

LỘC NHIỄU

Trần Công Toại

Trường Đại Học Kỹ Thuật

(Bài nhận ngày 04/03/1998)

TÓM TẮT: Lộc nhiễu trong xử lý ảnh số là một trong những hướng nghiên cứu được nhiều nhà khoa học trên thế giới quan tâm. Ngay tại Việt Nam, trong thời gian gần đây đã xuất hiện một số nhóm nghiên cứu về vấn đề này..

Bằng các lý thuyết kết hợp với một số phương pháp thực nghiệm, đề tài đưa ra một số phương pháp:

- Mô phỏng, tạo nhiễu trên ảnh đen trắng cùng các giải thuật như tạo nhiễu xung, nhiễu cộng phân bố laplace, nhiễu nhân phân bố gauss.
- Các phương pháp lọc ảnh như lọc trung bình, lọc thông thấp, lọc median, lọc midpoint, lọc MMSE ..

Và đây là một số nội dung được trình bày tóm tắt về vấn đề trên.

1. MÔ PHỎNG NHIỄU TRÊN CÁC ẢNH ĐEN TRẮNG.

1.1 Nhiễu trên các ảnh đen trắng:

Ảnh số thường bị tác động bởi nhiễu trong quá trình thu nhận ảnh (chụp ảnh) hoặc trong quá trình truyền ảnh. Nhiễu phát sinh khi thu nhận ảnh có thể phụ thuộc loại nhiễu quang điện (do cảm biến quang điện gây ra) hoặc thuộc loại nhiễu hạt trên phim khi chụp ảnh. Các loại nhiễu này hoàn toàn phụ thuộc vào tín hiệu ảnh.

Giả sử $f(x,y)$ là ảnh gốc được ghi lên phim dương bản (phim slide) và $g(x,y)$ là ảnh quan sát được khi chiếu $f(x,y)$ lên màn ảnh, $g(x,y)$ chính là kết quả của 1 phép toán trên $f(x,y)$ cộng thêm với tác động của nhiễu $n(x,y)$:

$$g(x,y) = c (f(x,y) - \gamma n(x,y)) \quad (1)$$

Còn đối với các bộ cảm biến quang điện, nhiễu quang điện phụ thuộc vào tín hiệu ảnh như sau:

$$g(x,y) = c_2 (b(x,y))^y + (c_2 (b(x,y))^y)^{1/2} n(x,y) \quad (2)$$

trong đó $b(x,y)$ là ảnh đưa vào bộ cảm biến và $g(x,y)$ là ảnh thu nhận được. Ngoài ra các bộ cảm biến quang điện cũng còn tạo ra nhiễu nhiệt $n_t(x,y)$ dưới dạng cộng Gauss có trị trung bình bằng 0. Do đó:

$$g(x,y) = c_2 (h(x,y)) \otimes f(x,y) + (c_2 (h(x,y))^y)^{1/2} n(x,y) + n_t(x,y) \quad (3)$$

trong đó $f(x,y)$ là ảnh gốc, $h(x,y)$ là đáp ứng xung của hệ thống quang lọc.

Một dạng nhiễu khác được hình thành khi truyền ảnh, gọi là nhiễu xung (peak pulse) còn được gọi là nhiễu muối tiêu (salt-pepper noise) hay nhiễu nhị phân (binary noise). Nhiễu xung, dù thuộc nhiễu khí quyển hay nhiễu nhân tạo, xuất hiện như các xung đen và trắng trên ảnh, được mô phỏng bởi:

$$g(x,y) = \begin{cases} z(x,y) & \text{với xác suất } p \\ f(x,y) & \text{với xác suất } 1-p \end{cases} \quad (4)$$

Trong đó nhiễu xung $z(x,y)$ xuất hiện với xác suất p . Giá trị của nhiễu xung có thể cố định (thí dụ như 0 và 255) hoặc có thể theo dạng phân bố đuôi dài như phân bố Laplace.

