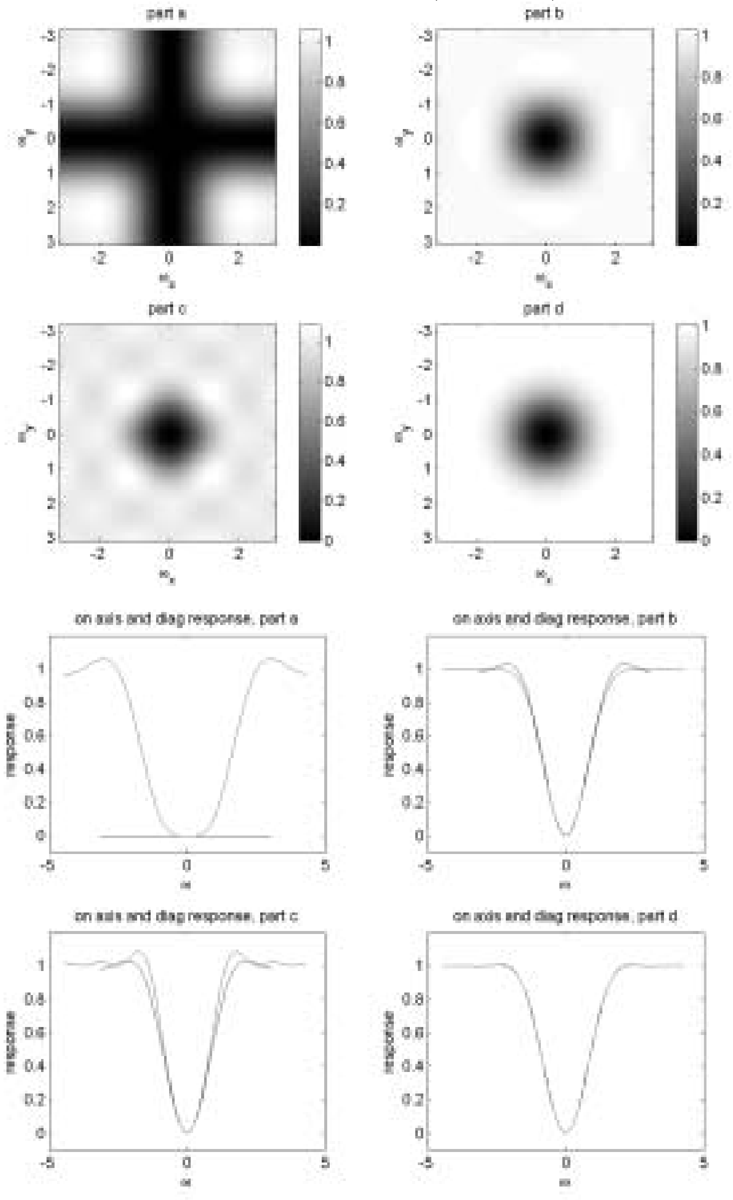
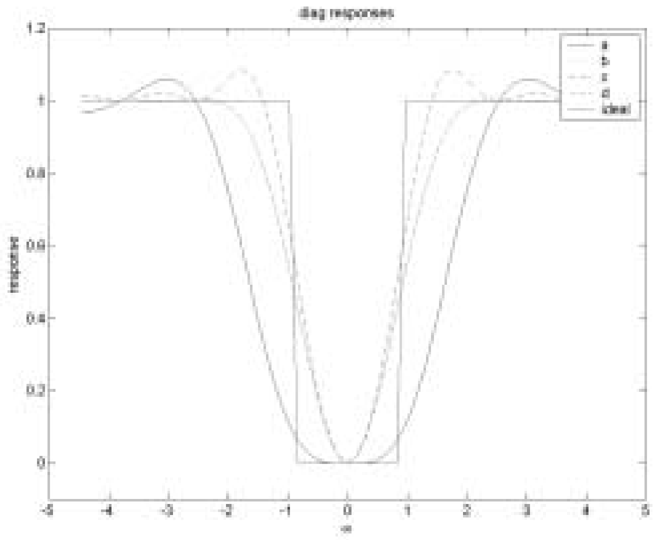
3. Filter design:
   a. Simply take the outer product *h'*h*.
   b. Use $\delta_2(n,m) - [\delta_1(n) - h(n)]'*[\delta_1(n) - h(n)]$.
   c. Here we follow the form of frequency sampling described in section 4.3.2 of the text. We sample at the N_1 x N_2 = 7 x7 locations and inverse DFT to get our desired filter.
   d. Here we use the "jinc" like formuli of Equation 4.10b in the text, but we modify as described in class to make sure the passband is approximately 1 and the stopband is approximately 0. We can also modify the filter with a circularly symmetric and smooth filter to reduce truncation artifacts (I did this).





