VISUALIZATION V-7: Spatial Quantization

By Laurence Hassebrook

10-10-2011

Most of the code is given to perform this visualization. Try it on your own image. Repeat the process and present your own images and specifications. There are two parts. The first part is ideal flat top sampling, re-alignment and averaging. The second part incorporates an ideal low pass filter to leave only the baseband. The resulting blurred flat top images are processed as before to recover the original image approximation. Figure numbers in the code may not match the figure numbers in this document.

Input well focused image. Let $N$ be the flat top sampling period and $N_2$ be the number of lateral spatial steps in each direction during the sampling process. The number of pixels per step is $dN$. Both $N_2$ and $dN$ are in units of pixels.

```matlab
clear all;
% create a series of spatially quantized images
N=10; % period of flat top sampling
N2=10; % number of motion steps
dN=floor(N/N2); % amount of motion per step
% input focused image reference
A_bmp=double(imread('data/Blurry367NOTsmall.JPG')); % load example.bmp image
Ar=A_bmp(:,:,1);
Ag=A_bmp(:,:,2);
Ab=A_bmp(:,:,3);
Abw=(Ar+Ag+Ab)/3;
figure(1);
imagesc(Abw);
colormap gray
axis image;
title('Image in focus')
```

The instructor’s coin was 699 rows by 706 columns and is shown in Fig. 1.
Figure 1: Original coin image.

The spectra of the coin is shown in Fig. 2 for the full resolution input. The code below also zero pads a downsampled version of the input and finds its spectra. Try running both spectra and compare with a couple of sentences of what you see different. Think about why they are so different?

```matlab
% downsample and flat top
Asize=size(Abw)
Bsize=floor(Asize/N)
Azpad=zeros(Asize);
for m=1:Bsize(1)
    for n=1:Bsize(2)
        m1=(m-1)*N+1;
        n1=(n-1)*N+1;
        if m1<1; m1=1;end;
        if m1>Asize(1); m1=Asize(1);end;
        if n1<1; n1=1;end;
        if n1>Asize(2); n1=Asize(2);end;
        Azpad(m,n)=Abw(m1,n1);
    end;  % n
end;  % m

%ADFT=abs(fft2(Azpad));
ADFT=abs(fft2(Abw));
ADFT(1,1)=0;
figure(2);
imagesc(fftshift(log(ADFT+max(max(ADFT))/1000)));
colormap vga;%gray;
axis image;
print -djpeg fig2
title('Input Image Spectrum')
```
The spectra in Fig. 2 shows a lowpass spectra that spans the frequency window. It is interesting to note that there will be aliasing in the Flat Top process. The horizontal and vertical lines are artifacts of the edge leakage of the image.

**IDEAL FLAT TOP SAMPLING AND REALIGNMENT**

Systematically and uniformly shift the input prior to flat top sampling. The Kronecker (kron) multiplication is used to simulate the flat top spatial image.

```matlab
% K=ones(N,N); % kernel for mapping back near original resolution
% i and j are offsets
B=zeros(Bsize(1),Bsize(2),N2*N2);
size(B)
for i=1:N2
    for j=1:N2
        index3=(i-1)*N2+(j-1)+1;
        for m=1:Bsize(1)
            for n=1:Bsize(2)
                m1=(m-1)*N+(i-1)*dN+1;
                n1=(n-1)*N+(j-1)*dN+1;
                if m1<1; m1=1;end;
                if m1>Asize(1); m1=Asize(1);end;
                if n1<1; n1=1;end;
                if n1>Asize(2); n1=Asize(2);end;
                B(m,n,index3)=Abw(m1,n1);
            end; % n
        end; % m
        % figure(index3+2);
        figure(3);
        % make the figures the original size
        C=kronecker(B(:,:,index3),K);
        Csize=size(C);
        imagesc(C);
        colormap gray;
        axis image;
    end; % j
end; % i
figure(3);
```

![Figure 2: Spectra of full resolution input image.](image)
imagesc(C);
colormap gray;
axis image;
title('Flat Top Sampled Image')
print -djjpeg fig3

Figure 3: Flat top sampled image.

Figure 3 shows one sample of the image sequence during the flat top sampling.