1.2 Tạo nhiễu:

Nhiễu có thể được lọc bỏ bởi một số phép toán lọc sẽ được trình bày trong phần này. Để mô phỏng quá trình lọc nhiễu, phải tạo ra nhiễu dưới nhiều dạng, dưới nhiều mức độ khác nhau. Ngoài ra, đưa nhiễu vào ảnh cũng là một cách tạo hiệu quả đặc biệt cho ảnh, mô phỏng ảnh như là đã được chụp với phim độ nhạy cao (1000 ASA chẳng hạn) hoặc như là đã được phóng to lên rất nhiều lần. Có thể tạo nhiễu dưới 3 dạng tiêu biểu:

- Nhiễu cộng: $g(x,y) = f(x,y) + n(x,y)$ (5)

- Nhiễu nhân: $g(x,y) = f(x,y) + f(x,y).n(x,y)$ (6)

- Nhiễu xung: theo (4)

Trong đó giá trị nhiễu có thể thuộc loại phân bố đều, phân bố Gauss hay phân bố Laplace. Để triển khai các giải thuật tạo nhiễu (GT 1, GT 2, GT 3), chỉ cần tạo ra các số ngẫu nhiên phân bố đều. Từ phân bố đều, dễ dàng suy ra các số ngẫu nhiên theo phân bố Gauss hoặc Laplace.

Thật vậy, giả sử x là biến ngẫu nhiên có hàm mật độ xác suất $f_x(x)$, nếu $y = g(x)$ là 1 phép toán biến đổi điểm thì hàm mật độ xác suất của y là:

$$f_y(y) = f_x(x_1) |dg(x_1)/dx| + \dots + |f_x(x_n) / dg(x_n)/dx| + \dots$$
 (7)

trong đó x_1, \dots, x_n, \dots là các nghiệm thực của phương trình $y = g(x)$. Trong nhiều trường hợp, phương trình này chỉ có 1 nghiệm duy nhất. Nếu biết trước $f_x(x)$ và $f_y(y)$, ta sẽ suy ra được hàm biến đổi $y = g(x)$ dựa theo (7).

Nếu như $f_x(x)$ là phân bố đều trong khoảng $[0,1]$ và $f_y(y)$ là phân bố Gauss chuẩn có giá trị trung bình bằng 0 và phương sai đơn vị, tức là:

$$f_y(y) = (1/\sqrt{2\pi}).\exp(-x^2)$$
 (8)

thì hàm biến đổi $y = g(x)$ sẽ được cho bởi:

$$y = g(x) = \sqrt{-2 \ln x \cos(2\phi)}$$
 (9)

hoặc $y = g(x) = \sqrt{-2 \ln x \sin(2\phi)}$ (10)

trong đó ϕ phân bố đều trên $(-\pi, \pi]$.

Tương tự, ta có hàm biến đổi từ phân bố đều trong khoảng $[0,1]$ sang phân bố Laplace:

$$y = \begin{cases} \ln(2x) & 0 \leq x \leq 1/2 \\ -\ln(2 - 2x) & 1/2 < x < 1 \end{cases}$$
 (11)

Với phân bố Laplace có dạng

$$f_y(x,y) = (1/2).\exp(-|y|)$$
 (12)

GT.1: Giải thuật tạo nhiễu xung.

Bước 1: Hiển thị ảnh vào đen trắng

Bước 2: Nhập xác suất hiện nhiễu p và giá trị lớn nhất $imax$ và nhỏ nhất $imin$ của nhiễu

Bước 3: Sao chép ảnh vào sang ảnh ra

Bước 4: Chọn ngẫu nhiên $100.p\%$ tổng số các cặp tọa độ (x,y) của ảnh). Thay thế các pixel ảnh ra tại các tọa độ đó bằng giá trị $imax$ hoặc $imin$.

Bước 5: Hiển thị ra để nhận xét đánh giá kết quả.

GT.2: Giải thuật tạo nhiễu cộng phân bố Laplace.

Bước 1: Hiển thị ảnh đen và trắng.

Bước 2: Nhập biên độ nhiễu A

Bước 3: Đối với mọi pixel (x,y) trên ảnh vào:

- Tạo nhiễu phân bố Laplace bằng cách nhân kết quả (11) với A.
- Thêm nhiễu vào theo (5) để tạo ảnh ra.

Bước 4: Hiển thị ảnh để nhận xét, đánh giá kết quả.;

GT.3: Giải thuật nhiễu nhân phân bố Gauss.

Bước 1: Hiển thị ảnh vào đen trắng

Bước 2: Nhập biên độ nhiễu A.

Bước 3: Đối với mọi pixel (x,y) trên ảnh vào:

- Tạo nhiễu phân bố Gauss bằng cách nhân kết quả (9) hoặc (10) với A.
- Thêm nhiễu vào ảnh theo (6) để tạo ảnh ra.

