|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***Лабораторна робота №* 1**  ***Тема:*** алгоритмічна компенсація випадкових похибок інформаційних сигналів у вимірювальному каналу  ***Мета роботи:*** отримати навики обробки випадкових похибок інформаційних сигналів у вимірювальному каналі за допомогою програми MATLAB.  **1.1. *Основні теоретичні відомості***  Отримання повних апріорних відомостей про вимірювальну інформацію є складною задачею, так як в загальному випадку початкова вимірювальна інформація (двовимірний масив 𝑓0(𝑥, 𝑦)) невідома, а для вимірювань доступне тільки спотворене відеозображення 𝑓*Н*(𝑛, 𝑚) на виході ПФВЗ. Тому для отримання апріорної інформації про вимірювальну інформацію пропонується використовувати методики, що розроблені в підрозділі 2.4.  Для оцінки ефективності застосування методів алгоритмічної компенсації похибок потрібно визначити критерії, за якими оцінюються результати компенсації. Відомі типові критерії оцінки вірності передачі та відтворення аналогових і цифрових відеозображень. Ці критерії орієнтовані на оцінку амплітудних похибок та візуальної якості відеозображень [161, 162, 216, 310, 315, 316]. Вказані критерії обчислюють середньоквадратичне значення похибки відтворення амплітуди відеосигналу на виході вимірювального каналу. Це є загальна оцінка похибок всього відеозображення. При цьому похибка відтворення кожного дискретного значення амплітуди відеосигналу розглядається незалежно від похибок відтворення сусідніх дискретних значень. Такий підхід дозволяє з високою точністю дати загальну оцінку похибок яскравості і кольору ОВ, що наявні на сформованому відеозображенні.  Однак, при оцінці похибок відеозображень з вимірювальною інформацією про ГП необхідно в першу чергу враховувати викривлення форми перепадів амплітуди відеосигналу, що відповідають контурам ОВ. Оскільки в амплітудних критеріях вказані викривлення не враховуються безпосередньо, то оцінка похибок відеозображень з вимірювальною інформацією на основі амплітудних критеріїв має низьку точність та об'єктивність. Окрім того, кількісну оцінку похибок ГП доцільно виражати в одиницях просторових координат відеозображення, а не в одиницях амплітуди відеосигналу. | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *3* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **1*.2. Виконання роботи***  Примітки: 1. Канал *Y* відповідає яскравості кольорового відеозображення, що обчислюється за формулою 𝑌 = 0,30𝑅 + 0,59𝐺 + 0,11𝐵, де 𝑅, 𝐺 і 𝐵 – відеосигнали в каналах червоного, зеленого та синього кольору.   1. Частота зрізу 𝜔10 визначається за умови |𝑊кв(𝑗𝜔1, 𝑗𝜔2)| = 0,5 при 𝜔2 = 0, частота зрізу 𝜔20 – за умови |𝑊кв(𝑗𝜔1, 𝑗𝜔2)| = 0,5 при 𝜔1 = 0. 2. \* – частота зрізу лежить за межами смуги перепускання 0 … 0,5 (д.т.)-1 для цифрового відеозображення, що представлено дискретними відліками.   1.2.1. Програма для виконання фільтрації.  clear all;  Dir\_Test='D:\My\_doc 02-11-10\MM\_3\Test\'; Dir\_Im='D:\My\_doc 02-11-10\MM\_3\CFA\_Natural\_Stone\'; FileMask='.bmp'; Type\_Real\_Im=1;  Ideal\_Im\_name='D:\My\_doc 02-11-10\MM\_3\Ideal\_Image\Ideal\_1982\_8s\_12s\_RGB.bmp'; M=1024; N=1024;  Y\_Background=27/255;  Y\_Figure=227/255;  Threshold\_Segm=(Y\_Background+Y\_Figure)/2; N\_Im=3;  N\_Frame\_Filt=10; Type\_Figure=2; Boun=10; Type\_Noise=1; AlfaX\_OV=0.700; NG=50;  L\_PK=12;  PSF\_Size=L\_PK+2; SNR=40; PDov=0.95;  Filtered\_Method=3; Filt\_M\_Rect=0.25; Filt\_N\_Rect=0.25;  Filt\_M\_Rect=round(Filt\_M\_Rect\*M); Filt\_N\_Rect=round(Filt\_N\_Rect\*N); Filt\_N\_Mask=3;  F\_Order\_1D=1;  Im\_Show=0;  Delta\_Show=1;  Im\_Write=0;  X\_Center=round(N/2); Y\_Center=round(M/2); RectXMax=round(M/sqrt(2)-2\*Boun-2\*L\_PK); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *4* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Варіант | Початкові дані для розрахунку | | | | | |
