PhD in Information Technology and Electrical Engineering

Università degli Studi di Napoli Federico II

PhD Student: Luca D'Amiano

XXX Cycle

Training and Research Activities Report – Third Year

Tutor: prof. Giovanni Poggi

1. Information

   Luca D'Amiano, MSc in Ingegneria delle Telecomunicazioni – Università di Napoli Federico II

   XXX Cycle - ITEE – Università di Napoli Federico II

   Tutor: Giovanni Poggi

2. Study and Training activities

   a. Courses

      -

   b. Seminars

      -

   c. External courses

      i. Workshop on Deep Learning for Visual Computing, Milano, 21/11/2016 (1 CFU)
      ii. IEEE EURASIP S3P 2017, Summer School on Signal Processing, "Signal Processing meets Deep Learning", Capri, 4-8/9/2017 (4 CFU)

						1	2	3	4	5	6			
	Estimated	Summary	Estimated	Summary	Estimated	bimonth	bimonth	bimonth	bimonth	bimonth	bimonth	Summary	Total	Check
Modules	18	19	9	10	12	0	0	0	0	0	4	**4**	**33**	**30-70**
Seminars	13	5,2	6	5,2	6	1	0	0	0	0	0	**1**	**11,4**	**10-30**
Research	34	36	42	42	45	9	10	10	10	10	8	**58**	**136**	**80-140**
	65	60,2	57	57,2	63	10	10	10	10	10	13	**63**	**180,4**	**180**

3. Research activity

Copy-Move Video forgery detection

Le attività svolte durante questo terzo anno di dottorato riguardanti la video forgery detection sono principalmente:

- Revisione dell'algoritmo di matching multirisoluzione
- Implementazione di una tecnica di post-processing per la rimozione di falsi allarmi.

Di seguito vengono brevemente descritti gli step principali dell'algoritmo. Segue poi una descrizione più dettagliata delle tecniche sopra elencate.

Step principali dell'algoritmo

L'obiettivo dell'algoritmo di detection è l'individuazione delle forgery di tipo comy-move nei video digitali. Come ogni algoritmo di copy-move detection è possibile decomporlo in tre step princiali:

- estrazione delle features
- matching
- post processing

Il primo passo consiste nel calcolare i descrittori locali del video. Per l'algoritmo proposto si è scelto di utilizzare i *Momenti di Zernike* che sono una specializzazione delle features calcolate tramite Circular Harmonic Transform (CHT). Di seguto viene riportata la formula per il calcolo dei *Momenti di Zernike:*

$$f(n,m) = \int_0^{\infty} \rho R^*_{n,m}(\rho) \times \left[\frac{1}{\sqrt{2\pi}} \int_0^{2\pi} I(\rho,\theta) e^{-jm\theta} d\theta \right] d\rho \quad (1)$$

$$R_{n,m}(\rho) = \begin{cases} \sqrt{C_n} \sum_{h=0}^{\frac{n-|m|}{2}} \frac{(-1)^h (n-h)!}{h!\left(\frac{n+|m|}{2}-h\right)!\left(\frac{n-|m|}{2}-h\right)!} \rho^{n-2h}, & \rho \leq 1 \\ 0, & otherwise \end{cases} \quad (2)$$

$$n - |m| \text{ is nonnegative and even}$$

In eq.(1) è mostrata la forma generale per il calcolo della CHT, mentre in eq.(2) è riportata l'espressione dei Momenti di Zernike.
Le fetures utilizzate garantiscono importanti proprietà di invarianza. In particolare, essendo una classe di CHT, se si considera il modulo delle features, esse risultano invarianti per rotazione. La scelta dei momenti di Zernike, invece, garantisce robustezza rispetto a piccole distorsioni dovute, ad esempio, ad aggiunta di rumore o compressione.
Queste features vengono calcolate frame per frame; il passo finale che conduce al calcolo dei descrittori utilizzati nella fase di matching consiste nel concatenare le features precedentemente calcolate secondo la seguente formula:

$$g(t) = \begin{cases} \frac{1}{\sqrt{2}} |f(t) + f(-t)| & t > 0 \\ |f(t)| & t = 0 \\ \frac{1}{\sqrt{2}} |f(t) - f(-t)| & t < 0 \end{cases}$$

Questo particolare metodo di concatenazione garantisce l'invarianza per flip-temporali.

