POLITECNICO DI MILANO
Corso di Laurea Magistrale in Ingegneria Informatica
Dipartimento di Elettronica, Informazione e Bioingegneria

ALGORITMO DI TAMPERING DETECTION OTTIMIZZATO TRAMITE SEGMENTAZIONE DELLA SCENA INQUADRATA

Relatore: Prof. Giacomo BORACCHI
Correlatore: Ing. Claudio MARCHISIO

Tesi di Laurea di:
Adriano GAIBOTTI, matricola 780200

Anno Accademico 2013-2014

A Sara

Sommario

Uno dei principali problemi, quando si ha a che fare con applicazioni di monitoraggio video, è quello di identificare quegli eventi che possano compromettere la corretta ripresa della scena da parte del sensore. Può capitare, ad esempio, che dell'acqua piovana si depositi sulla lente della camera, rendendo l'immagine acquisita sfocata, oppure che qualcuno sposti la camera in modo che essa non riprenda più la scena che stava monitorando. Il problema di individuare, in maniera automatica, questo tipo di eventi prende il nome di *tampering detection*. Nella letteratura scientifica lo studio di questo problema si è concentrato solamente sulle applicazioni di *videosorveglianza*, dove è necessario che la camera operi con una frequenza di acquisizione delle immagini elevata. Lo scopo della tesi è lo sviluppo di un algoritmo di tampering detection per sistemi di monitoraggio a basso consumo. Abbiamo considerato scenari, come ad esempio il monitoraggio ambientale, in cui non è necessario che la camera operi con un'acquisizione continua e, quindi, sia possibile limitare il carico computazionale acquisendo, ad esempio, un'immagine ogni minuto. In questi casi, se consideriamo il caso della ripresa di una strada, in cui passano delle macchine o dei pedoni, abbiamo un'elevata dinamicità che non permette di fare un confronto tra frame consecutivi per identificare gli eventi di nostro interesse. In aggiunta, abbiamo dei cambiamenti di luminosità, tra un'immagine e la successiva, più sostanziali rispetto al caso di acquisizione continua. La nostra proposta è quella di monitorare nel tempo degli indicatori semplici, calcolati considerando solamente il *contenuto visivo* delle singole immagini, in modo da ottimizzare le risorse hardware disponibili e ridurre la complessità computazionale. Data l'alta variabilità di questi indicatori abbiamo introdotto una *segmentazione* della scena ripresa, estratta durante una fase di *configurazione* dell'algoritmo, in modo da considerare solo le regioni in cui il monitoraggio risulta più efficace. La tesi è stata svolta durante uno stage presso *STMicroelectronics*, particolarmente interessata a sviluppare algoritmi intelligenti di processione immagini da integrare nei propri dispositivi hardware, e a scenari di impiego per questi. Prove sperimentali, fatte considerando diversi sistemi di acquisizione operanti con frame rate differenti, hanno confermato l'efficacia di utilizzare la segmentazione rispetto a considerare l'intera scena per individuare eventi di spostamento della camera.

Ringraziamenti

Ringrazio .....

Indice

Elenco delle figure

Elenco delle tabelle

Glossario

frame immagine acquisita dal sensore della camera. 11

framerate frequenza di acquisizione dei frame da parte della camera. 11

inquadratura porzione dello spazio fisico che viene acquisito dal sensore della camera. 7

scena luogo di interesse per una particolare applicazione di monitoraggio video. 7, 8

tampering letteralmente *manomissione*, nel contesto della tesi indica un qualsiasi evento che non permette una corretta ripresa della scena da parte della camera. 7

Capitolo 1

Introduzione

Negli ultimi anni le applicazioni di tipo multimediale sono aumentate in maniera esponenziale, soprattutto per quanto riguarda i contenuti video. L'abbassamento dei prezzi e delle dimensioni dei *sensori* e delle componenti hardware

Capitolo 2

Stato dell'arte

In questo capitolo elenchiamo quelle che sono le principali tecniche, presenti nella letteratura scientifica, utilizzate per identificare tentativi di manomissione su camere di videosorveglianza.

