Università degli Studi di Padova
Dipartimento di Scienze Statistiche

Corso di Laurea Magistrale in
Scienze Statistiche

Tesi di Laurea

SORVEGLIANZA STATISTICA DI IMMAGINI: CONFRONTO TRA DUE APPROCCI DI RIDUZIONE DIMENSIONALE

Relatore: Prof.ssa Giovanna Capizzi
Dipartimento di Scienze Statistiche

Laureando: Federico Tasinato
Matricola n. 1238227

Anno Accademico 2021/2022

Indice

Introduzione

La presente trattazione si inserisce nel contesto dell'analisi e della sorveglianza statistica di un particolare tipo di dato: le immagini. Queste possono essere classificate come un peculiare formato di *big data*, ovvero dati caratterizzati da una dimensionalità elevata che li contraddistingue dai semplici dati multivariati. Le immagini richiedono infatti tecniche di analisi statistica che siano in grado di gestire sia la dimensionalità sia la struttura spaziale intrinseca dei dati stessi. Sebbene esistano numerosi contributi in letteratura che trattano tale argomento, la maggior parte di essi è relativamente recente; la disponibilità di sistemi che garantiscono un acquisizione rapida e a basso costo di immagini per la sorveglianza statistica in tempo reale, infatti, non è sempre stata estesa, anzi. Solo in tempi recenti tali tecnologie si sono diffuse a tal punto da rendere disponibili grandi moli di dati nel formato di immagini. Anche per tale motivo, nonostante la letteratura annoveri diverse proposte di metodologie per la sorveglianza di immagini, solo in tempi recenti tale ambito della sorveglianza sta guadagnando sempre più interesse. In particolare, vengono qui considerati due approcci per la letteratura in tale ambito: questi infatti si differenziano da quelli esistenti in quanto uniscono efficacemente alla consueta elaborazione e modellazione delle immagini l'informazione spaziale intrinseca nei dati, migliorando significativamente degli approcci già disponibili. Lo scopo della presente trattazione è dunque quello di mettere a confronto queste due nuove procedure, ponendole in un *framework* comune così da poterne studiare il comportamento qualora esse si trovino a competere. A tal fine vengono quindi ipotizzati e generati dei *dataset* che descrivono gli scenari che è lecito poter riscontrare qualora ci si ritrovi ad analizzare un insieme di dati reali. I risultati dello studio condotto vengono infine riportati. È quindi opportuno sottolineare come il contributo che contraddistingue il presente lavoro sia dato dall'implementazione di una comparazione attualmente non presente in letteratura, che si propone di confrontare due procedure sviluppate per contesti diversi ma che possono tuttavia essere applicate a situazioni paragonabili in termini di problema

affrontato.

Il seguito della presente trattazione è così strutturato. Nel Capitolo 1 viene fornita una breve introduzione ai concetti basilari della sorveglianza statistica, con particolare attenzione ai *big data* e le immagini e ai sistemi che ne permettono l'acquisizione. Nel Capitolo 2.1 viene discusso brevemente lo stato dell'arte della letteratura riguardante gli approcci di *machine learning* applicati alla sorveglianza statistica. Nel Captolo 2.2 vengono quindi descritti gli approcci possibili alla sorveglianza delle immagini, a seconda di come queste vengano trattate e a seconda del particolare approccio di sorveglianza statistica adottato. Nel Capitolo 3 vengono quindi illustrate nello specifico le due procedure oggetto di studio, definendone il disegno e il funzionamento. Nel Capitolo 4 vengono descritti i passaggi necessari per la realizzazione del confronto, definendo il processo di generazione dei *dataset*, l'implementazione delle carte di controllo ed i risultati ottenuti dal confronto delle procedure. Infine, nel Capitolo 5 vengono riportate le conclusioni e alcune osservazioni finali in merito allo studio condotto, sintetizzando ciò che si può dedurre dai risultati ottenuti. Viene inoltre fornito in Appendice A il codice R completo che consente di generare gli scenari analizzati nei capitoli precedenti, implementare le carte di controllo ed eseguire quindi il confronto.

1 La sorveglianza statistica dei *big data*

1.1 Generalità sui concetti di sorveglianza statistica

Il controllo statistico di processo (*Statistical Process Control*, *SPC*) è un insieme di metodi statistici finalizzato alla sorveglianza di processi sequenziali per assicurarsi che questi operino in modo stabile e soddisfacente (Peihua, 2014). Nella sorveglianza di un processo è importante verificare se i prodotti finali ottenuti siano conformi alle specifiche prestabilite; se tale condizione risultasse violata, il processo dovrebbe essere interrotto il prima possibile. Nei casi in cui il processo operi in condizioni stabili secondo come previsto, esso si definisce essere in controllo statistico o semplicemente in controllo (*In-Control*, *IC*). Quando alcune componenti del processo vengono alterate o si comportano diversamente rispetto a quanto prestabilito, si osserva generalmente un cambiamento nella qualità degli output. Quando un processo viene interessato da questo fenomeno è considerato instabile, fuori controllo statistico o semplicemente fuori controllo (*Out-of-Control*, *OC*). L'obiettivo principale dell'*SPC* è quello stabilire dunque nel più breve tempo possibile se un processo sia in controllo o meno.

L'*SPC* è spesso divisa in due diverse fasi. Nella Fase I (offline) si cerca di avviare correttamente il processo in modo che operi stabilmente. Vista la scarsa conoscenza del processo, in questa fase l'analisi statistica coinvolta ha una natura esplorativa. Nella Fase II (online) si ritiene che il processo sia in controllo all'inizio della sorveglianza e l'obiettivo principale è sorvegliare il processo online per assicurarsi che continui ad operare in modo stabile. A tal fine, gli output vengono campionati sequenzialmente nel tempo e le relative osservazioni delle caratteristiche qualitative di interesse vengono sorvegliate utilizzando metodi statistici e grafici in grado di fornire un segnale se viene rilevata una deviazione significativa dai valori ottimali stabiliti. Dopo aver lanciato un allarme il processo deve essere interrotto immediatamente e deve essere condotta un'analisi per individuare le cause che hanno portato alla generazione dell'anomalia. Sia nella Fase I che nella Fase II, la procedu-

ra usuale prevede che venga scelto un parametro di interesse da sorvegliare che dovrebbe contenere quanta più informazione possibile sulla distribuzione delle caratteristiche di qualità del processo, oltre ad essere sensibile anche a qualsiasi spostamento distributivo ritenuto significativo. Viene quindi calcolata un'opportuna statistica riassuntiva in grado di riassumere le informazioni registrate sul parametro di interesse ; questa viene definita statistica di controllo e viene solitamente riportata in un grafico, in modo tale da avere un riscontro visivo dell'andamento delle prestazioni del processo nel tempo. L'insieme di statistiche di controllo e dei relativi grafici impiegati per la sorveglianza di un processo viene definito, nell'ambito dell'*SPC*, come carta di controllo: essa quindi rappresenta l'insieme degli strumenti statistici impiegati per la sorveglianza del processo. In letteratura sono stati sviluppati diversi tipi carte di controllo, a seconda della natura dell'oggetto di interesse della sorveglianza. Si parlerà dunque di (1 sorveglianza statistica univariata nel caso si sorvegli un singolo parametro (o un insieme di parametri in modo indipendente l'uno dall'altro), (2 sorveglianza statistica multivariata nel caso in cui le componenti di un vettore di parametri vengano considerate congiuntamente, o (3 sorveglianza statistica di profili nel caso in cui sia di interesse una relazione nei dati acquisiti sequenzialmente. In generale, per approfondimenti in merito ai concetti alla base dell'*SPC*, qui e per il seguito si rimanda a Peihua (2014).

1.2 La sorveglianza statistica di immagini

Attualmente, i progressi della misurazione online e delle tecnologie dei sensori utilizzati a tale scopo hanno permesso di raccogliere una mole ingente di dati in molte applicazioni (A. Wang, Xian et al., 2018). L'enorme quantità di misurazioni generate sequenzialmente da una vasta gamma di sensori è solitamente indicata come *big data*, dove il termine "*big*" può essere riferito sia alla complessità del formato dei dati (ad esempio immagini o video) sia all'alta frequenza di acquisizione degli stessi (ad esempio, fino a migliaia di osservazioni al secondo). Questo genere di dati differisce dai dati di na-

tura semplicemente multivariata in quanto la dimensionalità e la frequenza di rilevamento delle unità statistiche sono così elevate da richiedere metodi statistici specifici per poter trattare efficacemente tale tipologia di dati, riuscendo a sfruttare al meglio l'informazione contenuta in essi. Questi flussi di dati contengono infatti informazioni dettagliate ed in tempo reale sul processo sottostante, fornendo dunque opportunità per il miglioramento sia della sorveglianza statistica sia, di conseguenza, della qualità del processo stesso. Poiché i flussi di dati hanno un grande volume e sono generati ad alta velocità, l'analisi in tempo reale sarà obbligatoriamente influenzata dalle capacità computazionali, dalla velocità di elaborazione dei dati e dalla quantità di spazio di archiviazione disponibile dei calcolatori. Di conseguenza, questi vincoli *hardware* spesso richiedono di identificare, registrare ed utilizzare solamente alcune informazioni per l'analisi online. In questo contesto, sono necessari nuovi strumenti di sorveglianza statistica di processo per affrontare i problemi posti dall'utilizzo dei *big data*. Un esempio di dati ad alta dimensionalità è rappresentato dalle immagini o similmente dai video. L'uso di dati in formato di immagini ha attirato un crescente interesse industriale negli ultimi anni nelle applicazioni di sorveglianza dei processi (A. Wang, Xian et al., 2018). La disponibilità di sistemi di visione artificiale compatti, a basso costo e facili da integrare negli impianti di produzione (Capitolo 1.3) è notevolmente aumentata grazie ai continui progressi tecnologici e ciò rende di fatto possibile sia l'acquisizione che l'utilizzo di questi dati complessi. Le immagini richiedono una particolare attenzione per quanto ne riguarda la trattazione statistica e, in particolare, nel contesto della sorveglianza statistica di processo. Oltre al bisogno infatti di essere codificate in maniera adeguata per poter implementare i metodi di sorveglianza opportuni, necessitano di procedure statistiche in grado di gestire ed elaborare sia efficacemente che rapidamente i dati provenienti in tempo reale dal processo. La sorveglianza statistica di processi attraverso l'acquisizione online di immagini è il tema principale della presente trattazione. Va detto che l'uso di carte di controllo basate su immagini differisce dalle applicazioni di *SPC* più tradizionali. Queste differenze

possono essere attribuite a una serie di fattori, primo tra tutti il tipo di dati sorvegliati e la conseguente logica alla base dell'utilizzo delle carte stesse. È necessaria innanzitutto un'analisi preliminare delle immagini che comprende ad esempio operazioni come la riduzione del rumore, la compressione e altri tipi di elaborazione. Questa analisi è in genere molto più ampia e onerosa di quella richiesta nelle applicazioni standard dell'*SPC*. Inoltre, in alcune applicazioni, i grafici di controllo basati su immagini non vengono utilizzati per rilevare i cambiamenti nelle immagini nel tempo, ma per rilevare in che area di un'immagine si siano verificati eventuali difetti. In generale poi, per gli articoli attualmente presenti in letteratura che fanno riferimento alla sorveglianza statistica basata su immagini, spesso non viene fatta una chiara distinzione tra le due fasi caratteristiche dell'*SPC* tradizionale (Megahed, Woodall et al., 2011).

Ci sono poi molte decisioni pratiche che devono essere prese quando si utilizzano le immagini per la sorveglianza dei processi: la scelta del dispositivo di acquisizione, la frequenza di rilevamento, l'impostazione della scena nella quale le immagini verranno acquisite per garantire che l'illuminazione e l'allineamento del soggetto siano ottimali, il *software* da utilizzare per l'analisi e quali elaborazioni preliminari svolgere. Attualmente non esistono linee guida univoche per guidare chi si approccia a tali compiti (Megahed, Woodall et al., 2011). Poiché nella maggior parte dei casi non esiste un approccio standard alla sorveglianza delle immagini, la scelta di un metodo sarà guidata in larga misura dallo scopo della sorveglianza. In alcuni casi ad esempio, lo scopo è quello di rilevare dei difetti tramite il confronto tra un'immagine "ideale" e l'immagine osservata. La presenza di un cosiddetto *gold standard* di riferimento può essere di grande aiuto nello scegliere la metodologia da applicare, in quanto questa peculiarità può essere impiegata nella formulazione statistica della carta di controllo. In altre situazioni invece può essere impossibile condurre tale paragone con un riferimento ottimale, vista l'assenza di quest'ultimo. Un esempio è dato dalla sorveglianza di tessuti così come vengono trattati in Bui e Apley (2018), ossia come su-

perfici a struttura stocastica, che possono essere quindi visti come generati da un processo stocastico bidimensionale. Il confronto diretto di immagini stocastiche porterà ad identificare solamente come simili un eventuale *gold standard* e l'immagine confrontata. Poiché la configurazione specifica dei pixel varia stocasticamente da un'immagine all'altra in condizioni di stabilità del processo, in questo contesto ci sono infinite immagini che hanno lo stesso comportamento stocastico in controllo, pur risultando completamente diverse. La peculiare struttura del dato sorvegliato porta quindi in questo caso ad escludere a priori alcune metodologie che nell'esempio precedente sarebbero invece risultate ottimali, indirizzando la scelta della carta da applicare vero una particolare direzione.

1.3 I sistemi di visione artificiale

Prima di esporre la metodologia per la gestione delle immagini nell'ambito dell'*SPC*, è opportuno soffermarsi sui metodi che consentono l'acquisizione e quindi il conseguente utilizzo delle stesse.

Un sistema di visione artificiale (*Machine Vision System*, *MVS*) è un sistema informatico che utilizza uno o più dispositivi di acquisizione di immagini per fornire dati per l'analisi e l'interpretazione. Esso rende possibile l'acquisizione di immagini o video permettendone successivamente l'elaborazione e l'utilizzo per analisi d'interesse; tale impianto è infatti preposto per attività come l'acquisizione di immagini, l'elaborazione, la segmentazione e il riconoscimento di strutture particolari o *pattern* (Golnabi e Asadpour, 2007). Il ruolo principale di un sistema di visione artificiale è quello di trasformare i dati di un'immagine ottica in una serie di dati in formato numerico, che poi possono essere quindi manipolati per fini statistici e di sorveglianza. La Figura 1 riporta un semplice schema di come questa struttura operi. La luce proveniente da un'adeguata sorgente illumina la scena e un'immagine ottica viene generata grazie a degli appositi sensori. Fotocamere digitali o altri mezzi vengono utilizzati per convertire l'immagine ottica in un segnale elettrico che può essere convertito in un'immagine digitale definitiva. Operazioni come

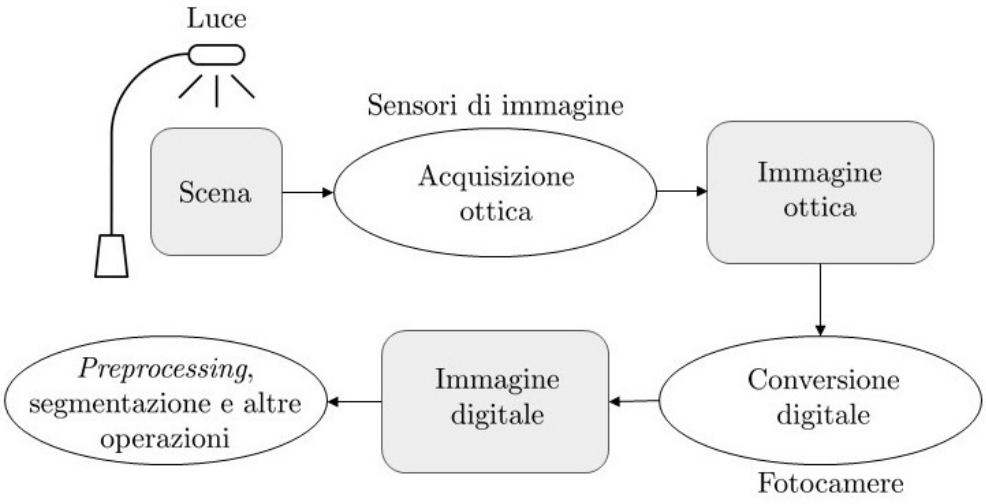


Figura 1: Schema di funzionamento di un sistema di visione artificiale.

l'elaborazione preliminare, la segmentazione e altre attività possono essere eseguite utilizzando questa immagine digitalizzata. Dopo che un'immagine è stata acquisita infatti, deve essere analizzata in modo che le informazioni possano essere estratte. Questo processo può essere diviso in tre livelli successivi: (1) processi a basso livello composti da operazioni di base di riduzione del rumore, compressione dell'immagine e miglioramento del contrasto, in cui gli input e gli output sono immagini; (2) processi a livello intermedio, in cui gli input sono immagini e gli output sono attributi estratti da immagini, come ad esempio bordi, soggetti o contorni; ed infine (3) operazioni ad alto livello, volte ad interpretare e dare dunque un senso ai dati.

I sistemi di visione artificiale sono ampiamente utilizzati in una varietà di settori a scopo di ispezione, dove i soggetti "conformi" devono essere separati da quelli "difettosi". Si possono classificare le applicazioni di ispezione tramite sistemi di visione artificiale in tre gruppi:

- applicazioni mediche, in cui i *MVSs* sono utilizzati per rilevare e diagnosticare anomalie in diverse parti del corpo (per un esempio in questo ambito si rimanda a Z. Q. Liu, Austin et al. (1996));

- applicazioni di trasporto e costruzione, in cui i sistemi di visione artificiale sono utilizzati nell'identificazione e nella misurazione di crepe, rientranze superficiali e sporgenze in diversi materiali da costruzione e strutture (si segnala ad esempio il lavoro svolto in Schmitt, Riddington et al. (2000));

- applicazioni industriali, per rilevare difetti dimensionali, strutturali o difetti superficiali dei prodotti fabbricati.

Nonostante alcune somiglianze tra visione umana e artificiale, ci sono differenze significative tra le due. Gli attuali sistemi di visione artificiale sono innanzitutto primitivi se confrontati con le capacità occhio-cervello umano poiché, ad esempio, gli attuali impianti sono suscettibili alle variazioni delle condizioni di illuminazione, riflessione e altri piccoli cambiamenti a cui l'occhio umano può facilmente adattarsi e compensare (Megahed, Woodall et al., 2011). Nonostante queste limitazioni, l'uso sistemi di visione artificiale è superiore all'ispezione visiva rispetto a (1) processi con elevati tassi di produzione; (2) l'esecuzione di più attività simultanee con oggetti diversi; (3) la capacità di coprire tutte le gamme dello spettro elettromagnetico, come nell'uso della risonanza magnetica e dei raggi X attraverso appositi strumenti di rilevazione; (4) la mancanza di suscettibilità alla fatica e alla distrazione. In alcuni casi poi l'uso di un *MVS* è più economico rispetto all'impiego di ispettori umani e si prevede che il costo di tali impianti continuerà a diminuire nel tempo, vista la progressiva e crescente adozione degli stessi da parte delle industrie. Questi vantaggi rendono l'uso dei sistemi di visione artificiale un'opzione valida e preferibile in molti casi all'ispezione visiva umana, il che aiuta a spiegare il loro uso diffuso e crescente in applicazioni industriali e mediche.

2 Stato dell'arte sulla sorveglianza statistica per *big data*

2.1 I metodi di *machine learning* per la sorveglianza statistica

In questo capitolo vengono presentati brevemente i principali approcci esistenti nella letteratura dell'*SPC* proposti per trattare problemi che coinvolgono dati multivariati e ad alta dimensionalità attraverso metodi di *machine learning*. Una prima distinzione da fare nella panoramica della metodologia *data-driven* di sorveglianza statistica individua due approcci generali possibili: metodi supervisionati e non supervisionati. La scelta del metodo di apprendimento statistico da utilizzare dipende spesso dalla struttura dei dati. Spesso, è necessaria una qualche forma di elaborazione preliminare dei dati prima di utilizzare l'una o l'altra classe di metodi di apprendimento statistico. Esistono alcuni modelli poi che possono essere utilizzati in modo supervisionato o non supervisionato.

L'apprendimento supervisionato si riferisce all'inferenza su una mappa da un insieme di variabili di input $\boldsymbol{x} = \{x_1, x_2, \ldots, x_p\}$ a una variabile di output y, dato un campione di addestramento $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ di coppie di dati generate secondo una distribuzione sconosciuta P_{xy} con densità $p(\boldsymbol{x}, y)$. L'obiettivo principale dell'apprendimento supervisionato è quello di stimare una funzione $H : \mathbb{X} \to Y$ tale che H classifichi correttamente le coppie non ancora osservate $(\boldsymbol{x}_i, y_i)$. Tra le metodologie di apprendimento supervisionato più comuni vi sono la regressione basata sul metodo dei minimi quadrati, dove H è lineare nei suoi parametri, la regressione logistica, le reti neurali artificiali (*Artificial Neural Networks*, *ANN*), le *Support Vector Machines* (*SVM*) e gli alberi decisionali. Diversi modelli di apprendimento supervisionato possono essere combinati per ottenere prestazioni predittive migliori di quelle che si potrebbero ottenere dall'adattamento di un singolo modello; la combinazione algoritmica di più modelli per migliorare le pre-

stazioni viene comunemente definita *ensemble modeling* (Weese, Martinez et al., 2016). Nella letteratura della sorveglianza statistica di processo, la maggior parte delle applicazioni dei modelli citati,– specialmente reti neurali, *SVM* e alberi decisionali – è incentrata sul disegno di carte di controllo per il riconoscimento di *pattern* anomali riscontrabili nelle realizzazioni del processo sorvegliato; questo tipo di applicazione prende il nome di *Control Chart Pattern Recognition* (*CCPR*).

Un approccio per ridurre la dimensionalità di un *dataset* è attraverso la costruzione di una variabile risposta (o un insieme più piccolo di nuove variabili) che "riassume" i dati contenuti all'interno del vettore p-variato originale. Nella letteratura dell'*SPC*, ci sono vari approcci principali per la riduzione della dimensionalità utilizzando metodi di regressione: il principale e più comune consiste nello considerare i residui di un opportuno modello che descrive i dati originali acquisiti attraverso una serie di variabili esplicative. In questo modo, la sorveglianza si limita ai residui stimati che hanno una distribuzione nota.

Il riconoscimento dei *pattern* anomali provvede all'identificazione e alla classificazione di comportamenti fuori controllo come tendenze, modelli ciclici e tipologie specifiche di cambiamenti di processo. Nella *CCPR*, un classificatore supervisionato viene addestrato per riconoscere specifici tipi di cambiamenti nel processo. L'input fornito al metodo di apprendimento scelto può essere costituito sia da variabili direttamente osservate che da combinazioni lineari o non lineari delle stesse. I modelli di reti neurali o *SVM* vengono utilizzati più spesso a causa della loro forte capacità predittiva, anche se questi possono essere difficili da interpretare; pertanto, si può pensare di utilizzare dei metodi basati sugli alberi decisionali così da ottenere modelli più interpretabili. Sebbene gran parte della letteratura in quest'area abbia tipicamente considerato dati a bassa dimensionalità, diversi autori si sono concentrati su dati con dimensioni superiori (ad esempio Deng, Runger et al. (2012), Dávila, Runger et al. (2011)). Deng, Runger et al. (2012) hanno fornito una panoramica dei principali sviluppi in questo settore in relazione

alla sorveglianza statistica.

La maggior parte delle applicazioni delle reti neurali e di modelli *SVM* nell'ambito dell'*SPC* rientra nell'area della *CCPR*, ma non costituisce l'unico impiego di queste classi di modelli nella sorveglianza statistica. Grazie alle loro proprietà predittive, le reti neurali artificiali possono essere generalmente utili in entrambe le fasi di rilevamento e analisi post-segnale dei guasti nel processo. Venkatasubramanian, Rengaswamy et al. (2003) hanno discusso l'utilizzo delle reti neurali in una rassegna di quelli che definiscono "metodi basati sulla storia del processo" per individuare e contemporaneamente assegnare alle corrette cause speciali un problema del processo. Va comunque osservato che (1) i modelli *ANN* addestrati su dati di processo storici non sono adeguati per essere generalizzati (Weese, Martinez et al., 2016), essendo stimati a partire da un campione di dati per riconoscere determinati cambiamenti nel processo e (2) ci sono stati pochi articoli pubblicati che considerano l'applicazione delle reti neurali artificiali a processi industriali reali. È importante notare che la prima osservazione vale per tutti i metodi di apprendimento supervisionato e non solo per le reti neurali.

Per quanto riguarda i modelli *SVM*, questi sono stati applicati nell'ambito dell'identificazione dei guasti o *fault detection*, ovvero l'identificazione della variabile o dei gruppi di variabili responsabili del cambiamento, sia esso nella struttura della media o della covarianza del processo, che hanno portato ad una situazione fuori controllo. C.-S. Cheng e H.-P. Cheng (2008) hanno confrontato i modelli *SVM* e *ANN* per l'identificazione dei guasti, dove il guasto in questo contesto era uno *shift* nella covarianza del processo, e hanno scoperto che i due metodi hanno portato a performance simili. Tuttavia, gli autori hanno consigliato di utilizzare le *SVM* poiché richiedono la stima di meno parametri rispetto alle reti neurali. Sebbene possano essere computazionalmente onerose, sia le applicazioni supervisionate che non supervisionate dei modelli *SVM* hanno un potenziale nel futuro dei metodi per l'*SPC* applicati ai *big data* (Weese, Martinez et al., 2016); in particolare, le *SVM* possono essere promettenti nello studio di dati di processi costituiti da

diversi tipi di dati, in quanto le macchine a vettore di supporto sono in grado di gestire contemporaneamente dati di diversa natura, come ad esempio variabili categoriali, discrete o continue.

L'apprendimento non supervisionato descrive un'area dell'apprendimento statistico che non beneficia della disponibilità di una variabile risposta. L'obiettivo dell'apprendimento non supervisionato è quello di individuare un particolare *pattern* caratteristico nella struttura delle variabili di input $\{x_1, x_2, \ldots, x_p\}$. Esempi di metodi di apprendimento non supervisionato includono l'analisi dei *cluster*, l'analisi delle componenti principali (*Principal Component Analysis*, *PCA*), i metodi a variabili latenti e i modelli mistura. I metodi di apprendimento non supervisionato sono applicabili quando si sa poco sul processo e non vengono fornite informazioni su ciò che costituisce un evento fuori controllo. Ciò rende i metodi di apprendimento non supervisionato particolarmente applicabili nella Fase I della sorveglianza statistica dei processi. Esistono due classi di metodi di apprendimento non supervisionato: metodi di riduzione della dimensionalità e metodi di classificazione a *cluster* e *one-class*. Gli approcci per la riduzione della dimensionalità si dividono a loro volta in due gruppi principali. Nel primo gruppo, l'attenzione si concentra sulla selezione di un sottoinsieme k di variabili, ignorando le restanti $p-k$. Un classico esempio di questo gruppo in statistica comporta la scelta dei predittori attraverso metodi di selezione delle variabili di regressione. Il secondo gruppo prevede la proiezione dell'insieme originale di variabili esplicative in un sottospazio di dimensioni inferiori. L'analisi dei componenti principali, i minimi quadrati parziali (*Partial Least Squares*, *PLS*) e l'analisi fattoriale sono tutti esempi di tali approcci. È importante notare che la scelta della riduzione della dimensionalità dipende e varia a seconda del caso in esame. In alcune applicazioni, può essere più significativo mantenere un sottoinsieme delle variabili originali in base a qualche criterio di classificazione se ciò faciliterà la sorveglianza, la diagnosi post-segnale d'allarme e il processo decisionale conseguente. Se non è necessario mantenere quella forma originale delle variabili, i metodi di proiezione possono essere più adatti.

Le tecniche di proiezione in sottospazi o di *feature extraction* per *dataset* con osservazioni p-dimensionali generalmente utilizzate si focalizzano su modelli basati su *PCA* o *PLS*. Recentemente, *PCA*, *PLS* e le estensioni che ne permettono l'impiego nelle carte di controllo sono state applicate a un certo numero di contesti ad alta dimensionalità differenti. In conclusione, l'uso di metodi simili alla *PCA* nella sorveglianza statistica dei processi: (1) può migliorare l'analisi rimuovendo l'influenza di variabili ridondanti o di disturbo; (2) può facilitare l'elaborazione dei dati; (3) può contribuire a migliorare l'interpretazione e la visualizzazione di *dataset* ampi; ed infine (4) può aiutare a comprendere la struttura di correlazione dei dati attraverso l'indagine dei pesi delle componenti.

I metodi di *clustering* possono essere basati su un modello scelto a priori, come la modellazione mistura, o metodi algoritmici come il metodo k-means o di *clustering* gerarchico agglomerativo. L'uso del *clustering* basato su modelli con carte di controllo di solito combina una tecnica di riduzione della dimensionalità con un approccio basato su modelli mistura. Una volta stimati il numero di *cluster* e i parametri distributivi e di mistura, queste informazioni vengono utilizzate per stabilire una regione di controllo per le normali condizioni operative del processo. Nelle applicazioni reali, l'esistenza di più *cluster* all'interno di un campione di riferimento potrebbe essere un indicatore di una situazione fuori controllo. Analogamente ai metodi di *clustering* basati sui modelli, gli approcci di classificazione ad una classe (*One-Class Classification*, *OCC*) per la sorveglianza statistica riformulano il problema della sorveglianza in un problema di classificazione che distingue tra osservazioni in controllo e fuori controllo. Poiché molti metodi di *clustering* e *OCC* non conservano l'ordine temporale dei dati, può essere difficile interpretare i segnali a potenziali eventi fuori controllo. I metodi di *clustering* che mirano a preservare l'ordinamento temporale dei dati sono stati applicati anche al problema del rilevamento di *outliers* nell'*SPC* multivariata (Weese, Martinez et al., 2016). Va notato comunque che i metodi di *clustering* descritti presuppongono che i dati, una volta raggruppati, possano essere archiviati

in memoria. Questo potrebbe non essere realizzabile nelle applicazioni che coinvolgono *big data*.

2.2 Gli approcci alla sorveglianza statistica di immagini

Le immagini vengono spesso – ma non sempre – analizzate attraverso un approccio statistico multivariato in grado di gestire l'alta dimensionalità di questo tipo di dato. Esistono infatti sia approcci univariati che multivariati, che gestiscono le immagini come un dato multivariato senza problemi derivanti dall'alta dimensionalità. Queste applicazioni sono generalmente riferite ad esempio a casi in cui le immagini sono in scala di grigi o binarie, oppure a bassa risoluzione, ma comunque più semplici da analizzare. In particolare, il processo di analisi statistica delle immagini attraverso l'uso di tecniche che consentono di gestire adeguatamente l'alta dimensionalità dei dati è comunemente noto nell'*SPC* come "analisi di immagini multivariate" (*Multivariate Image Analysis*, *MIA*).

L'acquisizione di un'immagine è il primo passo in qualsiasi applicazione di sistemi di visione artificiale. Un'immagine è rappresentata nella maggior parte dei casi come una funzione $f(x, y)$ o un vettore $\boldsymbol{f}(x, y)$, dove x e y rappresentano le coordinate spaziali sul piano dell'immagine e i valori in qualsiasi posizione (x, y) sono comunemente noti come intensità (Megahed, Woodall et al., 2011). Nelle immagini digitali x, y e le intensità assumono solo valori interi non negativi. All'aumentare della risoluzione del dispositivo di acquisizione delle, l'immagine viene divisa in un numero crescente di unità (cioè coppie di x e y) e quindi aumenta il numero totale di elementi (pixel) che la compongono. D'altra parte, i valori delle intensità $f(x, y)$ dipendono dal formato dell'immagine, a seconda cioè che essa sia in bianco e nero (in tal caso si definisce binaria o bitonale), in scala di grigi o a colori. Nel caso di un'immagine binaria, ogni pixel può avere solo due valori di intensità, ossia 0 (nero) o 1 (bianco). Se l'immagine è in scala di grigi a 8 bit, $f(x, y)$ può assumere qualsiasi valore intero compreso tra 0 (nero) e 255 (bianco). Infine, quando l'immagine è a colori, l'intensità di ogni pixel è rappresentata un

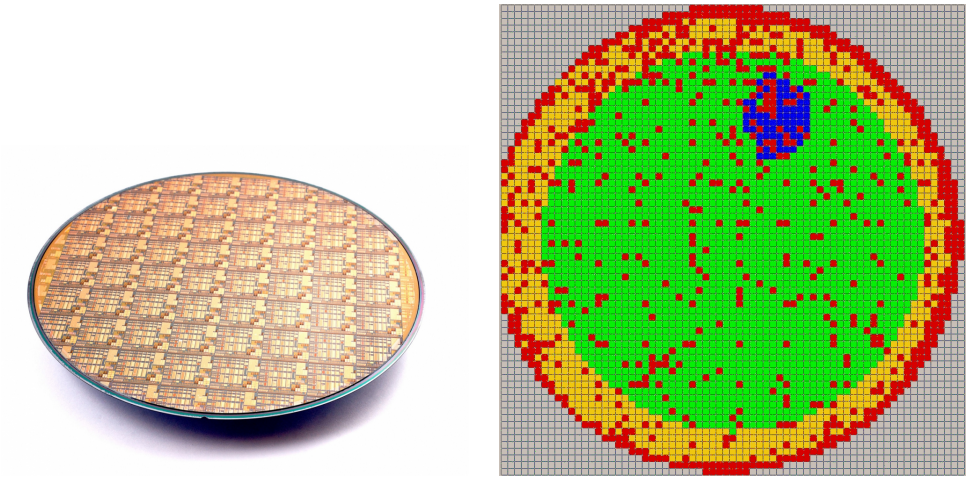
vettore di tre componenti corrispondenti alle concentrazioni dei colori rosso, verde e blu (*Red, Green and Blue*, *RGB*), con valori compresi tra 0 e 255 per ogni componente. Pertanto, in tutti e tre questi casi, un'immagine può essere visualizzata come una raccolta di osservazioni, univariate o multivariate, distribuite spazialmente e con dimensionalità che dipende sia dal numero di pixel che dal tipo di immagine. Per le immagini a 16 bit, i valori di intensità possono assumere valori compresi tra 0 e $2^{16} - 1$. Inoltre, va notato come i valori di intensità per i pixel vicini all'interno di un'immagine siano spesso altamente correlati; questa correlazione può comportare una notevole quantità di ridondanza nei dati, in cui le informazioni contenute in un determinato pixel sono abbastanza contenute, poiché la sua intensità può essere spesso prevista con precisione dai pixel vicini.

Le immagini nelle applicazioni industriali sono spesso immagini in scala di grigi o binarie. Queste immagini sono più semplici da ottenere e da analizzare. Inoltre, dal momento che in ambienti industriali con alti tassi di produzione il tempo è limitato per analizzare le immagini, ci sono vantaggi nell'utilizzo di metodi di analisi relativamente semplici. Esistono numerose applicazioni di carte di controllo per questo tipo di dato ad alta dimensionalità per rilevare i cambiamenti nelle performance del processo e aumentare l'efficienza dello stesso. In questo contesto, per performance si intende la capacità del processo di operare in controllo, in modo che l'output risultante rispetti le specifiche richieste. Quella che segue è una rassegna dei metodi proposti nel corso degli anni in letteratura per la gestione di dati nel formato di immagini nell'industria; non se ne discutono i dettagli in particolare, ma si forniscono alcuni esempi generali dei contesti applicativi considerati assieme al relativo approccio adottato.