Nella seconda fase dell'algoritmo viene calcolato il campo di nearest-neighbor (NN). Il NN indica, per ogni vettore di features, il match migliore. In formule:

$$\mathrm{NN}(s) = \arg\min_{s' \in \Omega} D(f(s), f(s'))$$

Dove Ω è il supporto del video, s è una generica posizione spazio temporale del video e D è una funzione che calcola la distanza fra la coppia di vettori di features.

Per il calcolo del NN si è stato implementato un algoritmo di matching che sarà descritto nel paragrafo successivo.

L'ultima fase dell'algoritmo di detection consiste nel post-processing, ovvero, utilizzando il NN calcolato al passo precedente, vengono individuate le regioni clonate del video. Nell'algoritmo di detection proposto questa fase è suddivisa in due sottofasi. La prima sottofase fornisce una prima versione della mappa di detection che verrà poi usata nella seconda sottofase per il calcolo della mappa finale.
La prima versione della mappa viene calcolata sogliando l'errore di fitting lineare. Date le caratteristiche del video, la mappa che si ottiene da questa sogliatura risulta essere affetta da una grande quantità di falsi allarmi. Per il calcolo della mappa finale viene effettuata una tecnica tesa all'eliminazione dei falsi allarmi. Questa tecnica sarà descritta nel paragrafo successivo.