e. In my implementations, filter c. has the tighted response but quite a bit of ringing (passband ripple) in spite of adding transition regions to the frequency sampling specification. Filters b. and d. have very similar responses in termps of bandwidth, though d. is slightly more symmetrical (circular). Filter a. has a very poor response – separable filters work well for low-pass designs but very poorly for high-pass and band-pass designs. Note the on-axis responses. All filters are 8 fold symmetric and d. looks nearly circularly symmetric, but it is not. Filter a., due to its separability, is the most computationally efficient. Filter b. is nearly as efficient as it can be implemented by low pass filtering the image (using separably low-pass filters) and then subtracting this from the original image. Filters c. and d. have some computational efficiencies related to 8 fold symmetry, but these are secondary to the separability advantages. If the response shapes, ripple, computational issues were all important, then filter b. would probably be best, but it really depends on the relative importance of these factors…

Code:

```
% solution problem 3, exam 1, EECS 556, 2003
n = 7; N=64; figure(1)
% part a
h1 = [-0.0141 -0.1111 -0.2319 0.7143 -0.2319 -0.1111 -0.0141];
ha = h1'*h1;
hpad = zeros([N N]);
padind = N/2+1-(n-1)/2:N/2+1+(n-1)/2;
hpad(padind,padind)=ha;
Ha = real(fftshift(fft2(fftshift(hpad))));
ax = [-N/2:N/2-1]*2*pi/N;
subplot(221)
imagesc(ax,ax,Ha); colormap gray; colorbar; axis image
xlabel('\omega_x'); ylabel('\omega_y'); title('part a')

% part b
l1 = [0 0 0 1 0 0 0] - h1;
lb = l1'*l1;
hb = -lb; hb((n+1)/2,(n+1)/2)=hb((n+1)/2,(n+1)/2)+1;
hpad(padind,padind)=hb;
Hb = real(fftshift(fft2(fftshift(hpad))));
subplot(222)
imagesc(ax,ax,Hb); colormap gray; colorbar; axis image
xlabel('\omega_x'); ylabel('\omega_y'); title('part b')
```

```
% part c
wsamp = [-3:3]/7*2*pi;
[wx,wy] = ndgrid(wsamp);
ww = abs(wx+i.*wy);
Hsamp = (ww - pi/7)*(7/2/pi).*(ww > pi/7).*(ww < 3*pi/7)...
    + (ww > 3*pi/7);
hc = fftshift(ifft2(ifftshift(Hsamp)));
hpad(padind,padind)=hc;
Hc = real(fftshift(fft2(fftshift(hpad))));
subplot(223)
imagesc(ax,ax,Hc); colormap gray; colorbar; axis image
xlabel('\omega_x'); ylabel('\omega_y'); title('part c')

% part d
nmind = [-3:3];
[nn,mm] = ndgrid(nmind);
rr = abs(nn+i.*mm)+eps;
wc = 2*pi/7;
ld = wc.*besselj(1,wc.*rr)./(2*pi*rr);
filt = (0.5 + 0.5*cos(pi*rr./3/sqrt(2)));
ld = filt.*ld;  % apodization
ld = ld./sum(ld(:));  %normalize
hd = -ld; hd((n+1)/2,(n+1)/2)=hd((n+1)/2,(n+1)/2)+1;
hpad(padind,padind)=hd;
Hd = real(fftshift(fft2(fftshift(hpad))));
subplot(224)
imagesc(ax,ax,Hd); colormap gray; colorbar; axis image
xlabel('\omega_x'); ylabel('\omega_y'); title('part d')

figure(2)
subplot(221)
[xx yy] = ndgrid([1:N]);
diagind = find(xx == yy);
plot(ax,Ha(N/2+1,:),ax.*sqrt(2),Ha(diagind))
xlabel('\omega');ylabel('response'); axis([-5 5 -0.1 1.2])
title('on axis and diag response, part a');
subplot(222)
plot(ax,Hb(N/2+1,:),ax.*sqrt(2),Hb(diagind))
xlabel('\omega');ylabel('response');axis([-5 5 -0.1 1.2])
title('on axis and diag response, part b');
subplot(223)
plot(ax,Hc(N/2+1,:),ax.*sqrt(2),Hc(diagind))
xlabel('\omega');ylabel('response');axis([-5 5 -0.1 1.2])
title('on axis and diag response, part c');
subplot(224)
plot(ax,Hd(N/2+1,:),ax.*sqrt(2),Hd(diagind))
xlabel('\omega');ylabel('response');axis([-5 5 -0.1 1.2])
title('on axis and diag response, part d');

figure(3)
plot(ax.*sqrt(2),Ha(diagind),'-',ax.*sqrt(2),Hb(diagind),':',...
    ax.*sqrt(2),Hc(diagind),'-.',ax.*sqrt(2),Hd(diagind),'--',...
    ax.*sqrt(2),abs(sqrt(2).*ax) > 2*pi/7,'-');
legend('a','b','c','d','ideal')
xlabel('\omega');ylabel('response');axis([-5 5 -0.1 1.2])
title('diag responses');
```