2.1 Tampering Detection

Nei moderni sistemi di videosorveglianza troviamo spesso algoritmi utilizzati per identificare particolari eventi all'interno della scena ripresa dalla camera. Ad esempio è possibile avere un software in grado di identificare le targhe delle automobili che superano il limite di velocità , oppure la presenza di oggetti incustoditi in una stazione [1]. Affinché questi algoritmi funzionino correttamente, è importante che l'*affidabilità* del sistema di acquisizione sia preservata. Per fare questo la letteratura scientifica offre molte tecniche che permettano l'identificazione automatica di eventi in grado di compromettere la corretta ripresa della scena da parte della videocamera.

2.1.1 Modello di background

Nella maggior parte dei lavori dedicati a questo problema, il concetto principale consiste nel confrontare ciascun frame con un modello che viene calcolato utilizzando i frame precedenti. Tale modello è ampliamente utilizzato in vari ambiti di visione artificiale e prende il nome di *modello di background*. Un metodo generale di calcolo del background è presentato in [2]. Indichiamo con $I_n(x, y)$ il valore, nel pixel di coordinate (x, y), della luminosità nell'i-esimo frame. Il valore del modello di background è calcolato in maniera ricorsiva secondo la seguente formula:

$$B_{n+1}(x,y) = \begin{cases} aB_n(x,y) + (1-a)I_n(x,y) & \text{, se (x, y) è fermo} \\ B_n(x,y) & \text{, se (x, y) si muove} \end{cases}, \quad (2.1)$$

dove $0 < a < 1$ è chiamato *parametro di aggiornamento* (*update parameter*). La discriminazione tra pixel che si muove e pixel che è fermo viene fatta confrontando due frame adiacenti:

$$|I_n(x,y) - I_{n-1}(x,y)| > T_n(x,y), \tag{2.2}$$

dove $T_n(x,y)$ è una soglia che permette di identificare un cambiamento sostanziale di luminosità nel pixel (x,y). Questa soglia viene aggiornata in maniera ricorsiva secondo la seguente formula:

$$T_{n+1}(x,y) = \begin{cases} aT_n(x,y) + (1-a)(c|I_n(x,y) - B_n(x,y)|) & \text{, se (x, y) è fermo} \\ T_n(x,y) & \text{, se (x, y) si muove} \end{cases}, \tag{2.3}$$

dove $c > 1$ e $0 < a < 1$. Lo stesso modello viene applicato in altri lavori ([7], [8]), mentre una variante molto usata consiste nell'estrarre il modello di background a partire dall'*estrazione dei contorni* da ciascun frame ([6], [5]).

2.1.2 Identificazione di occlusioni

L'evento di occlusione avviene quando un oggetto opaco viene posto vicino alla camera, in modo da coprire la scena ripresa. In [2] e [7] questo evento è associato a un cambiamento nella struttura dell'*istogramma* del frame occluso rispetto a quello del background. Infatti, nel caso di occlusioni, i valori dell'istogramma tendono a concentrarsi in un intervallo molto ristretto, verso i livelli più bassi della scala di grigi.
Un approccio simile è presente in [6], [5] e [4], in cui l'evento di occlusione è associato a un abbassamento dell'*entropia*:

$$E = -\sum_k p_k \ln(p_k), \tag{2.4}$$

dove p_k rappresenta la probabilità che il livello di grigio k sia presente all'interno dell'immagine.