I metodi univariati sono tra i primi ad essere stati sviluppati per poter gestire immagini binarie o in scala di grigi. In origine, Horst e Negin (1992) hanno discusso i vantaggi dell'utilizzo di carte di controllo con dati in formato di immagine in ambito industriale; in seguito, gli approcci di questo tipo che sono stati sviluppati prevedevano generalmente l'impiego di carte Shewhart

combinate. Grazie a questi schemi è possibile, ad esempio, impiegare dei sistemi di visione artificiale per campionare e misurare le caratteristiche di qualità (lunghezza, larghezza ed area) di un prodotto al fine di sorvegliarne statisticamente la produzione. Le carte di controllo combinate possono poi essere costruite per sorvegliare ogni pixel in modo individuale, disegnando tante carte di controllo quanti sono i pixel da sorvegliare. La costruzione di uno schema di sorveglianza per ogni pixel, tuttavia, non tiene conto della correlazione tra pixel vicini e il numero risultante di grafici di controllo può richiedere una quantità eccessiva di tempo computazionale. L'approccio univariato può essere preso in considerazione anche nel caso di immagini che non siano esclusivamente in scala di grigi: conoscendo infatti le tonalità che un'immagine acquisita può assumere, è possibile calcolare la differenza, per ogni singolo pixel, tra il valore registrato e quello previsto dalle specifiche stabilite per il processo. Sorvegliando tali errori è quindi possibile costruire ancora una volta uno schema di sorveglianza composto. Tuttavia, qualora venisse registrata una nuova tonalità di colore nelle immagini analizzate, questo approccio fornirebbe risultati distorti, non riuscendo ad adattarsi al nuovo valore non previsto. Infine, grazie agli approcci univariati è altresì possibile sorvegliare lo stato di usura degli strumenti di produzione, acquisendo immagini digitali degli oggetti di interesse opportunamente elaborate e private dello sfondo. Degli esempi riguardanti gli approcci descritti possono essere ritrovati in Tan, Chang et al. (1996), Armingol, Otamendi et al. (2003), Nembhard, Ferrier et al. (2003) e Liang e Chiou (2008).

Un altro tipo di approccio che estende naturalmente quello appena descritto nell'ambito univariato è dato dalle carte di controllo multivariate. In quest'ambito, spesso gli schemi impiegati, specialmente in ambito industriale, fanno uso di statistiche di controllo di tipo T^2 di Hotelling, sebbene le finalità di ogni approccio cambino a seconda del contesto applicativo. Un esempio è dato dalla sorveglianza dei difetti dei circuiti integrati di una mappa di *wafer*. Un *wafer* è l'elemento alla base della produzione di semiconduttori, in cui diverse centinaia di circuiti integrati sono montati contemporaneamen-



(a) (b)

Figura 2: Esempio di come appaiono (a) un *wafer* e (b) una mappa di *wafer*.

te su un *wafer*, mentre una mappa di *wafer* è una rappresentazione grafica utilizzata per mostrare le posizioni dei chip difettosi sul *wafer*. Per chiarezza, in Figura 2 sono riportati degli esempi di come appaiono un *wafer* (Figura 2(a)) e una mappa di *wafer* (Figura 2(b)). La logica alla base dell'adozione della carta di controllo T^2 di Hotelling in questo contesto invece di un tradizionale schema di controllo basato sui conteggi di Poisson si basa sul fatto che le posizioni dei difetti dei circuiti integrati su una mappa di *wafer* sono altamente correlate, poiché i chip mal funzionanti spesso si trovano in *cluster* o comunque sono disposti secondo schemi sistematici e non casuali (Megahed, Woodall et al., 2011). Il *pattern* delle anomalie rilevate inoltre può riflettere le cause dei difetti dei circuiti integrati. Spesso, i conteggi dei difetti in questo contesto non possono essere modellati da una distribuzione di Poisson e le carte basate su statistiche aventi questa distribuzione possono avere alti tassi di falso allarme. Il vantaggio di una carta di controllo multivariata sta dunque nel riuscire ad analizzare l'interazione tra il il numero di difetti e il loro raggruppamento, sorvegliando in modo efficiente le condizioni del processo di produzione dei *wafer*. L'estetica di un prodotto può anch'essa essere sorvegliata da uno schema multivariato. Impiegando

un apposito sistema di visione artificiale per rilevare e analizzare l'aspetto di prodotti fabbricati, è possibile infatti sorvegliare la qualità di oggetti il cui aspetto è contraddistinto dalla presenza di motivi particolari (come strisce o increspature), con caratteristiche variabili nelle loro forme e dimensioni. Dalle immagini acquisite è infatti possibile ottenere una stima quantitativa (multivariata) che esprime la qualità del prodotto analizzato, la quale può essere quindi sintetizzata da un'opportuna statistica T^2 di Hotelling, che consenta di stabilire se il processo stia generando oggetti conformi alle specifiche prestabilite. Un altro esempio di un contesto applicativo della sorveglianza per immagini tramite approcci multivariati può essere dato dalla progettare un sistema di ispezione visiva automatizzato, che potrebbe essere utilizzato nei sistemi di produzione di massa come parte integrante del processo di ispezione. Integrando tecnologie di elaborazione delle immagini e carte di controllo per processi multivariati, è infatti possibile registrare le caratteristiche di qualità d'interesse e riassumerle in modo adeguato tramite una statistica di controllo multivariata (di tipo T^2 di Hotelling o *MEWMA*). La carta che ne risulta avrà di conseguenza limiti di controllo calcolati sulla base di dati in controllo disponibili offline. Per degli approfondimenti in merito agli approcci appena accennati si rimanda a Tong, C. H. Wang et al. (2005), J. J. Liu e MacGregor (2006), Lin (2007a,b) e Lyu e Chen (2009).

La sorveglianza delle immagini può poi essere considerata come una naturale estensione dei metodi di sorveglianza di profili ai casi in cui le variabili esplicative indicano la posizione delle misurazioni di intensità (dei pixel ad esempio) all'interno dell'immagine. La sorveglianza di profili viene impiegata quando la qualità di un processo o di un prodotto può essere espressa in termini di una relazione tra una variabile risposta e una o più variabili esplicative. Ad ogni istante, i dati osservati possono essere descritti tramite una relazione, esprimibile ad esempio attraverso semplici metodi lineari, non lineari o non parametrici. In queste situazioni, invece di conservare in memoria e sorvegliare e l'intero set di osservazioni, è sufficiente sorvegliare i parametri del modello adattato che descrive la relazione di dipendenza tra le variabili.

Nelle applicazioni in esame che coinvolgono immagini, si tendono ad avere molti più dati, assieme ad una componente spaziale intrinseca di cui non è semplice tener conto e che richiede metodi di analisi più complessi. Nella sorveglianza di profili si deve anche decidere se una particolare caratteristica della relazione debba essere considerata o se l'intera funzione debba essere sorvegliata. Per sorvegliare l'intera funzione del profilo, è possibile modellare la relazione e adottare approcci diversi a seconda che vengano utilizzati metodi parametrici o non parametrici. Woodall e Montgomery (2014), Woodall, Spitzner et al. (2004) e Woodall (2007) hanno fornito revisioni dettagliate in merito a tale argomento, con spiegazioni di diverse applicazioni possibili per la sorveglianza di profili. Alcune applicazioni includono immagini 2D ad alta dimensionalità (K. Wang e Tsung, 2005) e scansioni di superfici 3D (Wells, Megahed et al., 2013). Un esempio di sorveglianza di immagini tramite profili può essere dato dall'impiego di un *Q-Q plot* che riflette la relazione tra l'immagine oggetto di analisi e un'immagine considerata come *gold standard*. Esso viene quindi usato per rilevare dei cambiamenti nelle immagini analizzate rispetto ad un'immagine "ideale". Tale approccio può essere usato ad esempio per rilevare difetti nei pannelli LCD per i display dei cellulari, anche se tale metodologia può servire solo come strumento per constatare la presenza di difetti, senza avere informazioni circa alla loro posizione o dimensione. Un altro contesto applicativo è dato, ad esempio, dalla necessità di sorvegliare la qualità di oggetti metallici prodotti tramite tornitura. I dati in tale contesto fanno infatti riferimento ad elementi tridimensionali che vanno analizzati in diversi aspetti qualitativi e che richiedono una particolare attenzione. Una possibilità è quella di combinare un modello di regressione con termine d'errore spazialmente correlato (per tener conto della struttura intrinseca dell'oggetto analizzato) con carte di controllo univariate e multivariate. Un'area di interesse per questo tipo d'approccio è infine dato dalla sorveglianza delle forme dei prodotti, dato che la forma degli oggetti fabbricati è spesso un aspetto piuttosto importante per la qualità degli stessi. Gli approcci di sorveglianza dei profili che utilizzano la forma degli oggetti come

tipologia di dato possono essere più efficienti dei metodi ingegneristici standard, perché in molti casi questi approcci non utilizzano tutte le informazioni presenti nei dati. Per ulteriori dettagli in merito agli esempi qui accennati si rimanda a K. Wang e Tsung (2005), Colosimo, Mammarella et al. (2010) e Woodall (2007).

Un metodo alternativo e diverso dai comuni sistemi di sorveglianza statistica è poi dato dalle carte di controllo spaziali. Gli autori delle procedure discusse nel seguito hanno infatti proposto carte di controllo "non standard", poiché, graficamente, l'asse orizzontale di ogni carta di controllo rappresenta una posizione nell'immagine e non l'istante di osservazione di quest'ultima. Le carte di controllo di questo tipo sono usate spazialmente spostando una maschera (o finestra) attraverso l'immagine, calcolando e riportando graficamente una statistica ogni volta che la maschera viene spostata. Solitamente le posizioni della maschera non si sovrappongono e viene implicitamente assunto che i difetti abbiano la stessa probabilità di verificarsi in una regione come in qualsiasi altra della stessa dimensione. La dimensione della maschera dipende dalla dimensione prevista dei difetti da rilevare, con regioni difettose più piccole che richiedono dunque maschere di dimensioni minori. Le carte di controllo spaziali, ad esempio, possono essere utili nei casi in cui si desideri ottenere una maggiore sensibilità in alcune posizioni specifiche delle immagini in cui è più probabile che si verifichino difetti. Un esempio in quest'ambito è dato dalla necessità di sorvegliare l'uniformità dei monitor LCD di alta qualità per rilevare il tipo, la dimensione e la posizione di eventuali difetti. Dividendo il pannello LCD in un certo numero di regioni diverse e ciascuna con un'area di dimensioni prefissate, è possibile confrontare ogni singola regione con le altre per identificare le aree significativamente diverse dalle altre aree in un pannello. Nonostante la capacità offerta da questa procedura di identificare il tipo, la posizione e la dimensione del difetto, risulta comunque difficile rilevare un'area difettosa che si sovrappone parzialmente a due o più aree di rilevamento. Un altro contesto nel quale si possono impiegare carte di controllo spaziali è dato dall'analisi di tessuti, per rilevare possibili difetti

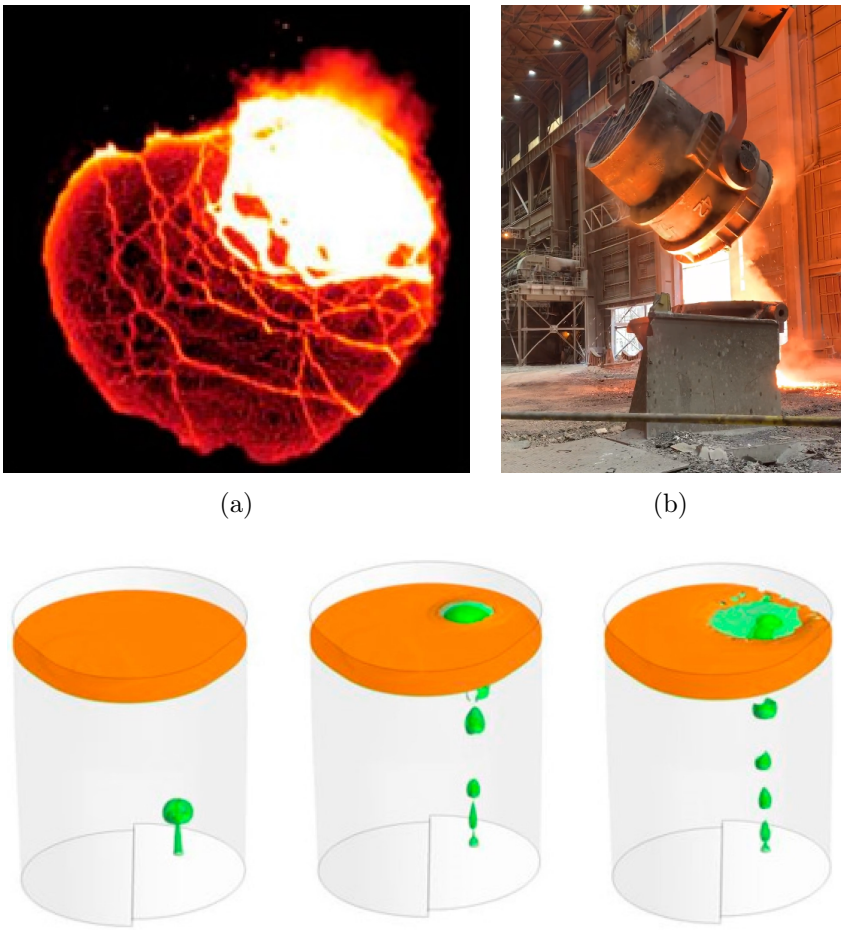
nella trama, assieme alla loro posizione. In tale contesto, possono essere usate tecniche di classificazione e regressioni con alberi decisionali per estrarre le caratteristiche significative che descrivono l'immagine analizzata. Dato che la presenza di un difetto nella *texture* provoca cambiamenti strutturali regolari, è possibile usare carta di controllo T^2 di Hotelling spaziale per combinare più caratteristiche della trama del tessuto e per rilevare i difetti. Usando questo metodo, è possibile rilevare i difetti associati al cambiamento della densità di tessitura, distribuiti lungo la larghezza o l'altezza di un'immagine. Inoltre, se viene eseguito un miglioramento del contrasto per aumentare la distinzione tra il tessuto e lo sfondo nell'immagine acquisita e si ruotano in modo opportuno queste ultime in modo che siano correttamente allineate, è possibile migliorare ulteriormente l'analisi e la sorveglianza che ne segue. È quindi possibile utilizzare una carta di controllo $\bar{x}$ spaziale sulla densità di tessitura come strumento per trovare regioni che sembrano contenere plausibilmente dei difetti. Per degli approfondimenti degli approcci e dei contesti qui descritti si può fare riferimento a Jiang, C.-C. Wang et al. (2005), Lu e Tsai (2004), Tunák e Linka (2008) e Tunák, Linka e Volf (2009).

Infine, un ultimo approccio disponibile per la sorveglianza delle immagini è dato da ciò che in letteratura viene definito come analisi multivariata delle immagini o, più brevemente, *MIA*. Essa descrive un insieme di tecniche che impiegano metodi statistici multivariati, come l'analisi delle componenti principali o i minimi quadrati parziali, per analizzare le immagini (Kourti, 2005). Prima di qualsiasi forma di sorveglianza statistica infatti c'è solitamente una fase di riduzione dimensionale, basata ad esempio sull'analisi delle componenti principali. Con la *multivariate image analysis* la maggior parte dell'analisi viene svolta nello spazio delle variabili latenti piuttosto che nello spazio dell'immagine. L'obiettivo dell'approccio, infatti, è quello di estrarre informazioni dall'immagine che sono legate alla qualità del prodotto e utilizzare tali informazioni per la sorveglianza. Questo approccio è diverso dalla tradizionale elaborazione delle immagini digitali, che comporta metodi per alterare l'immagine originale in qualche modo al fine di renderla utilizzabile

(ad esempio convertendola in scala di grigi) o per estrarre informazioni sui bordi dei soggetti o ancora sulla posizione di varie caratteristiche osservabili. Nel contesto *MIA* le immagini prevedono misurazioni effettuate per almeno tre diverse lunghezze d'onda (Megahed, Woodall et al., 2011); ciò significa che generalmente le immagini analizzate in quest'ambito possono anche essere a colori e trattate come tali senza doverle convertire in altri formati, o ancora possono essere immagini contenenti anche informazioni circa bande dello spettro visivo non percepibili a occhio nudo. Un esempio a tal proposito è dato dalle immagini spettrali che coinvolgono tre o più lunghezze d'onda. Aggregando i dati sulle posizioni dei pixel e usando la *PCA* per identificare le componenti principali che spiegano la maggior parte della variazione nei vettori delle intensità delle lunghezza d'onda, è possibile rintracciare i cambiamenti mostrandone le posizioni sull'immagine originale. In questo modo si riesce ad identificare il cambiamento nella frequenza e nella posizione dei pixel dell'immagine di partenza, facilitando così l'aspetto dell'analisi post-segnale e delle cause che hanno portato alla generazione del cambiamento. A tal proposito, Megahed, Woodall et al. (2011) hanno discusso l'uso dei metodi di proiezione con carte di controllo nel contesto dell'analisi multivariata delle immagini. In un'applicazione in ambito industriale, è possibile ad esempio impiegare la metodologia *MIA* per il controllo della qualità degli snack confezionati in linee di produzione. Basandosi ancora una volta sull'analisi delle componenti principali, si possono sviluppare dei modelli per prevedere il contenuto e la distribuzione del rivestimento sui prodotti. Ulteriori esempi in ambito industriale che possono richiedere un approccio *MIA* possono essere dati dalla necessità di sorvegliare l'intensità delle fiamme generate in una caldaia, per prevederne le prestazioni, o ancora l'esigenza di sorvegliare particolari processi produttivi, come quello di flottazione, uno dei procedimenti più utilizzati nella lavorazione dei minerali per separare i metalli preziosi dal minerale. In entrambi i casi, metodologie basate sull'analisi dei minimi quadrati parziali o delle componenti principali consentono di descrivere adeguatamente i dati nel formato di immagine acquisiti e di disegnare

di conseguenza un opportuno schema di sorveglianza. La *PCA* nell'ambito della *multivariate image analysis* può ancora essere impiegata per misurare e prevedere il fenomeno dell'"occhio di scoria" (*ladle eye phenomenon*) nella metallurgia (Figura 3(a)). Questo fenomeno si verifica dopo che la colata di metallo fuso è stata raccolta nella siviera (un recipiente preposto a contenere il metallo fuso, come in Figura 3(b)), quando del gas viene iniettato nel bagno di acciaio fuso attraverso uno o più fori posti nel fondo del contenitore per favorire la separazione tra metallo e scorie. La colonna di bolle generata risale in superficie e spinge lo strato di scorie verso la periferia della siviera. Se lo spessore dello strato di scorie in superficie è sufficientemente sottile, un'area di acciaio fuso si ritrova ad essere esposta all'atmosfera (Figura 3(c)). L'area di metallo fuso esposto in seguito a queste reazioni viene definito "occhio di scoria", ed è un elemento importante nella metallurgia, poiché l'"occhio" è un sito per la raccolta di ossigeno e azoto e le reazioni di raffinazione con le scorie sono limitate all'area di contatto scoria-metallo (Z. Liu, Li et al., 2017). Dopo una fase iniziale di analisi multivariata delle immagini acquisite, è possibile usare una carta di controllo T^2 di Hotelling per sorvegliare la validità delle previsioni effettuate (sulla base dei modelli fisici) dell'area dell'"occhio di scoria" e per valutare la presenza di valori anomali e/o una pre-elaborazione dell'immagine non adeguata o soddisfacente. Un ultimo esempio interessante è dato infine dall'uso dell'analisi delle componenti principali e della carta di controllo T^2 di Hotelling per sorvegliare il movimento dei maiali attraverso sequenze video. Ulteriori approfondimenti ed esempi di approcci *MIA* che fanno uso dell'analisi delle componenti principali o dei minimi quadrati parziali possono essere ritrovati in Bharati e MacGregor (1998), Yu, MacGregor et al. (2003), Yu e MacGregor (2004), J. J. Liu, MacGregor et al. (2005), Graham, Krishnapisharody et al. (2007), Facco, Bezzo et al. (2008) e Gronskyte, Kulahci et al. (2013)

Tra gli approcci *MIA* sviluppati nell'*SPC*, due di questi in particolare sono ritenuti di interesse per la trattazione che segue nei Capitoli successivi. Ciò che più accomuna questi due procedimenti consiste nel fatto che questi



(a) (b)

(c)

Figura 3: Le immagini riportano (a) la vista dall'alto di un "occhio di scoria", (b) una siviera mentre viene svuotata e (c) un'illustrazione di come il gas (in verde) immesso nella siviera risalga in superficie per creare l'"occhio". Fonte: Ramasetti, Visuri et al., 2019.

sono tra i primi, nella letteratura, ad estendere degli approcci esistenti per la sorveglianza statistica di immagini includendo anche l'informazione spaziale intrinsecamente contenuta in esse. Come verrà esposto in seguito, riuscire ad includere tale informazione nella modellazione e nella trattazione del problema della sorveglianza statistica di immagini comporta diversi miglioramenti in termini di reattività ed efficacia delle carte di controllo, riuscendo ad identificare più rapidamente e con maggior precisione se, quando e dove un evento fuori controllo si sia verificato. Entrambi i metodi possono essere classificati come approcci *MIA* di Fase II (online) alla sorveglianza di immagini, in quanto assumono note o stimate in modo sufficientemente accurato alcuni dei parametri che governano la dinamica del processo generatore dei dati ed inoltre perché operano una riduzione della dimensionalità dei dati oggetto della sorveglianza al fine di sorvegliare il processo, sebbene questo avvenga in modi differenti.

Nel caso del metodo proposto in Colosimo e Grasso (2018), la riduzione dimensionale dell'immagine avviene attraverso l'uso di una particolare formulazione delle componenti principali, specificamente sviluppata per il nuovo metodo proposto e che può essere vista come un'estensione spaziale di un metodo già esistente. Inoltre, tale metodologia viene sviluppata ed applicata alla sorveglianza di video, il che rappresenta un'ulteriore novità in quanto non erano presenti in letteratura dei procedimenti in grado di gestire tale tipologia di dati nel contesto della sorveglianza statistica. Tale metodologia viene descritta nel Capitolo 3.1.

Il metodo proposto da A. Wang, Xian et al. (2018) viene sviluppato in un contesto nel quale, ad ogni istante, l'intera immagine non è osservabile (a causa di vincoli *hardware*) e dunque non è possibile in nessun caso disporre della totalità dei dati per la sorveglianza del processo. La riduzione dimensionale viene quindi effettuata "in senso lato", scegliendo in modo intelligente quali aree dell'immagine sorvegliare, grazie ad un procedimento di campionamento adattivo che riesce ad identificare in quali regioni dell'immagine è più plausibile si sia verificato (o si stia verificando) un evento che potrebbe

portare il processo ad andare fuori controllo. In questo modo il numero di variabili considerate per la sorveglianza viene di fatto ridotto notevolmente senza che venga applicato uno dei metodi classici (o una sua estensione, come per l'altro metodo considerato) tra quelli presentati brevemente nel Capitolo 2.1. Questo secondo approccio viene descritto nel Capitolo 3.2.

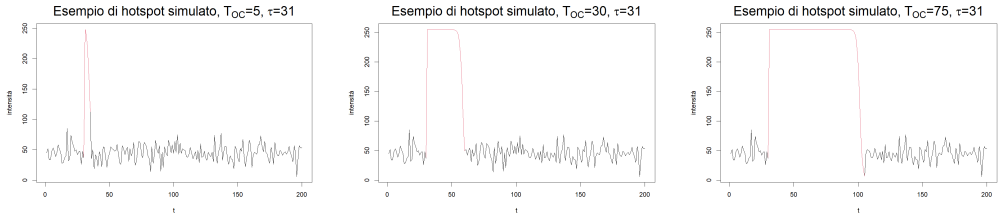
3 Disegno delle carte di controllo confrontate

Lo scopo della presente trattazione vuole essere quello di porre a confronto, per la prima volta in letteratura, il comportamento delle due carte di controllo presentate nei Capitoli 3.1 e 3.2, al fine di capire quali siano i vantaggi e gli svantaggi derivanti dall'utilizzo di una procedura piuttosto che l'altra. Per fare ciò, i due algoritmi vengono applicati ad un *dataset* comune; esso dovrà essere generato partendo dalle seguenti assunzioni. I dati consistono quindi in una sequenza video con *frames* in scala di grigi a 8 bit nella quale ad un certo istante τ si verifica un fenomeno anomalo. Tale anomalia può essere descritta come un evento isolato che porta un *cluster* di pixel vicini tra loro ad assumere valori di intensità irregolari rispetto a quelli previsti. Dunque, l'*hotspot* fuori controllo che si genera a partire dall'istante τ può essere descritto, ad esempio, tramite la seguente funzione sigmoidale (Colosimo e Grasso, 2018):

$$u_{m_{OC},n_{OC},j}(T_{OC}) = \frac{255}{1 + exp(0.95(j - 0.95T_{OC}))} \qquad j = 1, \ldots, T_{OC} \qquad (1)$$

dove T_{OC} è la durata del fenomeno fuori controllo e (m_{OC}, n_{OC}) è la posizione del pixel interessato dall'anomalia. La funzione prevede che il pixel in posizione (m_{OC}, n_{OC}) assuma un valore estremo di luminosità che decresce esponenzialmente per i successivi T_{OC} istanti di osservazione, fino a tornare ai valori in controllo. Un esempio dell'andamento dei valori di luminosità di un pixel soggetto ad un evento anomalo è riportato in Figura 4. In particolare qui si nota come la durata T_{OC} dell'*hotspot* influenzi sia la velocità che l'andamento con cui i valori osservati tornano a quelli in controllo: all'aumentare della persistenza dell'anomalia, aumenta il tempo per il quale il pixel interessato continua ad assumere valori anomali prima di tornare alla normalità.

Le due carte di controllo *ST-PCA* e *SASAM* vengono quindi impiegate per rilevare nella sequenza video l'insorgenza di un *hotspot*. Appare lecito assumere in questo contesto che ciò che potrebbe influenzare il comportamento delle carte è da identificarsi nella dimensione dei *frames* del video, la dimensione e la durata dell'anomalia. Al crescere della dimensione delle im-



(a) (b) (c)

Figura 4: Esempi di andamenti di intensità di un pixel soggetto ad un evento fuori controllo in $\tau = 31$ e della durata di T_{OC} =5, 30 e 75 rispettivamente per le Figure (a), (b) e (c). In rosso è evidenziato il periodo di tempo per il quale si osservano valori fuori controllo.

magini infatti aumenta la complessità dei dati da osservare, che richiedono un carico computazionale crescente. Bisogna dunque valutare se al crescere del numero totale p di pixel di ciascun *frame* si osservano performance costanti in termini computazionali o meno. D'altro canto, un'anomalia troppo breve o troppo circoscritta potrebbe essere difficile da rilevare, impedendo così il lancio di un allarme da parte delle procedure. Questi aspetti vanno quindi tenuti in considerazione per poter effettuare un confronto soddisfacente e per poter prendere in considerazione ogni scenario verosimilmente possibile.

3.1 *Spatially weighted T-mode Principal Component Analysis* (*ST-PCA*)

Il metodo proposto in Colosimo e Grasso (2018) si colloca nel contesto della sorveglianza di *streams* di immagini che vengono acquisite con un'alta frequenza grazie a sistemi di visione artificiale adeguati. Il caso di studio che ha motivato lo sviluppo di tale metodologia riguarda la produzione additiva a letto di polvere di metallo. Questo processo è guidato in modo digitale da programmi adeguati che riescono a definire le diverse sezioni dell'oggetto 3D che deve essere prodotto. Per ciascuna sezione, viene depositato su una piastra di base un sottile strato uniforme (di 30-50 μm di spessore) di polvere

metallica finissima, quindi una fonte di energia (fascio di elettroni o raggio laser) fornisce energia alla superficie fondendo o sinterizzando con precisione la polvere nella forma desiderata. Il processo viene ripetuto, depositando ulteriormente altra polvere strato dopo strato, finché il pezzo non è completo. Tra i vantaggi principali di questo metodo di produzione di parti metalliche vi sono (1) la riduzione dei costi delle attrezzature, fabbricando direttamente i pezzi senza bisogno di strumenti specifici altrimenti indispensabili; (2) la possibilità di ottenere parti dalle geometrie complesse; (3) l'opportunità di apportare cambiamenti al progetto rapidi e anche direttamente nel corso della produzione; (4) la compatibilità del metodo di essere integrato in altri processi di produzione (*Cos'è la produzione additiva in metallo?* 2021). La stabilità e la ripetibilità limitate del processo rappresentano ancora un grande ostacolo per l'adozione su vasta scala in ambito industriale di questa tecnologia (Colosimo e Grasso, 2018). Infatti, vari tipi di difetti possono originarsi durante il processo, a discapito della qualità del prodotto finale. In questo contesto, la visione artificiale ad alta velocità è potenzialmente in grado di identificare i fenomeni termici legati all'interazione laser-materiale che hanno luogo in lassi di tempo brevissimi, e di rilevare quindi l'insorgenza di difetti. In Figura 5 sono riportati degli esempi della tipologia di immagini che possono essere acquisite durante questo processo. L'obiettivo quindi è quello di individuare nel minor tempo possibile gli *hotspots* nei quali si sia verificato un surriscaldamento locale anomalo.

Assumendo dunque che vengano acquisite delle immagini con una frequenza f che renda possibile cogliere questi rapidi fenomeni, lo *stream* di un set di immagini può essere visto come un *array* a tre dimensioni, sia questo $U = \{\boldsymbol{U}_1, \boldsymbol{U}_2, \ldots, \boldsymbol{U}_J\} \in \mathbb{R}^{J \times M \times N}$. Qui $M \times N$ corrisponde alla dimensione (in numero di pixel) di ogni *frame* e J è il numero complessivo di immagini acquisite in un periodo di tempo pari a $T = J/f$. $\boldsymbol{U}_j \in \mathbb{R}^{M \times N}$ rappresenta la j-esima immagine, con $j = 1, \ldots, J$ e $\boldsymbol{u}_{m,n} = \{u_{m,n,1}, u_{m,n,2}, \ldots, u_{m,n,J}\}$ è il profilo delle intensità del pixel in posizione (m, n) sui J fotogrammi acquisiti, con $m = 1, \ldots, M$ e $n = 1, \ldots, N$. Per delle immagini ad 8 bit, la gamma

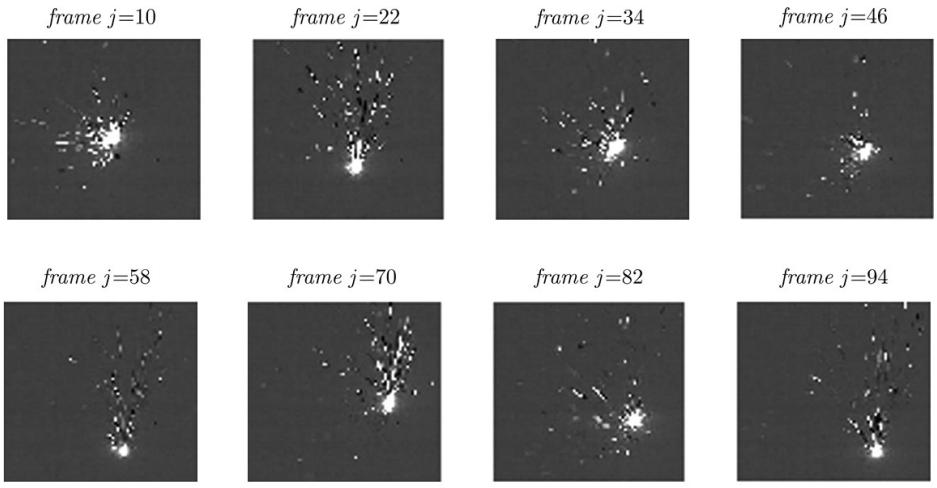


Figura 5: Esempi di *frames* da un video acquisito durante il processo di produzione a letto di polvere di metallo di un oggetto.

di intensità dei pixel è $u_{m,n,j} \in [0, 255]$. Inoltre, considerando in particolare il contesto specifico nel quale la nuova metodologia viene sviluppata, i pixel possono essere divisi in tre categorie: (1) pixel di sfondo, appartenenti a regioni in cui non si è verificata alcuna interazione laser-materiale, (2) pixel appartenenti alla zona di fusione normale, dove si è verificato un processo di saldatura in controllo, e (3) pixel *hotspot*, dove si è verificato un surriscaldamento locale fuori controllo. Tale classificazione, pur essendo riferita ad un caso di studio specifico, può essere generalizzata ed applicata a diversi contesti analoghi, in cui l'obiettivo sia quello di identificare *hotspots* anomali.

Al fine di poter identificare i possibili difetti, l'approccio comune consiste nell'adottare un metodo basato sull'analisi delle componenti principali, definito come *S-mode PCA*, in cui i pixel dell'immagine sono trattati come variabili ed i *frames* che vengono acquisiti ad ogni istante sono le osservazioni. In altre parole, nell'*S-mode PCA*, l'*array U* viene convertito nella matrice dei dati $\boldsymbol{X} \in \mathbb{R}^{J \times p}$, con $p = M \times N$, in cui ogni colonna identifica uno dei p pixel e ogni riga corrisponde ad un'immagine. Questo approccio viene

definito anche come analisi delle componenti principali vettorizzata o similmente *unfolded PCA*. L'*S-mode PCA* riesce a cogliere la correlazione spaziale dei pixel nel senso che riesce a stabilire quanto ogni *frame* risulti anomalo rispetto agli altri acquisiti; tuttavia, se si dovesse identificare un'immagine classificata come fuori controllo, il metodo non riuscirebbe in alcun modo a localizzare dove il difetto si sia verificato. L'informazione spaziale intrinseca dei dati quindi, nonostante non venga completamente trascurata, non viene sfruttata fino in fondo; questo si riflette nell'impossibilità di stabilire cosa abbia portato il processo ad andare fuori controllo, rendendo difficile l'adozione di correzioni utili al processo nel caso in cui un allarme venga lanciato.

Un approccio alternativo, chiamato *T-mode PCA*, cerca di colmare quest'ultima mancanza dell'*S-mode PCA* cercando di fornire anche l'indicazione spaziale circa il difetto che si viene a generare qualora venga colto e venga lanciato un allarme. Tale metodo prevede che U venga convertito nella matrice dei dati $\boldsymbol{X} \in \mathbb{R}^{p \times J}$, in cui le immagini acquisite sono trattate come variabili (ogni colonna di $\boldsymbol{X}$ identifica un *frame*) ed i pixel di ogni immagine acquisita rappresentano le osservazioni (ogni riga di $\boldsymbol{X}$ identifica un pixel). Questa definizione permette di cogliere l'autocorrelazione delle intensità di ciascun pixel sulle immagini ottenute. Le proiezioni dei dati originali sulle componenti principali che si decide di conservare appartengono allo spazio dell'immagine (dal momento che queste erano qui considerate come variabili) e quindi permettono una localizzazione spaziale delle regioni anomale. Tuttavia, seguendo questo approccio si perde l'informazione riguardante la correlazione spaziale tra i pixel dell'immagine, poiché la distanza effettiva tra questi ultimi non viene presa in considerazione. In Figura 6 è riportata in modo schematico la differenza nella conversione dell'*array U* effettuata dall'*S-mode PCA* e dalla *T-mode PCA*.

