mostrati i risultati di queste prime tre fasi di rimozione dei blob spuri applicate alle mappe binarie.

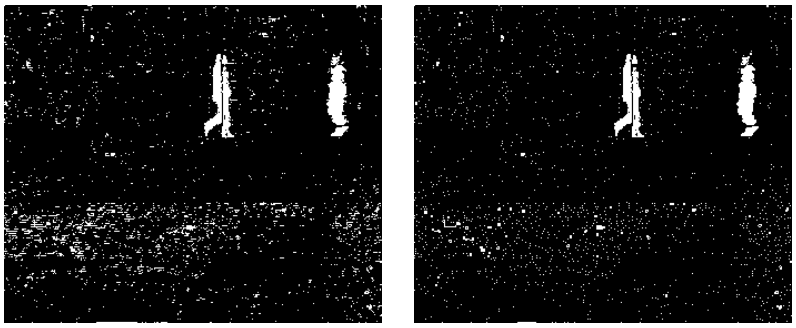

Figura 4: Immagine originale e dopo l'operatore spur.

4.4 Fusione dei blob

Le mappe sono state ripulite dalla gran parte del rumore presente in origine, ma le regioni associabili agli oggetti in movimento, oltre ad apparire disgregate in più blob, presentano in generale contorni frastagliati. Per questo motivo il passo successivo è l'applicazione di un algoritmo di fusione che raggruppi i blob che verosimilmente appartengono ad uno stesso oggetto e li renda un regione unica. Come criterio nella scelta delle regioni da fondere si è adottato quello della prossimità con l'ipotesi che tra tutti i blob in cui

[1] funzione di Matlab bwmorph
[2] funzione di Matlab bwfill

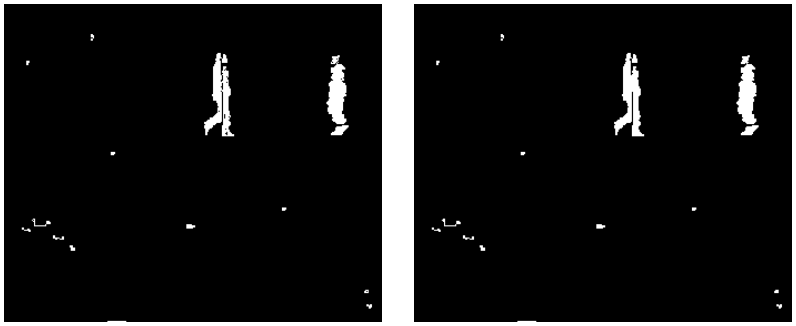

Figura 5: Immagine dopo il size filter e dopo l'operatore holes.

un oggetto è partizionato ve ne sia almeno uno con un'area sufficientemente grande. Di seguito sono riportati sinteticamente i passi di tale algoritmo.

Algoritmo di fusione

Siano B l'insieme di tutti i blob di una mappa binaria e $\sim_p$ la seguente relazione di prossimità tra blob :

$$b_i \sim_p b_j \Leftrightarrow (mindist(b_i, b_j) < T) \vee \exists b_k | (b_i \sim_p b_k) \wedge (b_k \sim_p b_j)$$

1. Selezione dei blob di area maggiore di una certa soglia.

$$S = \{b_i | b_i \in B \wedge Area(b_i) > a\}$$

2. Raggruppamento dei blob prossimi.

$$\forall s_i \in S, N_i = \{b_j | b_j \sim_p s_i\}$$

   Ne deriva che:

$$\forall i, j (N_i = N_j) \vee (Ni \cap Nj = \emptyset)$$

   Pur essendo $\sim_p$ una relazione di equivalenza, il tipo di raggruppamento effettuato non coincide con un partizionamento di B poiché non è sempre vero che $\bigcup N_i = B$.

3. Dilatazione ripetuta dei blob appartenenti a N_i fino ad ottenere un'unica componente connessa C_i. Dato che lo scopo di questa fase è di connettere regioni vicine che ragionevolmente corrispondono ad uno stesso oggetto, la dilatazione viene fatta esclusivamente all'interno dell'inviluppo convesso che contiene i blob di N_i in modo da evitare di espandere eccessivamente le regioni al di fuori del contorno dell'oggetto.

4. Irrobustimento delle connesioni deboli. Dopo il passo 3 è possibile trovare regioni connesse anche solo per un unico pixel. Per evitare casi del genere ed ottenere regioni più compatte si applica ad ogni C_i una operazione di *dilatazione* e una di *chiusura*. Quest'ultima, che consiste in una dilatazione seguita da una erosione, potrebbe da sola essere sufficiente a questo scopo. Sperimentalmente però si è notato come l'aggiunta di una dilatazione, pur comportando il rischio di una lieve espansione delle regioni oltre i confini reali degli oggetti, dia buone garanzie di non perdere pixel appartenenti al loro contorno.