Matching tramite algoritmo multirisoluzione

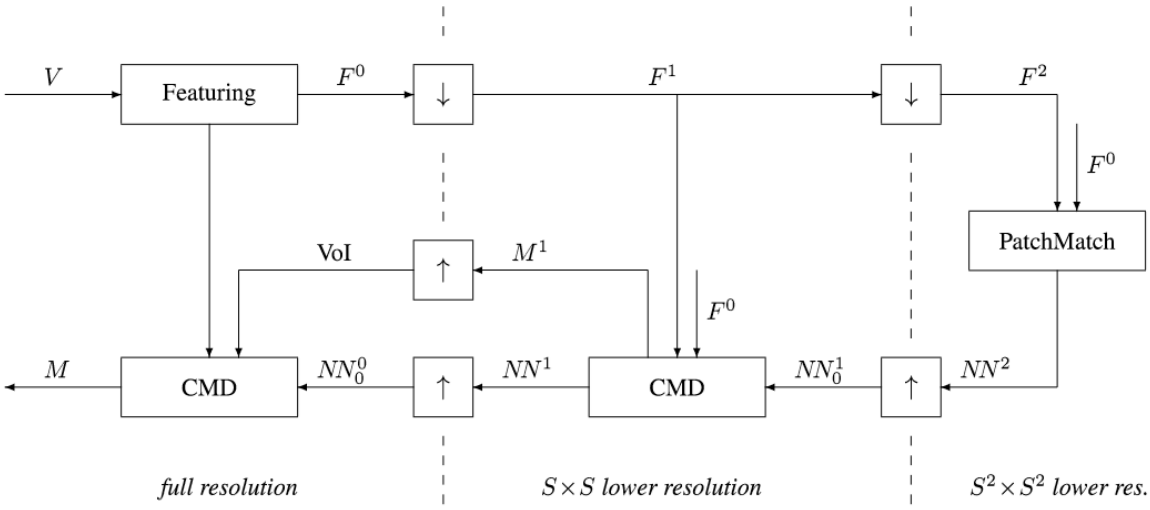

Figura 1: schema a blocchi dell'algoritmo proposto

In Fig. 1 è presentato lo schema a blocchi dell'algoritmo proposto.
L'algoritmo presenta una struttura a multirisoluzione. A valle dell'estrazione delle features si effettuano due sottocampionamenti del video. Sfruttando le caratteristiche dell'algoritmo di matching base (PatchMach con propagazione di ordine 1), è possibile effettuare il calcolo del NN nel dominio a più bassa risoluzione, utilizzando come campo di ricerca le features a massima risoluzione. Il NN così calcolato viene sovracampionato e utilizzato come dato per l'inizializzazione dell'algoritmo di matching a risoluzione intermedia. Anche in questo caso il matching viene effettuato utilizzando come campo di ricerca le features a massima risoluzione. Nell'ultimo step, con il NN calcolato a risoluzione intermedia si effettua una prima detection; se vengono rilevati volumi di interesse, viene effettuato un ultimo matching a massima risoluzione. In figura 2 è riportato lo pseudo-codice dell'algoritmo descritto.

Algorithm 1 Multi-resolution Video Copy-Move Detector

Require: V ▷ input video
Ensure: M ▷ output detection map
1: $F^0 = \text{FeatureExtract}(V)$ ▷ will work on features from now on
2: $F^1 = F^0 \downarrow S$ ▷ $S \times S$ downsampling
3: $F^2 = F^1 \downarrow S$ ▷ $S \times S$ downsampling
4: $NN^2 = \text{PatchMatch}(F^2, F^0)$ ▷ NN field at level 2
5: $NN_0^1 = NN^2 \uparrow S$ ▷ initial estimate of NN^1
6: $[M^1, NN^1] = \text{CMD}(F^1, F^0, NN_0^1)$ ▷ CMD at level 1
7: $M_0^0 = M^1 \uparrow S$ ▷ M_0^0 gives the VoI
8: $NN_0^0 = NN^1 \uparrow S$ ▷ initial estimate of NN^0
9: $[M, NN^0] = \text{CMD}(F^0, NN_0^0, \text{VOI})$ ▷ CMD at level 0 on VoI

Figura 2: pseudo-codice dell'algoritmo proposto

Il NN così calcolato viene utilizzato nel post-processing per il calcolo della mappa di detection finale.

Tecnica di Rimozione dei Falsi Allarmi

La prima versione della mappa di detection calcolata tramite la sogliatura dell'errore di fitting lineare, insieme al NN calcolato nella fase di matching, vengono utilizzati per il calcolo della mappa di detection finale.
Data la presenza di numerose strutture ridondanti, la prima versione della mappa di detection è affetta da numerosi falsi allarmi. La tecnica di rimozione di falsi allarmi sfrutta una peculiarità delle mappe di detection dei copy-move: se un vi è un clone nel video, la mappa di detection evidenzierà sia la regione falsificata che la regione sorgente da cui si è copiato. Sfruttando questa caratteristica, l'algoritmo proposto elimina dalla mappa di detection tutte le regioni che non hanno questa doppia corrispondenza. In figura 3 viene mostrato schematicamente l'approccio appena descritto: i rettangoli A, A' e B sono le regioni evidenziate dalla mappa di detection. Solo fra le regioni A e A' vi è una doppia corrispondenza; alla regione B invece corrisponde la regione C, che non è una zona evidenziata dalla mappa, e quindi viene eliminata.

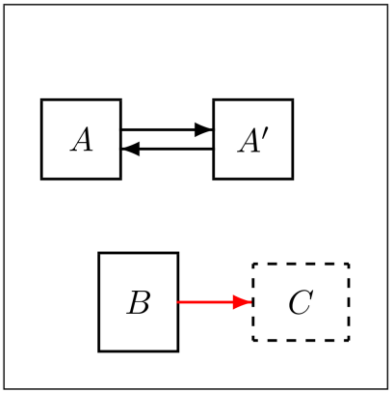

Figura 3: tecnica di rimozione dei falsi allarmi

In figura 4 vengono mostrate alcune mappe di detection calcolate su video reali. Le regioni verdi indicano le detection corrette, mentre le regioni rosse indicano i falsi allarmi presenti nella mappa. Risulta evidente la

riduzione dei falsi allarmi nelle mappe calcolate con la tecnica appena descritta (seconda riga) rispetto alle mappe calcolate utilizzando solo l'errore di fitting lineare (prima riga).