Con l'obiettivo quindi di cogliere sia la correlazione spaziale che temporale dei pixel contenuta nei dati, viene proposto un nuovo metodo, chiamato *ST-PCA* (*Spatially weighted T-mode PCA*). L'idea chiave di questo metodo consiste nell'incorporare la correlazione spaziale nell'operazione di proiezione

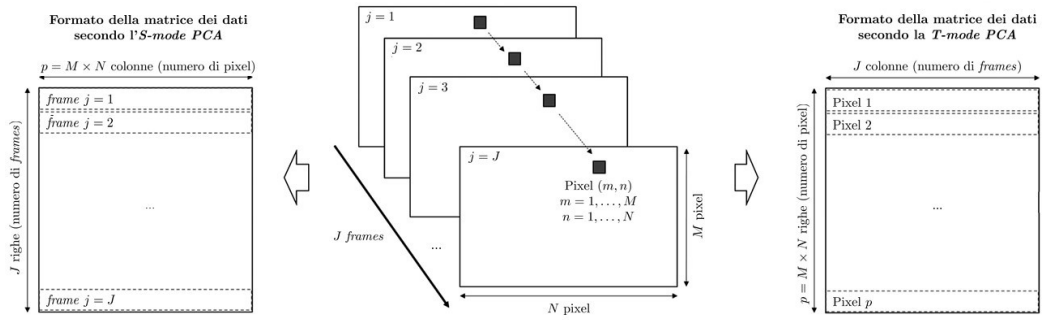


Figura 6: Conversione dell'*array U* secondo l'*S-mode PCA* (a sinistra) e la *T-mode PCA* (a destra).

che viene effettuata nella *T-mode PCA*. Tale informazione spaziale può essere incorporata nell'analisi delle componenti principali definendo una matrice quadrata $p \times p$ di pesi spaziali, $\boldsymbol{W}$. L'elemento in posizione (k, h) della matrice, $w_{k,h}$, quantifica la dipendenza spaziale tra il k-esimo e l'h-esimo pixel; quanto più grande è questo valore, tanto maggiore è la dipendenza.

L'idea di utilizzare dei pesi per definire la correlazione spaziale tra le osservazioni originali nel contesto dell'analisi delle componenti principali era già stata proposta. In tal senso, particolare attenzione va posta sull'approccio proposto in Stahlschmidt, Härdle et al. (2015), in cui gli autori suggeriscono un'estensione spazio-temporale della *PCA*, in cui il problema di massimizzazione che porta alla definizione delle componenti principali viene riformulato considerando sia la correlazione spaziale che temporale dei dati.

Per chiarezza espositiva, occorre richiamare brevemente la formulazione del problema nella definizione classica della *PCA*. Sia $\boldsymbol{Y} \in \mathbb{R}^p$ un vettore p-dimensionale di variabili casuali (centrate). L'obiettivo nell'analisi delle componenti principali consiste nel proiettare $\boldsymbol{Y}$ in nuovo sistema di coordinate tramite una combinazione lineare, definita come $\boldsymbol{\Phi} = \boldsymbol{Y}\boldsymbol{A}$, con $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ matrice di *loadings* di dimensione $p \times m_{pc}$, $m_{pc} \leq p$. Se m_{pc} è strettamente minore della dimensione p di partenza si ottiene una riduzione dimensionale. Le proiezioni $\boldsymbol{\phi}_1$ sulla prima coordinata sono ottenute tramite $\boldsymbol{\phi}_1 = \boldsymbol{Y}\boldsymbol{a}_1$, dove $\boldsymbol{a}_1$ può essere identificato grazie alla richiesta di massimiz-

zazione della varianza dei dati proiettati in questa nuova coordinata; tale condizione può essere espressa come:

$$\max_{a_1} \text{Var}(\boldsymbol{\phi}_1) = \max_{a_1} \text{Var}(\boldsymbol{Y}\boldsymbol{a}_1) = \max_{a_1} \boldsymbol{a}_1^\top n^{-1} \boldsymbol{Y}^\top \boldsymbol{Y}\boldsymbol{a}_1 = \max_{a_1} \boldsymbol{a}_1^\top \boldsymbol{\Sigma}\boldsymbol{a}_1 \quad (2)$$

imponendo il vincolo $\|\boldsymbol{a}_1\| = 1$ che garantisce l'identificabilità della soluzione e dove $\boldsymbol{\Sigma}$ indica la matrice di varianza e covarianza delle variabili centrate. La decomposizione spettrale di $\boldsymbol{\Sigma}$ permette la risoluzione del problema di ottimizzazione in Equazione (2), dal momento che l'autovettore associato all'autovalore più grande di $\boldsymbol{\Sigma}$ rappresenta il valore ottimale di $\boldsymbol{a}_1$. Similmente, le rimanenti colonne di $\boldsymbol{A}$ corrispondono agli autovettori rimanenti, in modo tale che la seconda colonna coincida con l'autovettore associato al secondo autovalore più grande e così via. Gli autovalori associati a ciascun autovettore indicano inoltre la percentuale di varianza spiegata da ciascuna componente principale.

Una prima estensione spaziale della *PCA* prevede l'introduzione nell'Equazione (2) di un termine in grado di tener conto della correlazione spaziale tra le osservazioni originali. Si definisce allora una statistica di autocorrelazione spaziale $I(\cdot)$ in grado di cogliere tale informazione, funzione sia dei dati a disposizione che di una matrice $\boldsymbol{W} = \{w_{r,s}\}_{r,s=1,\dots,N}$ di pesi che descrivono l'influenza spaziale dell'osservazione r sull'osservazione s, nel caso si abbiano N osservazioni disponibili. Il problema di massimizzazione può essere così espresso:

$$\begin{aligned} \max_{a_1} \text{Var}(\boldsymbol{\phi}_1) I(\boldsymbol{\phi}_1) &= \max_{a_1} \text{Var}(\boldsymbol{Y}\boldsymbol{a}_1) I(\boldsymbol{Y}\boldsymbol{a}_1) \\ &= \max_{a_1} \boldsymbol{a}_1^\top n^{-1} \boldsymbol{Y}^\top \boldsymbol{W}\boldsymbol{Y}\boldsymbol{a}_1 \\ &= \max_{a_1} \boldsymbol{a}_1^\top \boldsymbol{\Omega}\boldsymbol{a}_1 \end{aligned} \quad (3)$$

con $\boldsymbol{\Omega} = n^{-1}\boldsymbol{Y}^\top \boldsymbol{W}\boldsymbol{Y}$ che indica la matrice di correlazione spaziale e $\boldsymbol{W} = \{w_{r,s}\}_{r,s=1,\dots,N}$ è una matrice di pesi che descrivono il peso spaziale dell'osservazione r sull'osservazione s, nel caso si abbiano N osservazioni disponibili. Come nel caso precedente, il valore ottimale di $\boldsymbol{a}_1$ (e degli altri vettori $\boldsymbol{a}_j$,

$j = 1, \ldots, m$) può essere ottenuto grazie ad una decomposizione spettrale di $\boldsymbol{\Omega}$.

Nel caso si abbia a che fare con dati spazio-temporali, la *PCA* o qualsiasi sua variante spaziale potrebbe essere applicata ad ogni istante t di osservazione. In questo modo tuttavia qualsiasi correlazione temporale verrebbe ignorata e ad ogni istante si opererebbe un'analisi separata. L'estensione spazio-temporale dell'analisi delle componenti principali proposta da Stahlschmidt, Härdle et al. prevede invece il calcolo di una media temporale della matrice di correlazione spaziale a cui poi si applica una decomposizione spettrale. Di conseguenza, se da un lato si sfrutta la correlazione temporale, dall'altro ci si avvale del fatto che le misurazioni ripetute sulle unità spaziali (stabili nel tempo) rappresentano lo stesso contenuto informativo, mentre qualsiasi rumore aggiuntivo potrebbe variare nel tempo. Quindi la matrice di correlazione spaziale mediata nel tempo includerà un rapporto segnale-rumore più alto e presenterà autovettori stabili nel tempo (Stahlschmidt, Härdle et al., 2015). Questo procedimento risulta inoltre più veloce di qualsiasi applicazione ripetuta della *PCA* o di una sua variante spaziale, in quanto la decomposizione della matrice di correlazione deve essere condotta solo una volta invece di t volte. Formalizzando, l'estensione spazio-temporale dell'analisi delle componenti principali proposta massimizza la media temporale del prodotto tra la varianza e l'autocorrelazione spaziale dei punti dati proiettati $\boldsymbol{\Phi}$; per la prima componente $\boldsymbol{a}_1$, similmente poi per le rimanenti, si ha:

$$\begin{aligned} \max_{a_1} \frac{1}{T} \sum_{t=1}^{T} \mathrm{Var}(\boldsymbol{\phi}_{t,1}) I(\boldsymbol{\phi}_{t,1}) &= \max_{a_1} \frac{1}{T} \sum_{t=1}^{\top} \mathrm{Var}(\boldsymbol{Y}_t \boldsymbol{a}_1) I(\boldsymbol{Y}_t \boldsymbol{a}_1) \\ &= \max_{a_1} \boldsymbol{a}_1^\top \left(\frac{1}{T} \frac{1}{n} \sum_{t=1}^{T} \boldsymbol{Y}_t^\top \boldsymbol{W} \boldsymbol{Y}_t \right) \boldsymbol{a}_1 \\ &= \max_{a_1} \boldsymbol{a}_1^\top \boldsymbol{\Theta} \boldsymbol{a}_1 \end{aligned} \tag{4}$$

dove $\boldsymbol{\Theta}$ è la media temporale della matrice di correlazione spaziale. Se $\boldsymbol{W}$ è simmetrica, il vettore ottimale $\boldsymbol{v_j}$ può essere estratto come in precedenza tramite una decomposizione spettrale diretta di $\boldsymbol{\Theta}$. Come nella *PCA* ordina-

ria e nelle sue varianti spaziali, le proiezioni $\boldsymbol{\Phi}_t$ sono ottenute moltiplicando i dati originali $\boldsymbol{Y}_t$ con gli autovettori (invarianti nel tempo) contenuti in $\boldsymbol{A}$.

Riprendendo quindi l'approccio proposto in Stahlschmidt, Härdle et al. (2015) appena esposto, in Colosimo e Grasso (2018) si introducono dei pesi spaziali nella formulazione della *T-mode PCA*, in cui le diverse unità temporali (i *frames*) sono trattati come variabili e la decomposizione spettrale viene applicata alla matrice di varianza e covarianza di $\boldsymbol{X} \in \mathbb{R}^{p \times J}$, che già include l'informazione riguardante la correlazione temporale delle immagini. L'*ST-PCA* applica quindi la decomposizione alla matrice di varianza e covarianza campionaria pesata, definita come

$$\boldsymbol{S} = \frac{1}{p-1}(\boldsymbol{X} - \mathbf{1}\bar{\boldsymbol{x}})^\top \boldsymbol{W}(\boldsymbol{X} - \mathbf{1}\bar{\boldsymbol{x}}) \tag{5}$$

dove $\boldsymbol{X} \in \mathbb{R}^{p \times J}$ è la matrice dei dati *unfolded* nella formulazione secondo l'approccio *T-mode PCA*, $\bar{\boldsymbol{x}} \in \mathbb{R}^J$ è il vettore delle intensità medie osservate dei pixel in ogni *frame* e $\mathbf{1}$ è un vettore $p \times 1$ contenente solo valori unitari. La matrice $\boldsymbol{S}$ è una forma quadratica la cui decomposizione spettrale ha una soluzione in forma chiusa, dal momento che $\boldsymbol{W}$ è una matrice simmetrica di pesi (Colosimo e Grasso, 2018). Questo permette di proiettare i dati delle immagini in un nuovo sistema di coordinate mantenendo comunque la correlazione spaziale tra le osservazioni. Scomponendo la matrice di varianza e covarianza campionaria pesata, si può quindi scrivere:

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top \tag{6}$$

dove $\boldsymbol{\Lambda} = diag\{\lambda_1, \ldots, \lambda_J\}$ è la matrice diagonale contenente i J autovalori di $\boldsymbol{S}$ e $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_J)$ è una matrice ortogonale che viene definita, nel contesto dell'analisi delle componenti principali, come matrice dei *loadings* e che contiene gli autovettori normalizzati associati agli autovalori di $\boldsymbol{S}$. L'i-esimo *loading* $\boldsymbol{v}_i \in \mathbb{R}^J$ (ovvero l'i-esima colonna di $\boldsymbol{V}$) è un vettore di lunghezza J che associa un peso temporale ad ogni immagine acquisita; in altre parole, ogni componente principale può essere vista come una combinazione lineare di osservazioni nel tempo (ossia i *frames*). Il j-esimo *score* $\boldsymbol{z}_j \in \mathbb{R}^p$,

$p = M \times N$, $j = 1, \ldots, J$ (ovvero la j-esima colonna della matrice dei punti proiettati, $\boldsymbol{Z} = \boldsymbol{XV}$) è un vettore della stessa dimensione delle immagini originali e quindi racchiude l'informazione spaziale associata ad ogni componente principale. La selezione del numero $m_{pc} < J$ di componenti principali più rilevanti per l'analisi segue gli stessi criteri della *PCA* classica. In particolare, scegliere m_{pc} fissando una soglia sulla percentuale di varianza che si vuole spiegare permette di fare confronti equi tra diversi metodi basati sull'analisi delle componenti principali, dal momento che questi spiegherebbero la stessa percentuale di variabilità nei dati.

Per quanto riguarda la matrice di pesi spaziali $\boldsymbol{W}$, gli autori suggeriscono tre diverse definizioni:

$$\boldsymbol{W}_1 : \left\{ w_{i,j} = \frac{1}{d_{i,j}^2} \right\} \tag{7}$$

$$\boldsymbol{W}_2 : \left\{ w_{i,j} = 1 \; se \; d_{i,j} \leq r, \; w_{i,j} = 0 \; altrimenti \right\} \tag{8}$$

$$\boldsymbol{W}_3 : \left\{ w_{i,j} = \left(1 - \frac{d_{i,j}}{r^2}\right)^2 \; se \; d_{i,j} \leq r, \; w_{i,j} = 0 \; altrimenti \right\} \tag{9}$$

dove $d_{i,j}$ è la distanza euclidea tra l'i-esimo e il j-esimo pixel dell'immagine ed r è una distanza di riferimento fissata come soglia. La definizione di $\boldsymbol{W}_1$ comporta una diminuzione continua del peso proporzionale al quadrato della distanza tra i pixel. $\boldsymbol{W}_2$ e $\boldsymbol{W}_3$ sono rispettivamente le funzioni *kernel* discontinue *box-car* e *bi-square*, che impostano a zero il peso per punti lontani tra loro più di r. $\boldsymbol{W}_3$ accoppia una formulazione di peso decrescente con una discontinuità al valore fissato r. Gli elementi sulla diagonale principale di $\boldsymbol{W}$ sono fissati a zero ($w_{k,k} = 0$) e i pesi $w_{k,h}$, $h \neq k$, sono scalati per sommare a 1.

Gli autori propongono quindi di usare una statistica T^2 di Hotelling come indice sintetico per descrivere l'informazione contenuta nelle m_{pc} componenti principali più rilevanti selezionate all'interno delle J immagini acquisite. In particolare questa statistica è tale che $T_i^2 \in \mathbb{R}^p$, $i = 1, \ldots, p$, $p = M \times N$ ed

è definita come

$$T^2(m,n) = T_i^2 = \sum_{j=1}^{m_{pc}} \frac{z_{j,i}^2}{\lambda_j}, \qquad i = 1, \ldots, p \tag{10}$$

dove λ_j è il j-esimo autovalore e (m, n) sono le coordinate dell'i-esimo pixel. La statistica $T^2(m, n)$ associa quindi un valore T^2 ad ogni posizione dei pixel. Le aree dell'immagine caratterizzate da diversi modelli di autocorrelazione temporale risulteranno avere diversi livelli della statistica $T^2(m, n)$; questo dà la possibilità di localizzare modelli anomali sfruttando sia le dipendenze temporali che spaziali nel flusso di immagini. Inoltre, contrariamente alla *T-mode PCA*, ogni possibile disposizione dei pixel nell'immagine in una riga della matrice $\boldsymbol{X}$ produce lo stesso modello spaziale della statistica $T^2(m, n)$, grazie all'inclusione dell'informazione di correlazione spaziale nella stima della statistica.

Per combinare la localizzazione spaziale dei difetti con la capacità di sorvegliare la stabilità del processo nel tempo, l'*ST-PCA* deve essere aggiornata iterativamente ogni volta che un nuovo *frame* (o un gruppo di fotogrammi) diventano disponibili. Tale aggiornamento può avvenire in modo ricorsivo o tramite un approccio *moving window* (a finestra mobile). L'aggiornamento ricorsivo prevede di aumentare la matrice dei dati includendo ogni nuova osservazione non appena questa diventa disponibile, a meno che non venga rilevato uno stato fuori controllo, e il modello *PCA* e i parametri della carta di controllo vengono aggiornati di conseguenza. Nel contesto dell'*ST-PCA*, sia che $\boldsymbol{X}_{1:p,1:j}$ la matrice di dati che include j immagini (colonne). Quando un nuovo *frame* diventa disponibile, la matrice di dati aumenta, diventando $\boldsymbol{X}_{1:p,1:j+1}$ ed un nuovo modello *ST-PCA* basato su questa nuova matrice di dati viene calcolato. In questo modo, diversamente dal tradizionale approccio ricorsivo, viene aggiornato iterativamente il numero di colonne invece del numero di righe. L'aggiornamento tramite l'uso di una finestra mobile limita le osservazioni utilizzate nella stima del modello *PCA* a quelle più recenti, fissando a priori quante osservazioni impiegare ad ogni iterazione. L'idea consiste nell'aumentare iterativamente la matrice dei dati includendo la nuova

osservazione e scartando quella più vecchia, mantenendo fissa la dimensione L della finestra mobile. Nella contesto dell'*ST-PCA*, sia $\boldsymbol{X}_{1:p,j-L+1:j}$ la matrice di dati che include gli L fotogrammi più recenti (le colonne). Quando un nuovo *frame* diventa disponibile, la finestra mobile viene spostata in modo che l'*ST-PCA* venga applicata ad $\boldsymbol{X}_{1:p,j-L+2:j+1}$. Per entrambi i metodi descritti, se la statistica $T^2(m,n)$ risultante sembra indicare che il processo stia operando in condizioni di stabilità, la procedura viene ripetuta per il *frame* successivo, altrimenti viene lanciato un allarme. Tuttavia, qualora vengano resi disponibili di volta in volta gruppi di immagini anziché singoli *frames*, maggiore è la dimensione di questi *batch*, maggiore sarà il ritardo nel rilevare un possibile stato fuori controllo. La soluzione a finestra mobile può risultare meno onerosa dal punto di vista computazionale e quindi più conveniente per gestire in tempo reale video ad alta velocità. Il vantaggio di questo schema di aggiornamento consiste infatti nel mantenere costante il costo computazionale durante l'intero processo di sorveglianza, anche se le sue performance possono dipendere dalla selezione della dimensione L della finestra. Per quanto riguarda la scelta ottimale di L, gli autori sottolineano come l'approccio *moving window* non sia solamente molto più efficiente in termini computazionali dell'aggiornamento ricorsivo, ma garantisca anche di segnalare correttamente la presenza di un difetto dovuto ad uno stato di fuori controllo del processo. Inoltre, le performance della carta di controllo sono abbastanza robuste rispetto alla scelta di tale parametro. In particolare, L relativamente piccolo potrebbe portare a filtrare le dinamiche di processo più lunghe con hanno effetti negativi di conseguenza sulla caratterizzazione dell'autocorrelazione temporale dei profili dei pixel. D'altra parte, valori grandi di L non forniscono alcun vantaggio computazionale rispetto allo schema di aggiornamento ricorsivo. Per l'implementazione pratica quindi, il *trade-off* tra questi due effetti dovrebbe guidare la selezione del parametro.

In precedenza è stato fatto riferimento alla segnalazione dello stato fuori controllo del processo, lanciando un allarme in seguito all'analisi della statistica $T^2(m,n)$ ad un certo istante temporale. Occorre quindi definire un'*alarm*

rule che garantisca che ciò avvenga correttamente. La regola di allarme ha lo scopo di rilevare e localizzare automaticamente l'esistenza di *hotspots* nella struttura della statistica $T^2(m,n)$. L'algoritmo *k-means* rappresenta una tecnica efficace per suddividere un *dataset* in k gruppi caratterizzati dalla massima somiglianza entro i *cluster*. In questo caso, l'algoritmo *k-means* viene applicato alla struttura spaziale di $T^2(m,n)$, basandosi sulla seguente assunzione: in condizioni di controllo, sono previsti solo due gruppi, uno corrispondente ai pixel di sfondo e uno corrispondente alla normale area di fusione. In presenza di un *hotspot*, ci si aspetta un ulteriore *cluster*, come sintomo di uno stato fuori controllo. Pertanto, il *clustering k-means* viene applicato ai valori $T^2(m,n)$ impostando iterativamente un numero crescente k di gruppi. Si applica quindi un criterio per stabilire la significatività della suddivisione ottenuta, in modo tale da selezionare automaticamente il numero adeguato di *cluster* che meglio spiegano la partizione dei dati. A questo punto la statistica di controllo consiste semplicemente nel numero di *cluster* k, che, secondo il criterio di selezione, si adatta meglio al modello di $T^2(m,n)$. Se $k = 2$ il processo è giudicato in controllo, altrimenti, quando $k > 2$, viene lanciato un allarme. Nonostante sia un approccio *SPC* "non convenzionale" (Colosimo e Grasso, 2018), questa regola d'allarme basata sul *clustering* sfrutta la conoscenza ingegneristica del processo sorvegliato. Il numero di gruppi in controllo e fuori controllo ha infatti un'interpretazione fisica dal punto di vista del processo di produzione, la quale permette di interpretare agevolmente il segnale d'allarme e le relative cause che hanno portato alla segnalazione. Un metodo per la selezione di k consiste nel cercare un punto di gomito nelle somme delle distanze al quadrato entro i gruppi (*Sums of Squares Within-distances*, *SSWs*) dei valori $T^2(m,n)$ dei pixel appartenenti a ciascun *cluster* e il loro valore medio nel *cluster*, per diversi valori di k. L'indice $SSW(k)$ viene calcolato come segue:

$$SSW(k) = \frac{1}{k} \sum_{l=1}^{k} \sum_{(m,n)\in c_l} \left(T^2(m,n)_{c_l} - \bar{T}^2_{c_l} \right)^2, \qquad k = 1, 2, \ldots \tag{11}$$

dove $T^2(m,n)_{c_l}$ è il valore T^2 del pixel (m,n)-esimo appartenente al *cluster*

c_l e $\bar{T}^2_{c_l}$ è il valore T^2 medio del gruppo considerato. $SSW(k)$ è una funzione monotona decrescente del numero k di gruppi e il suo gomito identifica il valore k tale che un'ulteriore partizione non migliora significativamente il risultato del *clustering*. L'indice $SSW(k)$ è diviso per k al fine di penalizzare la "sovra-segmentazione", cioè la selezione di troppi *cluster*. Questa penalizzazione è guidata dalla conoscenza che solo due gruppi sono sufficienti per caratterizzare le osservazioni in controllo. Vale la pena notare che la $SSW(2)$, cioè la statistica SSW stimata quando si suppone che siano presenti $k = 2$ *cluster*, può essere vista essa stessa come una statistica di controllo per lo sviluppo di una regola d'allarme. Infatti, più è grande $SSW(2)$, maggiore è la probabilità che esistano più di due *cluster*. Tuttavia, la stima di un limite di controllo per questa statistica può rivelarsi difficile nella pratica, a causa della mancanza di una fase di *training* effettiva, poiché le dinamiche del processo cambiano continuamente.

Per verificare se la sorveglianza di processo trae vantaggi significativi dall'inclusione delle informazioni di correlazione spaziale nella formulazione della *T-mode PCA*, sono stati fatti degli studi di simulazione, introducendo artificialmente degli eventi *hotspot* di diversa durata e dimensione in un flusso di immagini reali. In Figura 7 sono riportati degli esempi di anomalie simulate nelle immagini reali. La *T-mode PCA* e l'*ST-PCA* sono state confrontate utilizzando come indicatore di performance l'*ARL* (*Average Run Length*, ovvero il numero di osservazioni (espresso in numero di *frames*) acquisite prima che un allarme venisse lanciato.

Confrontando le applicazioni dei due approcci è emerso che:

- L'*ST-PCA* è sempre più efficiente della *T-mode PCA* in termini di numero di componenti principali necessarie per spiegare una data percentuale di varianza. In particolare, lo schema di aggiornamento a finestra mobile permette di evitare un aumento continuo del numero m_{pc} di componenti principali che si conservano, dal momento che l'analisi si applica sempre ad un gruppo di fotogrammi di dimensione costante, con conseguenti benefici in termini di costo computazionale.

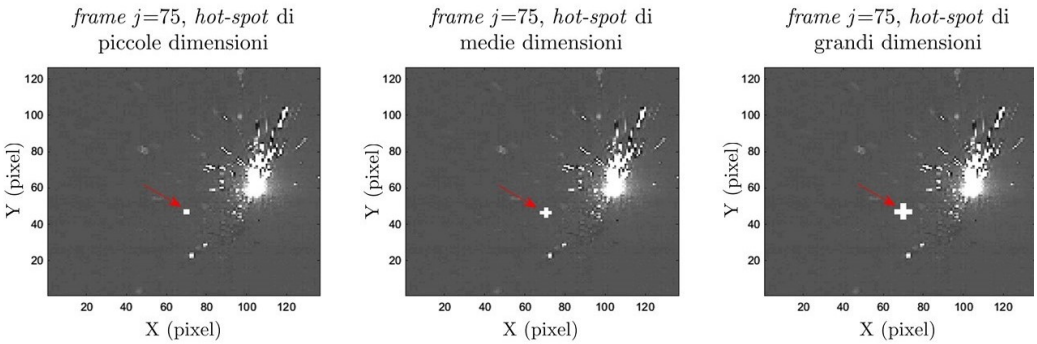


Figura 7: Esempi di un *frame* in cui degli *hotspot* di diversa dimensione sono stati simulati (le frecce indicano la posizione dell'anomalia).

- Per quanto riguarda le diverse specificazioni dei pesi spaziali $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{W}_3$ definiti rispettivamente in Equazione (7), (8), (9), si è notato che questi forniscono risultati comparabili in termini di performance della carta *ST-PCA*, il che evidenzia la robustezza del metodo proposto rispetto alla definizione dei pesi spaziali.

- Quando viene impiegato l'aggiornamento ricorsivo, l'*hotspot* viene sempre rilevato usando l'*ST-PCA*, anche se dopo una durata maggiore. Quando viene utilizzato l'aggiornamento a finestra mobile invece, l'*ST-PCA* è in grado di rilevare solo un *hotspot* di medie o grandi dimensioni. Questo è causato dal fatto che la dimensione scelta della finestra (pari a $L = 50$) è troppo piccola per riuscire a rilevare delle anomalie di gravità molto lieve: sarebbe necessaria in questo caso una finestra di dimensioni maggiori o l'adozione dell'approccio ricorsivo. inoltre, più grande è la dimensione dell'*hotspot*, più è veloce il suo rilevamento con entrambi i metodi confrontati. In generale però, l'approccio *ST-PCA* non solo è più veloce nel rilevare il difetto, ma è anche l'unico metodo che porta ad un allarme in tutti gli scenari fuori controllo simulati, indipendentemente dallo schema di aggiornamento iterativo.

- L'inclusione dell'informazione di correlazione spaziale migliora l'identificazione e la distinzione di *cluster* adiacenti che presentano valori di

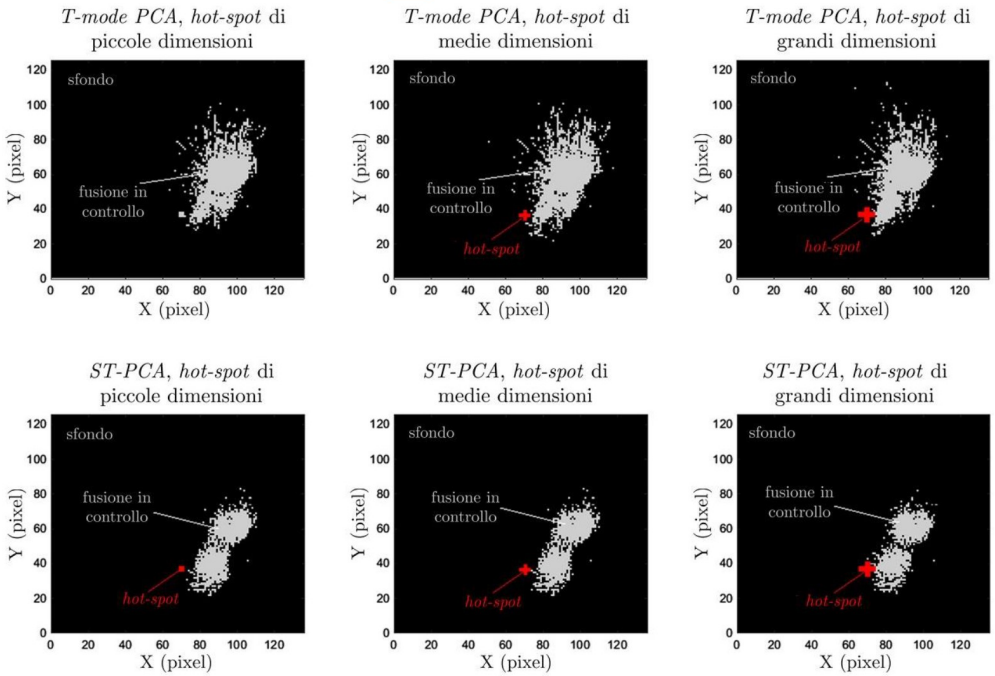


Figura 8: Confronto tra i risultati del raggruppamento *k-means* per gli approcci *T-mode PCA* ed *ST-PCA* (pesi spaziali definiti da $\boldsymbol{W}_1$, aggiornamento a finestra mobile con $L = 50$) selezionando una soglia pari al 50% di varianza spiegata dalle componenti principali.

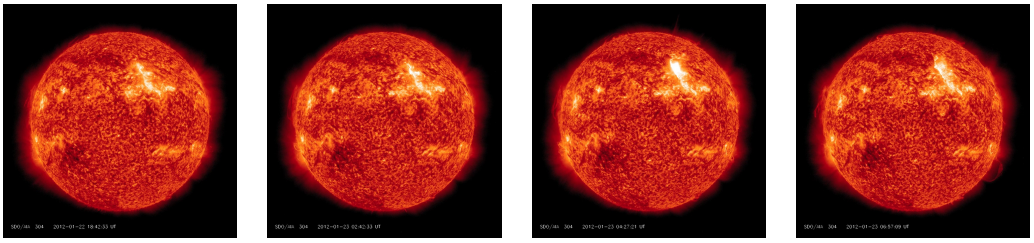
intensità dei pixel anomale. La Figura 8 confronta i risultati del *clustering k-means* per la *T-mode PCA* e l'*ST-PCA* nel caso si scelgano la specificazione $\boldsymbol{W}_1$ di Equazione (7) per i pesi spaziali e lo schema di aggiornamento a finestra mobile con $L = 50$; la soglia della varianza spiegata per la selezione di del numero di componenti principali è stata fissata al 50%. In questo caso, la *T-mode PCA* permette di rilevare solo gli *hotspot* di medie e grandi dimensioni. Questo conferma la maggiore efficacia dell'*ST-PCA* nel riconoscere questo tipo di eventi fuori controllo.

È stato condotto poi uno studio di sensitività rispetto alla scelta del parametro L che definisce la dimensione della finestra mobile nell'aggiornamento iterativo del modello durante la sorveglianza. Da questa analisi si

può dedurre che dimensioni troppo piccole delle finestre mobili possono compromettere la capacità di catturare il comportamento intrinseco del processo anche se, d'altra parte, dimensioni troppo grandi delle stesse possono ridurre il beneficio computazionale rispetto allo schema di aggiornamento ricorsivo. Quest'ultimo può essere usato all'inizio della sorveglianza per aumentare iterativamente la matrice dei dati $\boldsymbol{X}$ fino a una dimensione sufficiente L per modellare correttamente la dinamica del processo. In seguito, lo schema di aggiornamento a finestra mobile può essere impiegato per mantenere costanti la dimensione L della matrice ed il costo computazionale. Gli autori suggeriscono infatti di inizializzare l'analisi *ST-PCA* con un numero sufficiente di fotogrammi, nel senso che il primo modello stimato dovrebbe includere almeno 40-50 fotogrammi dall'inizio del processo (Colosimo e Grasso, 2018). Questo può ridurre il rischio di falsi allarmi causati da un sovradattamento a poche osservazioni iniziali. Inoltre, se l'aggiornamento del modello *ST-PCA* e la *cluster analysis* per ogni singolo *frame* non sono computazionalmente praticabili, è possibile applicare un aggiornamento "*batch-wise*": il modello viene aggiornato solo quando un nuovo insieme di B fotogrammi è reso disponibile, dove B dovrebbe essere scelto a seconda dei vincoli computazionali. Tuttavia, occorre notare che maggiore è B, minore è la potenziale reattività del metodo nel rilevamento dei difetti.

3.2 *Spatial-Adaptive Sampling And Monitoring* (*SASAM*)

La metodologia sviluppata da A. Wang, Xian et al. (2018) affronta il problema della sorveglianza statistica online di *big data streams*, ossia flussi di dati ad alta dimensionalità acquisiti con un alta frequenza e in grandi volumi. In particolare, l'attenzione viene posta al contesto della sorveglianza di dati nel formato di immagini. L'esempio che motiva lo studio e su cui viene infine applicato il metodo di sorveglianza sviluppato consiste nel rilevamento dei brillamenti solari (*solar flares*). In quest'ambito, vengono catturate ad alta frequenza delle immagini della superficie solare grazie alle telecamere presenti su alcune stazioni spaziali, in modo da rilevare la comparsa dei *flares*.



(a) (b) (c) (d)

Figura 9: Sequenza di fotogrammi tratta da un video acquisito a gennaio 2012 e che mostra un brillamento solare (in alto a destra nelle immagini) che raggiunge l'apice in (c). Fonte: *The Sun Flares with Activity* 2012.

Un brillamento solare è un'improvvisa eruzione di materia che esplode dalla superficie di una stella, sprigionando un'enorme quantità di energia che è in grado di influenzare fortemente il clima dello spazio circostante fino alle vicinanze della Terra. Tale fenomeno viene percepito come un lampo di luce sulla superficie solare nelle immagini che vengono acquisite ed il valore di luminosità di ogni pixel può essere considerato come un flusso di dati utile a rilevare questo tipo di evento. In Figura 9 è riportata una sequenza di fotogrammi provenienti da un video che ha catturato un brillamento, raggiungendo l'apice di intensità in Figura 9c (*The Sun Flares with Activity* 2012). Gli effetti potenzialmente dannosi o nocivi dei brillamenti includono il pericolo che i dispositivi elettronici presenti nell'area circostante vengano disattivati ed il rischio di esposizione alle radiazioni per i veicoli spaziali e gli astronauti. Per minimizzare le perdite, è importante rilevare rapidamente il verificarsi di questi eventi il più presto possibile. Per questo motivo, delle immagini solari ad alta risoluzione vengono catturate dai satelliti con un'alta frequenza di campionamento, producendo circa 1.5 terabyte di dati ogni giorno. Sebbene questi flussi di dati rappresentino un'opportunità senza precedenti per la sorveglianza online, essi pongono anche dei requisiti *hardware* critici per la comunicazione e l'elaborazione dei dati. A causa della limitata larghezza di banda di trasmissione dei dati infatti, solo flussi parziali di dati possono essere trasmessi alla Terra per l'analisi in tempo reale. In

questo modo, non tutti i dati dei *frames* sono accessibili per l'analisi, anche se questi *streams* vengono completamente memorizzati offline in un secondo momento.