La mappa binaria risultante dall'algoritmo di fusione sarà costituita dall'unione di tutte le componenti connesse C_i. Nelle figure riportate di seguito vengono mostrati i risultati passo-passo dell'applicazione dell'algoritmo alla mappa binaria precedentemente processata. A seguito della selezione (fig.6) si può notare che, per il primo raggruppamento, non è stata effettuata alcuna unificazione poiché c'è un unica regione (fig.7); diversamente, per il secondo, è stato necessario un passo di dilatazione per connettere le due regioni iniziali (fig.8).

Figura 6: Le regioni colorate risultato della selezione.

Figura 7: Dilatazione, irrobustimento e compattamento sul primo raggruppamento.

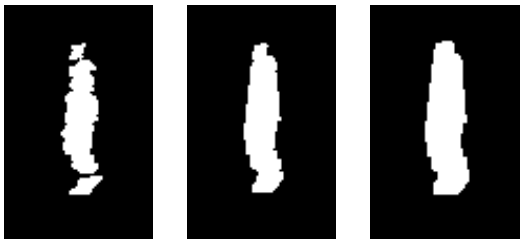

Figura 8: Dilatazione, irrobustimento e compattamento sul secondo raggruppamento.

Figura 9: Mappa binaria risultante.

L'ipotesi forte che si farà di qui in avanti è che ogni oggetto in movimento sia identificabile con un unica regione C_i pur non escludendo la possibilità che alcune di queste regioni non siano associabili a nessun oggetto, ma frutto di cause indesiderate. Per questo motivo sarà necessaria una successiva fase di raffinamento per discriminare i blob corrispondenti agli oggetti in movimento da quelli spuri.

4.5 Tracking degli oggetti in movimento

Mentre fino ad ora si è tentato di identificare i blob corrispondenti agli oggetti in movimento analizzando le mappe scorporate dal contesto della sequenza, l'obiettivo del tracking è discernere tali blob da quelli spuri in forza della coerenza temporale con cui essi preservano alcune loro proprietà morfologiche, cromatiche e di posa.

L'algoritmo ad hoc che si è realizzato a tale scopo prevede di percorrere le mappe binarie selezionando ad ogni passo (*selecting*) i blob con un'area superiore ad una certa soglia che vengono "inseguiti" (*tracking*) lungo la sequenza sulla base della massima similarità. Questa metodologia assume che un oggetto in movimento appaia con un'area consistente in almeno un frame. Se poi nel corso del video-shot la sua area diminuisse gradatamente non smetterebbe comunque di essere inseguito. Questo accorgimente permette

di gestire casi particolari come oggetti che si allontanano dall'osservatore o che progressivamente escono di scena. Viceversa per gestire casi di oggetti che aumentano via via la loro area avvicinandosi all'osservatore o entrando progressivamente nella scena, si è adottata una tecnica di inseguimento all'indietro (*backtracking*) in modo tale che non venga perso il tragitto precedente al loro selezionamento. Quando un nuovo blob viene selezionato si tenterà perciò sia di inseguirlo lungo la sequenza nella direzione del tempo sia in quella opposta. Se entrambi questi tentativi fallissero il blob verrebbe considerato spurio e quindi scartato.

Algoritmo di inseguimento

Sia TB_i l'insieme di blob inseguiti nella mappa B_i al tempo i:

- *selecting* dei blob in BM_1 e loro inserimento in TB_1
- Per i da 2 a N
  - *tracking* dei blob in TB_{i-1} nella mappa BM_i e inserimento in TB_i dei blob inseguiti
  - eliminazione da TB_{i-1} dei blob selezionati e mai inseguiti né in avanti (tracking), né all'indietro (backtracking)
  - *selecting* dei nuovi blob in BM_i e loro inserimento in TB_i
  - *backtracking* dei blob selezionati percorrendo la sequenza di mappe BM_j in senso inverso (per j da i-1 a 1) con eventuale inserimento in TB_j dei blob inseguiti
- eliminazione da TB_N dei blob selezionati e mai inseguiti all'indietro

Al termine di questo procedimento TB conterrà per ogni mappa binaria i blob che considero corrispondenti ad oggetti in movimento. Ciò consentirà, mascherando i frame originali, di ricostruire la sequenza di foreground.

Il tracking si realizza costruendo una matrice nxm che fornisce i coefficienti di similarità tra gli n blob appartenenti a TB_i nel frame F_i e gli m blob del frame successivo.

$$\mathcal{S}_{i,i+1} = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,m} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n,1} & s_{3,2} & \cdots & s_{n,m} \end{pmatrix}$$

L'elemento $s_{h,k}$ indica la similarità tra il blob h e il blob k e sarà calcolato mediante una funzione che verrà descritta in seguito.