| Корисний сигнал вимірювальної інформації | | | Випадкові викривлення (шум) | | |
| Параметр кореляційної функції 𝛼𝑥, (д.т.)-1 | Параметр кореляційної функції  𝛼𝑦, (д.т.)-1 | Середньо- квадратичне значення ам- плітуди 𝜎𝑓2 ,  д.р. | Параметр кореляційної функції 𝛼𝑥𝜉, (д.т.)-1 | Параметр кореляційної функції 𝛼𝑦𝜉, (д.т.)-1 | Середньо- квадратичне значення ам- плітуди  𝜎𝜉, д.р. |
| *4* | 0,064 | 0,037 | 28,6 | 1,552 | 0,881 | 3,9 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | | |
|  | | |  | | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RectYMin=round((M/sqrt(2)-2\*Boun-2\*L\_PK)/10); EllipsAMax=round((M/sqrt(2)-2\*Boun-2\*L\_PK)/2); EllipsBMin=round((M/sqrt(2)-2\*Boun-2\*L\_PK)/2/3);  A\_SKO\_Im\_M=zeros(N\_Im,1); A\_SNR\_dB\_Im\_M=zeros(N\_Im,1); A\_PSNR\_dB\_Im\_M=zeros(N\_Im,1); A\_SKO\_Im\_R=zeros(N\_Im,1); A\_SNR\_dB\_Im\_R=zeros(N\_Im,1); A\_PSNR\_dB\_Im\_R=zeros(N\_Im,1); B\_M\_XOR=zeros(N\_Im,1); B\_R\_XOR=zeros(N\_Im,1);  S\_Figure=zeros(N\_Im,1); S\_Figure\_M=zeros(N\_Im,1); S\_Figure\_R=zeros(N\_Im,1); Centroid\_X=zeros(N\_Im,1); Centroid\_X\_M=zeros(N\_Im,1); Centroid\_X\_R=zeros(N\_Im,1); Centroid\_Y=zeros(N\_Im,1); Centroid\_Y\_M=zeros(N\_Im,1); Centroid\_Y\_R=zeros(N\_Im,1); W\_Figure=zeros(N\_Im,1); W\_Figure\_M=zeros(N\_Im,1); W\_Figure\_R=zeros(N\_Im,1); H\_Figure=zeros(N\_Im,1); H\_Figure\_M=zeros(N\_Im,1); H\_Figure\_R=zeros(N\_Im,1); ED\_Figure=zeros(N\_Im,1); ED\_Figure\_M=zeros(N\_Im,1); ED\_Figure\_R=zeros(N\_Im,1);  % rand('state',sum(100\*clock)); randn('state',sum(100\*clock)); rand('state',0); randn('state',0);  ListFileInDir=dir(Dir\_Im); NumberOfFileInDir=length(ListFileInDir(:,1)); NumberOfFileToMask=0;  ListFileToMask=[];  for i=1:NumberOfFileInDir,  if strfind(ListFileInDir(i).name, upper(FileMask))>=1 NumberOfFileToMask=NumberOfFileToMask+1; ListFileToMask(NumberOfFileToMask).name=ListFileInDir(i).name;  end;  if strfind(ListFileInDir(i).name, lower(FileMask))>=1 NumberOfFileToMask=NumberOfFileToMask+1; ListFileToMask(NumberOfFileToMask).name=ListFileInDir(i).name;  end; end;  h\_waitbar = waitbar(0,'Obrobka zobrazhen'); for i\_Im=1:N\_Im waitbar(i\_Im/N\_Im,h\_waitbar);  Im\_Frame=zeros(M,N);  Im\_Frame\_M=zeros(M,N);  Im\_Frame\_R=zeros(M,N);  switch Type\_Figure case 1  RectX=round(RectXMax-(RectXMax-RectYMin)\*rand); RectY=round(RectXMax-(RectXMax-RectYMin)\*rand); RectTheta=180\*rand;  Im\_Frame\_J=zeros(RectY, RectX)+Y\_Figure;  Im\_Frame\_J = imrotate(Im\_Frame\_J,RectTheta,'bicubic','loose'); [M\_J, N\_J]=size(Im\_Frame\_J);  Im\_Frame(round(Y\_Center-M\_J/2+1):round(Y\_Center+M\_J/2), round(X\_Center- N\_J/2+1):round(X\_Center+N\_J/2))=Im\_Frame\_J;  Im\_Frame(Im\_Frame==0)=Y\_Background; Im\_Frame=mat2gray(Im\_Frame, [0 1]);  case 2  EllipsA=round(EllipsAMax-(EllipsAMax-EllipsBMin)\*rand); EllipsB=round(EllipsAMax-(EllipsAMax-EllipsBMin)\*rand); EllipsTheta=180\*rand;  Im\_Frame\_J=zeros(EllipsB\*2, EllipsA\*2); for j=1:EllipsB\*2  for i=1:EllipsA\*2  if abs(i-EllipsA)<=EllipsA\*sqrt(1-(EllipsB-j)^2/EllipsB^2) Im\_Frame\_J(j,i)=Y\_Figure;  end  end  end  Im\_Frame\_J = imrotate(Im\_Frame\_J,EllipsTheta,'bicubic','loose'); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *5* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| [M\_J, N\_J]=size(Im\_Frame\_J);  