In letteratura, sono state sviluppate diverse metodologie per la sorveglianza di flussi di *big data*. Molti di questi assumono come note le distribuzioni fuori controllo delle variabili che subiscono un cambiamento, così come si ipotizzano note quali variabili muteranno ad un certo istante. Sotto queste assunzioni, sono stati proposti diverse carte di controllo basate sul test rapporto di verosimiglianza che garantiscono alcune proprietà ottimali (A. Wang, Xian et al., 2018). Tuttavia, sapere sia quali variabili subiranno un cambiamento e sia le loro distribuzioni fuori controllo non è un'ipotesi realistica nella pratica. Inoltre, tutti i metodi sono stati proposti senza la considerazione dei vincoli *hardware*, ipotizzando cioè che tutte le osservazioni siano disponibili in ogni momento; la maggior parte di essi inoltre si concentra su "problemi non strutturati", in cui ad esempio non viene inclusa l'informazione spaziale del processo. Per affrontare la sorveglianza online di flussi di dati ad alta dimensionalità tenendo conto dei vincoli *hardware*, sono state proposte strategie di campionamento adattative che decidono in tempo reale quali flussi di dati parziali osservare durante la sorveglianza. Sebbene questi metodi disponibili si siano dimostrati promettenti, essi non riescono a considerare le informazioni spaziali delle variabili sorvegliate, il che può portare a dei ritardi nel rilevamento di situazioni ritenute fuori controllo. Ad esempio, nel problema del rilevamento dei brillamenti solari, la posizione dei pixel nell'immagine fornisce informazioni utili per rilevare il *flare*, dal momento che ogni brillamento influisce sui valori di luminosità di più pixel adiacenti. Un esempio di una metodologia di questo tipo si può ritrovare in K. Liu, Mei et al. (2015), in cui gli autori hanno sviluppato una procedura di campionamento adattivo (*Top-R based Adaptive Sampling*, *TRAS*), sotto l'ipotesi che solo $q < p$ variabili possano essere osservate in ogni momento. In particolare, la procedura *TRAS* assegna due statistiche *CUSUM* locali $W_{k,t}^{(1)}$ e $W_{k,t}^{(2)}$ ad ogni flusso di dati, per sorvegliare i potenziali *shift* positivi e negativi, rispettivamente, che

possono verificarsi sulla variabile k, $k = 1, \ldots, p$. La statistica di controllo viene allora definita come la somma delle r statistiche *CUSUM* più grandi osservate. Per affrontare il problema delle osservazioni mancanti, il metodo *TRAS* aggiorna i valori delle statistiche *CUSUM* in base al fatto che una certa variabile sia stata osservata o meno: quando la variabile k è osservata, le statistiche *CUSUM* locali sono aggiornate con la procedura standard, altrimenti viene aggiunta una "pseudo osservazione" sia a $W_{k,t}^{(1)}$ che a $W_{k,t}^{(2)}$. In questo modo, il metodo *TRAS* seleziona le variabili osservabili più sospette (cioè quelle con le statistiche *CUSUM* maggiori) da osservare al tempo $t+1$. È stato dimostrato che la procedura *TRAS* migliora significativamente l'efficienza di rilevazione rispetto a una procedura di campionamento casuale delle q variabili (K. Liu, Mei et al., 2015). Nonostante i vantaggi della procedura *TRAS* nei "problemi non strutturati", questo metodo non può essere efficacemente utilizzato per i "problemi strutturati", in cui sono disponibili le informazioni spaziali del processo sorvegliato. L'idea chiave della nuova metodologia proposta da A. Wang, Xian et al. (2018) consiste appunto nel riuscire ad includere tali informazioni spaziali per riuscire a sviluppare un modello di sorveglianza più efficace di quelli esistenti.

Si supponga quindi di essere interessati alla sorveglianza di p variabili che caratterizzano il flusso di dati, e siano queste indicizzate da k, $k = 1, \ldots, p$. Si può considerare l'informazione spaziale associata ad ogni variabile definendo $\boldsymbol{x}_k = (x_{k,1}, \ldots, x_{k,s}) \in \mathbb{R}^s$, dove s definisce la dimensione dello spazio nel quale possono essere collocate le variabili. Ad esempio, se $s = 2$ ogni variabile viene osservata su un piano bidimensionale, o ancora se $s = 3$ ogni variabile è associata ad un punto nello spazio tridimensionale. Per la sorveglianza di un flusso di immagini si considera dunque il caso particolare in cui si fissa $s = 2$ e si assume che le variabili siano spazialmente distribuite in una griglia rettangolare di dimensioni $M \times N$, tale che $p = M \times N$. Ogni punto di questa griglia identifica nello spazio una sola variabile in modo univoco; in questo contesto, si può pensare alle p variabili come i valori di intensità di ogni pixel che compone un'immagine. Ogni variabile può essere localizzata nello

spazio grazie a $\boldsymbol{x}_k = (x_{k,1}, x_{k,2})$, dove $x_{k,1} \in \{1, \ldots, M\}$ e $x_{k,2} \in \{1, \ldots, N\}$ sono relative le coordinate. I valori osservati delle p variabili al tempo t sono contenuti nel vettore $\boldsymbol{y}_t = (y_{1,t}, \ldots, y_{p,t})^\top$. Quando il processo è in controllo, si assume che $\boldsymbol{y}_t$ sia indipendente e identicamente distribuito nel tempo secondo una Normale p-variata a media nulla e varianza pari alla matrice identità di ordine p. Questa assunzione è spesso valida quando $\boldsymbol{y}_t$ identifica i residui di un modello spazio-temporale anziché le osservazioni grezze. Si è interessati a rilevare possibili scostamenti dalla media che si verificano in alcune variabili le cui posizioni spaziali associate sono raggruppate in una regione fuori controllo $\mathcal{C} \in \mathbb{R}^2$. Tuttavia, le informazioni esatte circa posizione, dimensione e forma della regione fuori controllo $\mathcal{C}$ sono ignote. In particolare, se si assume che ad un istante τ si verifichi un cambiamento nella media di un sottoinsieme di variabili, portando così il processo fuori controllo, si avrà che $\boldsymbol{y}_t$ si distribuisce come una Normale p-variata con media $\boldsymbol{\delta}$ e varianza $\boldsymbol{I}$ invariata. Formalmente,

$$\boldsymbol{y}_t \sim \begin{cases} N_p(\boldsymbol{0}, \boldsymbol{I}) & \text{per } t = 1, \ldots, \tau - 1 \\ N_p(\boldsymbol{\delta}, \boldsymbol{I}) & \text{per } t = \tau, \tau + 1, \tau + 2, \ldots \end{cases} \tag{12}$$

dove $\boldsymbol{0}$ è il vettore p-dimensionale di zeri, $\boldsymbol{\delta}$ è il vettore delle medie fuori controllo e $\boldsymbol{I}$ è la matrice identità di ordine p. Dato che si assume che solo un sottoinsieme di variabili possa andare fuori controllo ad un certo istante τ, va notato che solo alcune componenti di $\boldsymbol{\delta}$ saranno non nulle; dunque, per $t \geq \tau$, ad ogni variabile $y_{k,t}$, $k = 1, \ldots, p$ è associata una media fuori controllo, definita da

$$\mathbb{E}(y_{k,t}) = \delta \mathbb{1}_{\mathcal{C}}(\boldsymbol{x}_k) \tag{13}$$

dove $\mathbb{1}_{\mathcal{C}}(\cdot)$ è la funzione caratteristica che indica se una variabile appartiene o meno alla regione fuori controllo. Nella pratica, dal momento che si deve tener conto anche dei vincoli *hardware* che caratterizzano la sorveglianza di *big data streams*, si assume che solo q variabili tra le p totali siano osservabili ad ogni istante t, dove q è un valore intero fissato a priori.

In questo contesto, una strategia di campionamento spaziale online deve essere in grado di decidere quali variabili osservare in ogni momento per agevolare il rilevamento delle variabili fuori controllo, in modo tale da sviluppare uno schema di sorveglianza efficiente per individuare *cluster* di variabili dal comportamento anomalo. A tal scopo, viene proposto un metodo che integra un campionamento spaziale adattivo assieme ad uno schema di sorveglianza scalabile che rileva rapidamente gli *shift* che si verificano in un'area specifica dei flussi di immagini sorvegliati. Questa metodologia prende il nome di *SASAM* (*Spatial-Adaptive Sampling And Monitoring*). Per rilevare efficacemente le variabili fuori controllo raggruppate, lo schema di campionamento spaziale adattivo incorpora due strategie apparentemente contraddittorie:

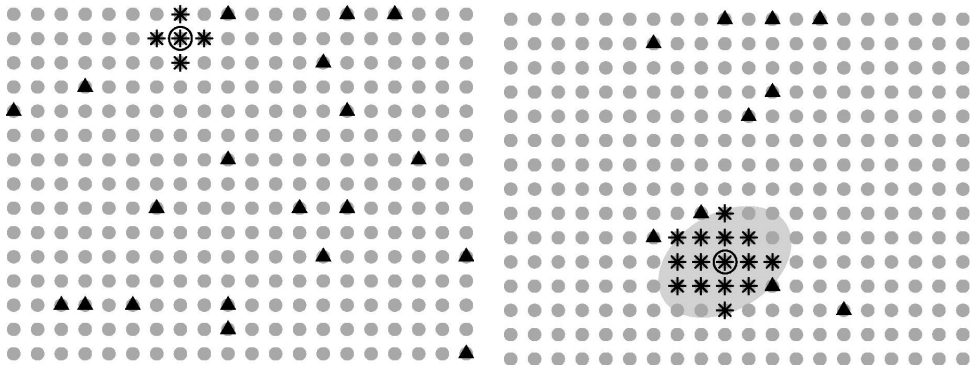
1. Strategia di ricerca estesa (*wide search strategy*), che mira a distribuire casualmente le risorse sull'intera area sorvegliata mentre il processo è in controllo. Dato che si assume che la posizione degli *shift* sia sconosciuta, la *wide search* assicura che la regione fuori controllo possa essere rapidamente osservata da almeno alcune delle risorse computazionali impiegate. Per questo primo tipo di campionamento, si suggerisce di selezionare dallo spazio sorvegliato, ad ogni istante, un insieme di variabili estratte in modo casuale.

2. Strategia di ricerca profonda (*deep search strategy*), il cui obiettivo è allocare un numero limitato di risorse alle variabili che sembrano essere collocate in una regione sospetta di essere fuori controllo. La logica è quella di identificare le variabili fuori controllo in modo mirato sfruttando le informazioni spaziali, accelerando così il rilevamento delle anomalie. Di conseguenza, se emerge l'evidenza che un si sia già verificato cambiamento in una variabile, allora anche le variabili vicine dovrebbero essere osservate per minimizzare il ritardo nel lanciare un allarme.

In teoria, le strategie di ricerca estesa e di ricerca profonda sono contraddittorie: la prima richiede che le variabili osservabili siano disperse sull'intera area

sorvegliata, mentre la seconda richiede che le variabili osservabili siano riunite in una regione specifica. Per superare questa incompatibilità, A. Wang, Xian et al. propongono di separare dinamicamente il totale q delle variabili osservabili in due gruppi, ognuno dei quali governato da una di queste due strategie ad ogni istante t. In particolare, nel seguito le variabili osservate in quanto selezionate dalle strategie *wide search* o *deep search* sono abbreviate come variabili-W (*W-variables*) e variabili–D (*D-variables*), rispettivamente. L'insieme delle variabili-W al tempo t è indicato con $\mathcal{O}_{W,t}$ ed ha cardinalità $|\mathcal{O}_{W,t}| = q_{W,t}$), mentre l'insieme delle variabili-D al tempo t è indicato con $\mathcal{O}_{D,t}$ con cardinalità $|\mathcal{O}_{D,t}| = q_{D,t}$); in questo modo, $q_{W,t} + q_{D,t} = q$. Poiché è logico aspettarsi che la scelta della strategia di ricerca sia legata alla probabilità che si sia verificato un cambiamento, i valori di $q_{D,t}$ e $q_{W,t}$ dovranno essere regolati dinamicamente, in base al valore osservato ad ogni istante della statistica di controllo. In particolare, una statistica di controllo che sembra indicare un'alta probabilità che si sia verificato un cambiamento nel processo porterà a selezionare un numero elevato di variabili-D ($q_{D,t}$) ed un numero ridotto di variabili-W ($q_{W,t}$). In altre parole, una volta che si individua un potenziale cambiamento nello spazio sorvegliato, una quantità maggiore di risorse sarà assegnata alla *deep search* per concentrarsi sull'osservazione della regione sospetta. Viceversa, una statistica di controllo indice di una bassa probabilità che si sia verificato uno *shift* porterà a bassi valori di $q_{D,t}$ e grandi valori di $q_{W,t}$. Pertanto, quando non si ha ragione di credere che si sia verificato un cambiamento nel processo, più risorse saranno assegnate alla ricerca estesa per cercare rapidamente una potenziale regione anomala. La Figura 10 fornisce un semplice esempio della selezione delle variabili osservabili dopo che si è verificato un cambiamento, in modo da avere una rappresentazione grafica che chiarisca il funzionamento dello schema di campionamento spaziale adattivo.

Una volta che sono state definite le meccaniche delle strategie di ricerca estesa e profonda per il campionamento dal processo, un ulteriore aspetto da considerare è rappresentato a questo punto dall'incorporare lo schema di



(a) (b)

Figura 10: Illustrazione dello schema di campionamento spaziale adattivo proposto da A. Wang, Xian et al. (2018). In questo esempio, ci sono $p = M \times N = 15 \times 20$ variabili distribuite su una griglia e le loro posizioni sono rappresentate da punti grigi. Tra queste, solo $q = 30$ possono essere osservate ad ogni istante. Quando il processo è in controllo e non ci sono segnali che indichino che si sia verificato un cambiamento, vengono assegnate un gran numero di variabili-W (indicate dai triangoli) e un numero ridotto di variabili-D (indicate dalle stelle), come mostrato in (a). In (b) invece, quando il processo è fuori controllo (con la regione anomala evidenziata dall'area ombreggiata) e la statistica di controllo ha suggerito la presenza di un'anomalia, vengono selezionate in modo intelligente più variabili-D e meno variabili-W.

campionamento con un metodo *SPC* scalabile per rilevare rapidamente le variabili fuori controllo raggruppate.

Per quanto riguarda l'impostazione del *layout* delle variabili-W, la procedura *SASAM* viene inizializzata selezionando tutte le variabili da osservare tramite il campionamento governato dalla strategia *wide search*, ossia $q_{W,t} = q$ al tempo $t = 1$. Quando $t > 1$ invece, si avrà $q_{W,t} = q - q_{D,t}$. L'obiettivo della strategia di ricerca estesa, come precedentemente esposto, è di distribuire le risorse di sorveglianza sull'intero dominio spaziale in modo che le variabili che subiscono un cambiamento possano essere identificate il prima possibile. Pertanto, un piano di campionamento per le variabili-W che selezioni i campioni nello spazio senza alti costi computazionali può essere ottenuto selezionando casualmente $q_{W,t}$ variabili-W dalle $p - q_{D,t}$ variabili che non sono state selezionate come variabili-D. Nonostante in A. Wang, Xian et al. (2018) gli autori abbiano provato altri piani di campionamento per la ricerca estesa, è stato notato che questi non portavano ad un significativo miglioramento in termini di performance. Al tempo $t > 1$, l'insieme $\mathcal{O}_{D,t}$ delle variabili-D è determinato basandosi sulle statistiche locali al tempo $t - 1$, $W_{k,t-1}^{(i)}$, $k = 1, \ldots, p$, $i = 1, 2$, mentre l'insieme $\mathcal{O}_{W,t}$ delle variabili-W è determinato dalla strategia di campionamento spaziale esteso (casuale). Una volta determinato il *layout* del campionamento delle q variabili osservabili, le osservazioni sono ottenute di conseguenza e sono quindi utilizzate per aggiornare le statistiche locali $W_{k,t}^{(i)}$, $k = 1, \ldots, p$, $i = 1, 2$ e la statistica di controllo S_t^{SASAM}. Se S_t^{SASAM} eccede un certo limite di controllo H, la procedura lancia un allarme; altrimenti, viene calcolato il numero e la disposizione delle variabili-D al prossimo istante e t diventa $t + 1$.

Ricordando poi che una delle idee innovative del metodo proposto è quella di aggiornare le statistiche locali di un flusso di dati utilizzando sia le osservazioni di una data variabile sia le informazioni di quelle vicine, viene

proposto l'aggiornamento delle statistiche locali tramite la formula:

$$W_{k,t}^{(1)} = \max\Bigg(W_{k,t-1}^{(1)} + \sum_{j \in \mathcal{O}_{W,t} \cup \mathcal{O}_{D,t}} K_h\big(|x_k - x_j|\big)\big(u_{min} y_{j,t} - u_{min}^2/2\big), 0\Bigg);$$
$$W_{k,t}^{(2)} = \max\Bigg(W_{k,t-1}^{(2)} + \sum_{j \in \mathcal{O}_{W,t} \cup \mathcal{O}_{D,t}} K_h\big(|x_k - x_j|\big)\big(-u_{min} y_{j,t} - u_{min}^2/2\big), 0\Bigg) \tag{14}$$

dove $W_{k,t-1}^{(1)}$ e $W_{k,t-1}^{(2)}$ indicano le statistiche locali del flusso di dati k al tempo $t-1$, con $W_{k,0}^{(1)} = W_{k,0}^{(2)} = 0$. Come in K. Liu, Mei et al. (2015), il parametro u_{min} indica l'entità minima di interesse del cambiamento che si intende individuare, che può essere determinata in base all'applicazione specifica. $K_h(\cdot)$ è una funzione *kernel* che riflette la dipendenza tra il cambiamento di una variabile e lo *shift* che avviene su un'altra variabile. In particolare, qui

$$K_h(x) = \left(1 - \left(\frac{x}{h}\right)^2\right)_+ \tag{15}$$

è la funzione *kernel* nella formulazione di Epanechnikov, in cui $(\cdot)_+$ denota l'operatore $\max(\cdot, 0)$ e h è il parametro che regola l'ampiezza di banda della funzione (*bandwidth*) ed è strettamente legato alla dimensione della regione fuori controllo. Si noti che se h viene scelto in modo tale da essere minore di $\min_{1 \le i < j \le p} |x_i - x_j|$, allora $K_h(x_i - x_j) = \delta_{ij}$ (dove $\delta_{ij} = 1$ se $i = j$ e $\delta_{ij} = 0$ se $i \neq j$) e di conseguenza lo schema di aggiornamento di Equazione (14) è equivalente a quello di uno schema *CUSUM* standard. Quando h è grande, l'osservazione della variabile j non solo avrà un effetto sull'aggiornamento della propria statistica locale, ma influenzerà anche le statistiche locali delle variabili vicine, purché la distanza tra le loro posizioni nello spazio sia inferiore ad h. L'integrazione del metodo *CUSUM* con il metodo di *smoothing* locale (tramite la funzione *kernel*) dovrebbe quindi riuscire a garantire una stima più accurata e rapida dello stato (*IC* o *OC*) del sistema. A questo punto, si decide di scegliere la più grande statistica locale come statistica di controllo della procedura proposta, ossia:

$$S_t^{SASAM} = \max_{\substack{1 \le k \le p \\ i = 1,2}} W_{k,t}^{(i)} \tag{16}$$

Un altro aspetto fondamentale della procedura *SASAM* è decidere ad ogni istante il numero e le posizioni delle variabili-D al tempo $t+1$, nel caso in cui al tempo t nessun allarme sia stato lanciato. Occorre ricordare che le variabili-D dovrebbero essere situate nella regione anomala sospetta e che il numero di variabili-D e di variabili-W vengono corretti basandosi sul valore osservato della statistica di controllo. Pertanto, le variabili-D saranno scelte in modo tale da essere la variabile k^*, che corrisponde alla più grande statistica locale registrata, e le variabili ad essa vicine. In particolare, sia $\pi(1), \ldots, \pi(p)$ una permutazione di $1, \ldots, p$ tale che

$$|x_{k^*} - x_{\pi(1)}| \leq |x_{k^*} - x_{\pi(2)}| \leq \ldots \leq |x_{k^*} - x_{\pi(p)}| \tag{17}$$

Dunque, per un fissato valore di $q_{D,t+1}$, le variabili-D al tempo $t+1$ saranno quelle indicizzate dall'insieme

$$\mathcal{O}_{D,t+1} = \big\{\pi(1), \ldots, \pi(q_{D,t+1})\big\} \tag{18}$$

Dal momento che il numero di variabili-W viene deciso di conseguenza (infatti $q_{W,t} = q - q_{D,t}$), ciò che risulta essere di maggiore importanza è la determinazione del numero di variabili-D al tempo $t+1$, ossia $q_{D,t+1}$.

Poiché le variabili-D sono raggruppate attorno alla variabile k^* e ci si concentra sul rilevamento degli *shift* che avvengono in una sola area dell'immagine, lo stato IC/OC di queste variabili-D dovrebbe essere simile. In altri termini, le osservazioni provenienti da diverse variabili-D possono essere considerate come campioni multipli tratti dallo stesso processo statistico. L'idea quindi è quella di scegliere un grande valore di $q_{D,t+1}$ quando la statistica di controllo S_t^{SASAM} è grande (quando cioè viene identificato uno sospetto stato fuori controllo) e fissare invece un piccolo valore di $q_{D,t+1}$ quando la statistica di controllo S_t^{SASAM} è piccola. La regolazione di $q_{D,t+1}$ avviene sulla base della funzione crescente $f_{\boldsymbol{\theta}}$ di S_t^{SASAM} e regolata dal parametro $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top$:

$$q_{D,t+1} = f_{\boldsymbol{\theta}}(S_t^{SASAM}) = \left\lceil \frac{q\theta_2\big(S_t^{SASAM} - H\theta_1\big)_+}{H(1-\theta_1)} \right\rceil \tag{19}$$

dove H indica il valore del limite di controllo della carta, $\theta_1 \in [0,1]$, $\theta_2 \in [0,1]$ e $\lceil \cdot \rceil$ indica l'arrotondamento per eccesso all'intero più vicino. L'espressione è una funzione lineare della statistica di controllo S_t^{SASAM} assumendo che $S_t^{SASAM} > H\theta_1$, dal momento che all'istante t non è stato lanciato un allarme. I parametri di regolazione della funzione possono essere così interpretati:

- θ_1 controlla la soglia per la statistica di controllo oltre la quale si iniziano ad allocare le variabili-D. In particolare, le variabili-D saranno assegnate se e solo se la statistica S_t^{SASAM} è maggiore del $100\theta_1\%$ del limite di controllo H, altrimenti tutte le variabili osservabili saranno variabili-W.

- θ_2 controlla il numero massimo di variabili-D. In particolare, al massimo il $100\theta_2\%$ delle q variabili osservabili può essere selezionato dal campionamento *deep search*. Inoltre, all'aumentare di θ_2 vengono assegnate più risorse di sorveglianza per la strategia di ricerca profonda piuttosto che per quella estesa. Il parametro θ_2 può essere interpretato anche come il rapporto tra il numero massimo di variabili-D e il numero totale di variabili osservabili, il che rende lo schema di controllo adeguato a diversi numeri di variabili osservabili.

Per poter implementare la procedura fin qui illustrata, occorre fissare i valori dei parametri che regolano e definiscono di conseguenza il comportamento della carta di controllo. In particolare, si tratta di ricercare dei valori ottimali per quattro parametri: (1) u_{min}, introdotto in Equazione (14), che indica l'intensità minima del cambiamento che si vuole individuare e che dipende dall'applicazione specifica; (2) h, che è il parametro che regola l'ampiezza di banda della funzione *kernel* $K_h(\cdot)$ definita in Equazione (15) ed è strettamente legato alla dimensione della regione fuori controllo; (3) θ_1 e (4) θ_2, che sono i parametri coinvolti nella funzione definita in Equazione (19) che regola la scelta del numero di variabili-D per l'osservazione successiva. La scelta di u_{min} dipende strettamente dall'applicazione sulla quale si intende operare, dal momento che, come richiamato sopra, definisce l'entità

minima ritenuta critica per il rilevamento di fenomeni anomali. Pertanto, la sua scelta deve essere guidata dalle conoscenze in possesso in merito al processo sorvegliato. Per quanto riguarda la scelta di h, si osserva che le prestazioni della carta di controllo proposta sono relativamente robuste ai diversi valori dell'ampiezza di banda della funzione *kernel* $K_h(\cdot)$. In merito ai parametri θ_1 e θ_2, si nota che in generale le performance della carta di controllo sono migliori laddove venga scelto un valore piccolo per θ_1, mentre θ_2 non ha un impatto significativo sul comportamento della carta. In particolare, quando θ_1 è grande, la procedura *SASAM* non assegna le variabili-D a meno che la statistica di controllo non sia molto grande e quindi riduce la possibilità che le variabili fuori controllo continuino ad essere osservate. Un valore piccolo di θ_1 aumenta dunque la capacità della carta di concentrarsi sull'osservazione delle regioni sospette, riducendo notevolmente il ritardo nel rilevamento di anomalie.

Infine, in merito al numero q di variabili osservabili, si nota come il ritardo dell'individuazione della regione fuori controllo diminuisce; questo è coerente con l'idea secondo la quale all'aumentare della quantità di informazioni acquisite in ogni momento, la procedura riesca ad essere più reattiva.

La nuova metodologia proposta viene infine applicata in A. Wang, Xian et al. (2018) al contesto illustrato inizialmente riguardante il rilevamento dei brillamenti solari, confrontandone il comportamento con la carta di controllo *TRAS*. In questo caso di studio, i primi 50 *frames* raccolti sono utilizzati per stimare i limiti di controllo per entrambe le carte; la sorveglianza inizia dunque al fotogramma osservato a $t = 51$. La procedura *SASAM* rileva il *flare* a $t = 90$, mentre la procedura *TRAS* lancia un allarme a $t = 95$. Nonostante entrambe queste le carte riescano ad individuare la presenza di un evento fuori controllo, il ritardo nel rilevamento della procedura *SASAM* è molto più breve di quello della procedura *TRAS*. Le immagini rappresentanti i *frames* e le posizioni delle variabili osservabili ai tempi $t = 81, 90, 95$ sono mostrate nella Figura 11, in cui ogni riga identifica, nell'ordine, un diverso istante di osservazione. Le Figure 11(a), (d) e (g) nella prima colonna mostrano le

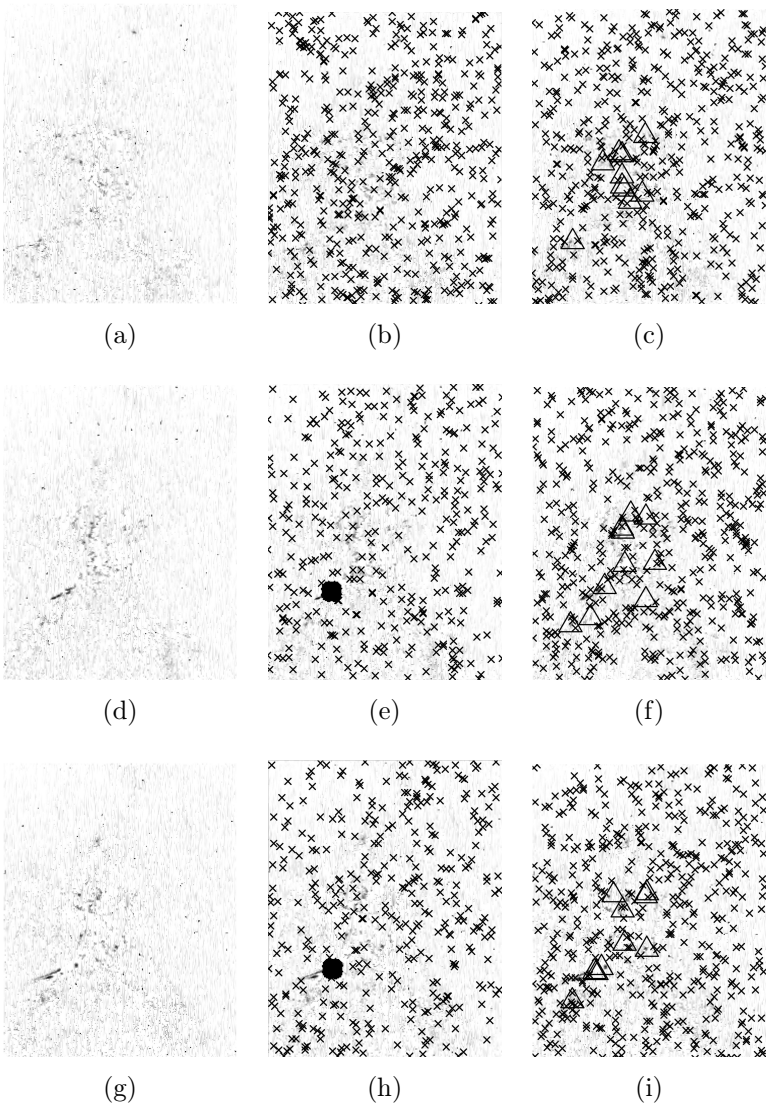


Figura 11: Immagini raffiguranti i *frames* con i contrassegni che indicano le posizioni delle variabili osservate. Il *dataset* è costituito da fotogrammi ciascuno contenente 67 744 pixel distribuiti su una griglia di dimensioni $M \times N = 292 \times 232$. Il *flare* solare si verifica nei dati tra gli istanti $t = 87$ e $t = 102$. Ogni riga corrisponde ad un istante di osservazione diverso, nell'ordine $t = 81, 90, 95$. La prima immagine di ogni riga riporta la posizione del *flare* solare. La seconda immagine mostra le posizioni delle variabili-W e delle variabili-D (contrassegnate dalla "$\times$") nella procedura *SASAM*. La terza immagine illustra le posizioni delle variabili osservabili della procedura *TRAS*; tra queste, quelle con una statistica locale tra le *top-r* sono contrassegnate da "$\triangle$".

immagini raccolte ai diversi istanti. Le Figure 11(b), (e) ed (h) nella seconda colonna riportano le posizioni delle variabili osservabili del metodo *SASAM*, ed infine, le Figure 11(c), (f) ed (i) nella terza colonna illustrano le posizioni delle variabili osservabili per il metodo *TRAS*. Queste immagini mostrano chiaramente che quando l'eruzione solare non si è verificata, sia la procedura *TRAS* che la procedura *SASAM* tendono a distribuire le variabili osservabili in modo casuale all'interno dell'immagine. Quando si verifica un brillamento solare, per la procedura *TRAS* vengono osservate solamente due variabili fuori controllo al tempo $t = 90$ (Figura 11(f)) e tuttavia non viene lanciato un allarme. Negli istanti successivi, sempre più osservazioni vengono identificate come fuori controllo nell'approccio *TRAS* finché non viene lanciato un allarme a $t = 95$. Di contro, l'algoritmo *SASAM*, una volta che identifica anche solo un'osservazione nella regione anomala, consente di allocare un gran numero di variabili-D nell'area circostante (Figura 11(e)) e la maggior parte di queste variabili-D risultano essere effettivamente fuori controllo. La procedura di aggiornamento in Equazione (14) utilizza quindi efficacemente le osservazioni di queste variabili-D per aggiornare le statistiche locali delle variabili vicine nella regione fuori controllo, il che si traduce in un notevole aumento della statistica di controllo S_t^{SASAM}. Per questi motivi, viene lanciato immediatamente un allarme. Dai risultati di questa applicazione, si può notare come la procedura *SASAM* assegni un numero maggiore di variabili osservabili alla sospetta regione anomala sfruttando adeguatamente le informazioni spaziali del processo. Questa peculiarità infatti migliora significativamente le performance di sorveglianza date dalla carta di controllo *SASAM*.

4 Confronto delle metodologie *ST-PCA* e *SA-SAM*

Come accennato in apertura del Capitolo 3, il *dataset* comune per mezzo del quale eseguire il confronto tra le procedure può variare in tre aspetti: (1) la dimensione dei singoli *frames* che determina il numero totale di pixel da sorvegliare, p, (2) la dimensione e (3) la durata dell'anomalia che potrebbero influenzare la reattività delle carte. Sono quindi state simulate delle sequenze video al variare di questi tre aspetti, in modo tale da poter considerare ogni possibile scenario ipotizzato. Tenendo conto che ogni immagine può essere considerata come una matrice $M \times N$ in cui ogni elemento corrisponde al valore di intensità osservato di ciascun pixel, all'aumentare del numero di righe e di colonne di tale struttura si otterranno *frames* più complessi. In particolare sono stati fissati tre possibili valori per il numero totale di pixel di un *frame*:

- $p = M \times N = 70 \times 72 = 5\,040$ corrisponde ad immagini di dimensioni modeste;
- $p = M \times N = 80 \times 125 = 10\,000$ porta alla costruzione di immagini di dimensione media;
- $p = M \times N = 100 \times 120 = 12\,000$ produce *frames* di dimensioni relativamente grandi.

Per quanto riguarda la dimensione dell'evento anomalo, vengono generati nei dati degli eventi anomali che interessano un gruppo di pixel vicini; la numerosità di tale gruppo influenza dunque la dimensione dell'*hotspot*. Un modo per riassumere in maniera concisa tale parametro può essere definito come "raggio", ovvero l'ampiezza del *cluster* di pixel interessati dall'anomalia; tale parametro viene nel seguito indicato con R_{OC}. Similmente a quanto fatto per il parametro p, sono stati scelti tre valori per questo nuovo parametro:

- $R_{OC} = 2$ identifica un *hotspot* di dimensioni minime, che coinvolge un numero esiguo di pixel;

- $R_{OC} = 5$ corrisponde ad un gruppo di pixel anomali di dimensioni medie e che si nota abbastanza facilmente visionando il video;
- $R_{OC} = 8$ porta alla creazione di un anomalia che è immediatamente riconoscibile ed individuabile osservando la sequenza video, essendo essa molto estesa.

Per quanto riguarda la durata, ancora una volta sono state scelti tre valori possibili, in modo tale da caratterizzare:

- un evento anomalo molto breve ($T_{OC} = 5$), che porta ad un picco nei valori di intensità dei pixel in pochi *frames*;
- un *hotspot* di persistenza media ($T_{OC} = 30$), crea un anomalia nei pixel più duratura e che può essere notata più facilmente;
- un fenomeno fuori controllo marcato ($T_{OC} = 75$), molto persistente nel tempo e facilmente individuabile anche dalla visione del video.