Ogni massimo di riga s_{h_m,k_m} corrisponderà ad una associazione tra il blob h_m e il blob k_m (criterio di massima similarità).

Per considerare inseguito un blob nel frame F_{i+1} è necessario che questo sia preferito da uno o più blob nel frame F_i, con una similarità superiore ad una soglia che permette di evitare accoppiamenti forzati, qualora non ci fossero blob sufficientemente simili.

Tra i casi particolari, si vuole gestire anche l'occlusione tra più oggetti in movimento.

Figura 10: F_i i blob sono separati; F_{i+1} i blob si uniscono.

Figura 11: F_{j-1} i blob sono ancora uniti; F_j i blob si separano.

Consideriamo ad esempio l'incrocio di due persone che camminano (fig.10-11) e vediamo come l'algoritmo controlla questo tipo di situazione. Due blob sono separati nel frame F_i, si uniscono in F_{i+1}, rimangono uniti fino a F_{j-1}, per poi distaccarsi nel frame F_j. Almeno uno dei due blob di F_i viene verosimilmente associato al blob di F_{i+1}, il solo presente in quel frame; successivamente fino a F_{j-1} il tracking, con buona probabilità, avverrà sull'unico blob esistente; nell'istante del distacco quest'ultimo sarà associato ad uno dei due blob di F_j, mentre l'altro verrà perso. Il blob non inseguito sarà però recuperato dalla nuova selezione nel frame F_{j+1}.

Le nuove selezioni e il backtracking affiancati al tracking garantisco inoltre la gestibilità delle occlusione di oggetti in movimento con parti statiche della scena e in maniera del tutto analoga anche l'entrata e l'uscita di oggetti.

Questo potrebbe essere il caso di un veicolo che per un tratto della sua corsa viene occluso da un palo (fig.12). L'oggetto, inizialmente intero, transita dietro ad una struttura fissa della scena e viene separato in due blob

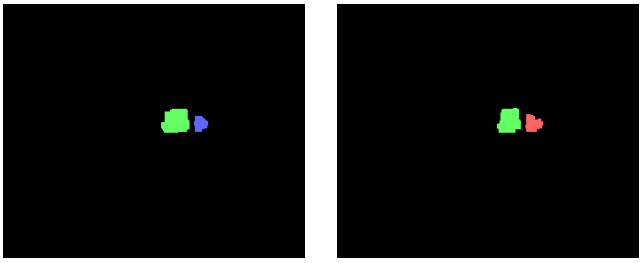

Figura 12: Tracking di un oggetto occluso da una parte statica della scena.

B_1 e B_2. Durante il passaggio l'area di B_1 tenderà a diminuire e quella di B_2 a crescere; pertanto inizialmente c'è il rischio di perdere B_2 per tutti i frame in cui la sua area è inferiore alla soglia stabilita. Il problema viene risolto grazie alla nuova selezione effettuata ad ogni frame, che permette di agganciare B_2 (blob rosso) non appena la sua area sarà superiore alla soglia; il conseguente backtracking, non essendo ad essa vincolato, riuscirà a recuperare i blob scartati nei frame precedenti (blob blu).

Si è visto come, affinché un blob venga considerato oggetto in movimento, sia necessario che la sua area superi una certa soglia per almeno un frame. Tale condizione può però non essere sufficiente in quanto non preclude a blob spuri di essere selezionati. Essi sono caratterizzati dalla loro incoerenza temporale ovvero compaiono da un frame all'altro in posizioni differenti e con caratteristiche morfologiche molto diverse. Per questo motivo si è scelto di eliminare tutti i blob selezionati che non vengono inseguiti (in avanti o all'indietro) per più di un frame (fig.13). Questo metodo funziona bene nella maggior parte dei casi, ma presenta talvolta effetti collaterali indesiderati: può capitare che un oggetto risulti frammentato in due o più blob per un solo frame; essi, anche se recuperati dalla nuova selezione, difficilmente verrebbero associati all'oggetto intero nel frame successivo o in quello precedente e andrebbero quindi eliminati. Tale perdita tuttavia, riguardando un unico frame, può essere considerata accettabile.

Figura 13: F_{i-1},F_i con blob di rumore,F_{i+1}

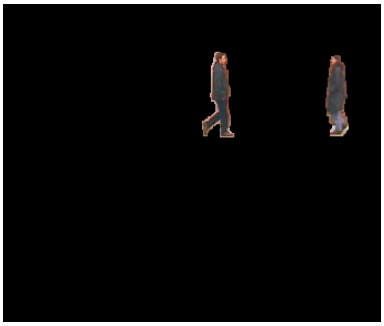

Figura 14: Esempio di immagine di foreground.

Il risultato dell'algoritmo di inseguimento, è l'insieme delle mappe finali che vengono utilizzate come maschere sui frame originali allo scopo di ottenere la sequenza di foreground (fig.14).