Bước 4: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

2. Một số phương pháp lọc ảnh trị nhiễu :

2.1 Lọc trung bình (Averaging).

Trong phép lọc trung bình không gian, giá trị pixel ở tâm cửa sổ sẽ được thay thế bởi trung bình cộng giá trị các pixel trong cửa sổ, nghĩa là:

$$g(x,y) = (1/N) \sum_{k,l \in w} f(x-k,y-l) \quad (13)$$

Trong đó N là số lượng pixel trong cửa sổ W. Theo (13), các hệ số của mặt nạ đều bằng 1/n. Kích thước mặt nạ thường được chọn là $n \times n = 3,5,7 \dots$

1	1	1		1/9	1/9		1/9
1	1	1		1/9	hoặc		1/9
1	1	1		1/9	1/9		1/9
1	1	1		1/9	1/9		1/9

Hình 1: Mặt nạ 3 x 3 dùng cho lọc trung bình

- Tạo nhiễu phân bố Gauss bằng cách nhân kết quả (9) hoặc (10) với A.
- Thêm nhiễu vào ảnh vào theo (6) để tạo ảnh ra.

Bước 4: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

2 Lọc trung bình (Averaging).

Trong phép lọc trung bình không gian, giá trị pixel ở tâm cửa sổ sẽ được thay thế bởi trung bình cộng giá trị các pixel trong cửa sổ, nghĩa là:

$$g(x,y) = (1/N) \sum_{k,l \in w} f(x-k,y-l) \quad (13)$$

Trong đó N là số lượng pixel trong cửa sổ W. Theo (13), các hệ số của mặt nạ đều bằng 1/N. Kích thước mặt nạ thường được chọn là $n \times n = 3,5,7 \dots$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \cdot (1/9) \text{ hoặc } \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Hình 1: Mặt nạ đề 3 x 3 dùng cho lọc trung bình

Người ta thường dùng lọc trung bình để làm dịu bớt ảnh hưởng của nhiễu đồng thời lọc lấy phần tần số thấp của ảnh, tạo cảm giác mờ nhòe. Thật vậy, giả sử ảnh $f(x,y)$ bị tác động bởi nhiễu cộng $n(x,y)$, theo (5), nghĩa là:

$$f(x,y) + n(x,y) \quad (14)$$

với $n(x,y)$ giả sử là nhiễu trắng Gauss có trị trung bình bằng 0 và phương sai σ^2 . Sau khi lọc trung bình, ta có:

$$g(x,y) = (1/N) \sum_{k,l} f(x-k,y-l) + n^*(x,y) \quad (15)$$

với $n^*(x,y)$ là kết quả lọc trung bình của $n(x,y)$, cũng có trị trung bình bằng 0 và phương sai σ^2/N , nghĩa là công suất nhiễu bị giảm đi N lần. Nếu ảnh không nhiễu $f(x,y)$ có giá trị không đổi trong phạm vi cửa sổ, thì $g(x,y)$ không đổi và do đó tỉ số tín hiệu trên tạp âm (SNR) tăng N lần. Thực tế $g(x,y)$ chỉ không đổi đối với các thành phần tần số thấp trên ảnh và thay đổi đối với các thành phần tần số cao, tạo cảm giác mờ nhòe. Kích thước cửa sổ (hay kích thước mặt nạ) nếu chọn càng lớn, cảm giác mờ nhòe càng tăng. GT.4 là giải thuật lọc trung bình ảnh đen trắng với mặt nạ kích thước $m \times n$.

GT.4 Giải thuật lọc trung bình.

Bước 1: Hiển thị ảnh vào đen trắng.

Bước 2: Nhập kích thước mặt nạ $m \times n$.

Bước 3: Biến đổi mọi giá trị pixel (x,y) trên ảnh vào (trừ một số pixel ở biên ảnh) theo (13) để tạo ra ảnh.

Bước 4: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

2.2 Lọc thấp qua không gian (Spatial LPF).

Để lọc nhiễu và làm mờ hình ảnh, cũng có thể dùng mạch lọc thấp qua (lpf) trong miền tần số. Có thể chuyển lọc LPF về miền không gian với đáp ứng xung tương ứng. Dựa vào đáp ứng xung này, chúng ta xây dựng được các mặt nạ lọc LPF không gian có dạng đối xứng đều với tâm có hệ số dương lớn, các hệ số xung quanh có hệ số dương nhỏ và tổng các hệ số sẽ đúng bằng 1 như hình 2 (a) và (b). Dạng hình chuông của đáp ứng xung càng nhọn tức là sai biệt giữa hệ số giữa và các hệ số xung quanh càng lớn, hình càng ít bị mờ nhòe. Nếu hình dạng chuông này quá lồi, lọc LPF gần như trở thành lọc trung bình.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} (1/10) \quad \text{(a)} \quad \begin{bmatrix} 1 & 2 & 1 \\ 1 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} (1/16) \quad \text{(b)}$$

Hình 2: Mặt nạ lọc thấp qua (LPF) không gian.