4.1 Generazione di video

Una volta stabiliti quali siano i parametri che devono variare e una volta che per questi sono stati fissati dei valori ritenuti rappresentativi di ciascun scenario, si può procedere alla simulazione delle sequenze video desiderate. In particolare e similmente al caso di studio affrontato in Colosimo e Grasso (2018), si è scelto di simulare ogni immagine che compone ciascun *dataset* in modo tale che i pixel possano essere classificati in due categorie se il processo opera in controllo: (1) pixel di sfondo, in cui non si sta verificando alcun evento di interesse e (2) pixel appartenenti ad un evento che altera momentaneamente la luminosità dei pixel coinvolti. La differenza principale consiste nella diversa luminosità e variabilità che si può osservare in queste due aree. Ricordando che le immagini sono codificate in 8 bit e che dunque i valori assumibili da ciascun pixel variano nell'intervallo $[0, 255]$, per i pixel di sfondo si osserveranno dei valori medi di 80 e poca variabilità, mentre i

pixel interessati da un evento avranno una luminosità media di 180 e una variabilità attorno maggiore attorno a tale valore. Nello specifico, si è scelto di simulare ogni pixel di ciascun *frame* da una mistura di due Normali. Indicando quindi con y il valore osservato per un generico pixel, esso può essere considerato come proveniente da $Y \sim N_{mix}(\boldsymbol{\mu}, \boldsymbol{\sigma^2}, \pi)$, $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top \in \mathbb{R}^2$, $\boldsymbol{\sigma^2} = (\sigma_1^2, \sigma_2^2)^\top \in \mathbb{R}_+^2$, $\pi \in (0, 1)$, avente la seguente funzione di densità:

$$\begin{aligned} f_Y(y; \boldsymbol{\mu}, \boldsymbol{\sigma^2}, \pi) &= f_Y(y; \mu_1, \mu_2, \sigma_1^2, \sigma_2^2, \pi) \\ &= \frac{\pi}{\sqrt{\sigma_1^2}} \phi\left(\frac{y - \mu_1}{\sqrt{\sigma_1^2}}\right) + \frac{(1-\pi)}{\sqrt{\sigma_2^2}} \phi\left(\frac{y - \mu_2}{\sqrt{\sigma_2^2}}\right) \end{aligned} \tag{20}$$

dove $\phi(\cdot)$ è la densità di una $N(0, 1)$. Tenendo conto di quanto detto in precedenza, in questo caso i parametri distributivi della distribuzione in Equazione (20) corrispondono a $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top = (80, 180)^\top$, $\boldsymbol{\sigma^2} = (\sigma_1^2, \sigma_2^2)^\top = (\frac{\mu_1}{8}, \frac{\mu_2}{8})^\top = (10, 22.5)^\top$ e $\pi = 0.75$. In questo modo, i pixel appartenenti allo sfondo in ogni frame corrisponderanno circa al 75% dei pixel totali. I valori di $\boldsymbol{\mu}$ e $\boldsymbol{\sigma^2}$ e π sono stati scelti in modo tale che le intensità dei pixel delle due diverse aree fossero comparabili ai valori osservati nei *dataset* impiegati in Colosimo e Grasso (2018) nel caso di studio esaminato. In questo modo, come si può notare dalla Figura 12, l'area di sfondo è caratterizzata da una variabilità limitata attorno al proprio valore medio, mentre i pixel coinvolti in un evento hanno intensità luminose maggiori e più variabili. È stato poi considerato un ulteriore aspetto per la generazione delle immagini: si è scelto di individuare in modo deterministico in ogni *frame* quali fossero le coordinate soggette al fenomeno che altera i valori di luminosità. Ad ogni istante di osservazione, dunque, non si ottengono dei valori distribuiti casualmente nello spazio di intensità maggiore per le aree interessate dell'evento, che altrimenti risulterebbero meno visibili dall'ispezione delle singole immagini o della sequenza video. In questo modo è possibile raggruppare in zone specifiche i pixel che non appartengono allo sfondo, in modo tale da rendere più visibile il fenomeno di interesse e far seguire a quest'ultimo un percorso prestabilito immagine dopo immagine. Va notato che l'imposizione della condizione per cui vengono scelte quali aree appartengono allo sfondo o

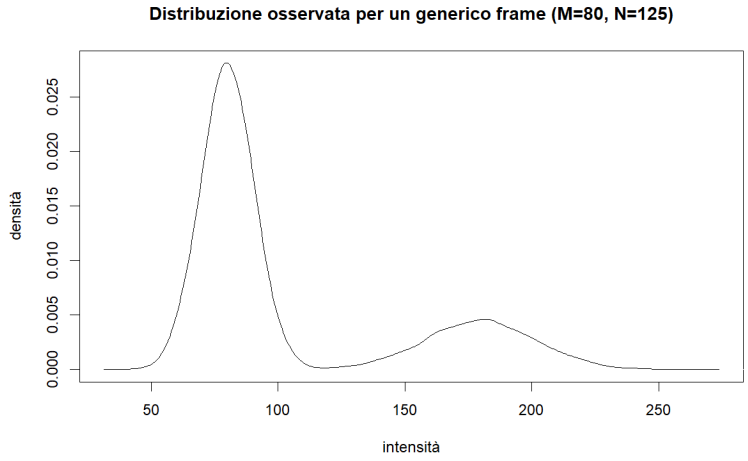


Figura 12: Distribuzione empirica per un generico *frame* di dimensioni $M \times N = 80 \times 125$ generato da una mistura di Normali con funzione di densità definita in Equazione (20).

meno non influenzano la distribuzione del vettore p-dimensionale di pixel di ogni immagine, ma aggiunge ai dati una struttura spaziale che varia (in modo deterministico) nel tempo, che altrimenti sarebbe stata assente se le aree fossero state determinate in modo del tutto casuale ed in modo indipendente ad ogni istante. In Figura 13 si può notare un esempio di alcuni *frames* successivi generati secondo il procedimento appena descritto. L'evento in questo caso corrisponde ad una sequenza di pixel adiacenti che aumentano di intensità per un certo periodo di tempo e secondo un *pattern* prestabilito che si ripete in modo ciclico per tutta la durata del video.

Una volta che sono stati generati i video in condizioni di controllo statistico al variare congiunto di M ed N, si può procedere all'introduzione del fenomeno anomalo nelle sequenze per ogni combinazione di p, R_{OC} e T_{OC}. In totale, dunque, considerando i valori dei parametri fissati, si avranno 3 video in controllo e 27 possibili scenari fuori controllo. Questi ultimi sono ottenuti modificando adeguatamente i video in controllo, inserendo da un certo *frame* e per la durata stabilita un'anomalia – generata mediante l'Equazione (1) – in un gruppo di pixel adiacenti la cui dimensione è regolata dal parametro

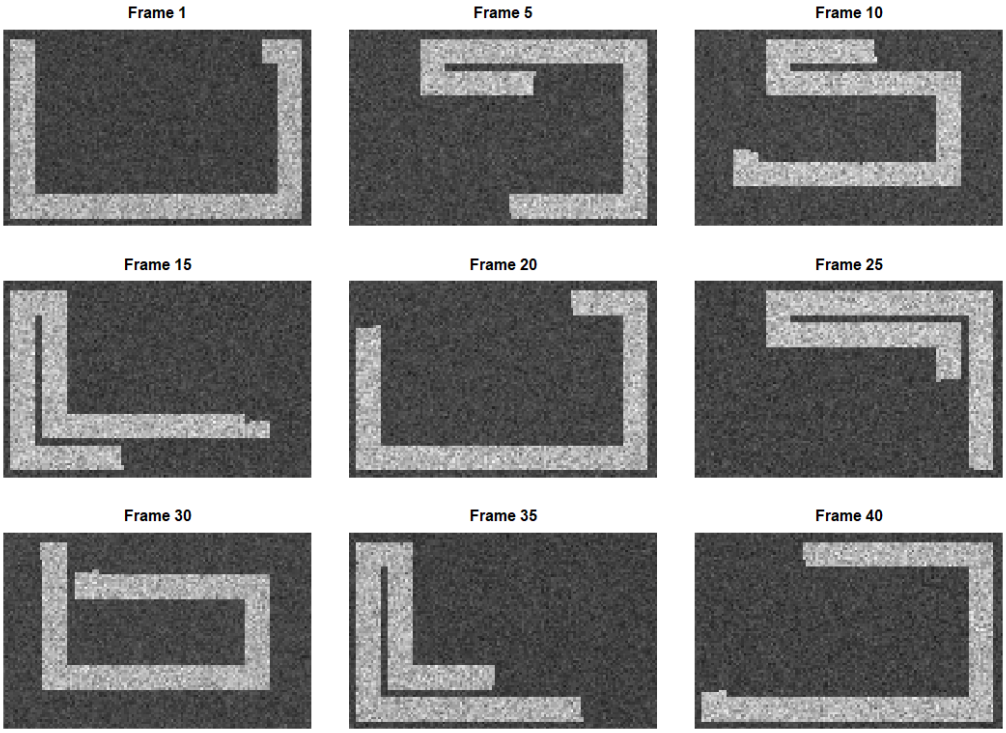


Figura 13: Esempi di *frames* appartenenti ad una sequenza video in controllo. Ciascun *frame* qui riportato ha dimensioni $M \times N = 80 \times 125$ ed è stato generato per simulazione da una mistura di Normali (Equazione (20)).

R_{OC} considerato. In particolare, si è scelto di simulare degli *hotspot* di forma variabile, in modo tale da renderli più facilmente individuabili alla visione del video o delle singole immagini interessate dall'evento. Questo può risultare utile specialmente per le anomalie di piccole o medie entità o di breve durata; in ogni caso, le diverse forme che caratterizzano gli *hotspot* simulati non influenzano le analisi ed i confronti tra procedure che seguono. Più precisamente, in corrispondenza di $R_{OC} = 2$ e $R_{OC} = 5$ le anomalie avranno una forma a croce ("*cross-shaped*"), mentre per $R_{OC} = 8$ l'evento fuori controllo, che avrà grandi dimensioni e quindi sarà facilmente individuabile, sarà un'area quadrata ("*square-shaped*"). Le coordinate centrali dei *cluster* di pixel anomali sono state scelte e fissate in modo casuale. Più precisamente, ad ogni valore di p è stata associata una coordinata che identifica il punto centrale nel quale avrà origine l'*hotspot*: in questo modo, ogni qualvolta si analizzano immagini di una dimensione $M \times N$ fissata, l'anomalia si troverà nella stessa area dell'immagine indipendentemente dalla dimensione e persistenza nel tempo. Dal momento che si devono analizzare molteplici scenari fuori controllo, tale scelta è motivata dal fatto di non voler creare eccessiva confusione con i diversi video generati. In particolare:

- se $p = 5\,040$, il pixel al centro dell'anomalia generata sarà in posizione $(m_{OC}, n_{OC}) = (35, 30)$, ossia nell'area centrale dell'immagine;
- se $p = 10\,000$, il pixel centrale dell'*hotspot* si troverà alle coordinate $(m_{OC}, n_{OC}) = (70, 100)$, cioè nella zona in alto a destra dei *frames*;
- se $p = 12\,000$, il pixel posto al centro dell'evento anomalo sarà alle coordinate $(m_{OC}, n_{OC}) = (20, 40)$, cioè nell'area in basso a sinistra delle immagini.

Nelle Figure 14, 15 e 16 si possono osservare alcuni dei *frames* interessati da un evento fuori controllo. In Figura 14 sono riportate delle immagini dai video generati imponendo $R_{OC} = 2$ e $T_{OC} = 30$ per ognuna delle possibili dimensioni $M \times N$: sono esempi di anomalie di piccola entità di media persistenza. In Figura 15 poi vengono mostrati i *frames* generati imponendo

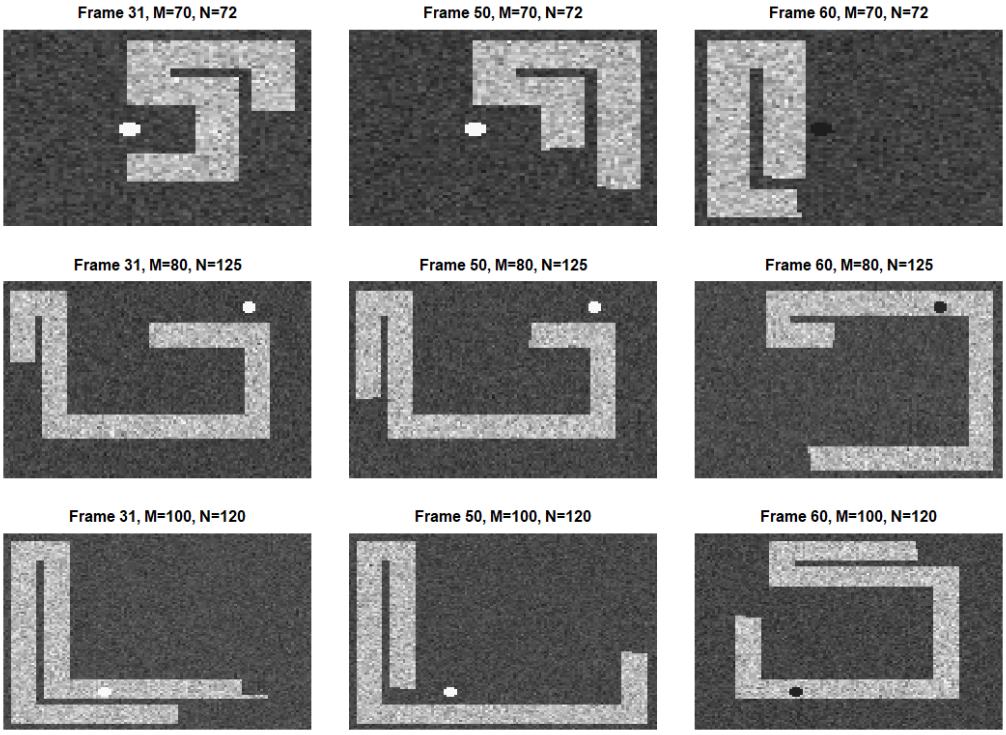


Figura 14: *Frames* generati imponendo $R_{OC} = 2$ (piccola entità) e $T_{OC} = 30$ (durata media) al variare di p. Nella prima riga si hanno immagini $M \times N = 70 \times 72$, nella seconda $M \times N = 80 \times 125$ e nella terza $M \times N = 100 \times 120$.

$R_{OC} = 5$ e $T_{OC} = 75$ al variare di p; questi sono esempi di anomalie di media entità e di durata estesa. In Figura 16 infine si possono notare delle immagini generate imponendo $R_{OC} = 8$ e $T_{OC} = 5$ per diversi valori di p; questi ultimi sono esempi di *hotspot* di grandi dimensioni ma di breve durata. In queste immagini si può notare come i pixel interessati dall'anomalia mostrino un picco nei valori di luminosità osservati che persiste nel tempo, per poi tornare gradualmente a dei valori più bassi ed in controllo.

In Appendice A.2 sono riportate le funzioni impiegate per ottenere e replicare i risultati esposti in questa sezione. Le procedure esposte in questa sezione e nelle prossime state implementate utilizzando RStudio (RStudio Team, 2015) sfruttando il *software* R in versione 4.0.0 (R Core Team, 2020).

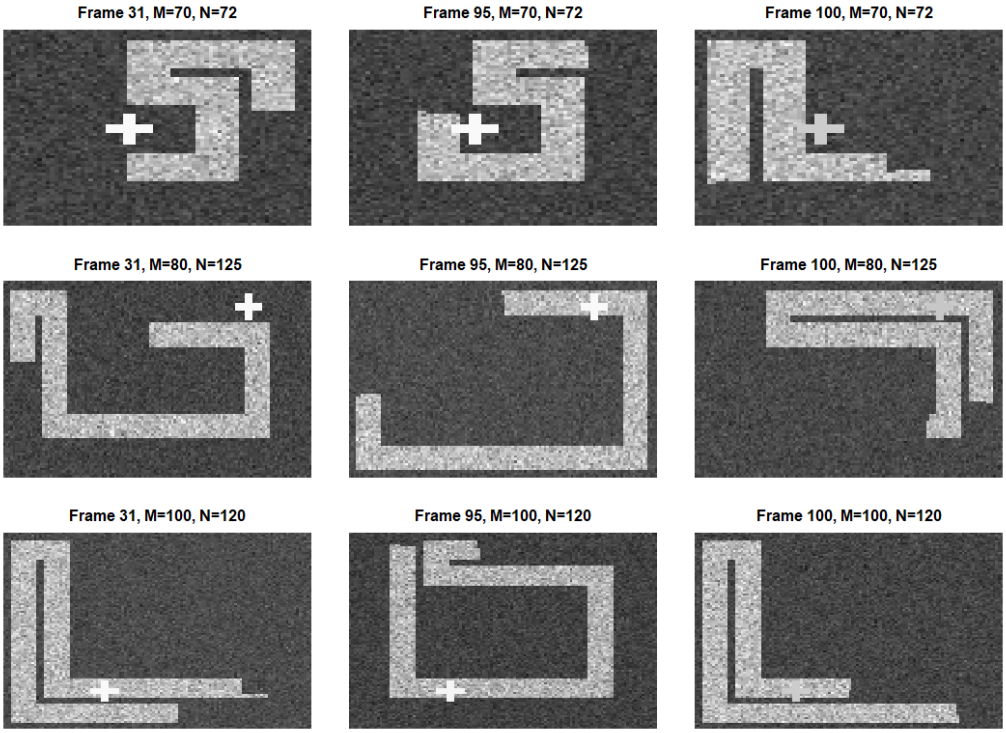


Figura 15: *Frames* generati imponendo $R_{OC} = 5$ (entità media) e $T_{OC} = 75$ (durata estesa) al variare di p. In ogni riga si hanno immagini contenenti rispettivamente 5 040, 10 000 e 12 000 pixel.

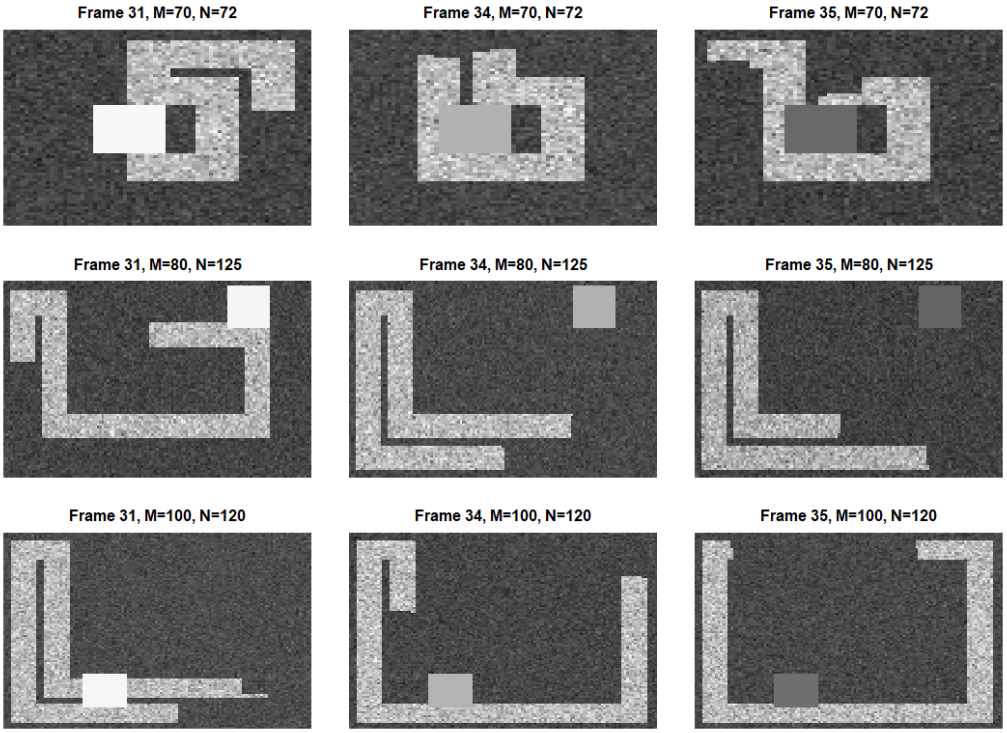


Figura 16: *Frames* generati imponendo $R_{OC} = 8$ (grandi dimensioni) e $T_{OC} = 5$ (durata breve) al variare di p. Nella prima riga si hanno immagini di dimensione 70×72, nella seconda 80×125 e nella terza 100×120.

4.2 Implementazione delle procedure *ST-PCA* e *SA-SAM*

Gli algoritmi *ST-PCA* e *SASAM*, descritti rispettivamente nei Capitoli 3.1 e 3.2, sono stati quindi implementati per poter sorvegliare delle sequenze video al fine di rilevare in queste degli eventi *hotspot* anomali. Entrambe le funzioni create sono caratterizzate dalla possibilità di ricevere in input sia il nome di un *file* video sia un *array* contenente le immagini. Le funzioni sono ideate in modo tale da elaborare i dati forniti in input per potersi ricondurre a quest'ultimo tipo di struttura. L'*array* avrà infatti dimensioni $M \times N \times J$: esso infatti sarà una raccolta di J matrici $M \times N$ che rappresentano i singoli fotogrammi del video. Nel caso venga fornito in ingresso il nome di un video salvato in un *file* esterno, sono state create delle apposite funzioni che riescono ad estrarre da esso le informazioni utili, salvare i singoli *frames* e creare l'*array* desiderato (Appendice A.1). Entrambe le funzioni poi includono l'opzione che venga fornita in input la matrice delle distanze tra i pixel: essa contiene le distanze di ciascun pixel da ogni altro presente nell'immagine e dipende unicamente dalla dimensione dei *frames* che si intendono analizzare. Tale elemento è di dimensione $p \times p$, dunque il numero di elementi che contiene aumenta velocemente all'aumentare della dimensione delle immagini. È inoltre una matrice simmetrica sulla cui diagonale (ossia gli elementi che indicano la distanza di ogni pixel da se stesso) vi sono unicamente valori nulli. Nonostante quindi ci si possa limitare al calcolo dei soli valori presenti nella parte triangolare superiore (o inferiore) e ottenere i rimanenti per simmetria, tale operazione rimane tra le più onerose in termini computazionali per entrambe le carte di controllo (si devono infatti calcolare almeno $p - 1$ distanze). Dal momento che per stimare tale elemento è solamente necessario essere a conoscenza della dimensione delle immagini da analizzare, appare sensato assumere come disponibile – in quanto calcolata offline – la matrice delle distanze tra i pixel, prima di avviare qualsiasi procedura di sorveglianza. Pertanto fornire tale matrice in input alle funzioni permette di alleggerire notevolmente il carico computazionale richiesto una

volta che inizia la sorveglianza.

Si considerano ora i due algoritmi delle carte di controllo individualmente. La funzione che permette di implementare l'algoritmo *ST-PCA* prevede che vengano forniti in input:

- `file.video` o `data`: i dati da sorvegliare; come spiegato sopra, possono essere sia il nome del video da analizzare sia un *array* che contiene i *frames*.

- `weight.type`: stabilisce quale definizione di matrice di pesi spaziali $\boldsymbol{W}$ utilizzare; può essere pari ad 1,2 o 3 a seconda che si utilizzi $\boldsymbol{W}_1$, $\boldsymbol{W}_2$ o $\boldsymbol{W}_3$ (Equazioni (7), (8), (9)).

- `W0.mat`: è la matrice delle distanze tra i pixel dell'immagine che può essere fornita in input.

- `SOGLIA_r`: influenza la definizione dei pesi spaziali $\boldsymbol{W}_2$ e $\boldsymbol{W}_3$; corrisponde al parametro r che stabilisce la distanza di riferimento fissata come soglia.

- `moving.window`: parametro logico che regola l'aggiornamento iterativo della matrice dei dati $\boldsymbol{X}$; se `moving.window=TRUE` allora si seguirà l'approccio a finestra mobile, altrimenti si seguirà l'approccio ricorsivo (Capitolo 3.1).

- `L`: se `moving.window=TRUE` è il parametro che regola l'ampiezza della finestra mobile per l'aggiornamento della matrice dei dati *unfolded*.

- `SOGLIA_var_pc`: soglia sulla variabilità spiegata dalle componenti principali $\boldsymbol{Z} = \boldsymbol{XV}$ da considerare; influenza il numero m_{pc} di componenti che vengono mantenute per l'analisi ad ogni iterazione.

- `Kmax`: parametro necessario per il criterio di scelta del corretto numero k di gruppi per il *clustering* delle statistiche riassuntive T_i^2, $i = 1, \ldots, p$.

- `plots`: parametro logico che stabilisce se stampare a schermo o meno dei grafici riassuntivi dell'analisi eseguita ad ogni istante di osservazione.

- `N_FRAME_START` e `STOP_FRAME`: stabiliscono da che *frame* far partire la sorveglianza e a quale fermarsi. In particolare, `N_FRAME_START` è utile qualora si desideri osservare un *batch* iniziale di *frames* prima di iniziare la sorveglianza, mentre `STOP_FRAME` è adatto per interrompere la sorveglianza una volta che è stato analizzato un numero prefissato di immagini.

- `seed`: serve per ottenere i medesimi risultati replicando l'analisi.

Tra i parametri elencati, `Kmax` richiede un'ulteriore spiegazione. Per il *clustering* delle statistiche T_i^2, $i = 1, \ldots, p$, la selezione del numero di gruppi k più adatto a descrivere i dati avviene ricercando un punto di gomito nelle somme delle distanze al quadrato entro i gruppi $SSW(k)$ (Equazione (11)). Un modo comune per trovare tale valore consiste nel calcolare le distanze, indicate da $D(k)$, tra $SSW(k)$ e la retta che collega i due punti estremi, $SSW(1)$ e $SSW(K)$, per un certo $K < p$ fissato e per $k = 1, \ldots, K$ (Grasso, Laguzza et al., 2017). Il gomito della funzione è quindi individuato da $\hat{k} = \arg\max_k D(k)$. Questo criterio è adatto quando il numero minimo di *cluster* è $k = 2$, che è il caso dell'applicazione in esame, in cui la distribuzione spaziale in controllo di $T^2(m, n)$ consiste in due gruppi e la presenza di raggruppamenti aggiuntivi è un sintomo della presenza di anomalie locali. Al variare di K dunque varia anche l'inclinazione della retta che congiunge $SSW(1)$ e $SSW(K)$, portando ad ottenere risultati distorti, qualora tale valore fosse scelto troppo grande o viceversa troppo piccolo (Grasso, Laguzza et al., 2017).

La funzione infine fornisce in output una lista di elementi:

- `alarm.time`, ossia l'istante in cui è stato lanciato l'allarme dalla procedura;

- `m_pc`, ossia un vettore che contiene l'elenco del numero di componenti principali conservate ad ogni iterazione;

- `ssw_k`, una matrice contenente nelle colonne i valori stimati di $SSW(k)$ ad ogni iterazione;

- `T2.stat`, una matrice contenente nelle colonne i valori stimati di T_i^2, $i = 1, \ldots, p$, ad ogni iterazione;

- `frame`, per avere un riferimento preciso tra frame analizzati e statistiche riassuntive fornite in output e che ritorna utile nel caso la sorveglianza non cominci dal primo *frame*;

- `Dk`, una matrice contenente nelle colonne i valori stimati di $D(k)$ ad ogni iterazione;

- `dati.clust.k`, cioè gli indici alle statistiche riassuntive T_i^2 che stabiliscono il gruppo di appartenenza di ogni elemento per il valore k scelto dal criterio.

L'implementazione dell'algoritmo *SASAM* avviene tramite una funzione che riceve in input i seguenti argomenti:

- `file.video`, `data`, `W0.mat`, `plots`, `STOP_FRAME` e `seed`: questi parametri hanno la stessa interpretazione di quelli forniti in input all'algoritmo *ST-PCA* descritto precedentemente; si rimanda alla precedente descrizione per tali valori.

- `q`: numero di variabili osservabili ad ogni istante espresso in termini relativi, ovvero solo il `q`% delle $M \times N$ variabili (i pixel) che compongono i fotogrammi è osservabile ad ogni istante, $1 < q \leq 100$. Se `q` è pari a 100 ciò equivale ad osservare completamente ogni *frame*.

- `theta`: corrisponde al vettore bidimensionale $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top$ che regola il funzionamento dello schema di campionamento nel tempo (Equazione (19)).

- `u_min`: entità minima di interesse del cambiamento che si intende individuare; corrisponde al parametro u_{min} introdotto in Equazione (14).

- `h`: corrisponde al parametro h che regola l'ampiezza di banda della funzione di Equazione (15); ha un'interpretazione simile al parametro `SOGLIA_r` della carta di controllo *ST-PCA*.

- `H`: limite di controllo della carta; deve essere calcolato prima di iniziare la sorveglianza tramite simulazione da dati in controllo disponibili offline.

Per il calcolo del limite di controllo H si può procedere tramite simulazione Monte Carlo e *bootstrap*, simulando dei campioni di dati a partire da *frames* in controllo disponibili offline. Il codice necessario per la stima di tale valore è riportato in Appendice A.4; in particolare, si è seguita la procedura proposta in K. Liu, Mei et al. (2015). Dato un valore ARL_U in controllo prestabilito, si stima H secondo il seguente processo:

1. Si fissano H_{min} e H_{max}, due valori rispettivamente piccolo e grande come limiti inferiore e superiore per H. Si fissa quindi $H = \frac{H_{min}+H_{max}}{2}$.

2. Si genera tramite *bootstrap* un *dataset* con J osservazioni (*frames*), estraendo con reinserimento dai dati in controllo disponibili offline.

3. Si esegue dunque la procedura *SASAM* e si registra l'indice della prima osservazione fuori controllo, RL (*run length*).

4. Si ripetono i passaggi 2–3 per N volte, il che porta ad ottenere N replicazioni *bootstrap* di RL. Si calcola quindi la media dei valori ottenuti, $\bar{RL}$.

5. Se $\bar{RL}$ è più grande di ARL_U, allora si fissa $H_{max} = H$, altrimenti $H_{min} = H$. Si aggiorna quindi il valore $H = \frac{H_{min}+H_{max}}{2}$.

6. Si ripetono i passaggi 2–5 finché la differenza tra $\bar{RL}$ e ARL_U è piccola.

Nell'applicazione considerata, si è fissato $ARL_U = 200$, il che equivale ad accettare di lanciare un falso allarme, dato che il processo è in controllo, con una probabilità del 0.5%.

A differenza dell'algoritmo *ST-PCA*, la procedura *SASAM* richiede che i dati da sorvegliare si distribuiscano normalmente, come descritto in Equazione (12), affinché si ottengano risultati non distorti. Per tale motivo, visto che i dati da sorvegliare sono stati simulati da una mistura di due distribuzioni Gaussiane (Equazione (12)), è opportuno sorvegliare i residui di un modello che descriva adeguatamente tali dati piuttosto che semplici osservazioni. Dal momento che si conosce il processo generatore dei dati, si può scegliere un modello di regressione che consenta di ottenere dei residui che rispettino le assunzioni distributive imposte dalla carta di controllo. Va notato che in assenza di tale informazione in merito al processo generatore dei dati che impedisce di preferire facilmente un modello piuttosto che un altro, occorre comunque trovare un modello adatto che porti ad ottenere dei residui normalmente distribuiti. Un esempio è dato dall'algoritmo *MOUSSE* (*Multiscale Online Union of SubSpaces Estimation*, Xie, Huang et al. (2012)), suggerito dagli autori stessi della procedura *SASAM*. Dal momento che l'implementazione e l'utilizzo di modelli spazio-temporali per la manipolazione di immagini esula dal tema esposto nella presente trattazione, si è optato per un modello più semplice e adatto ai dati generati per simulazione impiegati per l'analisi.

Sono poi state implementate due diverse versioni della carta di controllo *SASAM*: una bilaterale ed una unilaterale. La prima corrisponde alla procedura descritta nel Capitolo 3.2, mentre la seconda è costruita per identificare esclusivamente degli *shift* positivi nella media delle variabili osservate. Dal momento che si può assumere come nota la natura di un evento classificabile come anomalo e che porta dunque il processo in uno stato fuori controllo, è possibile usare solamente quest'ultima forma della carta di controllo per svolgere la sorveglianza. Ciò che differenzia le due procedure è da ricercare principalmente nella definizione delle statistiche spaziali $W_{k,t}^{(1)}$ e $W_{k,t}^{(2)}$ (Equazione (14)): queste infatti sono due statistiche *CUSUM* nelle quali viene considerata l'informazione spaziale intrinseca delle variabili osservate, ossia i pixel. Si può quindi considerare unicamente la statistica spaziale $W_{k,t}^{(1)}$, che

considera scostamenti positivi dalla media in controllo. Tale scelta si ripercuote sul calcolo del limite di controllo H: esso infatti assumerà valori diversi a seconda che la carta sia bilaterale o meno. Ciò nonostante, la procedura *bootstrap* descritta precedentemente per il calcolo di H rimane la stessa e non comporta quindi alcuna complicazione.

L'output della funzione infine consiste in una lista di elementi:

- `alarm.time`, ossia l'istante in cui è stato lanciato l'allarme dalla procedura;
- `St.list`, cioè un vettore che contiene l'elenco delle statistiche di controllo S_t^{SASAM} stimate ad ogni istante di osservazione;
- `W1_t`, una matrice contenente nelle colonne i valori stimati ad ogni iterazione di $W_{k,t}^{(1)}$. Se la carta di controllo è bilaterale viene fornita anche la matrice `W2_t`, la cui interpretazione è analoga;
- `qwt` e `qdt`, due vettori che raccolgono il numero di variabili sorvegliate estratte secondo la strategia di ricerca estesa e profonda, rispettivamente;
- `Owt` e `Odt`, due liste contenenti le coordinate delle variabili sorvegliate ed estratte rispettivamente dalle *wide search strategy* e *deep search strategy* per ogni fotogramma.

4.3 Risultati dell'analisi

Vengono ora riportati i risultati ottenuti dal confronto delle procedure *ST-PCA* e *SASAM* secondo le linee generali esposte in apertura del Capitolo 4. Per quanto riguarda le carte di controllo impiegate, i parametri di regolazione che decidono il comportamento sono stati impostati seguendo le linee generali fornite dagli autori delle procedure stesse, laddove possibile, in quanto non si assume come nota la natura (dimensione, posizione o durata) dell'anomalia che può manifestarsi in una sequenza video. In particolare, per la carta di

controllo *ST-PCA* è stato fissato $r = 5$, $m_{pc} = 0.8$, si è adottato l'approccio ricorsivo e la sorveglianza ha avuto inizio dopo che erano stati acquisiti 25 *frames*. Per quanto riguarda la definizione della matrice di pesi spaziali $\boldsymbol{W}$, si è notato che le diverse alternative proposte nelle Equazioni (7), (8), (9) non portano a differenze significative nei risultati ottenuti; i valori riportati nel seguito fanno tutti riferimento a quanto ottenuto scegliendo $\boldsymbol{W} = \boldsymbol{W}_2$. In merito alla scelta del parametro K che regola la selezione del numero di *cluster* individuati nella statistiche T_i^2, $i = 1, \ldots, p$, sono stati provati diversi valori compresi tra 8 e 20. È stato osservato che valori troppo bassi tendono a non fornire una discriminazione adeguata del numero di gruppi presenti nelle statistiche T_i^2, propendendo nel preferire $k = 2$ anche laddove sia evidente la presenza di un'anomalia nei dati. Viceversa, se K viene scelto eccessivamente grande, oltre ad ottenere un rallentamento nell'analisi dovuto al maggior numero di *cluster* da individuare, si è notato un aumento nel numero di falsi allarmi lanciati: la discriminazione tra diversi valori di k diventa fin troppo sensibile anche a valori o gruppi di valori che non sono di fatto estremi, fornendo una suddivisione dei dati troppo fine. Inoltre, si è osservato come K debba essere scelto progressivamente maggiore all'aumentare di p: valori di K che fornivano risultati soddisfacenti dal punto di vista dell'efficacia della sorveglianza risultavano essere troppo bassi per valori di p maggiori. In generale si è riscontrato che per $p = 5\,040$ e $p = 10\,000$, dei valori di K adeguati erano rispettivamente 9 e 10, mentre per $p = 12\,000$ tale parametro doveva essere impostato pari a 15 o 20 per ottenere dei risultati apprezzabili. I risultati riportati nel seguito fanno riferimento a quanto ottenuto impostando K come appena descritto e rappresentano perciò i risultati migliori ottenuti dalla sorveglianza degli scenari fuori controllo tramite la procedura *ST-PCA*. Per la carta di controllo *SASAM* invece sono stati fissati $h = 5$, in modo che la dimensione dell'area anomala ricercata dalle procedure fosse simile, q pari al 5% del numero totale di variabili p osservabili, $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top = (0.2, 0.5)^\top$ e $u_{min} = 1.5$. Per questi valori dei parametri di regolazione sono stati quindi calcolati i limiti di controllo H secondo quanto descritto in 4.2, ottenendo

tre diverse quantità per le tre diverse numerosità di variabili da sorvegliare.