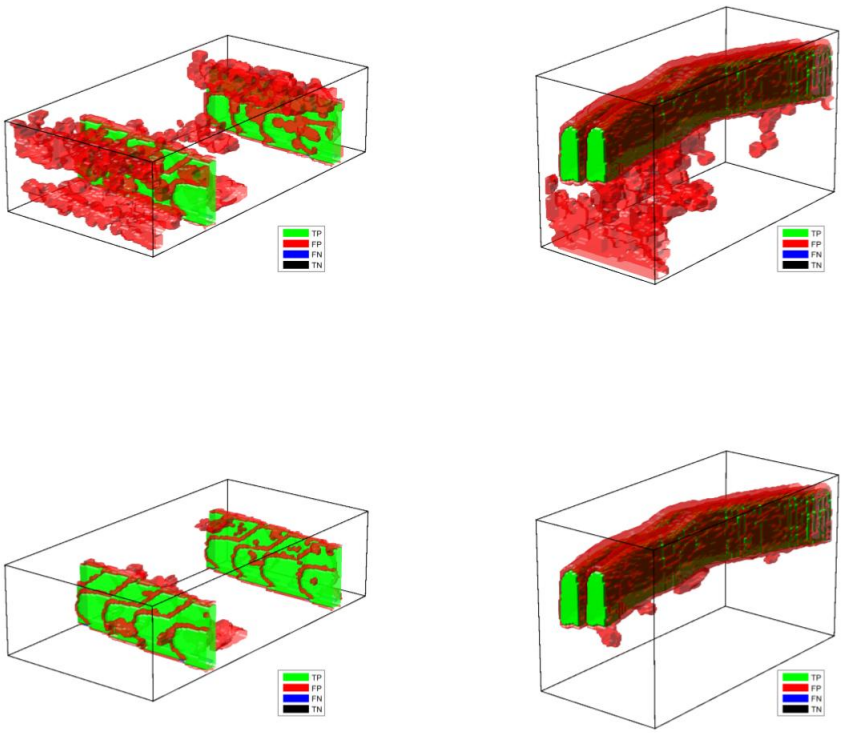

Figura 4: Confronto fra mappe di detection 3D

Patch-Based Optical-Driven SAR Despeckling

Nell'ambito del progetto PRIN mi sono occupato dell'implementazione di una tecnica di despeckling per immagini SAR. La tecnica si basa su un filtraggio nonlocal guidato da un'immagine ottica pilota co-registrata con l'immagine SAR. La tecnica è composta dai tre step usuali degli algoritmi di filtraggio nonlocal, ovvero, per ogni patch:

- **matching:** viene individuato un set di N patch simili alla patch target
- **filtering:** attraverso una media pesata si ottiene una stima della patch target
- **aggregation:** le stime che si sovrappongono vengono aggregate, pesando ogni contributo in base alla propria affidabilità.

La fase di matching è basata su una misura di distanza fra le patch definita come segue:

$$d(t,s) = \alpha d_S(t,s) + (1-\alpha) d_O(t,s)$$

Dove *t* e *s* sono i pixel di riferimento delle due patch da comparare, d_S e d_O sono rispettivamente la distanza calcolate per le patch dell'immagine SAR e la distanza calcolata per le patch dell'immagine ottica e $\alpha \in [0,1]$.