A partire dal mosaico, dal modello di moto 2-D globale e dalla sequenza di foreground sarà possibile la riscostruzione sintetica del video-shot originale.

Funzione di similarità

Per stabilire una misura quantitativa della simililarità tra due oggetti vengono considerate le loro caratteristiche di forma, di colore e di posa.
Vediamo quali sono per un generico oggetto:

- ***Area:*** numero di pixel dell'oggetto *(f^1)*;
- ***Solidità:*** rapporto tra area dell'oggetto e della regione convessa che lo racchiude *(f^2)*;
- ***Eccentricità:*** rapporto tra la distanza dei fuochi dell'ellisse che ha lo stesso momento secondo dell'oggetto e la lunghezza dell'asse maggiore dell'ellisse stessa *(f^3)*;
- ***Orientazione:*** angolo tra l'asse delle ascisse e l'asse maggiore dell'ellisse che ha lo stesso momento secondo dell'oggetto *(f^4)*; l'orientazione assume maggior significato quanto più l'oggetto é eccentrico.;
- ***colore medio:*** media dei colori dei pixel dell'oggetto; *(f^5)*;
- ***Scarto quadratico medio dei colori:*** scarto quadratico medio dei colori dei pixel dell'oggetto (f^6);
- ***Centroide:*** vettore di due elementi che specifica la posizione del centro di massa dell'oggetto (p).

Nel seguito indicheremo con k l'indice che identifica una particolare caratteristica. Siano f_i e f_j i pattern di caratteristiche relativi agli oggetti O_i e O_j:

$$f_i = [f_i^1 f_i^2 \cdots f_i^6]$$

$$f_j = \left[f_j^1 f_j^2 \cdots f_j^6\right]$$

La distanza tra f_i^k e f_j^k viene pesata in una funzione gaussiana:

$$G_k(f_i^k, f_j^k) = e^{-\frac{\|f_i^k - f_j^k\|^2}{2\sigma_{G_k}{}^2}}$$

Tale funzione, che decresce monotonicamente con la distanza nell'intervallo da 1 a 0, rappresenta il contributo della k-esima caratteristica al calcolo dell'indice di similartà $s_{i,j}$. I contributi di tutte le caratteristiche vengono combinati con opportuni pesi α_k che permettono di stabilire quanto ciascuna caratteristica incida sul valore finale del coefficiente:

$$C_{i,j} = \frac{\sum_{n=1}^{6} G_n(f_i^n, f_j^n)\bar{\alpha}_n}{\sum_{n=1}^{6} \bar{\alpha_n}}$$

La formula generale per il calcolo dell'indice $s_{i,j}$ sarà:

$$s_{i,j} = \frac{C_{i,j}}{(1 + \delta d_{i,j})}$$

Al denominatore si può notare la presenza di un ulteriore contributo $d_{i,j}$ che rappresenta la distanza tra la posizione del centroide dell'oggetto O_i prevista nel prossimo frame e quella reale dell'oggetto O_j. Tale previsione viene ottenuta basandosi sulle posizioni dell'oggetto O_i negli ultimi due frame supponedo che esso si muova di moto rettilineo uniforme. La costante δ rappresenta il peso di questa previsione (il valore scelto per l'applicazione é 0.02).

Si noti che l'indice di similarità varia tra 0 e 1 ed ha valore massimo quando $d_{i,j}$ é nulla.

Di seguito sono riportati i valori dei coefficienti σ_G e α_k utilizzati nell'applicazione.

caratteristica	$\bar{\sigma}_G$	$\bar{\alpha}_k$
$Area(f^1)$	$8.66A_d$	1.05
$Solidità(f^2)$	0.1	1
$Eccentricità(f^3)$	0.1	0.95
$MediaColore(f^5)$	44.34	1.05
$SqmColore(f^6)$	44.34	0.95

Come si puó notare mentre per le altre caratteristiche i valori di σ_G scelti sono costanti, per l'area non è così. Consideriamo, ad esempio, le aree di due oggetti; la differenza tra il numero di pixel dei due blob ad essi corrispondenti assume un diverso significato a seconda dell'entità del loro valore: una differenza di 5 pixel risulta significativa su aree molto piccole ma trascurabile per quelle di dimensioni maggiori. Per questo, quindi, il valore di σ_G é moltiplicato per A_d funzione dal valore dell'area.