Nelle Tabelle 1, 2, 3 sono riportati i valori degli istanti in cui è stato lanciato un allarme da ciascuna procedura a seconda dello scenario considerato. In ogni tabella la dimensione $p = M \times N$ delle immagini sorvegliate assieme alla posizione (m_{OC}, n_{OC}) dell'anomalia sono costanti. Vengono anche riportate le posizioni stimate delle anomalie, $(\hat{m}_{OC}, \hat{n}_{OC})$, intese come punto centrale dell'area fuori controllo rilevata, qualora un allarme sia stato lanciato. Per poter fare riferimento a ciascuno dei 27 video contenenti un'anomalia in modo agevole e per evitare di appesantire la notazione nelle tabelle, ciascun scenario fuori controllo verrà in queste identificato da una terna di valori. Quest'ultima sarà del tipo

$$(p_i, R_j, T_k) \qquad i, j, k = 1, 2, 3 \tag{21}$$

dove $p_i \in \{5\,040, 10\,000, 12\,000\}$ indica la dimensione ($p = M \times N$) delle immagini dello scenario considerato, $R_j \in \{2, 3, 8\}$ identifica l'estensione dell'anomalia (R_{OC}), ed infine $T_k \in \{5, 30, 75\}$ rimanda alla durata (T_{OC}) del fenomeno fuori controllo. Ciò che emerge dall'esame di questi risultati può essere così riassunto:

- la procedura *SASAM* riesce sempre, in qualsiasi scenario, a rilevare l'anomalia nel flusso dei dati e a lanciare un allarme in modo tempestivo; l'evento fuori controllo ha infatti inizio all'istante 31 mentre la carta di controllo segnala l'anomalia solamente al *frame* successivo in ogni scenario. Al contrario, l'*ST-PCA* non riesce sempre a rilevare l'anomalia; in particolare, indipendentemente dalla dimensione della stessa e delle immagini analizzate, se l'evento fuori controllo ha una durata troppo breve esso non viene rilevato in alcun scenario.

- la reattività della carta di controllo *ST-PCA* aumenta all'aumentare della dimensione dell'evento fuori controllo (R_{OC}): maggiore è l'estensione spaziale dell'anomalia, minore è il numero di *frames* successivi all'istante τ che devono essere acquisiti affinché un allarme venga lanciato e la procedura di conseguenza arrestata. Di contro, la reattività

Tabella 1: Risultati della sorveglianza di immagini di dimensione $M \times N = 70 \times 72$ ($p = 5\,040$ pixel totali), anomalia in posizione $(m_{OC}, n_{OC}) = (35, 30)$ di dimensione (R_{OC}) e persistenza (T_{OC}) variabili. In grassetto sono evidenziate le performance migliori e "–" indica che non è stato segnalato alcun allarme dalla procedura.

Scenario	Approccio	Istante del primo allarme (indice del *frame*)	$\hat{m}_{OC}$	$\hat{n}_{OC}$
(p_1, R_1, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	34.00	28.71
(p_1, R_1, T_2)	*ST-PCA*	53	35.00	30.00
	SASAM	**32**	34.00	29.00
(p_1, R_1, T_3)	*ST-PCA*	53	35.00	30.00
	SASAM	**32**	34.00	29.00
(p_1, R_2, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	32.97	29.90
(p_1, R_2, T_2)	*ST-PCA*	45	39.21	30.32
	SASAM	**32**	33.00	30.00
(p_1, R_2, T_3)	*ST-PCA*	45	39.21	30.32
	SASAM	**32**	33.00	30.00
(p_1, R_3, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	38.00	23.00
(p_1, R_3, T_2)	*ST-PCA*	33	35.75	30.31
	SASAM	**32**	38.00	22.89
(p_1, R_3, T_3)	*ST-PCA*	33	35.75	30.31
	SASAM	**32**	38.00	22.89

Tabella 2: Risultati della sorveglianza di immagini di dimensione $M \times N = 80 \times 125$ ($p = 10\,000$ pixel totali), anomalia in posizione $(m_{OC}, n_{OC}) = (70, 100)$ di dimensione (R_{OC}) e durata (T_{OC}) variabili. In grassetto sono evidenziate le performance migliori e "–" indica che non è stato segnalato alcun allarme dalla procedura.

Scenario	Approccio	Istante del primo allarme (indice del *frame*)	$\hat{m}_{OC}$	$\hat{n}_{OC}$
(p_2, R_1, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	72.00	98.82
(p_2, R_1, T_2)	*ST-PCA*	48	70.00	100.00
	SASAM	**32**	72.00	99.00
(p_2, R_1, T_3)	*ST-PCA*	48	70.00	100.00
	SASAM	**32**	72.00	99.00
(p_2, R_2, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	71.00	100.89
(p_2, R_2, T_2)	*ST-PCA*	42	70.53	100.00
	SASAM	**32**	70.95	100.95
(p_2, R_2, T_3)	*ST-PCA*	42	70.53	100.00
	SASAM	**32**	70.95	100.95
(p_2, R_3, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	69.95	100.79
(p_2, R_3, T_2)	*ST-PCA*	36	70.04	99.65
	SASAM	**32**	70.00	100.89
(p_2, R_3, T_3)	*ST-PCA*	36	70.04	99.65
	SASAM	**32**	70.00	100.89

Tabella 3: Risultati della sorveglianza di immagini di dimensione $M \times N = 100 \times 120$ ($p = 12\,000$ pixel totali), anomalia in posizione $(m_{OC}, n_{OC}) = (20, 40)$ di dimensione (R_{OC}) e persistenza (T_{OC}) variabili. In grassetto sono evidenziate le performance migliori e "–" indica che non è stato segnalato alcun allarme dalla procedura.

Scenario	Approccio	Istante del primo allarme (indice del *frame*)	$\hat{m}_{OC}$	$\hat{n}_{OC}$
(p_3, R_1, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	21.00	41.00
(p_3, R_1, T_2)	*ST-PCA*	53	20.00	40.00
	SASAM	**32**	21.00	40.93
(p_3, R_1, T_3)	*ST-PCA*	53	20.00	40.00
	SASAM	**32**	21.00	40.93
(p_3, R_2, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	20.00	38.94
(p_3, R_2, T_2)	*ST-PCA*	45	20.53	40.00
	SASAM	**32**	20.00	38.91
(p_3, R_2, T_3)	*ST-PCA*	45	20.53	40.00
	SASAM	**32**	20.00	38.91
(p_3, R_3, T_1)	*ST-PCA*	–	–	–
	SASAM	**32**	20.00	38.91
(p_3, R_3, T_2)	*ST-PCA*	39	20.70	40.15
	SASAM	**32**	20.00	38.92
(p_3, R_3, T_3)	*ST-PCA*	39	20.70	40.15
	SASAM	**32**	20.00	38.92

dell'algoritmo *SASAM* si mantiene costante al variare di p, R_{OC} e T_{OC} e resta maggiore rispetto a quella del *competitor* in ogni scenario.

- se viene lanciato un allarme, la posizione dell'anomalia viene stimata da entrambe le carte di controllo in modo abbastanza preciso e simile, avvicinandosi ai valori reali. Tale informazione aiuta nelle analisi e nel confronto effettuati dal momento che, essendo note le reali coordinate (m_{OC}, n_{OC}), essa permette di capire se è stato rilevato davvero l'evento fuori controllo o se si tratta solamente di un falso allarme.

Per quanto riguarda quest'ultimo punto, in Tabella 4 è riportato un confronto tra le precisioni delle due carte di controllo nello stimare la posizione dell'anomalia, qualora venga lanciato un allarme. In questo contesto, per "precisione" si intende la distanza spaziale tra la reale posizione dell'evento fuori controllo e quella stimata dalla procedura. In tal senso, quanto più tale distanza è piccola tanto più una carta di controllo può essere definita "precisa". Si può quindi notare come l'*ST-PCA* riesca tendenzialmente ad individuare con più accuratezza rispetto alla procedura concomitante il sito nel quale ha luogo il fenomeno fuori controllo. Le uniche eccezioni a tale comportamento si notano, ovviamente, nei casi in cui la procedura non riesca a lanciare un allarme e nei due scenari caratterizzati da video di dimensione ridotta contenenti anomalie di dimensioni medio-grandi e di durata medio-lunga. Si può quindi constatare che la carta di controllo *ST-PCA*, sebbene sia meno efficace nel rilevare un'anomalia e lanciare di conseguenza un allarme, risulta più efficiente nell'individuare la posizione dell'evento fuori controllo. Va comunque detto che entrambi gli algoritmi non forniscono stime eccessivamente distorte di tali coordinate e i valori di precisione riportati sono di ordine di grandezza comparabile.

A differenza della carta di controllo *SASAM*, l'*ST-PCA* fornice un'ulteriore informazione utile per le analisi qualora un allarme venga lanciato, ovvero una stima della dimensione dell'evento fuori controllo. Questa infatti può essere facilmente ottenuta in seguito al *clustering* delle statistiche T_i^2, $i = 1, \ldots, p$, all'istante $\hat{\tau}$, dove $\hat{\tau}$ è l'indice del *frame* al quale viene lancia-

Tabella 4: Precisione nella stima della posizione dell'anomalia rilevata, in termini di distanza dai veri valori di (m_{OC}, n_{OC}). In grassetto sono evidenziate le performance migliori e "–" indica che non è stato segnalato alcun allarme dalla procedura.

Scenario	*ST-PCA*	*SASAM*	Scenario	*ST-PCA*	*SASAM*
(p_1, R_1, T_1)	–	**1.629**	(p_2, R_2, T_3)	**0.530**	1.338
(p_1, R_1, T_2)	**0.000**	1.414	(p_2, R_3, T_1)	–	**0.794**
(p_1, R_1, T_3)	**0.000**	1.4142	(p_2, R_3, T_2)	**0.352**	0.8901
(p_1, R_2, T_1)	–	**2.036**	(p_2, R_3, T_3)	**0.352**	0.8901
(p_1, R_2, T_2)	4.222	**2.000**	(p_3, R_1, T_1)	–	**1.414**
(p_1, R_2, T_3)	4.222	**2.000**	(p_3, R_1, T_2)	**0.000**	1.366
(p_1, R_3, T_1)	–	**7.616**	(p_3, R_1, T_3)	**0.000**	1.366
(p_1, R_3, T_2)	**0.812**	7.714	(p_3, R_2, T_1)	–	**1.065**
(p_1, R_3, T_3)	**0.812**	7.714	(p_3, R_2, T_2)	**0.530**	1.093
(p_2, R_1, T_1)	–	**2.325**	(p_3, R_2, T_3)	**0.530**	1.093
(p_2, R_1, T_2)	**0.000**	2.236	(p_3, R_3, T_1)	–	**1.093**
(p_2, R_1, T_3)	**0.000**	2.236	(p_3, R_3, T_2)	**0.716**	1.082
(p_2, R_2, T_1)	–	**1.339**	(p_3, R_3, T_3)	**0.716**	1.082
(p_2, R_2, T_2)	**0.530**	1.338			

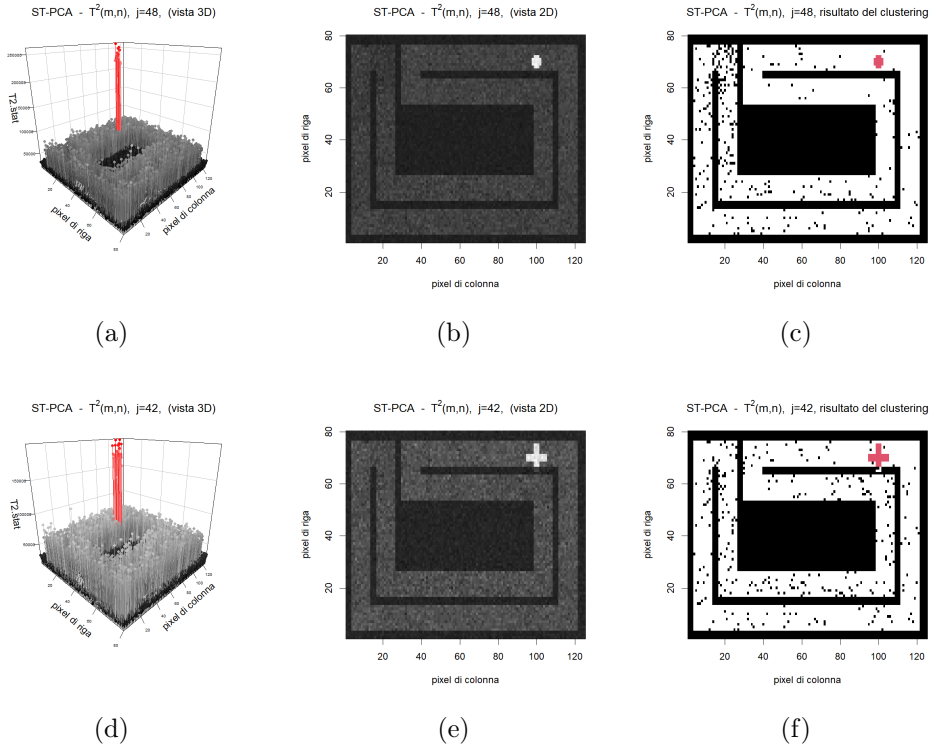


Figura 17: Due esempi di output della procedura *ST-PCA*, nel caso di immagini contenenti $p = 10\,000$ pixel. Le Figure (a), (b) e (c) fanno riferimento allo scenario in cui si ha luogo un evento anomalo di dimensioni ridotte e di durata media, mentre le Figure (d), (e) e (f) sono riferite all'analisi condotta sul video contenente un'anomalia di medie dimensione e di durata media.

to l'allarme, che coincide con una stima di τ, l'istante dal quale ha inizio il cambiamento nel processo. Se viene segnalata la presenza di un'anomalia infatti, l'*ST-PCA* fornisce informazioni precise circa i raggruppamenti che hanno portato all'interruzione della sorveglianza e, semplicemente guardando alla cardinalità dei *cluster* ottenuti ed in particolare a quello corrispondente ai valori maggiori di T_i^2, si può ottenere una stima della dimensione del fenomeno anomalo. In Figura 17 sono riportati due esempi di output che fornisce la procedura *ST-PCA*, che mostrano come, grazie al *clustering*, si riescano ad individuare la posizione e la dimensione dell'*hotspot*. Tale informazione

Tabella 5: Dimensioni effettive (in numero di pixel) delle anomalie presenti nei video analizzati e dimensioni stimate dalla procedura *ST-PCA*. Se non è stato lanciato alcun allarme viene invece riportato "–".

Scenario	Dimensioni effettive	Dimensioni stimate	Scenario	Dimensioni effettive	Dimensioni stimate
(p_1, R_1, T_1)	21	–	(p_2, R_2, T_3)	57	51
(p_1, R_1, T_2)	21	21	(p_2, R_3, T_1)	289	–
(p_1, R_1, T_3)	21	21	(p_2, R_3, T_2)	289	204
(p_1, R_2, T_1)	57	–	(p_2, R_3, T_3)	289	204
(p_1, R_2, T_2)	57	73	(p_3, R_1, T_1)	21	–
(p_1, R_2, T_3)	57	73	(p_3, R_1, T_2)	21	21
(p_1, R_3, T_1)	289	–	(p_3, R_1, T_3)	21	21
(p_1, R_3, T_2)	289	354	(p_3, R_2, T_1)	57	–
(p_1, R_3, T_3)	289	354	(p_3, R_2, T_2)	57	51
(p_2, R_1, T_1)	21	–	(p_3, R_2, T_3)	57	51
(p_2, R_1, T_2)	21	21	(p_3, R_3, T_1)	289	–
(p_2, R_1, T_3)	21	21	(p_3, R_3, T_2)	289	206
(p_2, R_2, T_1)	57	–	(p_3, R_3, T_3)	289	206
(p_2, R_2, T_2)	57	51			

circa le dimensioni effettive – o quantomeno stimate – dell'anomalia non è invece disponibile per la procedura *SASAM*. Essa può al massimo fornire il numero di variabili-W e di variabili-D allocate nel corso della sorveglianza e all'istante dell'allarme, ma informazioni precise in merito alla reale dimensione dell'*hotspot* non sono in tal caso reperibili. In Tabella 5 sono riportate le dimensioni effettive e stimate delle anomalie rilevate dall'*ST-PCA*. In particolare, si nota come essa riesca a stimare in modo accurato l'estensione di *hotspot* di piccole dimensioni, mentre tenda a sottostimare (o sovrastimare) la grandezza di anomalie di dimensioni medio-grandi, indipendentemente dalla dimensione delle immagini analizzate.

5 Conclusioni

I recenti progressi dei metodi impiegati per l'acquisizione di dati in tempo reale e delle tecnologie dei sensori impiegati a tale scopo offrono la possibilità di raccogliere quantità considerevoli di dati in molte applicazioni sia commerciali che industriali. Per queste ultime, in particolare, l'utilizzo di sistemi di visione artificiale nelle operazioni di produzione sta ridefinendo la natura delle applicazioni di sorveglianza di processo, portando all'adozione di nuove metodologie che consentano un'adeguata gestione ed analisi di *big data*. Un particolare formato di questi ultimi è rappresentato dalle immagini e dai video. In questo contesto i *MVSs*, oltre a rendere di fatto disponibili alle analisi grandi moli di dati, accrescono al contempo la necessità di metodi di controllo statistico dei processi basati sulle immagini (Colosimo e Grasso, 2018). Per l'analisi di dati in questo formato, sono stati proposti in letteratura molteplici approcci, classificabili in diverse categorie a seconda della metodologia impiegata per la gestione di questo particolare formato di dati. Si possono infatti ritrovare carte di controllo per la sorveglianza di immagini univariate, multivariate, basate sui profili oppure, deviando dagli approcci più "standard", spaziali. Infine, esistono le carte di controllo basate sull'analisi multivariata delle immagini (*MIA*), delle procedure che comprendono una fase di riduzione dimensionale seguita da opportuni metodi statistici multivariati per analizzare le immagini che riescono a tener conto dell'alta dimensionalità dei dati trattati. Tale ultima tipologia di approccio, in particolare, è stata oggetto centrale della presente trattazione. Dal momento che un obiettivo comune nella sorveglianza di immagini consiste nel determinare quando e dove un evento fuori controllo abbia avuto origine, sono state selezionate due procedure *MIA* che consentono di assolvere a tale scopo. Più precisamente, le carte di controllo in questione – *ST-PCA* e *SASAM* – costituiscono, nella letteratura dell'*SPC* per immagini, due approcci che migliorano degli algoritmi esistenti. Tale progresso avviene includendo nella modellazione anche l'informazione spaziale intrinseca di ogni *frame*, assente invece in precedenza. Nello specifico, l'*ST-PCA* include l'informazione spa-

ziale nella sorveglianza tenendo conto della distanza tra i pixel di ogni *frame* nella definizione della matrice di varianza e covarianza dei dati analizzati ad ogni istante temporale. Tale matrice viene quindi utilizzata per la stima delle componenti principali pesate sia spazialmente che temporalmente, a loro volta impiegate per ottenere delle statistiche riassuntive circa l'andamento temporale della luminosità di ogni pixel. Viene quindi proposta una regola d'allarme basata sul *clustering* di queste statistiche riassuntive, che sfrutta la conoscenza ingegneristica del processo sorvegliato per poter stabilire se esso sia stabile o meno. La procedura *SASAM*, d'altro canto, sfrutta l'informazione spaziale di ogni *frame* per definire uno schema di campionamento spaziale dei pixel da sorvegliare. Tale schema tenderà a distribuire casualmente le risorse della sorveglianza se il processo è in controllo, mentre porterà a concentrare le risorse di sorveglianza sulle osservazioni ritenute anomale qualora venga rilevata la presenza di un'irregolarità nei dati. Nei Capitoli 3.1 e 3.2 si è notato che, se paragonati ai metodi esistenti, i due approcci considerati costituiscono un miglioramento significativo in termini sia di efficienza che di reattività delle carte stesse.

Nel Capitolo 4.3 sono stati quindi riportati i risultati ottenuti mettendo a confronto queste due procedure, al fine di vedere quali fossero i vantaggi e gli svantaggi di ciascuna, qualora ci si trovi a dover applicare le due carte di controllo per sorvegliare uno stesso *dataset*. Un confronto come quello proposto in questa trattazione infatti non era ancora stato proposto in letteratura e si ritiene che paragonare queste due procedure, ponendole in un *framework* comune, possa evidenziare particolari vantaggi e svantaggi che derivano dallo scegliere un algoritmo piuttosto che un altro qualora ci si ritrovi ad affrontare problemi di sorveglianza come quello proposto. Il confronto consente poi di effettuare uno studio di sensitività dei parametri di regolazione delle carte, analizzando come queste si comportino per determinati valori prefissati dei parametri.

Dalle analisi condotte è emerso che l'*ST-PCA* mostra una reattività minore rispetto al *competitor* e non riesce a rilevare l'anomalia in ogni sce-

nario fuori controllo considerato. Tuttavia, rispetto all'algoritmo *SASAM*, nel caso un allarme venga lanciato la procedura offre un'analisi post-segnale più completa. È infatti possibile stimare sia la dimensione che la posizione dell'*hotspot* rilevato; quest'ultima, in particolar modo, è risultata essere sempre alquanto accurata, sia in termini relativi se paragonati all'altra procedura, sia in termini assoluti in quanto a precisione nella stima fornita. Un ulteriore vantaggio della carta di controllo *ST-PCA* è dato inoltre dalla sua scalabilità: l'approccio *moving window* permette infatti di mantenere costante il costo computazionale necessario per la sorveglianza nel tempo, anche in presenza di lunghe sequenze video. Nondimeno, la procedura risulta essere robusta rispetto alla specificazione di diversi pesi spaziali per poter tenere conto di tale tipo di informazione: al variare della matrice di pesi spaziali impiegata nella definizione della matrice di varianza e covarianza $\boldsymbol{S}$ (Equazione (5)) si ottengono risultati pressoché uguali, con variazioni minime e trascurabili. Un altro vantaggio di questa procedura è da ricercare poi nella particolare regola d'allarme stabilita: essendo basata sul *clustering* delle statistiche riassuntive calcolate ad ogni istante di osservazione, essa fa in modo che per la carta non sia di fatto necessario il calcolo di un limite di controllo, per il quale è solitamente richiesta la disponibilità di grandi moli di dati in controllo offline. La sola conoscenza ingegneristica del processo è infatti sufficiente per definire i parametri che regolano la scelta del numero adeguato di *cluster* e quindi per stabilire lo stato del processo. Tuttavia, in mancanza di tale informazione potrebbe risultare difficile l'impiego della carta di controllo per la sorveglianza; una soluzione potrebbe essere data dalla possibilità di sorvegliare una trasformazione iniziale dei dati – ad esempio i residui di un modello spaziale stimato per ogni *frame* – in modo tale da riuscire a definire nuovamente il numero di gruppi attesi in controllo e fuori controllo. Si è poi notato come, per la definizione del criterio di scelta del corretto numero di *cluster*, il valore K (che stabilisce il numero massimo di gruppi da individuare nei dati) aumenti velocemente all'aumentare della dimensione delle immagini sorvegliate; se da un lato questo comporta un rapido aumento del costo

computazionale richiesto per la sorveglianza, dall'altro può essere facilmente aggirato definendo un nuovo criterio di scelta del numero di *cluster*.

Se l'*ST-PCA*, come si è detto sopra, non riesce sempre ad individuare l'anomalia nei dati, la procedura *SASAM* è stata in grado di identificare in ogni scenario la presenza dell'*hotspot* in modo tempestivo e corretto. Sebbene infatti la stima della posizione dell'anomalia non sia tendenzialmente precisa quanto quella fornita dall'*ST-PCA*, essa non risulta essere così distorta da poter essere considerata erronea o fuorviante nell'analisi post-segnale. Inoltre, l'assenza, in questo caso, di una stima della dimensione (in numero di pixel) dell'anomalia è compensata dalla reattività della carta: le performance di questo secondo algoritmo sono consistenti e costanti in qualsiasi scenario considerato, riuscendo a rilevare anche gli eventi fuori controllo di breve durata che l'*ST-PCA* non riusciva a cogliere. Anche in questo caso, comunque, il costo computazionale della procedura riesce a mantenersi costante nel tempo. A differenza poi dell'*ST-PCA*, la carta di controllo *SASAM* richiede il calcolo di un limite di controllo per stabilire la regola d'allarme: in questo contesto, dunque, si rende necessaria la disponibilità di sequenze video in controllo salvate offline precedentemente all'inizio della sorveglianza, di modo che tale valore possa essere calcolato tramite simulazione Monte Carlo o *bootstrap*. Un'ultima dissimilarità rispetto al *competitor* consiste nella richiesta distributiva dei dati: mentre l'*ST-PCA* non postula alcuna distribuzione per l'implementazione della sorveglianza, la carta di controllo *SASAM* richiede che i dati si distribuiscano normalmente come definito in Equazione (12). Questo può essere ottenuto agevolmente considerando, per ogni immagine analizzata, i residui di un modello adeguato, anche se la scelta opportuna di quest'ultimo può diventare più complessa nel caso in cui si operi con sequenze video caratterizzate da una particolare struttura di correlazione spazio-temporale.

Appare quindi forzato stabilire quale carta di controllo debba essere preferita in termini assoluti all'altra. Da un lato, l'*ST-PCA* consente di sfruttare le conoscenze ingegneristiche del processo per la definizione di una regola

d'allarme che non richiede il calcolo dei limiti di controllo, senza necessitare peraltro di dati offline in controllo per avviare la procedura con dei parametri precedentemente stimati in modo consistente. Tale algoritmo rende possibile inoltre un'analisi post-segnale relativamente più esaustiva, fornendo non solo una stima precisa della posizione dell'anomalia rilevata, ma anche una stima della sua dimensione. Dall'altro lato la carta di controllo *SASAM* mostra una maggiore reattività, riuscendo sempre ad identificare l'anomalia indipendentemente dalla sua durata e dalla dimensione sia dei dati che dell'*hotspot* stesso. Essa tuttavia non fornisce un'analisi post-segnale tanto completa quanto quella dell'*ST-PCA*. A seconda dunque di quali aspetti siano ritenuti di maggiore interesse per l'analisi e a seconda delle condizioni nelle quali si opera la sorveglianza, andrà preferita una procedura piuttosto che l'altra.

A Codice R

Per l'implementazione delle seguenti funzioni sono stati utilizzati i pacchetti `av` (Ooms, 2022), `imager` (Barthelme, 2021), `jpeg` (Urbanek, 2021) e `flexmix` (Grün e Leisch, 2007, 2008; Leisch, 2004). È possibile inoltre ottenere dei grafici tridimensionali delle immagini analizzate servendosi delle funzioni presenti nel pacchetto `plot3D` (Soetaert, 2021).

A.1 Funzioni generali

```
## Estrarre i frames dai video e salvarli
frames_from_video <- function(video, destdir=tempfile(),
                              format="jpg", fps=NULL){
  #Modifica della funzione "av_video_images" per gestire
  #meglio i nomi delle immagini
  require(av)
  stopifnot(length(video) == 1)
  vfilter <- ifelse(length(fps), paste0("fps=fps=", fps), "null")
  framerate <- av_media_info(video)$video$framerate
  dir.create(destdir)
  codec <- switch(format, jpeg="mjpeg", jpg="mjpeg", format)
  output <- file.path(destdir, paste0("image_%d.", format))
  av_encode_video(input=video, output=output, framerate=framerate,
                  codec=codec, vfilter=vfilter, verbose=F)
  list.files(destdir, pattern=paste0("image_\\d.",
                                     format), full.names=TRUE)
}

## Salvare i frames di un video nel formato '.jpeg' o '.png' in
## una cartella
save_frames <- function(file.video, format="jpeg"){
  stopifnot(format=="jpeg" | format=="png")
  cat(paste0("Saving the frames of \'",file.video,
             "\' video in a folder using \'",format,"\' format\n"))
  frames_from_video(file.video, format=format,
                    fps=av_media_info(file.video)$video$framerate,
                    destdir=paste0(dirname,"/",
```

```
                                        file.video,"_",format))
}

## Importare i frames di un video salvandoli in un array (da file
## '.jpeg')
# Ottengo un array con 3 dimensioni:
#    (altezza) x (largezza) x nframes
#           M  x         N  x nframes
image_array_general<-function(file_video, M, N){
  # M: numero di righe dell'immagine     (ALTEZZA)
  # N: numero di colonne dell'immagine   (LARGEZZA)
  require(av); require(jpeg); require(imager)

  ## Importo i frames
  #numero di frames totali
  nframes <- av_media_info(file_video)$video$frames
  #array vuoto da riempire
  images <- array(NA,dim=c(M, N, 3, nframes))
  for(i in 1:nframes)
    images[,,,i] <-
    readJPEG(source=paste0(dirname,"/", file_video,
                           "_jpeg/image_",i,".jpeg"), native=F)

  ## Sistemo l'array
  #Tengo un solo canale (date che uso immagini in scala di grigi)
  #imager::renorm => per riscalare correttamente i valori
  #                  in [0,255]
  images <- imager::renorm(images[,,1,], min=0, max=255)
  images_correct <- array(NA, dim=c(M, N, nframes))
  for(i in 1:nframes) images_correct[,,i] <- (images[M:1, 1:N, i])
  images <- images_correct; rm(images_correct)
  #images[,,i] => i-esima immagine

  cat(paste0("Created array with the frames from the \'",
             file_video,"\' video\n"))

  return(images)
}
```

```
66
67 ## Distanza euclidea tra punti x e y di coordinate (x[1],x[2]),
68 ## (y[1],y[2])
69 dist_xy <- function(x,y){
70   as.numeric(((x[1]-y[1])^2+(x[2]-y[2])^2)^.5)
71 }
72
73 ## Grafico delle immagini, con una versione "adeguata" di image()
74 plot_img <- function(data, M, N, ...){
75   # data   vettore o matrice che possono essere visti come
76   #        un'immagine MxN
77
78   require(colorspace)
79   #utile per le palette di colori che possono essere specificate
80   #in '...'
81
82   data<-matrix(data,ncol=N,nrow=M)
83   image(1:N, 1:M, t(data), ...)
84 }
85
86 ## Grafico di un'area dell'immagine
87 plot_selection <- function(data, M, N, x=c(1,N), y=c(1,M),
88                            original=F, ...){
89   # data  matrice o vettore con l'immagine M x N originale
90   # M     numero di righe dell'immagine     (ALTEZZA)
91   # N     numero di colonne dell'immagine   (LARGEZZA)
92   # x     coordinate degli estremi (orizzontali) della selezione
93   # y     coordinate degli estremi (verticali) della selezione
94
95   data<-matrix(data, nrow=M, ncol=N)
96
97   if(original){
98     cat('original=TRUE: plotting the original image too')
99     par(mfrow=c(1,2))
100    image(1:N,1:M, t(data), ...)
101    segments(x[1], y[1], x[1], y[2], col='green')
102    segments(x[1], y[2], x[2], y[2], col='green')
103    segments(x[2], y[2], x[2], y[1], col='green')
```

```
104     segments(x[2], y[1], x[1], y[1], col='green')
105   }
106
107   image(x[1]:x[2], y[1]:y[2],
108         t(data[y[1]:y[2], x[1]:x[2]]),
109         xlab=paste0(x[1],':',x[2]), ylab=paste0(y[1],':',y[2]),...)
110
111   par(mfrow=c(1,1))
112 }
113
114 ## Aggiungere punti all'immagine disegnata con 'plot_img'
115 add_points <- function(coord, M, N, ...){
116   # coord  matrice di coordinate dei punti da inserire nel grafico:
117   #        - prima colonna = coordinate di RIGA
118   #        - seconda colonna = coordinate di COLONNA
119   # M      numero di righe dell'immagine     (ALTEZZA)
120   # N      numero di colonne dell'immagine   (LARGEZZA)
121   if(is.null(coord)) return()
122   coord<-matrix(coord,ncol=2)
123   points(coord[,2], coord[,1], ...)
124 }
125
126 ## Salvare le coordinate dei pixel di un'immagine
127 coord.fun <- function(immagine){
128   cbind(riga=rep(1:nrow(immagine),ncol(immagine)),
129         colonna=sort(rep(1:ncol(immagine),nrow(immagine))))
130   #riga => indici di riga
131   #colonna => indici di colonna
132 }
133
134 ## Funzione per ottenere W0.mat, la matrice delle distanze
135 ## "standard" tra i pixel
136 dist.mat.fun <- function(coord){
137   # coord  matrice di coordinate dei punti da inserire nel grafico:
138   #        - prima colonna = coordinate di RIGA
139   #        - seconda colonna = coordinate di COLONNA
140
141   w <- matrix(0,ncol=nrow(coord),nrow=nrow(coord))
```

```
142
143   cat(paste0("Beginning the calculation of ",
144              sum(1:(nrow(coord)-1)),
145              " distances (this could take a while)\n",
146              "Cycle ends after ",(nrow(coord)-1)," iterations\n"))
147   for(i in 1:(nrow(coord)-1)){
148     for(j in (i+1):nrow(coord)){
149       w[i,j] <- w[j,i] <-
150         ((coord[i,1]-coord[j,1])^2+(coord[i,2]-coord[j,2])^2)^.5
151     }
152     if((i%%100)==0)cat(paste0(i,"\t"))
153   }
154   cat("done\n")
155   return(w)
156 }
157
158 ## Aggiornamento della matrice dei dati: aggiungere colonne di
159 ## dati in versione unfolded
160 newimage.mw.fun <- function(data, newdata, L){
161   # data     matrice dei dati nella versione "unfolded" (p x J)
162   # newdata  la nuova immagine che viene acquisita,
163   #          matrice di dimensione (M x N)
164
165   temp <- cbind(data,as.vector(newdata))
166   #Finche' non ho L colonne aggiungo e basta
167   # => ritorno la matrice che ottengo aggiungendo colonne
168   if(is.vector(data) | ncol(temp)<=L ) return(temp)
169   #Se ho piu' di L colonne => ritorno solo le ultime L colonne
170   else return(temp[,(ncol(temp)-L+1):ncol(temp)])
171 }
```