2.1.3 Identificazione di spostamenti della camera

2.1.4 Identificazione di sfocature

Capitolo 3

Impostazione del problema di ricerca

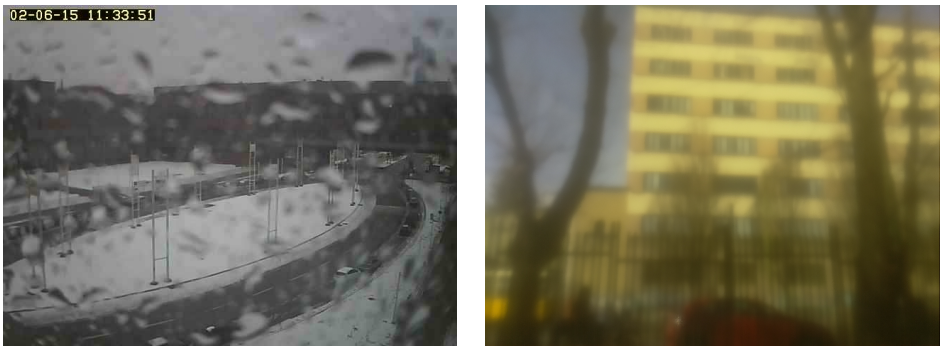
In questo capitolo descriviamo il problema, affrontato dall'algoritmo di tampering detection, in maniera formale e rigorosa. Il primo paragrafo illustra il modello delle osservazioni e gli eventi che siamo interessati a identificare, mentre il secondo paragrafo formalizza il concetto di tampering detection.

3.1 Modello delle osservazioni

Il nostro campo di osservazione si concentra su quegli eventi che si interpongono tra la scena ripresa da una camera e il sensore che acquisisce le immagini. Non vogliamo, cioè, identificare degli eventi particolari che avvengono nella scena, come un oggetto lasciato incustodito [1], bensì vogliamo identificare quegli eventi tali per cui il sensore non è più nelle condizioni di riprendere, in maniera ottimale, la scena, quali ad esempio sfocature o spostamenti della camera. Nel seguito cerchiamo di dare una definizione formale di questi eventi.

3.1.1 Scena e posizione della camera

Prima di tutto definiamo, in maniera rigorosa, i termini che verranno utilizzati nel seguito della trattazione.
Lo scenario che consideriamo è quello di una camera che deve riprendere una particolare *scena*. La posizione e la rotazione della camera determinano l'*inquadratura* della scena. In un'applicazione di monitoraggio video consideriamo che la camera, in condizioni di funzionamento ottimale, mantenga la stessa inquadratura nel tempo e che si in grado di acquisire, in maniera nitida, tutti gli elementi di interesse presenti nella scena. Definiamo tampering un qualsiasi evento che determina un cambio di inquadratura o che non



(a) Pioggia sull'obiettivo (b) Deodorante spray

Figura 3.1: Esempi di sfocature

permette la corretta acquisizione di parte o totalità della scena. Possiamo classificare gli eventi di tampering in quattro categorie:

- Sfocature
- Spostamenti della camera
- Occlusioni
- Guasti della camera

3.1.2 Sfocatura

Il fenomeno della sfocatura avviene quando un elemento trasparente o semitrasparente si interpone tra la lente della camera e la scena ripresa, oppure quando viene cambiata la messa a fuoco, causando una perdita nei dettagli della scena ripresa.
Nella figura 3.1 sono mostrati degli esempi in cui sono presenti delle sfocature. Queste possono essere di origine diversa:

- dovute a *cause naturali*, come ad esempio dell'acqua piovana che si deposita sulla lente (figura 3.1(a)), o la condensa dovuta all'umidità e alle basse temperature, oppure un raggio di sole incidente sull'obiettivo della camera;
- per *intervento dell'uomo*, che a sua volta può avvenire in maniera intenzionale (e in questo caso si può parlare di *manomissione*) oppure non intenzionale. Ad esempio, si può direttamente intervenire sulla messa a fuoco, nel caso sia possibile cambiarla manualmente; oppure (come nel caso della figura 3.1(b)) è possibile applicare una sostanza semitrasparente sulla lente della camera, come il gas di un deodorante spray.