Im\_Frame(round(Y\_Center-M\_J/2+1):round(Y\_Center+M\_J/2), round(X\_Center- N\_J/2+1):round(X\_Center+N\_J/2))=Im\_Frame\_J;  Im\_Frame(Im\_Frame<Y\_Background)=Y\_Background; Im\_Frame=mat2gray(Im\_Frame, [0 1]);  case 3  if i\_Im<=NumberOfFileToMask Im\_Frame=rgb2gray(im2double(imread(Ideal\_Im\_name)));  Im\_Frame\_M=rgb2gray(im2double(imread(strcat(Dir\_Im,ListFileToMask(i\_Im).name)))); M=size(Im\_Frame,1); N=size(Im\_Frame,2);  Im\_Frame\_R=zeros(M,N); else  N\_Im=i\_Im-1; break; end;  end; % switch Type\_Figure  if Type\_Figure<3 PSF\_DVC=fspecial('gaussian',PSF\_Size,L\_PK/6); Im\_Frame=imfilter(Im\_Frame, PSF\_DVC, 'circular'); switch Type\_Noise  case 0  Disp\_Gaussian=1/10^(2\*SNR/20); Im\_Frame\_M=imnoise(Im\_Frame,'gaussian',0,Disp\_Gaussian);  case 1  gamma1=rand(NG,1); gamma2=rand(NG,1); gamma4=rand(NG,1)\*2\*pi; psbp=rand(NG,1);  for lll=1:NG  if psbp(lll)>=0.5 psbp(lll)=1; else psbp(lll)=-1; end end;  ksi\_k=psbp.\*sqrt(-log(gamma1)); e1=cos(gamma4); e2=sin(gamma4); % îðò x îðò y eta\_mod=AlfaX\_OV.\*sqrt(1./gamma2.^2-1); eta1\_k=eta\_mod.\*e1; eta2\_k=eta\_mod.\*e2;  h2 = waitbar(0,strcat('Wait ... generated Im Noise ',num2str(i\_Im))); Im\_Noise=zeros(M,N);  for lll=1:NG  waitbar(lll/NG,h2); for jjj=1:M for iii=1:N  Im\_Noise(jjj,iii)=Im\_Noise(jjj,iii)+sqrt(2)\*ksi\_k(lll)\*sin(eta1\_k(lll)\*iii+eta2\_k(lll)\*j jj+pi/4);  end; end; end; close(h2);  Im\_Noise=1/10^(SNR/20)/sqrt(NG)\*Im\_Noise; Im\_Frame\_M=Im\_Frame+Im\_Noise;  end % switch Type\_Noise end; % if Type\_Figure<3  switch Filtered\_Method case 1  Snoise2=abs(fft2(Im\_Frame\_M-Im\_Frame, M\*2, N\*2)).^2; Simage2=abs(fft2(Im\_Frame, M\*2, N\*2)).^2; HF=zeros(M\*2,N\*2); HF=fftshift(Simage2./(Simage2+Snoise2)); h17=fspecial('average',17);  HF=imfilter(HF,h17); HF=zeros(M\*2,N\*2); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *6* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| HF((M-2\*Filt\_M\_Rect+1):(M+2\*Filt\_M\_Rect),(N 2\*Filt\_N\_Rect+1):(N+2\*Filt\_N\_Rect))=1; Fourie\_Im=fftshift(fft2(Im\_Frame\_M, M\*2, N\*2)); iFourie\_Im=real(ifft2(fftshift(HF.\*Fourie\_Im))); Im\_Frame\_R=iFourie\_Im(1:M, 1:N);  case 2  HF=zeros(M,N);  HF((round(M/2)-Filt\_M\_Rect+1):(round(M/2)+Filt\_M\_Rect),... (round(N/2)-Filt\_N\_Rect+1):(round(N/2)+Filt\_N\_Rect))=1;  [f1 f2]=freqspace([M N], 'meshgrid'); hf=real(fsamp2(f1,f2,HF,[Filt\_N\_Mask Filt\_N\_Mask])); hf=hf/sum(sum(hf)); Im\_Frame\_R=imfilter(Im\_Frame\_M,hf);  case 3  Snoise2=abs(fft2(Im\_Frame\_M-Im\_Frame, M\*2, N\*2)).^2; Simage2=abs(fft2(Im\_Frame, M\*2, N\*2)).^2; HF=zeros(M\*2,N\*2); HF=fftshift(Simage2./(Simage2+Snoise2)); h17=fspecial('average',17);  HF=imfilter(HF,h17); Fourie\_Im=fftshift(fft2(Im\_Frame\_M, M\*2, N\*2)); iFourie\_Im=real(ifft2(fftshift(HF.\*Fourie\_Im))); Im\_Frame\_R=iFourie\_Im(1:M, 1:N);  case 4  Snoise=abs(fft2(Im\_Frame\_M-Im\_Frame)).^2; Simage=abs(fft2(Im\_Frame)).^2; HF=zeros(M,N); HF=fftshift(Simage./(Simage+Snoise)); h17=fspecial('average',17); HF=imfilter(HF,h17);  [f1 f2]=freqspace([M N], 'meshgrid'); hf=real(fsamp2(f1,f2,HF,[Filt\_N\_Mask Filt\_N\_Mask])); hf=hf/sum(sum(hf)); Im\_Frame\_R=imfilter(Im\_Frame\_M,hf);  case 5  F\_1D=linspace(0,1,round(N/2)); HF\_1D=zeros(1,round(N/2)); HF\_1D(1:Filt\_N\_Rect)=1; [B\_1D,A\_1D]=yulewalk(F\_Order\_1D,F\_1D,HF\_1D); for jjj=1:M  Im\_Frame\_R(jjj,:) = filter(B\_1D, A\_1D, Im\_Frame\_M(jjj,:)); end;  case 6  F\_1D=linspace(0,1,round(N/2));  Snoise1=abs(fft(Im\_Frame\_M(round(M/2),:)-Im\_Frame(round(M/2),:), N)).