A.2 Simulazione di immagini e video

```
1 ## Funzione per creare un hotspot nel tempo
2 hotspot.fun <- function(durata){
3   #funzione per inserire un hotspot in una successione pixel
4   # durata = numero di frames successivi in cui inserire l'hotspot
```

```
  t <- 1:durata
  255/(1+exp(.2*(t-.95*durata)))
}

## Funzione per inserire un hotspot in un array di immagini
hotspot.injection <- function(data, m, n, raggio, tau, durata,
                              shape=''){
  #Funzione per inserire un fenomeno hotspot centrato in un pixel
  #in posizione (m,n) di forma 'shape' e dimensione regolata da
  #'raggio'
  #
  # data    array (M x N x nframes) di immagini nei quali inserire
  #         l'hotspot
  #
  # m,n     coordinate del pixel centrale dell'hotspot
  #
  # raggio  dimensione anomalia
  #
  # tau     frame dal quale inizia l'anomalia
  #
  # durata  numero di frames successivi in cui inserire l'hotspot
  #
  # shape   'square' o 'cross' o 'bigcross', determina la forma
  #         dell'hotspot

  #'SQUARE-SHAPED' HOTSPOT
  if(shape=='square'){
    for(i in (m-raggio):(m+raggio)){
      for(j in (n-raggio):(n+raggio)){
        data[i, j, tau:(tau+durata-1)] <- hotspot.fun(durata)
      }
    }
    return(data)
  }

  #'CROSS-SHAPED' HOTSPOT
  if(shape=='cross'){
    # m.coord<-((m-raggio):(m+raggio))[-(raggio+1)]
```

```
43    # n.coord<-((n-raggio):(n+raggio))[-(raggio+1)]
44    mn.coord<-cbind(c(m, rep(m,2*raggio),
45                      ((m-raggio):(m+raggio))[-(raggio+1)]),
46                    c(n, ((n-raggio):(n+raggio))[-(raggio+1)],
47                      rep(n, 2*raggio)))
48
49    for(i in 1:nrow(mn.coord)){
50      data[mn.coord[i,1], mn.coord[i,2], tau:(tau+durata-1)] <-
51        hotspot.fun(durata)
52    }
53    return(data)
54  }
55
56  #'BIG-CROSS-SHAPED' HOTSPOT
57  if(shape=='bigcross'){
58    #Se shape='bigcross' ma raggio=1, richiamo la funzione e
59    #uso shape='cross'
60    if(raggio<2){
61      cat(paste("'raggio' is too small for a 'bigcross'-shaped",
62                "hotspot: using shape = 'cross'\n"))
63      data <-
64        hotspot.injection(data, m, n, raggio, tau, durata,
65                          shape='cross')
66    }
67
68    else{
69      mn.coord<-cbind(c(rep(m-1, 2*raggio+1),
70                        rep(m,   2*raggio+1),
71                        rep(m+1, 2*raggio+1),
72                        rep((m-2):(m-raggio),3),
73                        rep((m+2):(m+raggio),3)),
74                      c(rep((n-raggio):(n+raggio),3),
75                        rep(n-1, raggio-1),
76                        rep(n,   raggio-1),
77                        rep(n+1, raggio-1),
78                        rep(n-1, raggio-1),
79                        rep(n,   raggio-1),
80                        rep(n+1, raggio-1)) )
```

```
81
82        for(i in 1:nrow(mn.coord)){
83          data[mn.coord[i,1], mn.coord[i,2], tau:(tau+durata-1)] <-
84            hotspot.fun(durata)
85        }
86      }
87      return(data)
88    }
89
90    #'SINGLE-PIXEL' HOTSPOT
91    else{
92      cat(paste("'shape' does not have one of the default values:\n",
93                " => returning a single-pixel hotspot in data[",
94                m,", ",n,", ",tau,":",(tau+durata-1),"]"))
95      data[m, n,tau:(tau+durata-1)] <- hotspot.fun(durata)
96      return(data)
97    }
98  }
99
100 ## Coordinate per un percorso di un "evento" nelle immagini
101 coord.percorso.fun <- function(x.percorso=8, y.percorso=5, M, N){
102   # M            altezza dell'immagine (numero di righe)
103   # N            larghezza dell'immagine (numero di colonne)
104   # x.percorso   larghezza del percorso dell'"evento"
105   # y.percorso   distanza dai margini dell'"evento"
106
107   cat(paste0('Creating a path for a ',M,' x ',N,
108              ' image, output will be path coordinates'))
109
110   #Coordinate del percorso
111   x <- x.percorso; y <- y.percorso
112   coord.percorso<-c()
113   for(i in (M-y-1):(1+y)){
114     coord.percorso <- rbind(coord.percorso,
115                             cbind(rep(i,x), (1+y):(x+y))) }
116   for(i in (y+x+1):(N-y-x-1)){
117     coord.percorso <- rbind(coord.percorso,
118                             cbind((1+y):(x+y), rep(i,x))) }
```

```
119   for(i in (1+y):(M-y-1)){
120     coord.percorso <- rbind(coord.percorso,
121                             cbind(rep(i,x), (N-y-x):(N-y-1))) }
122   for(i in (N-y-x-1):(3*y+3*x+1)){
123     coord.percorso <- rbind(coord.percorso,
124                             cbind((M-y-1):(M-y-x), rep(i,x))) }
125   for(i in (M-y-1):(M-2*y-2*x)){
126     coord.percorso <- rbind(coord.percorso,
127                             cbind(rep(i,x),
128                                   (3*y+2*x+1):(3*y+3*x))) }
129   for(i in (3*y+3*x+1):(N-2*y-2*x-1)){
130     coord.percorso <- rbind(coord.percorso,
131                             cbind((M-2*y-x-1):(M-2*y-2*x),
132                                   rep(i,x))) }
133   for(i in (M-2*y-x-1):(2*y+x+1)){
134     coord.percorso <- rbind(coord.percorso,
135                             cbind(rep(i,x),
136                                   (N-2*y-2*x):(N-2*y-x-1))) }
137   for(i in (N-2*y-2*x-1):(2*y+2*x+1)){
138     coord.percorso <- rbind(coord.percorso,
139                             cbind((2*y+x+1):(2*y+2*x),
140                                   rep(i,x))) }
141   for(i in (2*y+x+1):(M-y-1)){
142     coord.percorso <- rbind(coord.percorso,
143                             cbind(rep(i,x),
144                                   (2*y+x+1):(2*y+2*x))) }
145   for(i in (2*y+x):(y+x+1)){
146     coord.percorso <- rbind(coord.percorso,
147                             cbind((M-y-1):(M-y-x),
148                                   rep(i,x))) }
149
150   colnames(coord.percorso) <- c('riga','colonna')
151
152   return(coord.percorso)
153 }
154
155 ## Generazione di frames mistura di Normali
156 frames.genaration.fun <- function(nframes, M, N,
```

```
157                                   OC=FALSE, m=M/2, n=N/2,
158                                   raggio=1, tau=nframes/2,
159                                   durata=1, shape='',
160                                   mu=c(80,200), sig=c(10,25),
161                                   prop=c(.75, .25),
162                                   x.percorso=10, y.percorso=3,
163                                   saveJPEG=FALSE,
164                                   destdir=tempfile(),
165                                   seed=888){
166
167   # Genero un array di 'nframes' immagini di dimensione M x N
168   # (M righe, N colonne)
169   #
170   # Decido se tutte le immagini devono essere IC o se inserire
171   # un'anomalia dal frame 'tau' di durata 'durata', in posizione
172   # (m,n), di dimensione regolata da 'raggio' e di forma 'shape'
173   # ('square', 'cross', 'bigcross')
174
175   # nframes                           numero di frames da generare
176   #
177   # M, N                              dimensione dei frames
178   #
179   # OC                                generare anomalia?
180   #
181   # m, n, raggio, tau, durata, shape  stessi parametri di
182   #                                   hotspot.injection()
183   #
184   # mu, sig, prop                     parametri distributivi per
185   #                                   la generazione
186   #
187   # x.percorso, y.percorso            stessi parametri di
188   #                                   coord.percorso.fun()
189   #
190   # saveJPEG                          salvare i frames generati in
191   #                                   file '.jpeg'?
192   #
193   # destdir=tempfile()                directory in cui salvare
194   #                                   le immagini
```

```
195
196 require(imager); require(jpeg)
197
198 cat(paste0("Beginning the generation of ",nframes,
199            " images each measuring M x N = ",M," x ",N,
200            " pixels:\n"))
201
202 #Preparazione dell'array vuoto per inserire i frames
203 out <- array(NA, dim=c(M, N, nframes))
204
205 #Dimensione dell'evento coordinate del percorso
206 cat(' - ')
207 coord.percorso <- coord.percorso.fun(x.percorso, y.percorso, M,N)
208 cat(': done\n')
209
210 #Dimensioni di evento e percorso
211 dim.evento <- M * N * prop[2]
212 dim.percorso <- nrow(coord.percorso)
213
214 #Creo una matrice di coordinate d'"appoggio" del percorso
215 coord.temp <- coord.percorso
216
217 #Genero i frames
218 set.seed(seed)
219 cat(paste0(' - Creating an array of ',nframes,
220            ' (M x N = ',M,' x ',N,') images'))
221 for(i in 1:nframes){
222   #Genero background
223   temp <- matrix(rnorm(M*N, mu[1], sig[1]), ncol=N, nrow=M)
224
225   #Creazione di una matrice adeguata che descriva che percorso
226   #deve seguire l'evento
227   coord.temp <- rbind(coord.temp,coord.percorso)
228   coord.evento <- coord.temp[1:dim.evento,]
229
230   #Inserimento dell'evento alle coordinate opportune appena
231   #salvate
232   temp[coord.evento] <- rnorm(dim.evento, mu[2], sig[2])
```

```
233
234     #Salvataggo dell'i-esimo frame
235     out[,,i]<-temp
236
237     #Aggiornamento della matrice di coordinate d'"appoggio"
238     coord.temp<-coord.temp[-(1:(dim.evento/8+1)),]
239   }
240   cat(': done\n')
241
242   #Se OC=TRUE si inserisce un'anomalia nei dati
243   if(OC){
244     cat(paste0("Injecting hotspot in data[",
245                m,", ",n,", ",tau,":",(tau+durata-1),"]\n"))
246     out <- hotspot.injection(out, m, n, raggio, tau, durata, shape)
247   }
248
249   #Se saveJPEG = TRUE si salvano le immagini in una cartella nella
250   #directory specificata
251   if(saveJPEG){
252     dir.create(destdir)
253     for(i in 1:nframes){
254       out.cimg <- imager::as.cimg( t(out[M:1, , i]) )
255       imager::save.image(im = out.cimg, quality = 1,
256                          file = paste0(destdir,
257                                        "/image_",i,".jpeg"))
258     }
259     cat(paste0('Images saved in ', destdir,'\n'))
260   }
261
262   return(out)
263 }
264
265 ## Generazione di un video a partire da un array contenente
266 ## i frames
267 save.video.fun <- function(data, destdir=tempfile()){
268   # data      array (M x N x nframes) contenente i frames da cui
269   #           generare il video
270   #
```

```
271   # destdir  directory in cui salvare il file video generato
272   #          (contiene il nome del file)
273
274   require(imager)
275
276   if(!is.array(data)){
277     warning(paste('data must be a (M x N x nframes) array',
278                   'containing the frames that will be used to',
279                   'generate the video, returning NULL'))
280     return(NULL)
281   }
282
283   M <- dim(data)[1]  #altezza dei frames (numero di righe)
284   N <- dim(data)[2]  #larghezza dei frames (numero di colonne)
285   nframes <- dim(data)[3] #numero totale di frames
286
287   data.cimg <- array(NA,dim=c(N, M, nframes))
288   for(i in 1:nframes){
289     data.cimg[,,i] <- t(data[M:1,,i])
290   }
291   data.cimg <- as.cimg(data.cimg[,,1:nframes])
292
293   imager::save.video(im=data.cimg, fname=destdir)
294
295   cat(paste0('A file video has been saved in the directory ',
296              destdir,'\n'))
297 }
298
299 ## Modellazione un vettore di dati proveniente da una mistura di
300 ## due Normali al fine di ottenerne i residui
301 get.residual.gmm2 <- function(data, seed=888){
302   #Modello per un vettore di dati proveniente da una mistura di
303   #due Normali con un modello mistura gaussiano (GMM)
304
305   # data  vettore di dati proveniente da una mistura di due Normali
306
307   if(!is.vector(data)){
308     warning('data must be a vector, returning NULL')
```

```
309     return(NULL)
310   }
311
312   require(flexmix)
313
314   set.seed(seed)
315
316   #si stimano due modelli (e' rapido), perche' a volte la stima puo'
317   #non essere corretta
318   fit.flex1 <- flexmix(data~1, k=2)
319   fit.flex2 <- flexmix(data~1, k=2)
320
321   if(fit.flex1@logLik > fit.flex2@logLik) fit.flex <- fit.flex1
322   else fit.flex <- fit.flex2
323
324   if(!fit.flex@converged){
325     warning('EM algorithm did not converged, returning NULL')
326     return(NULL)
327   }
328
329   res <- id <- rep(NA,length(data))
330
331   res[fit.flex@cluster==1] <- data[fit.flex@cluster==1] /
332     sd(data[fit.flex@cluster==1]) -
333     fit.flex@components$Comp.1[[1]]@parameters$coef /
334     fit.flex@components$Comp.1[[1]]@parameters$sigma
335
336   res[fit.flex@cluster==2] <- data[fit.flex@cluster==2] /
337     sd(data[fit.flex@cluster==2]) -
338     fit.flex@components$Comp.2[[1]]@parameters$coef /
339     fit.flex@components$Comp.2[[1]]@parameters$sigma
340
341   id[fit.flex@cluster==1] <- 1
342   id[fit.flex@cluster==2] <- 2
343
344   out <- list(
345     res=res, id=id,
346     size=c(as.integer(fit.flex@size[1]),
```

```
            as.integer(fit.flex@size[2])),
    mean.est=c(
      as.numeric(fit.flex@components$Comp.1[[1]]@parameters$coef),
      as.numeric(fit.flex@components$Comp.2[[1]]@parameters$coef)),
    sd.est=c(
      as.numeric(fit.flex@components$Comp.1[[1]]@parameters$sigma),
      as.numeric(fit.flex@components$Comp.2[[1]]@parameters$sigma))
  )

  # res        residui (standardizzati) del GMM
  #
  # id         indici della distribuzione generatrice stimata
  #            (1 o 2) dei residui
  #
  # size       stima del GMM della dimensione dei due gruppi
  #
  # mean.est   stima del GMM delle medie
  #
  # sd.est     stima del GMM della dev.std

  return(out)
}
```

A.3 Codice *ST-PCA*

```
## W1, W2 e W3 in funzione della matrice delle distenze euclidee
## tra pixel
w.fun <- function(dist.mat, mode=0, SOGLIA_r=5, scaled=TRUE){
  # dist.mat   matrice delle distanze tra gli elementi
  # mode       che peso usare
  # SOGLIA_r   valore della soglia 'r' da usare in W2 o W3
  # scaled     gli elementi delle matrici calcolate devono sommare
  #            ad 1?

  if(!(mode %in% 1:3)){
    warning(paste("'mode' does not have a fixed value (1,2 or 3).",
                  "The output will be the matrix provided",
```

```
                  "in input"))
    return(dist.mat)
  }

  if(mode==1){
    cat("Beginning the calculation of W1... ")
    temp <- 1/(dist.mat^2)
    diag(temp) <- 0
    if(scaled){ cat("scaling... "); temp<-temp/sum(temp) }
    cat("done\n")
    return(temp)
  }

  if(mode==2){
    cat("Beginning the calculation of W2... ")
    temp <- matrix(0,ncol=ncol(dist.mat),nrow=nrow(dist.mat))
    temp[dist.mat <= SOGLIA_r] <- 1
    diag(temp) <- 0
    if(scaled){ cat("scaling... "); temp<-temp/sum(temp) }
    cat("done\n")
    return(temp)
  }

  if(mode==3){
    cat("Beginning the calculation of W3... ")
    temp <- matrix(0,ncol=ncol(dist.mat),nrow=nrow(dist.mat))
    temp[dist.mat <= SOGLIA_r] <-
      (1 - dist.mat[dist.mat<=SOGLIA_r]/SOGLIA_r^2)^2
    diag(temp) <- 0
    if(scaled){ cat("scaling... "); temp<-temp/sum(temp) }
    cat("done\n")
    return(temp)
  }
}

## S - matrice di varianza-covarianza pesata (spazialmente)
s.varcov.fun <- function(data,
                         W.mat=diag(1, ncol=nrow(data),
```

```
51                                      nrow=nrow(data))){
52   # data   matrice dei dati nella versione unfolded T-mode (p x J)
53   # W.mat  matrice di pesi spaziali (p x p) (default: Identita')
54
55   #Output: stima della matrice di varianza-covarianza campionaria
56   #pesata (J x J)
57
58   temp <- ((nrow(data) - 1)^(-1)) *
59     t(apply(data, 2, function(x){ x - mean(x) })) %*%
60     W.mat %*%
61     (apply(data, 2, function(x){ x - mean(x) }))
62
63   return(temp)
64 }
65
66 ## Implementazione della carta di controllo ST-PCA
67 STPCA.fun <- function(file.video=NULL, data=NULL, weight.type=0,
68                       W0.mat=NULL, SOGLIA_r=5,
69                       moving.window=TRUE, L=50, SOGLIA_var_pc=.8,
70                       Kmax=8, plots=TRUE, N_FRAME_START=1,
71                       STOP_FRAME=456, seed=888){
72
73   require(av)
74
75   #Leggo un batch iniziale di N_FRAME_START frames e poi inizio
76   #a sorvegliare. La sorveglianza inizia dal frame
77   #(N_FRAME_START+1)-esimo
78
79   ## PARAMETRI GENERALI in input alla funzione:
80   # file.video          nome del video da sorvegliare
81   #
82   # data                array (M x N x nframes) contenente i
83   #                     frames da sorvegliare
84   #
85   # weight.type         scelta della matrice di peso spaziale,
86   #                     puo' essere 0,1,2,3
87   #  weight.type=0 => W = identita', nessun peso
88   #  weight.type=1 => W = W1
```

```
89  #  weight.type=2 => W = W2
90  #  weight.type=3 => W = W3
91  #
92  # W0.mat              matrice della distanza tra pixel, puo'
93  #                     essere passata in input per risparmiare
94  #                     (molto) tempo
95  #
96  # SOGLIA_r            influenza la definizione delle matrici di
97  #                     pesi spaziali W2 e W3
98  #
99  # moving.window       uso l'approccio moving window se TRUE,
100 #                     altrimenti ricorsivo
101 #
102 # L                   ampiezza della finestra mobile
103 #
104 # SOGLIA_var_pc       soglia per la varianza spiegata dalle
105 #                     componenti principali che si tengono
106 #
107 # Kmax                numero di cluster da trovare in T2.stat
108 #                     (al massimo sono 3, se ne fanno di piu' per
109 #                     l'indice utilizzato per trovare k ottimale)
110 #
111 # plots               mostrare in output dei grafici riassuntivi?
112 #
113 # STOP_FRAME          frame al quale l'analisi si ferma se non e'
114 #                     stato lanciato un allarme (serve solo per
115 #                     provare la funzione o se si sa dove si
116 #                     dovrebbe lanciare l'allarme)
117 #
118 # seed                per ottenere replicare gli stessi risultati
119
120 ## INIZIALIZZAZIONE
121 if(is.null(file.video) & is.null(data)){
122   warning(paste("(STCPA): Neither \'file.video\' nor \'data\'",
123                 " was given in input, returning NULL\n"))
124   return(NULL)
125 }
126
```

```
127  #Se ho un array (M x N x nframes):
128  if(!is.null(data)){
129    nframes <- dim(data)[3]    #numero di frames
130    M <- dim(data)[1]          #altezza, numero di RIGHE
131    N <- dim(data)[2]          #larghezza, numero di COLONNE
132  }
133
134  #Se ho il nome di un file in ingresso:
135  if(!is.null(file.video)){
136    #numero di frames
137    nframes <- av_media_info(file.video)$video$frames
138    #altezza, numero di RIGHE
139    M <- av_media_info(file.video)$video$height
140    #larghezza, numero di COLONNE
141    N <- av_media_info(file.video)$video$width
142
143    cat(paste("\n(STCPA): Creating an array with data from the",
144              "input video file "))
145    #Salvo i frames del video in un array:
146    # - leggendo i file jpeg dei frames che avevo gia' salvato
147    #   in precedenza:
148    if(dir.exists(paste0(dirname,"/",file.video,"_jpeg"))){
149      cat("using existing \'jpeg\' files\n")
150      data <- image_array_general(file.video, M=M, N=N)
151    }
152
153    # - creando i file jpeg dei frames del video
154    else{
155      cat("saving \'jpeg\' files\n")
156      save_frames(file.video, format='jpeg')
157      data <- image_array_general(file.video, M=M, N=N)
158    }
159    cat("(STCPA): done\n")
160  }
161
162  cat("(STCPA): ACTIVE PARAMETERS:\n")
163  print(data.frame(PARAMETER = c("file.video", "L",
164                                 "Monitoring starts at frame",
```

```
                                    "M", "N", "nframes",
                                    "W0.mat in input?",
                                    "weight.type","SOGLIA_r",
                                    "SOGLIA_var_pc"),
                    VALUE = c(ifelse(is.null(file.video),"NULL",
                                     file.video),
                              L, N_FRAME_START,
                              M, N, nframes,
                              !is.null(W0.mat),
                              weight.type, SOGLIA_r,
                              SOGLIA_var_pc)))

frame.index<-1  #indice del frame analizzato
#dati in versione 'unfolded'
dati.unfolded <- as.vector(data[,,frame.index])

#Coordinate dei pixel dei frame, di dimensione (M*N x 2)=(p x 2)
coord.dati<-coord.fun(data[,,frame.index])

cat(paste0("\n(STCPA): Reading the first ",N_FRAME_START,
           " frames\n"))
#Leggo i primi N_FRAME_START frames
for(frame.index in 2:N_FRAME_START){
  dati.unfolded<-newimage.mw.fun(dati.unfolded,
                                 newdata=data[,,frame.index],
                                 L=N_FRAME_START) }

#Operazione che richiede (molto) tempo: calcolo le distanze tra
#i pixel. Ottengo una matrice (M*N x M*N) = (p x p)
if(is.null(W0.mat)){
  cat(paste("\n(STCPA): Creating W0.mat (matrix of distances",
            "between pixels in each frame); it is the most",
            "time-consuming process\n"))
  W0.mat<-dist.mat.fun(coord.dati)
}

#Visualizzo a schermo che W0.mat e' stata fornita in input
if(!is.null(W0.mat)){
```

```
203     cat(paste("\n(STCPA): W0.mat (matrix of distances between",
204               "pixels in each frame) given in input\n"))
205   }
206
207   cat(paste0("\n(STCPA): Creating W.mat (spatial weight matrix), ",
208              "weight.type=",weight.type,"\n"))
209   W.mat <- w.fun(W0.mat, mode=weight.type, SOGLIA_r=SOGLIA_r,
210                  scaled=TRUE)
211
212   #Inizializzo dei vettori in cui salvare i vari valori dell'output
213   mpc.list <- c()            #numero di pc conservate ad ogni passo
214   ssw.list <- c()            #le ssw del clustering ad ogni istante
215   T2.list <- c()             #le T2.stat ai vari istanti
216   index <- c()               #per avere riferimenti esatti ai frames
217   Dk.list <- c()             #valori della statistica Dk
218   dati.clust.k <- c()        #appartenenza al gruppo per k scelto
219
220   ## SORVEGLIANZA: FRAMES SUCCESSIVI
221   #Ripeto questa procedura finche' non lancio un allarme o
222   #finche' non finisco i frames
223   k<-2
224   while(k == 2  &  frame.index < nframes){
225     #Finche' i cluster che individuo sono 2 (processo IC) acquisisco
226     #un nuovo frame e procedo
227
228     #Aumento l'indice del frame a cui sono arrivato
229     frame.index <- frame.index + 1
230     #Leggo i nuovi dati
231     dati.unfolded <- cbind(dati.unfolded,
232                            as.vector(data[,,frame.index]))
233
234     cat(paste0("(STCPA): Frame ",frame.index,": "))
235
236     ## APPROCCIO RICORSIVO: se moving.window = FALSE si aggiungono
237     #   frames senza toglierne, non si deve fare altro in tal caso
238
239     ## APPROCCIO MOVING WINDOW: finche' non ho L colonne aggiungo e
240     #  basta. Se ho piu' di L colonne => considero solo le ultime L
```

```
241
242     if(moving.window & ncol(dati.unfolded)>L)
243       dati.unfolded <-
244       dati.unfolded[,(ncol(dati.unfolded)-L+1):ncol(dati.unfolded)]
245
246     dati.unfolded <- scale(dati.unfolded)
247
248     #Matrice di varianza-covarianza pesata,
249     #dimensioni (L x L) o (frame.index x frame.index)
250     cat("S")
251     S.dati.unf <- s.varcov.fun(dati.unfolded, W.mat)
252     cat("... ")
253
254     #Matrice diagonale con gli autovalori di S, stesse dimensioni
255     #di S
256     Lambda <- diag(eigen(S.dati.unf)$value)
257
258     #Matrice dei LOADINGS, gli autovettori, stesse dimensioni di S
259     V <- eigen(S.dati.unf)$vectors
260
261     #Matrice con le ST-PCA, di dimensioni (M*N x L) = (p x L) o
262     # (M*N x frame.index) = (p x frame.index)
263     Z <- dati.unfolded %*% V
264
265     #Percentuale di varianza spiegata dalle componenti principali
266     perc.var.pc <- diag(Lambda) / sum(diag(Lambda))
267     #Numero di componenti principali da conservare
268     m_pc <- min(which(cumsum(perc.var.pc) >= SOGLIA_var_pc))
269
270     #Statistica di controllo: T2(m,n) = T2_i
271     T2.stat <-
272       apply(Z, 1, function(z){
273         sum( z[1:m_pc]^2 / diag(Lambda)[1:m_pc] ) })
274
275     ## K-MEANS CLUSTERING di T2.stat
276     set.seed(seed)
277     cat("kmeans")
278     gruppi <- sapply(1:Kmax, function(x){
```

```
279       kmeans(T2.stat, centers=x, nstart=50,
280              algorithm='MacQueen', iter.max=5000)})
281
282     if(any(gruppi[,1:Kmax]$ifault) == 4){
283       cat('ALGORITHM DID NOT CONVERGE, interrupting execution')
284       x <- list(alarm.time=NA, m_pc=mpc.list, ssw_k=ssw.list,
285                 T2.stat=T2.list, frame=index, Dk=Dk.list,
286                 dati.clust.k=dati.clust.k)
287       return(x)
288     }
289
290     ssw_k <- sapply(1:Kmax, function(x){
291       (gruppi[,x]$tot.withinss)/x })
292
293     # Aggiorno k: k=argmax(D_k)
294     Dk <- ((ssw_k[Kmax] - ssw_k[1]) / (Kmax - 1)) * (1:Kmax) +
295       (ssw_k[1] * Kmax - ssw_k[Kmax]) / (Kmax - 1) - ssw_k
296     k <- which.max(Dk)
297     cat("... ")
298
299     ## GRAFICI
300     if(plots){
301       cat("plotting... ")
302       par(mfrow=c(2,2))
303       plot_img(data[,,frame.index], col=hcl.colors(256, 'Grays'),
304                M=M, N=N, xlab='X (pixels)', ylab='Y (pixels)',
305                cex.main=.9, main=paste('Frame',frame.index))
306
307       plot_img(T2.stat, main=paste("T2.stat, frame", frame.index),
308                col=hcl.colors(256, 'Grays'), M=M, N=N)
309
310       plot(Dk, type='b', pch=20, col=4, main='D(k)',
311            xlab='k', ylab='D(k)')
312       abline(v=k, col=2, lty='dashed')
313
314       plot(ssw_k, type='b', pch=20, col=4, main='SSW(k)',
315            xlab='k', ylab='SSW(k)')
316       abline(v=k, col=2, lty='dashed')
```

```
317
318       par(mfrow=c(1,1))
319     }
320
321     #Aggiorno i vettori che ritorno come risultato
322     mpc.list <- c(mpc.list, m_pc)
323     ssw.list <- cbind(ssw.list, ssw_k)
324     T2.list <- cbind(T2.list, as.vector(T2.stat))
325     index <- c(index, frame.index)
326     Dk.list <- cbind(Dk.list, Dk)
327     dati.clust.k <- cbind(dati.clust.k, gruppi[,k]$cluster)
328     cat("done\n")
329
330     #Mi fermo se raggiungo il frame STOP_FRAME (se so che un evento
331     #fuori controllo e' avvenuto nei dati prima di quel momento)
332     if(frame.index == STOP_FRAME){
333       cat(paste0("\n(STCPA) ",frame.index," frames out ",nframes,
334                  " total frames analyzed: no alarm ",
335                  "(stop due to STOP_FRAME parameter)\n"))
336       x <- list(alarm.time=NA, m_pc=mpc.list, ssw_k=ssw.list,
337                 T2.stat=T2.list, frame=index, Dk=Dk.list,
338                 dati.clust.k=dati.clust.k)
339       return(x)
340     }
341   }
342
343   cat("\n(STCPA): Ciclo while terminato\n")
344   #Fine del ciclo: due possibili casi:
345   # 1 - ho interrotto perche' processo OC
346   if(k == 3){
347     cat(paste0("\nALARM AT FRAME ",frame.index,"!\n"))
348     x <- list(alarm.time=frame.index, m_pc=mpc.list,
349               ssw_k=ssw.list,T2.stat=T2.list,
350               frame=index, Dk=Dk.list,
351               dati.clust.k=dati.clust.k)
352     return(x)
353   }
354
```

```
355   # 2 - tutto IC
356   cat(paste0("\n(STCPA) ",frame.index," frames out ",nframes,
357               " total frames analyzed: no alarm\n"))
358   x <- list(alarm.time=NA, m_pc=mpc.list, ssw_k=ssw.list,
359             T2.stat=T2.list, frame=index, Dk=Dk.list,
360             dati.clust.k=dati.clust.k)
361   return(x)
362 }
```