Riprendendo [3], questo fenomeno può essere modellato come un operatore di *degradazione* D applicato a un'immagine y, considerata priva di errori, i.e.,

$$z = \mathcal{D}[y]. \tag{3.1}$$

In particolare, all'interno dell'operatore $\mathcal{D}$ si può considerare il contributo dovuto a un operatore di *sfocatura* B (dall'inglese *blur*) e un termine aleatorio η corrispondente al rumore, i.e.,

$$z(x) = \mathcal{D}[y](x) = \mathcal{B}[y](x) + \eta(x), \qquad x \in \mathcal{X} \tag{3.2}$$

dove abbiamo indicato con x le coordinate dei *pixel* dell'immagine e $\mathcal{X}$ è l'insieme dei pixel che formano l'immagine. Considerando il caso continuo, possiamo assumere la sfocatura $\mathcal{B}$ come un operatore *lineare*,

$$\mathcal{B}[y](x) = \int_{\mathcal{X}} y(s)h(x,s)ds, \tag{3.3}$$

dove $h(x,s)$ rappresenta la *risposta all'impulso* (*point spread function* (PSF)) della sfocatura sul pixel x, il cui risultato consiste nel rendere le differenze di intensità, tra pixel adiacenti, più morbide (*smooth*). Nel caso in cui la sfocatura sia applicata sulla totalità dell'immagine (come nel caso della figura 3.1(b)), allora è possibile modellare l'operatore di blur come una *convoluzione*[1]:

$$\mathcal{B}[y](x) = \int_{\mathcal{X}} y(s)h(s-x)ds, \tag{3.4}$$

dove $h(.)$ è un filtro gaussiano o uniforme.
Nel caso più generale possiamo considerare che la camera acquisisca un sequenza di N osservazioni $\{z_i\}, i = 1, \ldots, N$, quindi la formula 3.2 si può riscrivere come

$$z_i(x) = \mathcal{D}_i[y_i](x) = \mathcal{B}_i[y_i](x) + \eta(x), \qquad x \in \mathcal{X}. \tag{3.5}$$

La sequenza delle immagini $\{y_i\}, i = 1, \ldots, N$, può variare in maniera significativa nel suo contenuto, anche nel caso in cui la vista sia la stessa. Un esempio è illustrato nella figura 3.2, in cui le immagini riprese dalla camera sono acquisite ogni minuto. Nonostante l'inquadratura non cambi tra le acquisizioni, il contenuto delle singole immagini varia parecchio, dato il continuo passaggio di automobili e pedoni. Questo problema fa sì che l'identificazione delle sfocature non possa avvenire facendo un semplice confronto tra frame consecutivi, in quanto avremmo un numero troppo elevato di falsi positivi. Infatti, nel caso in cui avessimo un riscontro negativo dal confronto tra due frame ($z_i \neq z_{i+1}$), sarebbe difficile capire se è cambiato il contenuto delle immagini ($y_i \neq y_{i+1}$) o l'operatore di sfocatura ($\mathcal{B}_i \neq \mathcal{B}_{i+1}$).

[1] Il blur convoluzionale è quello che abbiamo utilizzato per generare, in maniera sintetica, sequenze con immagini sfocate.