Nella fase di filtraggio la stima della patch viene fatta come segue:

$$\widehat{u}(t+\delta) = \sum_{i=1}^{N_p} w_i \cdot v(s_i+\delta) \quad \forall \delta \in \mathbb{P}$$

Dove v è l'intensità dell'immagine SAR e i pesi del filtraggio vengono calcolati come segue:

$$w_i = C \exp\left(-\lambda_S d_S(t,s_i) - \lambda_O d_O(t,s_i)\right)$$

Con *C* costante di normalizzazione.

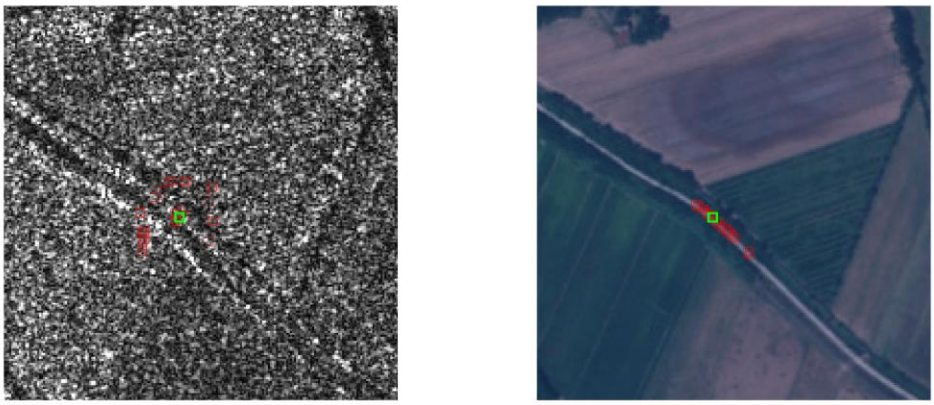

Figura 5: Immagine SAR (sinistra) e ottica (destra). Nella fase di matching per ogni target patch (verde) vengono calcolate le N patch più simili (rosse)

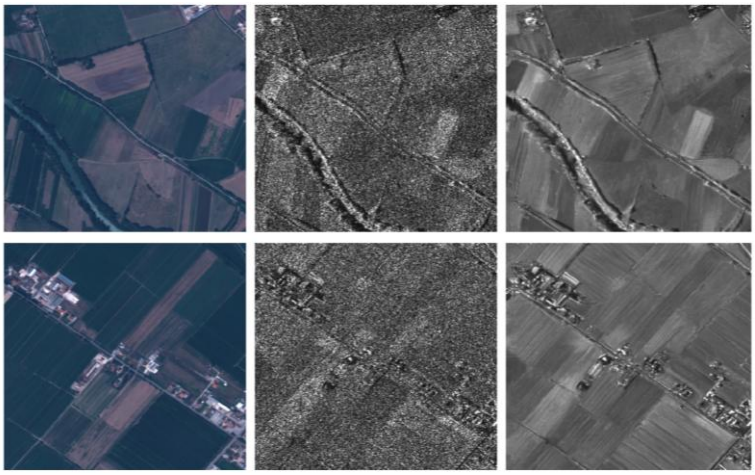

Figura 5: Da sinistra a destra: immagine ottica, immagine SAR, immagine filtrata usando la tecnica proposta.

Products

Conference

- D'Amiano, L., Cozzolino, D., Poggi G., & Verdoliva, L. (2015, June). "Video forgery detection and localizzation based on 3D patchmatch". Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on (pp.1-6) IEEE.
- Gaetano, R., Cozzolino, D., D'Amiano, L., Verdoliva, L., Poggi G. "FUSION OF SAR-OPTICAL DATA FOR LAND COVER MONITORING". International Geoscience and Remote Sensing Symposium (IGARSS), July 2017.

Journal

- D'Amiano, L., Cozzolino, D., Poggi G., & Verdoliva, L. "A PatchMatch-based Dense-field Algorithm for Video Copy-Move Detection and Localization". IEEE Transactions on Circuits and Systems for Video Technology (in review)

4. Conferences and Seminars
   -

5. Activity abroad
   -

6. Tutorship
   -