Per l'orientazione è necessario un discorso a parte. L'orientazione é definita come angolo tra l'asse delle ascisse e l'asse maggiore dell'ellisse che racchiude l'oggetto; due oggetti risultano orientati allo stesso modo a meno di un angolo di 180 gradi, se sono sufficientemente eccentrici; si é quindi scelto di utilizzare una funzione periodica per il calcolo del suo contributo:

$$H(f_i^4, f_j^4) = \cos\left(\frac{2\pi(f_i^4 - f_j^4)}{360}\right)^2$$

Il peso dell'orientazione dovrà risultare tanto maggiore quanto piú alto risulterà il valore minimo delle eccentricità dei due oggetti:

$$\alpha_4 = \frac{\tanh(16(min(f_i^3, f_j^3) - 0.5) + 1)}{2}$$

5 RISULTATI

Di seguito vengono illustrati alcuni risultati sperimentali ottenuti su sequenze video acquisite con una telecamera digitale commerciale. Come primo esempio riportiamo la sequenza "Michele Lorena S.Zeno" nella quale si riprende, sullo fondo della facciata della chiesa di San Zeno a Verona, una persona che entra nella scena, sale una scala e incontra un'altra persona, entrata nel frattempo nell'inquadratura, assieme alla quale prosegue nella stessa direzione. La telecamera effettua un movimento rotazionale seguendo il tragitto della prima persona che entra sulla scena. A questa stessa sequenza si e già fatto riferimento nei paragrafi precedententi. La figura 1 mostra il mosaico ad essa relativo che, come si può notare, non contiene alcuna traccia degli oggetti in movimento. In figura 15 si vuole evidenziare la somiglianza tra un frame originale della sequenza e lo stesso frame ricostruito sfruttando il mosaico, opportunamente allineato, e l'immagine di foreground.

Figura 15: Immagine Originale e immagine ricostruita

Nel secondo esempio due persone, in scena sin dal primo frame, si incrociano di fronte all'Arena di Verona e proseguono il loro tragitto in direzioni opposte; la telecamera effettua un panning seguendo il tragitto di una delle due persone. La figura 16, relativamente ad uno dei frame di questa sequenza, mostra quello originale, quello riscostruito, la mappa risultante dall'applicazione della norma alla loro differenza nello spazio RGB e il risultato dell'estrazione delle dei due oggetti in movimento.

Come misura quantitativa della bontà della ricostruzione si è utilizzato il *Peak Signal-to-Noise Ratio* (PSNR) tra i frame della sequenza originale e quelli ricostruiti. Se I_o è l'immagine originale di dimensioni nxm e I_r è la stessa immagine ricostruita esso è definito come segue:

$$PSNR(I,J) = 20 \log \frac{255}{\sqrt{\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}(I_o(i,j) - I_r(i,j))^2}}$$

I grafici in figura 17 e 18 mostrano l'andamento del PSNR relativamente alle due sequenze di prova.

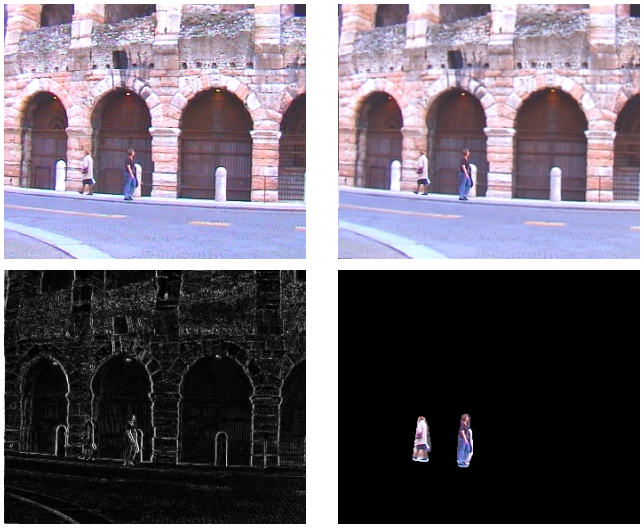

Figura 16: Originale, ricostruita, differenza, oggetti in foreground.