^2; Simage1=abs(fft(Im\_Frame(round(M/2),:), N)).^2; HF\_1D=Simage1./(Simage1+Snoise1);  h17=fspecial('average',17); HF\_1D=imfilter(HF\_1D(1:round(N/2)),h17); [B\_1D,A\_1D]=yulewalk(F\_Order\_1D,F\_1D,HF\_1D); B\_1D=B\_1D\*sum(A\_1D)/sum(B\_1D);  for jjj=1:M  Im\_Frame\_R(jjj,:) = filter(B\_1D, A\_1D, Im\_Frame\_M(jjj,:)); end;  case 7  Im\_Frame\_R=zeros(M,N);  h1\_waitbar = waitbar(0,'Obrobka neruchomych zobrazhen'); if Type\_Figure<3  for iii=1:N\_Frame\_Filt waitbar(iii/N\_Frame\_Filt,h1\_waitbar); switch Type\_Noise | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *7* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| case 0  Disp\_Gaussian=1/10^(2\*SNR/20); Im\_Frame\_M=imnoise(Im\_Frame,'gaussian',0,Disp\_Gaussian);  case 1  gamma1=rand(NG,1); gamma2=rand(NG,1); gamma4=rand(NG,1)\*2\*pi; psbp=rand(NG,1);  for lll=1:NG  if psbp(lll)>=0.5 psbp(lll)=1; else psbp(lll)=-1; end; end;  ksi\_k=psbp.\*sqrt(-log(gamma1)); e1=cos(gamma4); e2=sin(gamma4); % îðò x îðò y eta\_mod=AlfaX\_OV.\*sqrt(1./gamma2.^2-1); eta1\_k=eta\_mod.\*e1; eta2\_k=eta\_mod.\*e2;  h2 = waitbar(0,  strcat('Wait ... generated Im Noise ',num2str(iii))); Im\_Noise=zeros(M,N);  for lll=1:NG  waitbar(lll/NG,h2);  for jjj=1:M for iii=1:N Im\_Noise(jjj,iii)=Im\_Noise(jjj,iii)+sqrt(2)\*ksi\_k(lll)  \*sin(eta1\_k(lll)\*iii+eta2\_k(lll)\*jjj+pi/4); end; end;  end; close(h2);  Im\_Noise=1/10^(SNR/20)/sqrt(NG)\*Im\_Noise; Im\_Frame\_M=Im\_Frame+Im\_Noise;  end; % switch Type\_Noise Im\_Frame\_R=Im\_Frame\_R+Im\_Frame\_M;  end; else  for iii=1:N\_Frame\_Filt waitbar(iii/N\_Frame\_Filt,h1\_waitbar); if iii<=NumberOfFileToMask  Im\_Frame\_M=rgb2gray(im2double(imread( strcat(Dir\_Im,ListFileToMask(iii).name))  ));  Im\_Frame\_R=Im\_Frame\_R+Im\_Frame\_M; else  N\_Frame\_Filt=iii-1; break; end;  end; end;  Im\_Frame\_R=Im\_Frame\_R/N\_Frame\_Filt; close(h1\_waitbar);  end; % switch Filtered\_Method  Pixel\_Value=zeros(M,N); Pixel\_Value\_M=zeros(M,N); Pixel\_Value\_R=zeros(M,N); if Y\_Figure<Threshold\_Segm  Pixel\_Value(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame(1+Boun:M-Boun,1+Boun:N- Boun), 0, Threshold\_Segm);  Pixel\_Value\_M(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame\_M(1+Boun:M- Boun,1+Boun:N-Boun), 0, Threshold\_Segm);  Pixel\_Value\_R(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame\_R(1+Boun:M- Boun,1+Boun:N-Boun), 0, Threshold\_Segm);  else  Pixel\_Value(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame(1+Boun:M-Boun,1+Boun:N- Boun), Threshold\_Segm, 1);  Pixel\_Value\_M(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame\_M(1+Boun:M- Boun,1+Boun:N-Boun), Threshold\_Segm, 1);  Pixel\_Value\_R(1+Boun:M-Boun,1+Boun:N-Boun)=roicolor(Im\_Frame\_R(1+Boun:M- Boun,1+Boun:N-Boun), Threshold\_Segm, 1); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *8* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| end;  [L num]=bwlabel(Pixel\_Value, 8);  [L\_M num\_M]=bwlabel(Pixel\_Value\_M, 8);  [L\_R num\_R]=bwlabel(Pixel\_Value\_R, 8);  feats=imfeature(L, 'Area', 'Centroid', 8); feats\_M=imfeature(L\_M, 'Area', 'Centroid', 8); feats\_R=imfeature(L\_R, 'Area', 'Centroid', 8);  Areas=zeros(num); CentX=zeros(num); CentY=zeros(num); Areas\_M=zeros(num\_M); CentX\_M=zeros(num\_M); CentY\_M=zeros(num\_M); Areas\_R=zeros(num\_R); CentX\_R=zeros(num\_R); CentY\_R=zeros(num\_R); for i=1:1:num  Areas(i)=feats(i).Area; CentX(i)=feats(i).Centroid(1); CentY(i)=feats(i).Centroid(2);  end;  