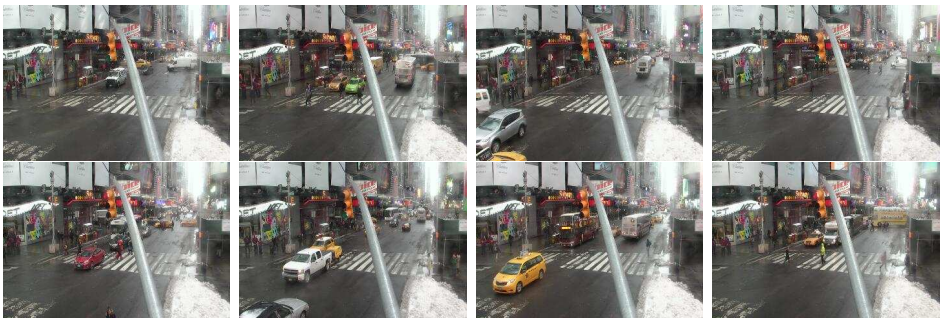
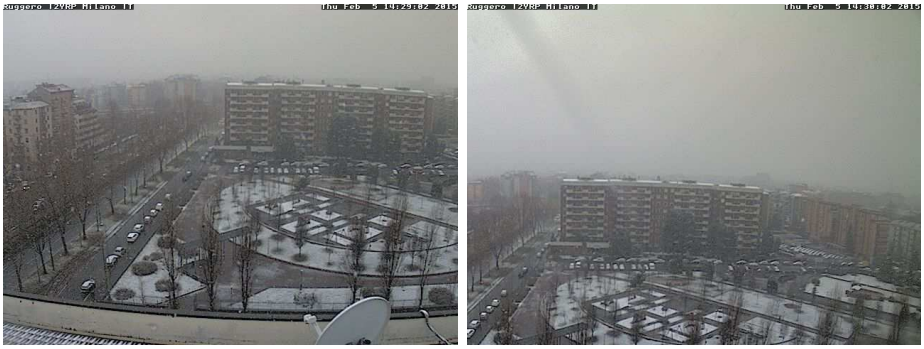


Figura 3.2: Sequenza di otto frame consecutivi acquisiti ogni minuto



(a) Vista originale

(b) Vista in seguito allo spostamento della camera

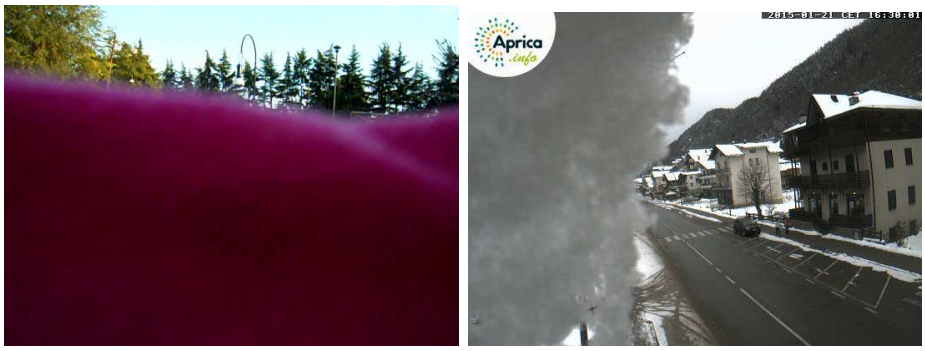
Figura 3.3: Esempio di spostamento della camera

3.1.3 Spostamento della camera

Lo spostamento della camera avviene quando cambia la sua vista. Le cause possono essere, ancora una volta, di tipo naturale, ad esempio una raffica di vento che sposta la camera, oppure dovute a un intervento malevolo da parte di un uomo. Un esempio di spostamento della camera è mostrato nella figura 3.3. Possiamo formalizzare il concetto di spostamento della camera nel modo seguente: consideriamo la sequenza $\{y_i\}$ di immagini generate da una camera in una certa posizione, e la sequenza $\{w_i\}$ di immagini generate dalla stessa camera in una posizione differente.
Possiamo, dunque, considerare la sequenza di immagini $\{z_i\}$ in cui avviene uno spostamento della camera all'istante T^* nel seguente modo:

$$z_i(x) = \mathcal{S}_i[y_i](x) = \begin{cases} y_i(x) + \eta(x) & \text{per} \quad i < T^* \\ w_i(x) + \eta(x) & \text{per} \quad i \geqslant T^* \end{cases}, \tag{3.6}$$

dove $\eta(x)$ è un rumore stazionario.
In questa formulazione abbiamo considerato lo spostamento come un fenomeno *istantaneo*; in generale, possiamo considerare una fase *transitoria* in



(a) Occlusione volontaria (b) Occlusione dovuta a deposito di neve

Figura 3.4: Esempi di occlusione

cui l'inquadratura della camera cambia a ogni frame acquisito, fino a raggiungere la posizione finale. Dato che, nella nostra applicazione, la camera opera con framerate bassi (come ad esempio un'immagine al minuto), possiamo considerare lo spostamento come istantaneo e, quindi, tenere come riferimento il modello della formula 3.6.
Anche per lo spostamento della camera vale la considerazione fatta nel caso della sfocatura: il contenuto delle immagini varia con il passare del tempo, quindi identificare lo spostamento confrontando frame consecutivi nel tempo genera un alto numero di falsi positivi.