Phương pháp lọc thấp qua không gian như vậy được xác định qua tích chập :

$$g(x,y) = \sum_{k,l \in W} h(k,l) \cdot f(x-k, y-l) \quad (16)$$

với $h(k,l)$ là các hệ số của mặt nạ LPF. Giải thuật lọc LPF không gian ảnh đen trắng có dạng tương tự GT4, nhưng dùng phép biến đổi (16) và mặt nạ lọc cho bởi hình 2.

2.3 Lọc median (Median Filter).

Lọc median là loại lọc thấp quan không gian phi tuyến rất hiệu quả để lọc bỏ nhiễu xung, nhiễu phân bố Laplace; nhưng vẫn giữ lại được các cạnh, các đường biên của vật thể, cũng như các chi tiết nhỏ trên ảnh. Trong phép lọc median, giá trị pixel ở tâm cửa sổ sẽ được thay thế bằng giá trị giữa (giá trị median) trong số các giá trị pixel thuộc cửa sổ đó, nghĩa là :

$$g(x,y) = \text{median}\{f(x-k, y-l), (k,l) \in W\} \quad (17)$$

Lọc median là 1 công cụ rất hữu hiệu để lọc bỏ các pixel cô lập mà vẫn duy trì được độ phân giải không gian như ảnh gốc. Tuy nhiên lọc median không hiệu quả bằng lọc trung bình khi lọc nhiễu cộng Gauss và tác dụng lọc của lọc median cũng sẽ không hiệu quả khi pixel bị nhiễu trong phạm vi 1 cửa sổ lớn hoặc hơn bằng phân nửa tổng số pixel trong cửa sổ đó.

GT.5: Giải thuật lọc median căn bản.

Bước 1: Hiển thị ảnh vào đen trắng.

Bước 2: Nhập kích thước mặt nạ $m \times n$.

Bước 3: Ứng dụng từng pixel (x,y) trên ảnh vào (trừ một số pixel ở biên ảnh) :

- Sắp xếp giá trị các pixel trong phạm vi cửa sổ tâm (x,y) theo thứ tự lớn dần hoặc nhỏ dần.

- Lấy giá trị giữa (giá trị median) để tạo ảnh ra.

Bước 4: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

Lưu ý rằng lọc median, cũng như mọi phép toán lọc không gian khác, là *phép toán chạy* (running operator), nghĩa là trong quá tính toán tạo ảnh ra, tâm (x,y) của cửa sổ di chuyển khắp mặt phẳng ảnh. Như vậy 2 cửa sổ kế cận nhau, có tâm (x,y) và $(x+1, y)$ chẳng hạn sẽ có phần chồng lặp lẫn nhau. Cụ thể, khi cửa sổ kích thước $m \times n$ tâm (x,y) di chuyển 1 pixel qua vị trí $(x+1, y)$, $m \cdot n - n$ pixel bị loại bỏ và được thay thế bằng n pixel mới. Giải thuật của Huang vận dụng đặc điểm vừa nêu để lọc median với tốc độ nhanh hơn rất nhiều. GT6 là bản cải biên của giải thuật lọc median này.

GT.6 Giải thuật lọc median chạy:

Bước 1: Hiển thị ảnh vào đen trắng.

Bước 2: Nhập kích thước mặt nạ $m \times n$

Bước 3: Tính histogram nội bộ của $n \cdot m$ pixel trong cửa sổ đầu tiên và tìm giá trị median của cửa sổ đó. Tính số lượng pixel N có mức xám nhỏ hơn giá trị median.

Bước 4: Di chuyển qua cửa sổ kế tiếp bằng cách loại bỏ n pixel và thêm vào n pixel mới. Hiệu chỉnh lại histogram. Đếm số lượng pixel N có mức xám nhỏ hơn giá trị median ở bước 3.