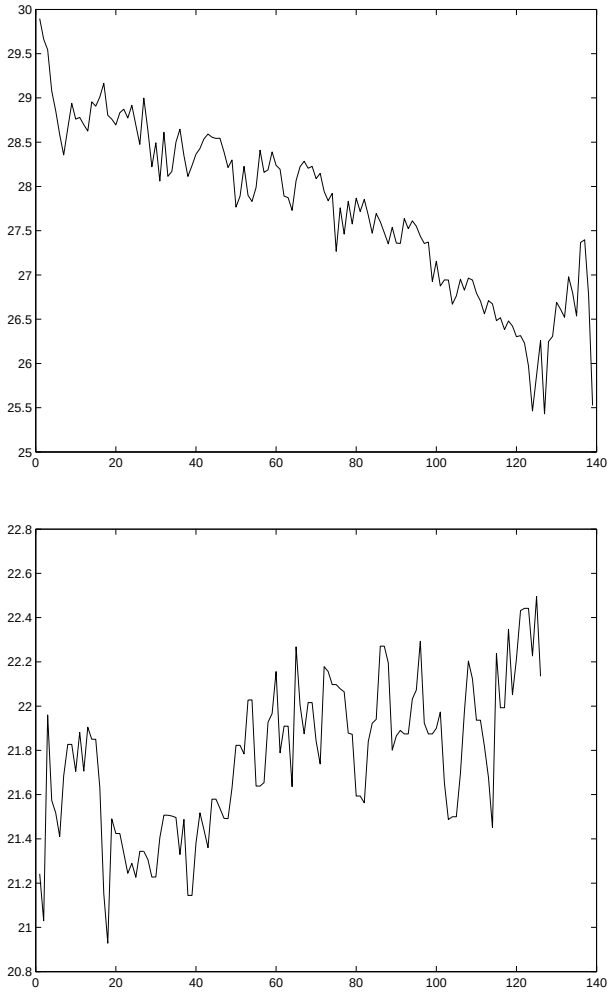

Figura 17: PSNR delle sequenze "Michele Lorena S.Zeno" e "Incrocio Arena".

6 APPLICAZIONI

Uno dei principali campi di applicazione del metodo di rappresentazione proposto è il *video coding*. Lo standard MPEG-4 per la codifica dei dati multimediali prevede un approccio content-based in cui vengano identificati e codificati separatamente gli oggetti video presenti sulla scena. Si noti che lo standard non descrive come gli oggetti video debbano essere estratti, ma semplicemente quale sia la loro rappresentazione. Trattando gli oggetti in movimento individuati nella sequenza di foreground e il mosaico statico dello sfondo come singoli oggetti video, si può notare la perfetta compatibilità tra lo standard MPEG-4 e il metodo di rappresentazione stratificata proposto. Tale rappresentazione consente di trasmettere separatamente i vari oggetti che compongono una sequenza video. Il mosaico, inviato un'unica volta, permetterà di ottenere il background di ogni signolo frame. Gli oggetti in movimento verranno poi trasmessi separatamente assieme alle trasformazioni (omografie) che peretteranno di allineare opportunamente lo sfondo e gli oggetti in ogni frame. La ricostruzione potrà anche avvenire annullando totalmente il movimento della telecamera al fine di ottenere una sequenza con vista panoramica statica della scena.

La separazione dello sfondo dagli eventuali oggetti in movimento suggerisce la possibilità di inserire virtualmente oggetti sintetici nella scena. Questo è un tipico caso di *video editing*, un'altra delle possibili applicazioni del metodo ti rappresentazione illustrato, di cui riportiamo un esempio in figura 18. In questa sequenza una automobile entra in scena, percorre un tratto di strada seguita dalla telecamera ed esce. La rimozione degli oggetti in movimento ha consentito di inserire sullo sfondo della scena un elemento che prima non esisteva (in questo caso uno striscione con la scritta "VIPS").

Per poter inserire correttamente lo striscione è necessario prima effettuare sul mosaico una operazione di rettificazione metrica che ha lo scopo di sintetizzare una vista frontale-parallela del piano 3D su cui verrà effettuata l'applicazione. Un piano 3D e la sua immagine prospettica sono legate da una omografia che è definita dalla posizione relativa di quattro punti nel piano mondo. L'operatore mappa la proiezione sul mosaico dei quattro punti 3D (punti rossi nella prima immagine) sulla proiezione degli stessi punti su un piano parallelo e frontale (punti rossi nella seconda immagine). Si sono scelti come punti 3D gli estremi di due lampioni giacenti su un piano parallelo alla ringhiera. Su di essa dopo aver effettuato la rettificazione del mosaico dove verrà collocato lo striscione. In seguito si applica all'immagine risultante la trasformazione prospettica inversa per riportarlo nelle condizioni iniziali (terza immagine). Per ottenere la sequenza con il nuovo sfondo basterà effettuare il procedimento di ricostruzione sostituendo al mosaico originale quello modificato. Nella quarta immagine viene mostrato un frame della sequenza ricostruita con vista panoramica statica.

Per poter vedere altri risultati si rimanda alla pagina web:

http://zeus.sci.univr.it/∼marzotto

sulla quale si potranno trovare i test eseguiti su diverse sequenze oltre a vari mosaici della città di Verona.

Figura 18: Mosaico originale, rettificato e modificato, finale, esempio di frame ricostruito.

7 CONCLUSIONE

Allo stato attuale i risultati ottenuti sono da ritenersi soddisfacenti, ma sono possibili ulteriori miglioramenti intervenendo a vari livelli.

Nei test effettuati si è riscontrata una forte dipendenza tra la fase di costruzione del mosaico e la successiva segmentazione; in alcuni casi la buona riuscita di quest'ultima risultava già inizialmente compromessa dall'elevata rumorosità delle immagini differenza tra mosaico e frame originali. La registrazione globale potrebbe essere la soluzione ideale per migliorare sensibilmente la qualità del mosaico; essa contribuirebbe a compensare gli errori causa dei disallineamenti, influendo positivamente anche sulle fasi successive.