3.1.4 Occlusione e guasti della camera

Il fenomeno dell'occlusione avviene quando un oggetto opaco si pone a ridosso della camera, impedendo la visione di una parte, se non la totalità, della scena. L'effetto di questo evento è quello di impedire l'acquisizione della scena nella sua totalità. Un esempio di occlusione ce l'abbiamo quando qualcuno copre la camera con un cappello, o mette un foglio di giornale davanti alla lente del sensore, oppure può succedere che della neve si accumuli sull'ottica del sensore. Il guasto del sensore invece comporta un aumento del rumore presente nell'immagine.
Queste problemi non sono stati affrontati durante la tesi, comunque le tecniche messe a punto per rilevare spostamenti e sfocature possono essere utilizzate anche per rilevare questo tipo di eventi.

3.2 Tampering detection

L'algoritmo di tampering detection consiste nell'analizzare la sequenza di osservazioni $\{z_i\}, i = 1, \ldots, N$ in modo da determinare un possibile cambiamento nell'operatore di degradazione $\mathcal{D}$ o in quello di spostamento $\mathcal{S}$. Come abbiamo illustrato precedentemente, discriminare tra cambiamenti avvenuti

a livello di contenuto della scena e cambiamenti dovuti a tampering può diventare complicato confrontando direttamente i frame tra loro.

Capitolo 4

Soluzione proposta

Capitolo 5

Realizzazioni sperimentali e valutazione

5.1 Acquisizione dei dataset

5.2 Risultati

Capitolo 6

Direzioni future di ricerca e conclusioni

Bibliografia

[1] `http://www.mitan.it/security-solution/videosorveglianza/sistemi-di-videosorveglianza-e-registrazione/`. Visitato il giorno 09/03/2015.

[2] Anil Aksay, Alptekin Temizel, and A. Enis Cetin. Camera tamper detection using wavelet analysis for video surveillance. In *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pages 558–562. IEEE, 2007.

[3] Cesare Alippi, Giacomo Boracchi, Romolo Camplani, and Manuel Roveri. Detecting external disturbances on the camera lens in wireless multimedia sensor networks. *Instrumentation and Measurement, IEEE Transactions on*, 59(11):2982–2990, 2010.

[4] Damian Ellwart, Piotr Szczuko, and Andrzej Czyżewski. Camera sabotage detection for surveillance systems. In *Security and Intelligent Information Systems*, pages 45–53. Springer, 2012.

[5] Pedro Gil-Jiménez, R. López-Sastre, Philip Siegmann, Javier Acevedo-Rodríguez, and Saturnino Maldonado-Bascón. Automatic control of video surveillance camera sabotage. *Nature Inspired Problem-Solving Methods in Knowledge Engineering*, pages 222–231, 2007.

[6] Sebastien Harasse, Laurent Bonnaud, Alice Caplier, and Michel Desvignes. Automated camera dysfunctions detection. In *Image Analysis and Interpretation, 2004. 6th IEEE Southwest Symposium on*, pages 36–40. IEEE, 2004.

[7] Ali Saglam and Alptekin Temizel. Real-time adaptive camera tamper detection for video surveillance. In *Advanced Video and Signal Based Surveillance, 2009. AVSS'09. Sixth IEEE International Conference on*, pages 430–435. IEEE, 2009.

[8] Theodore Tsesmelis, Lars Christensen, Preben Fihl, and Thomas B Moeslund. Tamper detection for active surveillance systems. In *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*, pages 57–62. IEEE, 2013.