for i=1:1:num\_M  Areas\_M(i)=feats\_M(i).Area; CentX\_M(i)=feats\_M(i).Centroid(1); CentY\_M(i)=feats\_M(i).Centroid(2);  end;  for i=1:1:num\_R  Areas\_R(i)=feats\_R(i).Area; CentX\_R(i)=feats\_R(i).Centroid(1); CentY\_R(i)=feats\_R(i).Centroid(2);  end;  [rows cols]=size(Pixel\_Value); idx=find(Areas>1000&CentX>50&CentY>50&CentX<(cols-50)&CentY<(rows-50));  idx\_M=find(Areas\_M>1000&CentX\_M>50&CentY\_M>50&CentX\_M<(cols-50)&CentY\_M<(rows-50)); idx\_R=find(Areas\_R>1000&CentX\_R>50&CentY\_R>50&CentX\_R<(cols-50)&CentY\_R<(rows-50)); Pixel\_Value=ismember(L, idx);  Pixel\_Value\_M=ismember(L\_M, idx\_M); Pixel\_Value\_R=ismember(L\_R, idx\_R);  [L num]=bwlabel(Pixel\_Value, 8);  [L\_M num\_M]=bwlabel(Pixel\_Value\_M, 8);  [L\_R num\_R]=bwlabel(Pixel\_Value\_R, 8);  feats=imfeature(L, 'Area', 'Centroid', 'BoundingBox', 'EquivDiameter', 8); feats\_M=imfeature(L\_M, 'Area', 'Centroid', 'BoundingBox', 'EquivDiameter', 8); feats\_R=imfeature(L\_R, 'Area', 'Centroid', 'BoundingBox', 'EquivDiameter', 8);  A\_SKO\_Im\_M(i\_Im)=255.0\*sqrt(sum(sum((Im\_Frame(Boun+1:M-Boun,Boun+1:N-Boun)- Im\_Frame\_M(Boun+1:M-Boun,Boun+1:N-Boun)).^2))/((M-2\*Boun)\*(N-2\*Boun))); A\_SKO\_Im\_R(i\_Im)=255.0\*sqrt(sum(sum((Im\_Frame(Boun+1:M-Boun,Boun+1:N-Boun)- Im\_Frame\_R(Boun+1:M-Boun,Boun+1:N-Boun)).^2))/((M-2\*Boun)\*(N-2\*Boun)));  A\_SNR\_dB\_Im\_M(i\_Im)=10\*log10(sum(sum(Im\_Frame(Boun+1:M-Boun,Boun+1:N- Boun).^2))/sum(sum((Im\_Frame(Boun+1:M-Boun,Boun+1:N-Boun)-Im\_Frame\_M(Boun+1:M- Boun,Boun+1:N-Boun)).^2))); A\_SNR\_dB\_Im\_R(i\_Im)=10\*log10(sum(sum(Im\_Frame(Boun+1:M-Boun,Boun+1:N- Boun).^2))/sum(sum((Im\_Frame(Boun+1:M-Boun,Boun+1:N-Boun)-Im\_Frame\_R(Boun+1:M- Boun,Boun+1:N-Boun)).^2)));  A\_PSNR\_dB\_Im\_M(i\_Im)=10\*log10(1.0\*(M-2\*Boun)\*(N-2\*Boun)/sum(sum((Im\_Frame(Boun+1:M- Boun,Boun+1:N-Boun)-Im\_Frame\_M(Boun+1:M-Boun,Boun+1:N-Boun)).^2))); A\_PSNR\_dB\_Im\_R(i\_Im)=10\*log10(1.0\*(M-2\*Boun)\*(N-2\*Boun)/sum(sum((Im\_Frame(Boun+1:M- Boun,Boun+1:N-Boun)-Im\_Frame\_R(Boun+1:M-Boun,Boun+1:N-Boun)).^2))); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *9* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S\_Figure(i\_Im)=feats.Area;  Centroid\_X(i\_Im)=feats.Centroid(1); Centroid\_Y(i\_Im)=feats.Centroid(2); W\_Figure(i\_Im)=feats.BoundingBox(3); H\_Figure(i\_Im)=feats.BoundingBox(4); S\_Figure\_M(i\_Im)=feats\_M.Area;  Centroid\_X\_M(i\_Im)=feats\_M.Centroid(1); Centroid\_Y\_M(i\_Im)=feats\_M.Centroid(2); W\_Figure\_M(i\_Im)=feats\_M.BoundingBox(3); H\_Figure\_M(i\_Im)=feats\_M.BoundingBox(4); S\_Figure\_R(i\_Im)=feats\_R.Area;  Centroid\_X\_R(i\_Im)=feats\_R.Centroid(1); Centroid\_Y\_R(i\_Im)=feats\_R.Centroid(2); W\_Figure\_R(i\_Im)=feats\_R.BoundingBox(3); H\_Figure\_R(i\_Im)=feats\_R.BoundingBox(4); ED\_Figure(i\_Im)=feats.EquivDiameter; ED\_Figure\_M(i\_Im)=feats\_M.EquivDiameter; ED\_Figure\_R(i\_Im)=feats\_R.EquivDiameter;  B\_M=xor(Pixel\_Value, Pixel\_Value\_M); B\_R=xor(Pixel\_Value, Pixel\_Value\_R); if Im\_Show==1  figure; subplot(2, 3, 1); imshow(Pixel\_Value); title('Segm. pochatkovogo zobrazhennya');  subplot(2, 3, 2); imshow(Pixel\_Value\_M); title('Segm. zobrazhennya z vykryvlennyamy');  subplot(2, 3, 5); imshow(Pixel\_Value\_R); title('Segm. filtr. zobrazhennya'); subplot(2, 3, 3); imshow(B\_M); title('Vidminnist pochatkovogo i vykr.'); subplot(2, 3, 6); imshow(B\_R); title('Vidminnist pochatkovogo i filtr.');  end;  B\_M\_XOR(i\_Im)=sum(sum(B\_M(Boun+1:M-Boun,Boun+1:N-Boun))); B\_R\_XOR(i\_Im)=sum(sum(B\_R(Boun+1:M-Boun,Boun+1:N-Boun)));  