Il tracking gestisce la presenza di più oggetti, le eventuali occlusioni fra loro o con parti statiche della scena e l'entrata o l'uscita durante la sequenza; tuttavia gli oggetti in movimento non vengono trattati come entità indipendenti con una propria individualità. Questo impedisce di risalire alla storia di ogni singolo oggetto in modo da poter ricostruire la sequenza togliendo o aggiungendo oggetti a piacimento. Una via da percorrere potrebbe essere quella di irrobustire il tracking non limitandosi ad una analisi delle similarità tra coppie di frame consecutivi, ma sfruttando una visione globale della sequenza (come ad esempio l'approccio proposto in [2]) per riuscire a stimare le traiettorie degli oggetti.

8 RINGRAZIAMENTI

Vogliamo ringraziare la dott.ssa Francesca Odone che ha collaborato alla stesura della prima versione in linguaggio C del programma.

Ringraziamo inoltre Andrea Colombari per il codice Matlab relativo al feature-tracker di Kanade-Lucas-Tomasi da noi utilizzato.

Infine un grazie caloroso a tutti i componenti del laboratorio VIPS per il costante supporto.

9 MANUALE UTENTE

L'applicazione é costituita da funzioni Matlab versione 6.0 ($'nomefunzione.m'$). Alcune funzioni sono state realizzate in linguaggio C compilate con la funzione mex che consente di eseguirle all'interno dell'ambiente Matlab come librerie condivise ($MEX - files$).

Di seguito si riportano le fasi del procedimento, le chiamate alle funzioni che le realizzano specificando i relativi parametri. Per ulteriori dettagli tecnici utilizzare l'help delle singole funzioni digitando '*help nomefunzione*' in ambiente Matlab.

- **Inizializzazione del vettore dei parametri**

  Il vettore dei parametri (P) specifica le caratteristiche del filmato in ingresso da comprimere oltre a definire tutte le modalità di funzionamento delle fasi del procedimento. I parametri sono semplici se hanno valore binario oppure complessi se é necessario specificare un valore. Dove i parametri non sono specificati vengono assegnati valori di default.

  Parametri semplici:

  v: Verbose mode.

  c: Immagini a colori.

  f: Salva la sequenza con le feature marcate.

  C : Salva la sequenza con i blob colorati.

  a : Legge la sequenza di ingresso come file in formato avi.

  N : Viene effettuata la normalizzazione fotometrica dei frames.

  e : Si tiene conto degli edge per ridurre l'effetto del disallineamento.

  Parametri complessi:

  i : Sequenza in ingresso.

  n : Lunghezza della sequenza.

  p : Primo frame.

  I : Directory di input.

  o : Prefisso al nome dei frames in uscita.

  O : Directory di output.

  M : Massimo numero di features cercate.

  m : Minimo numero di features inseguite.

  s : Periodo di sottocampionamento (nel blending del mosaico).

  b : Metodo di blending (m(edian), l(ast), f(irst), a(verage)).

F : Formato dell'immagine (occorre specificare l'estensione del file).

r : Frame di riferimento.

R : Frame di riferimento per la normalizzazione fotometrica.

Esempio di chiamata:

$$
\begin{array}{ll}
P = init\_par(& \text{'i','nomesequenza',} \\
 & \text{'I','/sequenzeinput/',} \\
 & \text{'O','../test/sequenze/',} \\
 & \text{'o','outseq',} \\
 & \text{'M', 50,} \\
 & \text{'n', 431,} \\
 & \text{'p', 1,} \\
 & \text{'c',} \\
 & \text{'a',} \\
 & \text{'s', 3,} \\
 & \text{'C',} \\
 & \text{'b','m',} \\
 & \text{'F','bmp',} \\
 & \text{'r', 65,} \\
 & \text{'v');}
\end{array}
$$

- **Lettura dei frame con eventuale ridimensionamento**

  I frame possono essere letti da un file AVI non compresso oppure da una sequenza di immagini; é possibile ridimensionare con un fattore di scala e ritagliare ogni immagine specificando rispettivamente un numero compreso tra 0 e 1 e un vettore che indica il numero di pixel da eliminare sui quattro lati. Tutti i frame della sequenza verranno memorizzati in $Images$ $(cellarray)$.

  Esempio di chiamata:

$$Images = read\_frames(P, 1, [1\ 1\ \ 6\ 6]);$$

- **Costruzione del mosaico**

  La costruzione del mosaico avviene sulla base dei parametri specificati in P e produce i seguenti files come record Matlab:

  - $mos\_img\_nomesequenza.tif$: immagine del mosaico;
  - $mos\_rc\_nomesequenza$: file contenente le dimensioni dell'immagine (campi $'m\_row'$,$'m\_col'$), i riferimenti del mosaico (campi $'m\_ref\_row'$,$'m\_ref\_row'$) e l'immagine (campo $'image'$);
  - $mos\_tensor\_nomesequenza$: file contenente il vettore delle omografie tra ciascun frame e il frame di riferimento scelto.