% spectra of last image
CDFT=abs(fft2(C));
CDFT(1,1)=0;
figure(4);
imagesc(fftshift(log(CDFT+max(max(CDFT))/1000)));
colormap vga; % gray;
axis image;
print -djjpeg fig4

Figure 4: Flat top spectra.

The flat top spectra is shown in Fig. 4. Note the null locations for a flat top sampling with 100% duty cycles. Circle the baseband and the first replicated spectras. Indicate the null locations. The
reconstruction is performed by reversing the miss-alignment offsets back to overlay the images properly. Once aligned, they are averaged to recover the estimate of the original input image.

% average the figures back together
Cout=zeros(Csize);
for i=1:N2
    for j=1:N2
        % page
        index3=(i-1)*N2+(j-1)+1;
        C=kron(B(:,:,index3),K);
        for m=1:Csize(1)
            for n=1:Csize(2)
                % offset coordinates
                m1=m-(i-1)*dN;
                n1=n-(j-1)*dN;
                if m1<1
                    m1=1;
                end;
                if m1>Csize(1)
                    m1=Csize(1);
                end;
                if n1<1
                    n1=1;
                end;
                if n1>Csize(2)
                    n1=Csize(2);
                end;
                Cout(m,n)=Cout(m,n)+C(m1,n1);
            end; % n
        end; % m
        figure(5);
        imagesc(Cout);
        colormap gray;
        axis image;
    end; % j
end; % i

Figure 5: Reconstructed coin from averaging realigned images.
2. IDEAL FLAT TOP BASEBAND REALIGNMENT

In this section of the visualization, the flat top sampled image for each offset is filtered with an ideal filter to pass only the baseband spectra. Perform this filtering and repeat the process performed in the previous section and answer any of the same questions.

```matlab
% form a ideal flattop reconstruction filter to simulate blurring
figure(6);
Nhx=1+2*Csize(2)/N
Mhy=1+2*Csize(1)/N
H=irect(Mhy,Nhx,Csize(1),Csize(2));
imagesc(fftshift(H));
colormap gray;
axis image;
title('Ideal BaseBand Filter')
```

**Figure 6:** Baseband filter based on N and size values.

```matlab
% i and j are offsets
Asize=size(Abw)
Bsize=floor(Asize/N)
B=zeros(Bsize(1),Bsize(2),N2*N2);
size(B)
for i=1:N2
    for j=1:N2
        index3=(i-1)*N2+(j-1)+1;
        for m=1:Bsize(1)
            for n=1:Bsize(2)
                m1=(m-1)*N+(i-1)*dN+1;
                n1=(n-1)*N+(j-1)*dN+1;
                if m1<1; m1=1; end;
                if m1>Asize(1); m1=Asize(1); end;
                if n1<1; n1=1; end;
                if n1>Asize(2); n1=Asize(2); end;
                B(m,n,index3)=Abw(m1,n1);
```

end; % n
end; % m
%   figure(index3+2);
figure(7);
% make the figures the original size
C=kron(B(:,:,index3),K);
CFT=fft2(C);
C=abs(ifft2(CFT.*H));
Csize=size(C);
imagesc(C);
colormap gray;
axis image;
end; % j
end; % i

Figure 7: example of flat top sampled image with baseband filtering.

% spectra of last image
CDFT=abs(fft2(C));
CDFT(1,1)=0;
figure(8);
imagesc(fftshift(log(CDFT+max(max(CDFT))/1000)));
colormap vga;%gray;
axis image;
title('Filtered Flat Top Spectra')
Figure 8: Baseband spectra.

% average the figures back together
Cout=zeros(Csize);
for i=1:N2
    for j=1:N2
        %page
        index3=(i-1)*N2+(j-1)+1;
        C=kron(B(:,:,index3),K);
        for m=1:Csize(1)
            for n=1:Csize(2)
                % offset coordinates
                ml=m-(i-1)*dN;
                nl=n-(j-1)*dN;
                if ml<1
                    ml=1;
                end;
                if ml>Csize(1)
                    ml=Csize(1);
                end;
                if nl<1
                    nl=1;
                end;
                if nl>Csize(2)
                    nl=Csize(2);
                end;
                Cout(m,n)=Cout(m,n)+C(ml.nl);
            end; % n
        end; % m
    end; % j
end; % i
Figure 9: Re-aligned and reconstructed image.