Bước 5: Bắt đầu từ giá trị median của cửa sổ trước đó, di chuyển lên (hoặc xuống) trên thang xám nếu N nhỏ (hoặc lớn hơn) giá trị $(mn + 1)/2$ và hiệu chỉnh N cho đến khi đạt được giá trị median.

Bước 6: Lặp lại từ bước 4 cho đến cuối dòng

Bước 7: Lặp lại từ bước 3 cho đến dòng cuối của ảnh.

Bước 8: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

Giải thuật GT6 cần $2m$ phép so sánh khi tính 1 giá trị pixel của ảnh ra, trong khi giải thuật GT5 với các sắp xếp thứ tự loại Quicksort thông thường cũng phải cần $2m^2 \log m$ phép so sánh nếu $n \approx m$, do đó GT6 thực hiện nhanh hơn GT5 rất nhiều.

2.4 Lọc midpoint (Midpoint Filter).

Như đã đề cập ở trên, lọc trung bình có khả năng triệt nhiễu cộng Gauss tốt hơn lọc median, trong khi lọc median lại giữ được các cạnh, các đường biên của các vật thể và loại bỏ nhiễu xung tốt hơn lọc trung bình. Dạng tổng quát của cả 2 loại lọc trên được gọi là lọc trung bình hiệu chỉnh theo α (α - trimmed mean filter):

$$g(x, y) = (1 / N(1 - 2\alpha)) \cdot \sum_{j=\alpha N + 1}^{N - \alpha N} f(j) \quad (18)$$

trong đó, $j=1,2 \dots N$ là giá trị các pixel trong cửa sổ tâm (x,y) được xếp thứ tự từ nhỏ đến lớn. (18) có khuynh hướng loại bỏ các mức xám lớn nhất và nhỏ nhất, rồi lấy trung bình cộng các mức xám còn lại. Số lượng các mức xám lớn nhất và nhỏ nhất bị loại bỏ phụ thuộc vào α , $0 \leq \alpha \leq 0,5$. Nếu $\alpha = 0$, không có mức xám nào bị bỏ bớt, do đó (18) chính là lọc trung bình. Còn nếu α tiến gần đến 0,5 tất cả các mức xám lớn, nhỏ đều bị loại bỏ ngoại trừ mức xám median, như vậy (18) trở thành lọc median. Lọc trung bình hiệu chỉnh theo α kết hợp được các ưu điểm của cả lọc trung bình lẫn lọc median, tuy nhiên loại lọc này vẫn tỏ ra kém hữu hiệu khi lọc nhiễu có dạng phân bố đều. Trong trường hợp này lại cần dùng phép toán lọc midpoint có đặc điểm trái ngược lại: lấy trung bình cộng giá trị mức xám lớn nhất và nhỏ nhất:

$$g(x, y) = \left[f_{(1)} + f_{(N)} \right] / 2 \\ = (1 / 2) \left[\max_{(k,1) \in W} \{f(x - k, y - 1)\} + \min_{(k,1) \in W} \{f(x - k, y - 1)\} \right] \quad (19)$$

Ngoài ra dùng lọc nhiễu, lọc midpoint cũng có tác dụng lọc mờ vì cũng là một loại lọc thấp qua không gian phi tuyến. Giải thuật midpoint căn bản tương tự như GT5, nhưng thay vì tính giá trị median, chúng ta tính trung bình cộng giá trị lớn nhất và nhỏ nhất theo (19) để tạo ra ảnh

2.5 Lọc tối thiểu sai số trung bình bình phương thích nghi.

(Adaptive Minimal Mean Squara Error Filter)

Các phép toán lọc phi tuyến nêu trên chỉ đạt hiệu quả tối ưu đối với từng loại nhiễu, và đôi lúc, đối với từng loại tín hiệu ảnh cụ thể. Hình ảnh thường được mô phỏng như 1 quá trình ngẫu nhiên không dừng, có các giá trị trung bình, độ lệch chuẩn ... thay đổi theo từng

vùng trên ảnh. Hơn nữa, độ lệch chuẩn của nhiễu, cũng như hàm mật độ xác suất của nhiễu cũng thay đổi từ ứng dụng này sang ứng dụng khác. Đôi khi trong cùng ứng dụng, nhưng đặc tính của nhiễu cũng lại thay đổi từ ảnh này sang ảnh khác. Vì ít khi biết trước được các đặc tính của ảnh cũng như đặc tính của nhiễu, nên các mạch lọc không thích nghi nêu trên thường tỏ ra không thích hợp cho các trường hợp tổng quát trong thực tế.