Esempio di chiamata:

$$[\mathrm{M},\ \mathrm{T},\ \mathrm{F}] = mosaic(P, Images);$$

In uscita le variabili M, T ed F contengono rispettivamente il record del mosaico (senza i campi $'m\_row'$ e $'m\_col'$ che specificano le dimensioni), il vettore delle omografie e il vettore dei frame utilizzato per costruire il mosaico (entrambi *cell array*).

- **Costruzione delle mappe binarie**

  La costruzione delle mappe binarie dei frame, in generale di una sottosequenza (si possono specificare il frame iniziale e il frame finale), avviene sulla base dei parametri specificati utilizzando il mosaico, il vettore delle omografie ed i frame originali. In uscita si avranno nell'ordine:

  - Il vettore delle immagini con le regioni etichettate.
  - Il vettore delle immagini con le regioni etichettate allineate rispetto al riferimento scelto.
  - Il vettore con il numero delle regioni di ciascuna immagine binaria.
  - Il vettore dei frames originali allineati rispetto al riferimento scelto.
  - Il vettore degli spiazzamenti rispetto al riferimento scelto per ogni frame.
  - Il vettore dei frames originali della sottosequenza specificata (utile solo se non si considerano tutti i frame).
  - Il vettore omografie di trasformazione rispetto al riferimento della sottosequenza specificata (utile solo se non si considerano tutti i frame).

  Esempio di chiamata:

  $$[Regions\_lab, Regions\_lab\_warped, num\_reg,\ Images\_warped, ref, FRAME, TN] = global\_register\_bmap(P, M, Tensor, Images, NS, NE);$$

- **Estrazione degli oggetti in movimento**

  L'estrazione degli oggetti in movimento avviene sulla base dei parametri specificati utilizzando le immagini con le regioni etichettate originali e allineate, il vettore dei numeri di regioni di ciascun frame, il vettore dei frame originali della sottosequenza con i relativi riferimenti, e il numero di frame su cui effettuare l'estrazione. Il risultato

sará il salvataggio su file della sequenza di immagini di foreground ($outnameNNN.tif$) e, se specificato nei parametri, la sequenza di mark per vedere i risultati del tracking ($mark\_outnameNNN.tif$).

Esempio di chiamata:

$$extract\_moving\_objs(P, RL, RLW, num\_reg, Fr, IW, ref, N);$$

Sono state realizzate due funzioni per la ricostruzione della sequenza:

- **Costruzione e salvataggio della sequenza originale**

  Questa funzione ricostruisce, una volta effettuata la fase di estrazione, la sequenza originale specificando, oltre ai parametri iniziali, la lunghezza della sottosequenza da costruire e il fps desiderato. Il vettore film specifica quali filmati salvare (film(1)=1 : salvataggio sequenza mark ($film\_mark\_nomesequenza.avi$), film(2)=1 : salvataggio sequenza foreground ($film\_fore\_nomesequenza.avi$), film(3)=1 : salvataggio sequenza ricostruita ($film\_rec\_nomesequenza.avi$). In uscita viene restituito il vettore (*cell array*) dei frames della sequenza.

  Esempio di chiamata:

  $$[Reconstructed\_image] = reconstruct\_sequence(P, N, film, fps);$$

- **Costruzione e salvataggio della sequenza sintetica con telecamera fissa e visione panoramica**

  É possibile costruire, una volta effettuata la fase di estrazione, una sequenza con visione panoramica senza movimento della telecamera ($film\_st\_rec\_nomesequenza.avi$) specificando, oltre ai parametri iniziali, la lunghezza della sottosequenza da costruire e il fps desiderato.

  Esempio di chiamata:

  $$rec\_static\_sequence(P, N, fps);$$

Indice

Riferimenti bibliografici

[1] M. Irani, S. Hsu, and P. Anandan. Mosaic based video compression. In *SPIE Conference on Electronic Imaging*, Feb 1995.

[2] I. Cohen and G. Medioni. Detecting and tracking moving objects for video surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 319–325, 1999.

[3] P. Giaccone and G. Jones. Segmentation of global motion using temporal probabilistic classification. In *British Machine Vision Conference*, pages 619–628, 1998.

[4] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu. Efficient representations of video sequences and their applications. *Signal processing: Image Communication*, 8(4):327–351, May 1996.

[5] F. Odone, A. Fusiello, and E. Trucco. Robust motion segmentation for content-based video coding. In *6th RIAO (Recherche d'Informations Assiste par Ordinateur) International Conference*, 2000.

[6] J. Davis. Mosaics of scenes with moving objects. In *Computer Vision and Pattern Recognition Conference*, pages 354–360, 1998.