PROVA SCRITTA DI ELABORAZIONE DEI SEGNALI MULTIMEDIALI del 16.07.12
(Ingegneria delle Telecomunicazioni)
Tempo: 2 ore e mezza. NON è consentito l'uso di materiale didattico e appunti propri.

EX. 1 Si vuole implementare lo *switching median filter*, una versione più efficiente del filtro mediano che realizza il filtraggio dell'immagine rumorosa limitandosi ad elaborare solo i pixel che effettivamente possono essere considerati come rumore. In particolare, il filtro per ogni pixel genera la maschera binaria $z(i,j)$ per stabilire se un pixel va o meno elaborato come:

$$z(i,j) = \begin{cases} 1 & |m(i,j) - x(i,j)| > T \\ 0 & \text{altrimenti} \end{cases}$$

dove $m(i,j)$ è il valore mediano nella finestra scorrevole $k \times k$ e $x(i,j)$ il valore del pixel centrale. Scrivete una function dal prototipo `function y = smf(x,k,T)` in cui realizzate il filtro descritto.

Realizzate un esperimento applicando rumore sale e pepe all'immagine lena.jpg con $p = 0.2$ al variare di $k = 3, 5, 7, 9, 11$ valutando per ogni k empiricamente il valore di T. Infine confrontate, mediante un grafico, l'algoritmo con il filtro mediano standard in termini di PSNR al variare di k, mostrando le immagini filtrate con i due approcci per $k = 5$.

EX. 2 Il calcolo del *block artifact grid (BAG)* (griglia degli artefatti dovuti all'algoritmo di compressione JPEG) permette in modo abbastanza semplice di scoprire se un'immagine è stata manipolata mediante operazioni del tipo copy-and-paste. L'algoritmo è tanto più efficiente quanto più è basso il fattore di qualità. Nel file BAG.y (240×360, float) è contenuto il BAG dell'immagine aerei.jpg, che va ulteriormente elaborato per scoprire dove è stata effettuata la manomissione. In particolare, la detection prevede che per ogni blocco 8×8 distinto del BAG si valuti il seguente parametro e lo si assegni a tutti i valori del blocco:

$$b = \max\{\sum_{m=2}^{7} BAG(m,n) | 2 \le n \le 7\} - \max\{\sum_{m=2}^{7} BAG(m,n) | n = 1,8\} +$$
$$\max\{\sum_{n=2}^{7} BAG(m,n) | 2 \le m \le 7\} - \max\{\sum_{n=2}^{7} BAG(m,n) | m = 1,8\}$$

Una volta ottenuta l'immagine elaborata in questo modo, rendetela binaria attravero un'operazione di thresholding e applicate un'opportuna operazione morfologica per eliminare pixel spuri sparsi. La mappa binaria così ottenuta vi dovrebbe permettere di individuare facilmente l'area che è stata ritoccata, verificate il risultato osservando l'immagine originale contenuto in aerei_originale.jpg.

EX. 3 (6 CFU) Si consideri il file video akiyo_016_cif.y (288×352, uchar), dopo aver prelevato la prima e la sedicesima frame, scrivete una funzione `mvf = crossSearch(cur, ref)` che effettua la stima del movimento utilizzando la SSE come criterio da minimizzare con strategia di ricerca cross-search. I blocchi devono avere dimensione 16×16 e i vettori devono avere componenti comprese tra -15 e 15. La tecnica consiste nel cercare prima la componente orizzontale e poi quella verticale del vettore di movimento. In altre parole, si valuta prima l'indice di riga r_{min} tale che il vettore $(r_{min}, 0)$ abbia la minima SSD tra i vettori $(r, 0)$ nella finestra di ricerca; poi si valuta c_{min} tale che il vettore (r_{min}, c_{min}) minimizzi la SSD rispetto a tutti gli altri vettori (r_{min}, c). Utilizzate la funzione displayMVF.m fornita per visualizzare i vettori calcolati e confrontateli con quelli ottenuti tramite ricerca full-search, il cui codice è contenuto nel file me.m.

EX. 3 (9 CFU) Data l'immagine binaria testo.y (256×512, uint8), scrivete nello script ex3.m il codice matlab per calcolare e confrontare tra loro l'entropia del I ordine con l'entropia lungo blocchi 1×2, 2×1 e 2×2.