A.4 Codice *SASAM*

```
1  ## Kernel di Epanichinov della statistica spaziale W_t
2  Kh.fun <- function(x,h) max( (1 - (x/h)^2 ) , 0)
3
4  ## Funzione per calcolare una volta sola Kh.mat per l'aggiornamento
5  ## delle statistiche spaziali W(i)
6  Kh.mat.fun <- function(dist.mat, h){
7    cat("Beginning the calculation of Kh.mat, h=",h,"... ",sep='')
8    temp <-  (1 - (dist.mat/h)^2 )
9    temp[temp<0] <- 0
10   cat("done\n")
11   return(temp)
12 }
13
14 ## Implementazione della carta di controllo SASAM bilaterale
15 SASAM.bilat.fun <- function(file.video=NULL, data=NULL,
16                              W0.mat=NULL, q=10, theta=c(0.1,0.5),
17                              u_min=1.5, h=5, H, plots=FALSE,
18                              STOP_FRAME=NULL, seed=888){
19
20   ## PARAMETRI GENERALI in inupt della funzione:
21   # file.video              nome del video da sorvegliare
22   #
23   # data                    array (M x N x nframes) contenente i
24   #                         frames da sorvegliare
25   #
26   # W0.mat                  matrice delle distanze tra pixel
```

```
27  #
28  # q                       numero di variabili osservabili ad ogni
29  #                         istante, espresso in termini relativi,
30  #                         cioe' solo il q% delle M*N variabili e'
31  #                         osservabile (1 < q < 100)
32  #
33  # theta=(theta1,theta2)   parametri che regolano la carta
34  #
35  # u_min                   magnitudine minima di interesse del
36  #                         cambiamento
37  #
38  # h                       bandwidth (per la funzione kernel Kh)
39  #
40  # H                       limite di controllo calcolato in
41  #                         precedenza per simulazione da dati IC
42  #
43  # plots                   mostrare dei grafici riassuntivi?
44  #
45  # STOP_FRAME              frame al quale l'analisi si ferma se
46  #                         non e' stato lanciato un allarme (serve
47  #                         solo per provare la funzione e se si sa
48  #                         dove si dovrebbe lanciare l'allarme)
49
50  require(av); require(imager)
51
52  ## INIZIALIZZAZIONE
53  #Controllo sull'input
54  if(is.null(file.video) & is.null(data)){
55    warning(paste("(SASAM): Neither 'file.video' nor 'data'",
56                  "was given in input, returning NULL"))
57    return(NULL)
58  }
59
60  #Se ho un array (M x N x nframes):
61  if(!is.null(data)){
62    nframes <- dim(data)[3]    #numero di frames
63    M <- dim(data)[1]          #altezza, numero di RIGHE
64    N <- dim(data)[2]          #larghezza, numero di COLONNE
```

```
65  }
66
67  #Se ho il nome di un file in ingresso:
68  if(!is.null(file.video)){
69    #numero di frames
70    nframes <- av_media_info(file.video)$video$frames
71    #altezza, numero di RIGHE
72    M <- av_media_info(file.video)$video$height
73    #larghezza, numero di COLONNE
74    N <- av_media_info(file.video)$video$width
75
76    cat(paste("\n(SASAM): Creating an array with data from the",
77              "input video file "))
78    #Salvo i frames del video in un array:
79    # - leggendo i file jpeg dei frames che avevo gia' salvato in
80    #   precedenza:
81    if(dir.exists(paste0(dirname,"/",file.video,"_jpeg"))){
82      cat("using existing \'jpeg\' files\n")
83      data <- image_array_general(file.video, M=M, N=N)
84    }
85
86    # - creando i file jpeg dei frames del video
87    else{
88      cat("saving \'jpeg\' files\n")
89      save_frames(file.video, format='jpeg')
90      data <- image_array_general(file.video, M=M, N=N)
91    }
92    cat("(SASAM): done\n")
93  }
94
95  #Calcolo il numero di variabili osservabili q
96  q <- min(c(ceiling(M*N*(q/100)), M*N))
97
98  cat("(SASAM): ACTIVE PARAMETERS:\n")
99  print(data.frame(PARAMETER = c("file.video","q", "M", "N",
100                                 "nframes", "theta1", "theta2",
101                                 "u_min", "h"),
102                   VALUE = c(ifelse(is.null(file.video),"NULL",
```

```
                                        file.video), q, M, N,
                                nframes, theta[1], theta[2],
                                u_min, h)))

if(is.null(STOP_FRAME)) STOP_FRAME <- nframes

t <- 1  #indice del frame analizzato
#dati in versione 'unfolded'
dati.unfolded <- as.vector(data[,,t])

#Coordinate dei pixel dei frames ( (M*N x 2) = (p x 2) )
coord.dati <- coord.fun(data[,,t])

#Operazione che richiede (molto) tempo: calcolo le distanze tra
#pixel: w_ij = |x_i - x_j|, argomento della funzione Kh() in
#W(i)_t. Ottengo una matrice (M*N x M*N) = (p x p)
if(is.null(W0.mat)){
  cat(paste("\n(SASAM): Creating W0.mat (matrix of distances",
            "between pixels in each frame); it is the most",
            "time-consuming process\n"))
  W0.mat <- dist.mat.fun(coord.dati)
}

#Visualizzo a schermo che W0.mat e' stata fornita in input
if(!is.null(W0.mat)){
  cat(paste("\n(SASAM): W0.mat (matrix of distances between",
            "pixels in each frame) given in input\n"))
}

cat(paste0("\n(SASAM): Creating Kh.mat (matrix containing ",
           "kernel values), h=",h,"\n"))
Kh.mat <- Kh.mat.fun(dist.mat=W0.mat, h=h)

#Valori iniziali dei parametri per t=1:
p <- length(dati.unfolded)
qdt <- 0
Odt.index<-NULL
coord.Odt<-NULL
```

```
141  W1_t <- matrix(NA, nrow=p, ncol=nframes)
142  W2_t <- matrix(NA, nrow=p, ncol=nframes)
143  Wi_0 <- rep(0,p)
144  #e' comune per i=1,2 al tempo 0, serve per il calcolo di St
145
146  allarme <- FALSE #per capire se lancio un allarme
147  MAX_t <- min(nframes, STOP_FRAME)
148
149  #Liste per gli output:
150  qw.list <- c()
151  qd.list <- 0
152  St.list <- c()
153  Owt.coord.list <- list()
154  Odt.coord.list <- list()
155  Odt.coord.list[[1]] <- 0
156
157  ## CICLO: INIZIO LA SORVEGLIANZA
158  set.seed(seed)
159  while(t <= MAX_t  &  !allarme){
160    #Finche' il processo e' IC acquisisco un nuovo frame e procedo
161
162    #Leggo i nuovi dati
163    dati.unfolded <- as.vector(data[,,t])
164    #Ottengo i residui dal modello per i dati
165    res <- get.residual.gmm2(dati.unfolded)$res
166    dati.unfolded <- scale(res)
167
168    cat(paste0("(SASAM): Frame ",t,": "))
169
170    #Aggiornamento di qwt
171    qwt <- q-qdt
172    qw.list <- c(qw.list, qwt)
173    cat(paste0('qW=',qwt,', qD=',qdt,'... '))
174
175    #Owt: random spatial sampling sui pixel (escludendo quelli
176    #in Odt)
177    if(!is.null(Odt.index)){ #da t=2 in poi
178      Owt.index <-
```

```
179         sample((1:nrow(coord.dati))[-Odt.index], qwt, replace=F)
180     }
181     else{
182       Owt.index <- sample(1:nrow(coord.dati), qwt, replace=F)
183     }
184
185     coord.Owt <- coord.dati[Owt.index,]
186
187     Owt.coord.list[[t]] <- coord.Owt
188
189     #Ottengo le q osservazioni yt a seconda dello schema di
190     #campionamento
191     y_wt <- dati.unfolded[Owt.index]
192     y_dt <- dati.unfolded[Odt.index]
193     yt <- c(y_wt, y_dt)
194     coord.y <- rbind(coord.Owt, coord.Odt)
195
196     #Elemento che contiene gli indici di 'coord.dati'
197     #corrispondenti ai q pixel estratti
198     Owdt.index <- c(Owt.index, Odt.index)
199
200     #Aggiornamento delle statistiche locali W(1)_t W(2)_t:
201     cat("W(1)")
202     W1_t[,t] <- cbind(Wi_0,W1_t)[,t] +
203       apply(Kh.mat, 1, function(x){
204         sum(x[Owdt.index] * (u_min*yt-(u_min^2)/2)) })
205     W1_t[,t][W1_t[,t] < 0] <-0
206
207     cat("... W(2)")
208     W2_t[,t] <- cbind(Wi_0,W1_t)[,t] +
209       apply(Kh.mat, 1, function(x){
210         sum(x[Owdt.index] * (-u_min*yt-(u_min^2)/2)) })
211     W2_t[,t][W2_t[,t] < 0] <-0
212
213     ## Statistica di controllo St
214     cat("... St")
215     St <- max(W1_t[,t], W2_t[,t])
216     St.list <- c(St.list, St)
```

```
217
218     if(plots){
219       par(mfrow=c(2,2))
220
221       plot_img(imager::renorm(W1_t[,t], min=0, max=255), M=M, N=N,
222                col=hcl.colors(256, 'Grays'), xlab='X (pixels)',
223                ylab='Y (pixels)', main=expression('W'[t]^{'(1)'}))
224       plot_img(imager::renorm(W2_t[,t], min=0, max=255), M=M, N=N,
225                col=hcl.colors(256, 'Grays'), xlab='X (pixels)',
226                ylab='Y (pixels)', main=expression('W'[t]^{'(2)'}))
227
228       plot_img(data[,,t], col=hcl.colors(256, 'Grays'), M=M, N=N,
229                xlab='X (pixels)', ylab='Y (pixels)',  cex.main=.9,
230                main=paste('Frame',t,'\nVariabili-W in blu,',
231                           'varibili-D in rosso'))
232       add_points(coord.Owt, col=4, pch=20, cex=.75, M=M, N=N)
233       if(qdt!=0){ add_points(coord.Odt, col=2, pch=20,
234                              cex=.75, M=M, N=N)}
235
236       plot(St.list,type='b', pch=20, xlab='t',
237            ylab=expression(S['t']),
238            main=expression(paste("Andamento di ", S['t'])) )
239       abline(h=H,col=2)
240
241       par(mfrow=c(1,1))
242     }
243
244     #Confronto con il limite di controllo H
245     if(St > H){
246       allarme <- TRUE #allarme
247       cat(": OC!\n")
248     }
249
250     else{
251       cat(": IC => updating parameters")
252       #determino qdt+1 e Odt+1
253       kstar.index <- which.max(c(W2_t[,t], W1_t[,t]))
254       kstar.index <-
```

```
255         ifelse(kstar.index<=p, kstar.index, kstar.index-p)
256       perm <- sapply(1:p,function(j){
257         dist_xy(coord.dati[kstar.index,],coord.dati[j,])
258       })
259       perm.sort <- sort(perm,index.return=T)
260
261       qdt <- ceiling( (q*theta[2] * max(0,(St-H*theta[1]))) /
262                         (H*(1-theta[1])) )
263       Odt.index <- perm.sort$ix[1:qdt]
264       coord.Odt <- coord.dati[Odt.index,]
265       cat("... done\n")
266
267       qd.list <- c(qd.list, qdt)
268       Odt.coord.list[[(t+1)]] <- coord.Odt
269     }
270
271     #Aumento l'indice del frame a cui sono arrivato
272     t <- t + 1
273   }
274
275   cat("\n(SASAM): Cycle while ended\n")
276   #fine del ciclo: due possibili casi:
277   # 1 - ho interrotto perche' processo OC
278   if(allarme){
279     cat(paste0("\nALARM AT FRAME ",t-1,"!\n"))
280     x <- list(alarm.time=(t-1), St=St.list, W1_t=W1_t[,1:(t-1)],
281               W2_t=W2_t[,1:(t-1)], qwt=qw.list, qdt=qd.list,
282               Owt=Owt.coord.list, Odt=Odt.coord.list)
283     return(x)
284   }
285
286   # 2 - tutto IC
287   cat(paste0("\n(SASAM) ",t-1," frames out ",nframes,
288              " total frames analyzed: no alarm\n"))
289   x <- list(alarm.time=NA, St=St.list, W1_t=W1_t[,1:(t-1)],
290             W2_t=W2_t[,1:(t-1)], qwt=qw.list, qdt=qd.list,
291             Owt=Owt.coord.list, Odt=Odt.coord.list)
292   return(x)
```

```
}

## Implementazione della carta di controllo SASAM unilaterale
## (per shift positivi)
SASAM.unilat.fun <- function(file.video=NULL, data=NULL,
                             W0.mat=NULL, q=10, theta=c(0.1,0.5),
                             u_min=1.5, h=5, H=500, plots=FALSE,
                             STOP_FRAME=NULL, seed=888){

  ## PARAMETRI GENERALI in inupt della funzione:
  # file.video              nome del video da sorvegliare
  #
  # data                    array (M x N x nframes) contenente i
  #                         frames da sorvegliare
  #
  # W0.mat                  matrice delle distanze tra pixel
  #
  # q                       numero di variabili osservabili ad ogni
  #                         istante, espresso in termini relativi,
  #                         cioe' solo il q% delle M*N variabili e'
  #                         osservabile (1 < q < 100)
  #
  # theta=(theta1,theta2)   parametri che regolano la carta
  #
  # u_min                   magnitudine minima di interesse del
  #                         cambiamento
  #
  # h                       bandwidth (per la funzione kernel Kh)
  #
  # H                       limite di controllo calcolato in
  #                         precedenza per simulazione da dati IC
  #
  # plots                   mostrare dei grafici riassuntivi?
  #
  # STOP_FRAME              frame al quale l'analisi si ferma se
  #                         non e' stato lanciato un allarme (serve
  #                         solo per provare la funzione e se si sa
  #                         dove si dovrebbe lanciare l'allarme)
```

```
331
332
333 require(av); require(imager)
334
335 ## INIZIALIZZAZIONE
336 if(is.null(file.video) & is.null(data)){
337   warning("(SASAM): Neither \'file.video\' nor \'data\' was",
338           "given in input, returning NULL\n")
339   return(NULL)
340 }
341
342 #Se ho un array M x N x nframes:
343 if(!is.null(data)){
344   nframes <- dim(data)[3]   #numero di frames
345   M <- dim(data)[1]         #altezza, numero di RIGHE
346   N <- dim(data)[2]         #larghezza, numero di COLONNE
347 }
348
349 #Se ho il nome di un file in ingresso:
350 if(!is.null(file.video)){
351   #numero di frames
352   nframes <- av_media_info(file.video)$video$frames
353   #altezza, numero di RIGHE
354   M <- av_media_info(file.video)$video$height
355   #larghezza, numero di COLONNE
356   N <- av_media_info(file.video)$video$width
357
358   cat(paste("\n(SASAM): Creating an array with data from the",
359             "input video file "))
360   #Salvo i frames del video in un array:
361   # - leggendo i file jpeg dei frames che avevo gia' salvato in
362   #   precedenza:
363   if(dir.exists(paste0(dirname,"/",file.video,"_jpeg"))){
364     cat("using existing \'jpeg\' files\n")
365     data <- image_array_general(file.video, M=M, N=N)
366   }
367
368   # - creando i file jpeg dei frames del video
```

```
369     else{
370       cat("saving \'jpeg\' files\n")
371       save_frames(file.video, format='jpeg')
372       data <- image_array_general(file.video, M=M, N=N)
373     }
374     cat("(SASAM): done\n")
375   }
376
377   #Calcolo il numero di variabili osservabili q
378   q <- min(c(ceiling(M*N*(q/100)), M*N))
379
380   cat("(SASAM): ACTIVE PARAMETERS:\n")
381   print(data.frame(PARAMETER = c("file.video", "q", "M", "N",
382                                  "nframes", "theta1", "theta2",
383                                  "u_min", "h"),
384                    VALUE = c(ifelse(is.null(file.video),"NULL",
385                                     file.video), q, M, N,
386                              nframes, theta[1], theta[2],
387                              u_min, h)))
388
389   if(is.null(STOP_FRAME)) STOP_FRAME <- nframes
390
391   t <- 1  #indice del frame analizzato
392   #dati in versione 'unfolded'
393   dati.unfolded <- as.vector(data[,,t])
394
395   #Coordinate dei pixel dei frames ( (M*N x 2) = (p x 2) )
396   coord.dati <- coord.fun(data[,,t])
397
398   #Operazione che richiede (molto) tempo: calcolo le distanze tra
399   #pixel: w_ij = |x_i-x_j|, argomento della funzione Kh() in
400   #W(i)_t. Ottengo una matrice (M*N x M*N) = (p x p)
401   if(is.null(W0.mat)){
402     cat(paste("\n(SASAM): Creating W0.mat (matrix of distances",
403               "between pixels in each frame); it is the most",
404               "time-consuming process\n"))
405     W0.mat <- dist.mat.fun(coord.dati)
406   }
```

```
407
408 #Visualizzo a schermo che W0.mat e' stata fornita in input
409 if(!is.null(W0.mat)){
410   cat(paste("\n(SASAM): W0.mat (matrix of distances between",
411             "pixels in each frame) given in input\n"))
412 }
413
414 cat(paste0("\n(SASAM): Creating Kh.mat (matrix containing ",
415            "kernel values), h=",h,"\n"))
416 Kh.mat <- Kh.mat.fun(dist.mat=W0.mat, h=h)
417
418 #Valori iniziali dei parametri per t=1:
419 p <- length(dati.unfolded)
420 qdt <- 0
421 Odt.index <- NULL
422 coord.Odt <- NULL
423 W1_t <- matrix(NA, nrow=p, ncol=nframes)
424 Wi_0 <- rep(0,p)
425 #e' comune per i=1,2 al tempo 0, serve per il calcolo di St
426 #
427 allarme <- FALSE #per capire se lancio un allarme
428 MAX_t <- min(nframes, STOP_FRAME)
429
430 #Liste per gli output:
431 qw.list <- c()
432 qd.list <- 0
433 St.list <- c()
434 Owt.coord.list <- list()
435 Odt.coord.list <- list()
436 Odt.coord.list[[1]] <- 0
437
438 ## CICLO: INIZIO LA SORVEGLIANZA
439 set.seed(seed)
440 while(t <= MAX_t  &  !allarme){
441   #Finche' il processo e' IC acquisisco un nuovo frame e procedo
442
443   #Leggo i nuovi dati
444   dati.unfolded <- as.vector(data[,,t])
```

```
445     #Ottengo i residui dal modello per i dati
446     res <- get.residual.gmm2(dati.unfolded)$res
447     dati.unfolded <- scale(res)
448
449     cat(paste0("(SASAM): Frame ",t,": "))
450
451     #Aggiornamento di qwt
452     qwt <- q-qdt
453     qw.list <- c(qw.list, qwt)
454     cat(paste0('qW=',qwt,', qD=',qdt,'... '))
455
456     #Owt: random spatial sampling sui pixel (escludendo quelli in
457     #Odt)
458     if(!is.null(Odt.index)){ #da t=2 in poi
459       Owt.index <-
460         sample((1:nrow(coord.dati))[-Odt.index], qwt, replace=F)
461     }
462     else{
463       Owt.index <- sample(1:nrow(coord.dati), qwt, replace=F)
464     }
465
466     coord.Owt <- coord.dati[Owt.index,]
467
468     Owt.coord.list[[t]] <- coord.Owt
469
470     #Ottengo le q osservazioni yt a seconda dell schema di
471     #campionamento
472     y_wt <- dati.unfolded[Owt.index]
473     y_dt <- dati.unfolded[Odt.index]
474     yt <- c(y_wt, y_dt)
475     coord.y <- rbind(coord.Owt, coord.Odt)
476
477     #Elemento che contiene gli indici di 'coord.dati'
478     #corrispondenti ai q pixel campionati
479     Owdt.index <- c(Owt.index, Odt.index)
480
481     #Update della statistica locale W(1)_t:
482     cat("W(1)")
```

```
483     W1_t[,t] <- cbind(Wi_0,W1_t)[,t] +
484       apply(Kh.mat, 1, function(x){
485         sum(x[Owdt.index] * (u_min*yt-(u_min^2)/2)) })
486     W1_t[,t][W1_t[,t] < 0] <-0
487
488     ## Statistica di controllo St
489     cat("... St")
490     St <- max(W1_t[,t])
491     St.list <- c(St.list, St)
492
493     if(plots){
494       par(mfrow=c(2,2))
495
496       plot_img(imager::renorm(W1_t[,t], min=0, max=255), M=M, N=N,
497                col=hcl.colors(256, 'Grays'), xlab='X (pixels)',
498                ylab='Y (pixels)', main=expression('W'[t]^{'(1)'}))
499
500       plot_img(data[,,t], col=hcl.colors(256, 'Grays'), M=M, N=N,
501                xlab='X (pixels)', ylab='Y (pixels)',  cex.main=.9,
502                main=paste0('Frame ',t,'\nVariabili-W in blu, ',
503                            'varibili-D in rosso'))
504       add_points(coord.Owt,col=4,pch=20,cex=.75,M=M, N=N)
505       if(qdt!=0){ add_points(coord.Odt, col=2, pch=20,
506                              cex=.75, M=M, N=N)}
507
508       plot(St.list,type='b',pch=20,xlab='t',
509            ylab=expression(S['t']),
510            main=expression(paste("Andamento di ", S['t'])) )
511       abline(h=H,col=2)
512
513       par(mfrow=c(1,1))
514     }
515
516     #Confronto con il limite di controllo H
517     if(St > H){
518       allarme <- TRUE #allarme
519       cat(": OC!\n")
520     }
```

```
521
522     else{
523       cat(": IC => updating parameters")
524       #determino qdt+1 e Odt+1
525       kstar.index <- which.max(W1_t[,t])
526       perm <- sapply(1:p, function(j){
527         dist_xy(coord.dati[kstar.index,], coord.dati[j,]) })
528       perm.sort<-sort(perm,index.return=T)
529
530       qdt <- ceiling( (q*theta[2] * max(0,(St-H*theta[1])) ) /
531                         (H*(1-theta[1])) )
532       Odt.index <- perm.sort$ix[1:qdt]
533       coord.Odt <- coord.dati[Odt.index,]
534       cat("... done\n")
535
536       qd.list <- c(qd.list, qdt)
537       Odt.coord.list[[(t+1)]] <- coord.Odt
538     }
539
540     #aumento l'indice del frame a cui sono arrivato
541     t <- t + 1
542   }
543
544   cat("\n(SASAM): Cycle while ended\n")
545   #fine del ciclo: due possibili casi:
546   # 1 - ho interrotto perche' processo OC
547   if(allarme){
548     cat(paste0("\nALARM AT FRAME ",t-1,"!\n"))
549     x <- list(alarm.time=(t-1), St=St.list, W1_t=W1_t[,1:(t-1)],
550               qwt=qw.list, qdt=qd.list, Owt=Owt.coord.list,
551               Odt=Odt.coord.list)
552     return(x)
553   }
554
555   # 2 - tutto IC (non dovrebbe verificarsi coi dati che ho)
556   cat(paste0("\n(SASAM) ",t-1," frames out ",nframes,
557              " total frames analyzed: no alarm\n"))
558   x <- list(alarm.time=NA, St=St.list, W1_t=W1_t[,1:(t-1)],
```

```
559             qwt=qw.list, qdt=qd.list, Owt=Owt.coord.list,
560             Odt=Odt.coord.list)
561   return(x)
562 }
563
564 ## Stima bootstrap del limite di controllo H
565 #Inizializzazione dei parametri:
566 Nsim <- 10000         #numero di simulazioni bootstrap
567 nframes.bs <- 1000    #quanti frames generare tramite bootstrap
568 RL.list <- c()
569 RL.stima.list <- c() #sequenza delle run length stimate
570 H.list <- c()         #sequenza dei limiti di controllo stimati
571 cond <- TRUE
572 eps <- 2
573 ARL_u <- 200      #ARL in controllo desiderato
574 Hmin <- 0
575 Hmax <- 200
576
577 #In memoria sono gia' presenti gli elementi:
578 # dati.IC => array M x N x nframes contenente dati in controllo
579 # W0.mat  => matrice delle distanze tra pixel
580
581 seed <- 1234
582 set.seed(seed)
583
584 #Ripeto finche' RL e ARL_u differiscono di poco
585 while(cond){
586   #cond=TRUE se la diffferenza tra RL e ARL e' grande e quindi
587   #bisogna continuare
588
589   #Ottengo tramite bootstrap una stima di RL
590   sim.index <- 1
591   while(sim.index <= Nsim){
592
593     #Si genera tramite bootstrap dataset di nframes.bs osservazioni
594     #usando i frames in controllo (dati.IC); si implementa quindi
595     #SASAM e si ricava un valore di RL
596     RL <- SASAM.unilat.fun(
```

```
597       data=dati.IC[,,sample(1:dim(dati.IC)[3], nframes.bs, replace=T)],
598       W0.mat=W0.mat, q=5, theta=c(0.2, 0.5), u_min=1.5, h=5,
599       H=(Hmin+Hmax)/2, seed=888)$alarm.time
600
601     if(is.null(RL)) RL <- ncol(dati.boot) #se non c'e' stato allarme
602     RL.list <- c(RL.list, RL)
603
604     sim.index <- sim.index+1
605   }
606
607   #Aggiornamento di Hmin o Hmax per la prossima iterazione
608   if(mean(RL.list) > ARL_u){ Hmax <- (Hmin+Hmax)/2 }
609   else{ Hmin <- (Hmin+Hmax)/2 }
610
611   #Aggiornamento di cond: se la differenza e' ancora grande si continua
612   if( abs(mean(RL.list)-ARL_u) > eps){ cond <- TRUE }
613   else{ cond <- FALSE }
614   H.list <- c(H.list, (Hmin+Hmax)/2)
615   RL.stima.list <- c(RL.stima.list, mean(RL.list))
616 }
617
618 #Stima bootstrap del limite di controllo:
619 H.stima <- H.list[length(H.list)]
```

Bibliografia

Armingol, José María, F. Javier Otamendi, Arturo de la Escalera, José Manuel Pastor e Francisco José Jimenez Rodríguez (2003). «Statistical Pattern Modeling in Vision-Based Quality Control Systems». In: *Journal of Intelligent and Robotic Systems* 37.3, pp. 321–336. DOI: 10.1023/A:1025489610281.

Barthelme, Simon (2021). *imager: Image Processing Library Based on 'CImg'*. R package version 0.42.11. URL: https://CRAN.R-project.org/package=imager.

Bharati, Manish H. e John Frederick MacGregor (1998). «Multivariate Image Analysis for Real-Time Process Monitoring and Control». In: *Industrial and Engineering Chemistry Research* 37.12, pp. 4715–4724. DOI: 10.1021/IE980334L.

Bui, Anh Tuan e Daniel W. Apley (2018). «A Monitoring and Diagnostic Approach for Stochastic Textured Surfaces». In: *Technometrics* 60.1, pp. 1–13. DOI: 10.1080/00401706.2017.1302362.

Cheng, Chuen-Sheng e Hui-Ping Cheng (2008). «Identifying the Source of Variance Shifts in the Multivariate Process Using Neural Networks and Support Vector Machines». In: *Expert Systems with Applications: An International Journal* 35.1-2, pp. 198–206. DOI: 10.1016/J.ESWA.2007.06.002.

Colosimo, Bianca Maria e Marco Grasso (2018). «Spatially Weighted PCA for Monitoring Video Image Data with Application to Additive Manufacturing». In: *Journal of Quality Technology* 50.4, pp. 391–417. DOI: 10.1080/00224065.2018.1507563.

Colosimo, Bianca Maria, Federica Mammarella e Stefano Petrò (2010). «Quality Control of Manufactured Surfaces». In: *Frontiers in Statistical Quality Control 9*, pp. 55–70. DOI: 10.1007/978-3-7908-2380-6_4.

Cos'è la produzione additiva in metallo? (2021). URL: https://www.renishaw.it/it/cose-la-produzione-additiva-in-metallo--15240 (visitato il 18/11/2021).

Dávila, Saylisse, George Runger e Eugene Tuv (2011). «High-dimensional Surveillance». In: *Artificial Neural Networks and Machine Learning, ICANN 2011* 6792 LNCS.PART 2, pp. 245–252. DOI: 10.1007/978-3-642-21738-8_32.

Deng, Houtao, George Runger e Eugene Tuv (2012). «System Monitoring with Real-Time Contrasts». In: *Journal of Quality Technology* 44.1, pp. 9–27. DOI: 10.1080/00224065.2012.11917878.

Facco, Pierantonio, Fabrizio Bezzo, José A. Romagnoli e Massimiliano Barolo (2008). «Using Digital Images for Fast, Reliable, and Automatic Characterization of Surface Quality: a Case Study on the Manufacturing of Smiconductors». In: *Workshop on Nanomaterials Production, Characterization and Industrial Applications*. Milano, Italia.

Golnabi, Hossein e Seyyed Hossein Asadpour (2007). «Design and Application of Industrial Machine Vision Systems». In: *Robotics and Computer-Integrated Manufacturing* 23.6, pp. 630–637. DOI: 10.1016/J.RCIM.2007.02.005.

Graham, Kevin J., Krishnakumar Krishnapisharody, Gordon A. Irons e John Frederick MacGregor (2007). «Monitoring Ladle Eye Dynamics Using Multivariate Statistical Methods». In: *AISTech - Iron and Steel Technology Conference Proceedings* 1, pp. 1369–1379.

Grasso, Marco, Vittorio Laguzza, Quirico Semeraro e Bianca Maria Colosimo (2017). «In-Process Monitoring of Selective Laser Melting: Spatial Detection of Defects Via Image Data Analysis». In: *Journal of Manufacturing Science and Engineering, Transactions of the ASME* 139.5. DOI: 10.1115/1.4034715.

Gronskyte, Ruta, Murat Kulahci, Line Katrine e Harder Clemmensen (2013). «Monitoring Motion of Pigs in Thermal Videos». In: *Workshop on Farm Animal and Food Quality Imaging*, pp. 31–36.

Grün, Bettina e Friedrich Leisch (2007). «Fitting Finite Mixtures of Generalized Linear Regressions in R». In: *Computational Statistics & Data Analysis* 51.11, pp. 5247–5252. DOI: 10.1016/j.csda.2006.08.014.

Grün, Bettina e Friedrich Leisch (2008). «FlexMix Version 2: Finite Mixtures with Concomitant Variables and Varying and Constant Parameters». In: *Journal of Statistical Software* 28.4, pp. 1–35. DOI: 10.18637/jss.v028.i04. URL: https://www.jstatsoft.org/v28/i04/.

Horst, Robert L. e Michael Negin (1992). «Vision System for High-Resolution Dimensional Measurements and On-Line SPC: Web Process Application». In: *IEEE Transactions on Industry Applications* 28.4, pp. 993–997. DOI: 10.1109/28.148468.

Jiang, Bernard Chen-Chun, Chien-Chih Wang e H.-C. Liu (2005). «Liquid Crystal Display Surface Uniformity Defect Inspection Using Analysis of Variance and Exponentially Weighted Moving Average Techniques». In: *International Journal of Production Research* 43.1, pp. 67–80. DOI: 10.1080/00207540412331285832.

Kourti, Theodora (2005). «Application of Latent Variable Methods to Process Control and Multivariate Statistical Process Control in Industry». In: *International Journal of Adaptive Control and Signal Processing* 19.4, pp. 213–246. DOI: 10.1002/acs.859.

Leisch, Friedrich (2004). «FlexMix: A General Framework for Finite Mixture Models and Latent Class Regression in R». In: *Journal of Statistical Software* 11.8, pp. 1–18. DOI: 10.18637/jss.v011.i08. URL: https://www.jstatsoft.org/v11/i08/.

Liang, Yu Teng e Yih Chih Chiou (2008). «Vision-Based Automatic Tool Wear Monitoring System». In: *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, pp. 6031–6035. DOI: 10.1109/WCICA.2008.4592857.

Lin, Hong-Dar (2007a). «Automated Visual Inspection of Ripple Defects Using Wavelet Characteristic Based Multivariate Statistical Approach». In: *Image and Vision Computing* 11.25, pp. 1785–1801. DOI: 10.1016/J.IMAVIS.2007.02.002.

— (2007b). «Computer-Aided Visual Inspection of Surface Defects in Ceramic Capacitor Chips». In: *Journal of Materials Processing Technology* 189.1-3, pp. 19–25. DOI: 10.1016/J.JMATPROTEC.2006.12.051.

Liu, J. Jay e John Frederick MacGregor (2006). «Estimation and Monitoring of Product Aesthetics: Application to Manufacturing of "Engineered Stone" Countertops». In: *Machine Vision and Applications* 16.6, pp. 374–383. DOI: 10.1007/S00138-005-0009-8.

Liu, J. Jay, John Frederick MacGregor, Carl Duchesne e Gianni Bartolacci (2005). «Flotation Froth Monitoring Using Multiresolutional Multivariate Image Analysis». In: *Minerals Engineering* 18.1, pp. 65–76. DOI: 10.1016/J.MINENG.2004.05.010.

Liu, Kaibo, Yajun Mei, Jianjun Shi e H. Milton Stewart (2015). «An Adaptive Sampling Strategy for Online High-Dimensional Process Monitoring». In: *Technometrics* 57.3, pp. 305–319. DOI: 10.1080/00401706.2014.947005.

Liu, Zhi Qiang, Timothy J. Austin, Christopher David L. Thomas e John G. Clement (1996). «Bone Feature Analysis Using Image Processing Techniques». In: *Computers in Biology and Medicine* 26.1, pp. 65–76. DOI: 10.1016/0010-4825(95)00044-5.

Liu, Zhongqiu, Linmin Li e Baokuan Li (2017). «Modeling of Gas-Steel-Slag Three-Phase Flow in Ladle Metallurgy: Part I. Physical Modeling». In: *ISIJ International* 57.11, pp. 1971–1979. DOI: 10.2355/isijinternational.ISIJINT-2016-710.

Lu, Chi-Jie e Du-Ming Tsai (2004). «Automatic Defect Inspection for LCDs Using Singular Value Decomposition». In: *The International Journal of Advanced Manufacturing Technology 2004 25:1* 25.1, pp. 53–61. DOI: 10.1007/S00170-003-1832-6.

Lyu, Jr Jung e Ming Nan Chen (2009). «Automated Visual Inspection Expert System for Multivariate Statistical Process Control Chart». In: *Expert Systems with Applications* 36.3 PART 1, pp. 5113–5118. DOI: 10.1016/J.ESWA.2008.06.047.

Megahed, Fadel M., William H. Woodall e Jaime A. Camelio (2011). «A Review and Perspective on Control Charting with Image Data». In: *Journal of Quality Technology* 43.2, pp. 83–98. DOI: 10.1080/00224065.2011.11917848.

Nembhard, Harriet Black, Nicola J. Ferrier, Tim A. Osswald e Juan R. Sanz-Uribe (2003). «An Integrated Model for Statistical and Vision Monitoring in Manufacturing Transitions». In: *Quality and Reliability Engineering International* 19.6, pp. 461–476. DOI: 10.1002/QRE.517.

Ooms, Jeroen (2022). *av: Working with Audio and Video in R*. R package version 0.7.0. URL: https://CRAN.R-project.org/package=av.

Peihua, Qiu (2014). *Introduction to Statistical Process Control*. Chapman & Hall/CRC Texts in Statistical Science Series. Chapman e Hall/CRC. ISBN: 9781439847992.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Ramasetti, Eshwar Kumar, Ville Valtteri Visuri, Petri Sulasalmi, Timo Fabritius, Jari Savolainen, Mingming Li e Lei Shao (2019). «Numerical Modelling of the Influence of Argon Flow Rate and Slag Layer Height on Open-Eye Formation in a 150 Ton Steelmaking Ladle». In: *Metals 2019, Vol. 9, Page 1048* 9.10, p. 1048. DOI: 10.3390/MET9101048.

RStudio Team (2015). *RStudio: Integrated Development Environment for R*. RStudio, Inc. Boston, MA. URL: http://www.rstudio.com/.

Schmitt, K. M., John R. Riddington, Rupert C.D. Young, David M. Budgett e C. R. Chatwin (2000). «Image Processing Applied to Brick Quality Control». In: *International Journal of Advanced Manufacturing Technology* 16.6, pp. 434–440. DOI: 10.1007/s001700050175.

Soetaert, Karline (2021). *plot3D: Plotting Multi-Dimensional Data*. R package version 1.4. URL: https://CRAN.R-project.org/package=plot3D.

Stahlschmidt, Stephan, Wolfgang K. Härdle e Helmut Thome (2015). «An Application of Principal Component Analysis on Multivariate Time-stationary Spatio-temporal Data». In: *Spatial Economic Analysis* 10.2, pp. 160–180. DOI: 10.1080/17421772.2015.1023339.

Tan, Jinglu, Z. Chang e Fit Hung Hsieh (1996). «Implementation of an Automated Real-Time Statistical Process Controller». In: *Journal of Food Process Engineering* 19.1, pp. 49–61. DOI: 10.1111/J.1745-4530.1996.TB00380.X.

The Sun Flares with Activity (2012). URL: https://earthobservatory.nasa.gov/images/76998/the-sun-flares-with-activity (visitato il 08/12/2021).

Tong, Lee Ing, Chung Ho Wang e Chih Li Huang (2005). «Monitoring Defects in IC Fabrication Using a Hotelling T^2 Control Chart». In: *IEEE Transactions on Semiconductor Manufacturing* 18.1, pp. 140–147. DOI: 10.1109/TSM.2004.836659.

Tunák, Maroš e Aleš Linka (2008). «Directional Defects in Fabrics». In: *Research Journal of Textile and Apparel* 12.2, pp. 13–22. DOI: 10.1108/RJTA-12-02-2008-B002.

Tunák, Maroš, Aleš Linka e Petr Volf (2009). «Automatic Assessing and Monitoring of Weaving Density». In: *Fibers and Polymers* 10.6, pp. 830–836. DOI: 10.1007/S12221-009-0830-1.

Urbanek, Simon (2021). *jpeg: Read and write JPEG images*. R package version 0.1-9. URL: https://CRAN.R-project.org/package=jpeg.

Venkatasubramanian, Venkat, Raghunathan Rengaswamy, Surya N. Kavuri e Kewen Yin (2003). «A Review of Process Fault Detection and Diagnosis Part III: Process History Based Methods». In: *Computers and Chemical Engineering* 27.3, pp. 327–346. DOI: 10.1016/S0098-1354(02)00162-X.

Wang, Andi, Xiaochen Xian, Fugee Tsung e Kaibo Liu (2018). «A Spatial-Adaptive Sampling Procedure for Online Monitoring of Big Data Streams». In: *Journal of Quality Technology* 50.4, pp. 329–343. DOI: 10.1080/00224065.2018.1507560.

Wang, Kaibo e Fugee Tsung (2005). «Using Profile Monitoring Techniques for a Data-Rich Environment with Huge Sample Size». In: *Quality and Reliability Engineering International* 21.7, pp. 677–688. DOI: 10.1002/qre.711.

Weese, Maria, Waldyn Martinez, Fadel M. Megahed e L. Allison Jones-Farmer (2016). «Statistical Learning Methods Applied to Process Monitoring: An Overview and Perspective». In: *Journal of Quality Technology* 48.1, pp. 4–27. DOI: 10.1080/00224065.2016.11918148.

Wells, Lee J., Fadel M. Megahed, Cory B. Niziolek, Jaime A. Camelio e William H. Woodall (2013). «Statistical Process Monitoring Approach for High-Density Point Clouds». In: *Journal of Intelligent Manufacturing* 24.6, pp. 1267–1279. DOI: 10.1007/S10845-012-0665-2.

Woodall, William H. (2007). «Current Research on Profile Monitoring». In: *Production* 17.3, pp. 420–425. DOI: 10.1590/S0103-65132007000300002.

Woodall, William H. e Douglas C. Montgomery (2014). «Some Current Directions in the Theory and Application of Statistical Process Monitoring». In: *Journal of Quality Technology* 46.1, pp. 78–94. DOI: 10.1080/00224065.2014.11917955.

Woodall, William H., Dan J. Spitzner, Douglas C. Montgomery e Shilpa Gupta (2004). «Using Control Charts to Monitor Process and Product Quality Profiles». In: *Journal of Quality Technology* 36.3, pp. 309–320. DOI: 10.1080/00224065.2004.11980276.

Xie, Yao, Jiaji Huang e Rebecca Willett (2012). «Multiscale Online Tracking of Manifolds». In: *2012 IEEE Statistical Signal Processing Workshop, SSP 2012* 2, pp. 620–623. DOI: 10.1109/SSP.2012.6319777.

Yu, Honglu e John Frederick MacGregor (2004). «Monitoring Flames in an Industrial Boiler Using Multivariate Image Analysis». In: *AIChE Journal* 50.7, pp. 1474–1483. DOI: 10.1002/AIC.10164.

Yu, Honglu, John Frederick MacGregor, Gabe Haarsma e Wilfred Bourg (2003). «Digital Imaging for Online Monitoring and Control of Indu-

strial Snack Food Processes». In: *Industrial and Engineering Chemistry Research* 42.13, pp. 3036–3044. DOI: 10.1021/IE020941F.