if Im\_Write==1 Name\_Frame=strcat(Dir\_Test,'Test\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Im\_Frame,Name\_Frame); Name\_Frame=strcat(Dir\_Test,'Test\_Noise\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Im\_Frame\_M,Name\_Frame); Name\_Frame=strcat(Dir\_Test,'Filtered\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Im\_Frame\_R,Name\_Frame); Name\_Frame=strcat(Dir\_Test,'Segm\_Test\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Pixel\_Value,Name\_Frame); Name\_Frame=strcat(Dir\_Test,'Segm\_Test\_Noise\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Pixel\_Value\_M,Name\_Frame); Name\_Frame=strcat(Dir\_Test,'Segm\_Filtered\_',num2str(i\_Im,'%d'),'.bmp'); imwrite(Pixel\_Value\_R,Name\_Frame);  end;  if Im\_Show==1  figure; subplot(2, 2, 1); imshow(Im\_Frame); title('Pochatkove zobrazhennya'); subplot(2, 2, 2); imshow(Im\_Frame\_M); title('Zobrazhennya z vykryvlennyamy'); subplot(2, 2, 3); imshow(Im\_Frame\_R); title('Filtrovane zobrazhennya'); subplot(2, 2, 4); imshow(Pixel\_Value\_R); title('Segm. filtr. zobrazhennya');  end;  end; % for i\_Im=1:N\_Im close(h\_waitbar);  if Delta\_Show==1  Max\_A\_SKO\_Im\_M=max(A\_SKO\_Im\_M); Max\_A\_SKO\_Im\_R=max(A\_SKO\_Im\_R); Mean\_A\_SKO\_Im\_M=mean(A\_SKO\_Im\_M); Mean\_A\_SKO\_Im\_R=mean(A\_SKO\_Im\_R); Min\_A\_SKO\_Im\_M=min(A\_SKO\_Im\_M); Min\_A\_SKO\_Im\_R=min(A\_SKO\_Im\_R); Delta\_Dov\_M=sort(A\_SKO\_Im\_M, 'descend'); Delta\_Dov\_R=sort(A\_SKO\_Im\_R, 'descend'); MaxDov\_A\_SKO\_Im\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_A\_SKO\_Im\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov))); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *10* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Max\_A\_SNR\_dB\_Im\_M=max(A\_SNR\_dB\_Im\_M); Max\_A\_SNR\_dB\_Im\_R=max(A\_SNR\_dB\_Im\_R); Mean\_A\_SNR\_dB\_Im\_M=mean(A\_SNR\_dB\_Im\_M); Mean\_A\_SNR\_dB\_Im\_R=mean(A\_SNR\_dB\_Im\_R); Min\_A\_SNR\_dB\_Im\_M=min(A\_SNR\_dB\_Im\_M); Min\_A\_SNR\_dB\_Im\_R=min(A\_SNR\_dB\_Im\_R); Delta\_Dov\_M=sort(A\_SNR\_dB\_Im\_M, 'descend'); Delta\_Dov\_R=sort(A\_SNR\_dB\_Im\_R,  'descend');  MaxDov\_A\_SNR\_dB\_Im\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_A\_SNR\_dB\_Im\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov)));  Max\_A\_PSNR\_dB\_Im\_M=max(A\_PSNR\_dB\_Im\_M); Max\_A\_PSNR\_dB\_Im\_R=max(A\_PSNR\_dB\_Im\_R); Mean\_A\_PSNR\_dB\_Im\_M=mean(A\_PSNR\_dB\_Im\_M); Mean\_A\_PSNR\_dB\_Im\_R=mean(A\_PSNR\_dB\_Im\_R); Min\_A\_PSNR\_dB\_Im\_M=min(A\_PSNR\_dB\_Im\_M); Min\_A\_PSNR\_dB\_Im\_R=min(A\_PSNR\_dB\_Im\_R); Delta\_Dov\_M=sort(A\_PSNR\_dB\_Im\_M, 'descend'); Delta\_Dov\_R=sort(A\_PSNR\_dB\_Im\_R,  'descend');  MaxDov\_A\_PSNR\_dB\_Im\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_A\_PSNR\_dB\_Im\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov)));  Delta\_S\_Figure\_M=(S\_Figure\_M-S\_Figure); D1\_S\_Figure\_M=Delta\_S\_Figure\_M.\*100./S\_Figure;  Delta\_S\_Figure\_R=(S\_Figure\_R-S\_Figure); D1\_S\_Figure\_R=Delta\_S\_Figure\_R.\*100./S\_Figure;  Delta\_Centroid\_X\_M=Centroid\_X\_M-Centroid\_X; D1\_Centroid\_X\_M=Delta\_Centroid\_X\_M.\*100./Centroid\_X; Delta\_Centroid\_X\_R=Centroid\_X\_R-Centroid\_X; D1\_Centroid\_X\_R=Delta\_Centroid\_X\_R.\*100./Centroid\_X; Delta\_Centroid\_Y\_M=Centroid\_Y\_M-Centroid\_Y; D1\_Centroid\_Y\_M=Delta\_Centroid\_Y\_M.\*100./Centroid\_Y; Delta\_Centroid\_Y\_R=Centroid\_Y\_R-Centroid\_Y; D1\_Centroid\_Y\_R=Delta\_Centroid\_Y\_R.