Mạch lọc thích nghi, với khả năng xác định gần đúng trị trung bình và độ lệch chuẩn của tín hiệu ảnh, độ lệch chuẩn của nhiễu trên 1 cửa sổ, sẽ suy ra được giá trị xấp xỉ gần đúng của ảnh không nhiễu. Giả sử ảnh bị nhiễu $g(x,y)$ được tạo bởi ảnh không nhiễu $f(x,y)$ và nhiễu cộng $n(x,y)$ theo (5) nghĩa là :

$$g(x,y) = f(x,y) + n(x,y) \quad (20)$$

Xấp xỉ gần đúng của $f(x,y)$ với sai số trung bình bình phương tối thiểu cho bởi:

$$\hat{f}(x,y) = (1 - \sigma_n^2 / \sigma_g^2) g(x,y) + (\sigma_n^2 / \sigma_g^2) m^g \quad (21)$$

Trong đó σ_n , σ_n , m^g là xấp xỉ gần đúng độ lệch chuẩn của tín hiệu ảnh và trị trung bình của tín hiệu ảnh. Trong những mảng ảnh lớn tương đối thuần nhất, $\sigma_n \approx \sigma_g$, nên theo (21), $\hat{f}(x,y) \approx m^g$. Tại các cạnh vật thể $\sigma_g \gg \sigma_n$, mạch lọc hầu như không tác dụng $\hat{f}(x,y) \approx g(x,y)$. Như vậy lọc MMSE thích nghi sẽ lọc mờ các mảng ảnh tương đối thuần nhất để loại nhiễu, giữ lại các cạnh, các đường viền của vật thể trên ảnh.

GT7 : Giải thuật tạo lọc MMSE thích nghi

Bước 1: Hiển thị ảnh vào đen trắng

Bước 2: Nhập độ lệch chuẩn của nhiễu và kích thước mặt nạ $m \times n$

Bước 3: Ứng dụng từng pixel (x,y) trên ảnh vào (trừ một số pixel ở biên ảnh):

- Tính trị trung bình và độ lệch chuẩn nội bộ của các giá trị pixel trong cửa sổ tâm (x,y) .

- Tính $g(x,y)$ theo (21) để tạo ảnh ra.

Bước 4: Hiển thị ảnh ra để nhận xét, đánh giá kết quả.

3. KẾT LUẬN

Đối với một hình ảnh thu nhận được có khi không có nhiễu, có khi có nhiễu, thậm chí trên cùng một bức ảnh có thể tồn tại nhiều loại nhiễu khác nhau. Mỗi một phương pháp lọc nhiễu chỉ có hiệu quả đối với một vài loại nhiễu nhất định. Vấn đề là làm thế nào để đánh giá được nhiễu đó là nhiễu gì để chọn phương pháp lọc nhiễu thích hợp. Một hình ảnh có khi ít chi tiết, có khi nhiều chi tiết, đôi lúc các chi tiết trên hình ảnh phân bố không đều, để đảm bảo về mặt thời gian lọc. Việc chọn cửa sổ lọc nhiễu, phương pháp lọc nhiễu là cả một vấn đềNội dung bài viết này một phần nào giải đáp một số vấn đề liên quan đến các suy nghĩ trên.

Hy vọng nhận được nhiều đóng góp của quý thầy cô, các bạn nhằm vấn đề đưa ra được phong phú và ngày càng hoàn thiện hơn.

NOISE FILTER

Tran Cong Toai

ABSTRACT: Digital image processing has a broad spectrum of applications, such as remote sensing via satellites and other spaceserafts, image transmission and storage for business applications, nudical processing, radar, sonar and acoustic image processing, robotics and automated in spection of industrial parts.

Noise filter includes gray level and contrast manipulation, noise reproduction, edge crispening and sharpening, filetering, interpolation and magnification, pseudocoloring, and so on.

In this article we only mention image restoration and image enhancement (noise smoothing, median filtering, low-pass, bandpass, high-pass filtering, edge detector).

TÀI LIỆU THAM KHẢO

[1] Maher A. Sid-Ahmed

Image Processing . Theory, Algorithms, And Architectures
International Editions 1995

[2] A. Murat Jekalp - University Of Rochester

Digital Video Processing

Prentice- Hall, Inc - 1995

[3] Jae S.Lim

Two-Dimensional Signal And Image Processing

Prentice-Hall internatimal editions 1996