\*100./Centroid\_Y;  Max\_Delta\_S\_Figure\_M=max(abs(Delta\_S\_Figure\_M)); Max\_Delta\_S\_Figure\_R=max(abs(Delta\_S\_Figure\_R));  Delta\_Dov\_M=sort(abs(Delta\_S\_Figure\_M), 'descend'); Delta\_Dov\_R=sort(abs(Delta\_S\_Figure\_R), 'descend');  MaxDov\_Delta\_S\_Figure\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_Delta\_S\_Figure\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov)));  Mean\_Delta\_S\_Figure\_M=mean(Delta\_S\_Figure\_M); Mean\_Delta\_S\_Figure\_R=mean(Delta\_S\_Figure\_R);  Std\_Delta\_S\_Figure\_M=std(Delta\_S\_Figure\_M,1); Std\_Delta\_S\_Figure\_R=std(Delta\_S\_Figure\_R,1);  Max\_Delta\_Centroid\_M=max(abs([Delta\_Centroid\_X\_M; Delta\_Centroid\_Y\_M])); Max\_Delta\_Centroid\_R=max(abs([Delta\_Centroid\_X\_R; Delta\_Centroid\_Y\_R])); Delta\_Dov\_M=sort(abs([Delta\_Centroid\_X\_M; Delta\_Centroid\_Y\_M]), 'descend'); Delta\_Dov\_R=sort(abs([Delta\_Centroid\_X\_R; Delta\_Centroid\_Y\_R]), 'descend'); MaxDov\_Delta\_Centroid\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov)));  MaxDov\_Delta\_Centroid\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov))); Mean\_Delta\_Centroid\_M=mean([Delta\_Centroid\_X\_M; Delta\_Centroid\_Y\_M]); Mean\_Delta\_Centroid\_R=mean([Delta\_Centroid\_X\_R; Delta\_Centroid\_Y\_R]); Std\_Delta\_Centroid\_M=std([Delta\_Centroid\_X\_M; Delta\_Centroid\_Y\_M],1); Std\_Delta\_Centroid\_R=std([Delta\_Centroid\_X\_R; Delta\_Centroid\_Y\_R]);  Max\_D1\_S\_Figure\_M=max(abs(D1\_S\_Figure\_M)); Max\_D1\_S\_Figure\_R=max(abs(D1\_S\_Figure\_R));  Delta\_Dov\_M=sort(abs(D1\_S\_Figure\_M), 'descend'); Delta\_Dov\_R=sort(abs(D1\_S\_Figure\_R), 'descend');  MaxDov\_D1\_S\_Figure\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_D1\_S\_Figure\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov)));  Mean\_D1\_S\_Figure\_M=mean(D1\_S\_Figure\_M); Mean\_D1\_S\_Figure\_R=mean(D1\_S\_Figure\_R); Std\_D1\_S\_Figure\_M=std(D1\_S\_Figure\_M,1); Std\_D1\_S\_Figure\_R=std(D1\_S\_Figure\_R,1); | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *11* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Max\_D1\_Centroid\_M=max(abs([D1\_Centroid\_X\_M; D1\_Centroid\_Y\_M])); Max\_D1\_Centroid\_R=max(abs([D1\_Centroid\_X\_R; D1\_Centroid\_Y\_R])); Delta\_Dov\_M=sort(abs([D1\_Centroid\_X\_M; D1\_Centroid\_Y\_M]), 'descend'); Delta\_Dov\_R=sort(abs([D1\_Centroid\_X\_R; D1\_Centroid\_Y\_R]), 'descend'); MaxDov\_D1\_Centroid\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov)));  MaxDov\_D1\_Centroid\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov))); Mean\_D1\_Centroid\_M=mean([D1\_Centroid\_X\_M; D1\_Centroid\_Y\_M]); Mean\_D1\_Centroid\_R=mean([D1\_Centroid\_X\_R; D1\_Centroid\_Y\_R]); Std\_D1\_Centroid\_M=std([D1\_Centroid\_X\_M; D1\_Centroid\_Y\_M],1); Std\_D1\_Centroid\_R=std([D1\_Centroid\_X\_R; D1\_Centroid\_Y\_R]);  Delta\_W\_Figure\_M=(W\_Figure\_M-W\_Figure); D1\_W\_Figure\_M=Delta\_W\_Figure\_M.\*100./W\_Figure;  Delta\_W\_Figure\_R=(W\_Figure\_R-W\_Figure); D1\_W\_Figure\_R=Delta\_W\_Figure\_R.\*100./W\_Figure;  Delta\_H\_Figure\_M=(H\_Figure\_M-H\_Figure); D1\_H\_Figure\_M=Delta\_H\_Figure\_M.\*100./H\_Figure;  Delta\_H\_Figure\_R=(H\_Figure\_R-H\_Figure); D1\_H\_Figure\_R=Delta\_H\_Figure\_R.\*100./H\_Figure;  Max\_Delta\_L\_Figure\_M=max(abs([Delta\_W\_Figure\_M; Delta\_H\_Figure\_M])); Max\_Delta\_L\_Figure\_R=max(abs([Delta\_W\_Figure\_R; Delta\_H\_Figure\_R])); Delta\_Dov\_M=sort(abs([Delta\_W\_Figure\_M; Delta\_H\_Figure\_M]), 'descend'); Delta\_Dov\_R=sort(abs([Delta\_W\_Figure\_R; Delta\_H\_Figure\_R]), 'descend'); MaxDov\_Delta\_L\_Figure\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov)));  MaxDov\_Delta\_L\_Figure\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov))); Mean\_Delta\_L\_Figure\_M=mean([Delta\_W\_Figure\_M; Delta\_H\_Figure\_M]); Mean\_Delta\_L\_Figure\_R=mean([Delta\_W\_Figure\_R; Delta\_H\_Figure\_R]); Std\_Delta\_L\_Figure\_M=std([Delta\_W\_Figure\_M; Delta\_H\_Figure\_M],1); Std\_Delta\_L\_Figure\_R=std([Delta\_W\_Figure\_R; Delta\_H\_Figure\_R],1);  Max\_D1\_L\_Figure\_M=max(abs([D1\_W\_Figure\_M; D1\_H\_Figure\_M])); Max\_D1\_L\_Figure\_R=max(abs([D1\_W\_Figure\_R; D1\_H\_Figure\_R])); Delta\_Dov\_M=sort(abs([D1\_W\_Figure\_M; D1\_H\_Figure\_M]), 'descend'); Delta\_Dov\_R=sort(abs([D1\_W\_Figure\_R; D1\_H\_Figure\_R]), 'descend'); MaxDov\_D1\_L\_Figure\_M=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov)));  MaxDov\_D1\_L\_Figure\_R=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov))); Mean\_D1\_L\_Figure\_M=mean([D1\_W\_Figure\_M; D1\_H\_Figure\_M]); Mean\_D1\_L\_Figure\_R=mean([D1\_W\_Figure\_R; D1\_H\_Figure\_R]); Std\_D1\_L\_Figure\_M=std([D1\_W\_Figure\_M; D1\_H\_Figure\_M],1); Std\_D1\_L\_Figure\_R=std([D1\_W\_Figure\_R; D1\_H\_Figure\_R],1);  B\_M\_XOR=B\_M\_XOR./(ED\_Figure.\*pi); B\_R\_XOR=B\_R\_XOR./(ED\_Figure.\*pi); Max\_B\_M\_XOR=max(abs(B\_M\_XOR)); Max\_B\_R\_XOR=max(abs(B\_R\_XOR)); Delta\_Dov\_M=sort(abs(B\_M\_XOR), 'descend'); Delta\_Dov\_R=sort(abs(B\_R\_XOR),  'descend');  MaxDov\_B\_M\_XOR=Delta\_Dov\_M(1+round(N\_Im\*(1-PDov))); MaxDov\_B\_R\_XOR=Delta\_Dov\_R(1+round(N\_Im\*(1-PDov)));  Mean\_B\_M\_XOR=mean(B\_M\_XOR); Mean\_B\_R\_XOR=mean(B\_R\_XOR); Min\_B\_M\_XOR=min(B\_M\_XOR); Min\_B\_R\_XOR=min(B\_R\_XOR);  end; | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *12* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1.2.2. Виконуємо програму та отримаємо розрахунки фільтрації зображення.    Рис. 1.1. Результат виконання програми    Рис. 1.2. Спектр вихідного зображення | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *13* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Рис. 1.4. Спектр зображення з шумом    Рис. 1.5. Спектр зображення після фільтрації | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *14* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1.2.3. Розрахунки програмою MATLAB.  Розмір зображення рядка та стовпці д.т. 1024 x 1024 Фігура 1 – прямокутник  Кількість кадрів 30  ширина погр кривої (перепаду яскравості в рядку) д.т. 12 розмір маски фільтра д.т. 3  вів змаза під час руху відеокамери <=20 д.т. 0 кут змаза під час руху відеокамери град. 0 сигнал шум у дБ 40  Ампл погріш без фільт після фільт СКО серед д.у. 2.550 1.440  SNR серед дБ 28.518 33.842  PSNR серед дБ 40.001 45.325  Параметри похибки геометричних ознак д.т.  Погріш без фільт, д.т. площа% центр х центр y ширина висота Максимальна 0.031 0.017 0.030 2.000 1.000  Макс дов 0.021 0.017 0.022 1.000 1.000  Серед зн -0.001 0.001 -0.001 0.067 0.067  СКО 0.010 0.008 0.010 0.573 0.249  Погріш невпадання контуру без фільт після фільт макс д.т./довжину контуру 0.088 0.057  макс дов д.т./довжину конт 0.077 0.054  серед д.т./довжину контуру 0.060 0.037  СКО д.т./довжину контуру 0.010 0.010  ***Висновок:*** на данній лабораторній роботі було отримано навики обробки випадкових похибок інформаційних сигналів у вимірювальному каналі за допомогою програми MATLAB. | | | | | | |
|  |  |  |  |  | *МММТ.420.004.004 – ЗЛ1* | *Арк.* |
|  |  |  |  |  | *15* |
| *Змн.* | *Арк.* | *№ докум.* | *Підпис* | *Дата* |