4. For each parts a., b., d., and e., the output image size is 62x62 and the center pixel is (32,32). See attached Matlab code for implementations of these parts. Resultant figure for all parts are identical to within numerical error (see plots).

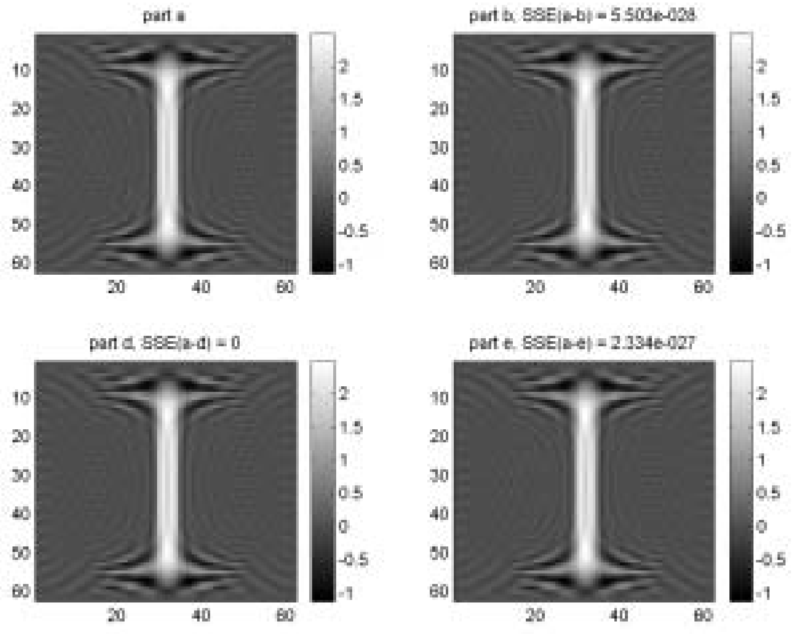
   c. Letting $r1d(p) = r1d(n + mN) = rsqr(n,m)$ and we also define $h1d(q) = h1d(k + lN) = h(k,l)$. Observe that $N \neq 5$ so h must be considered to be zero padded out to column length N. The convolution is:

$$y(n,m) = \sum_{k=-2}^{2} \sum_{l=-2}^{2} h(k,l) rsqr(n-k, m-l)$$

$$y1d(n + mN) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{l=-2}^{2} h1d(k + lN) r1d(n - k + (m-l)N)$$

$$y1d(p) = \sum_{q=-\frac{5N}{2}}^{\frac{5N}{2}-1} h1d(p) r1d(p + qN)$$

   One issue is that this convolution attaches the top of column *m* to the top of column (*m*+1). This is similar, though not exactly the same as circular convolution in the vertical direction. Thus, it is necessary to pad the column space of each out to *N* = 58+5-1 = 62. This is not necessary for the top of the 1st column and the bottom of the last column. So, the minimum length of *h1d* will be 4x*N*+5 = 253. The minimum length of *r1d* is 57x*N*+58 = 3,592. The output of the convolution will be 253+3,592-1=3,844=$(62)^2$, the same size output from the 2D convolution.



Code:

```
% Exam1, problem 4, 2003
N = 58;
[yy,xx]=ndgrid([-N/2:N/2-1]);
ang = pi/6;
sz = sqrt(2).*sin(ang);
xr = xx.*cos(ang) + yy.*sin(ang);
yr = -xx.*sin(ang) + yy.*cos(ang);
rks = sinc(xr.*sz).*sinc(yr.*sz);
rsqr = real(fftshift(fft(fftshift(rks))));  % the image


h = [ 0  -1 -1 -1  0;
      -1 -1 -1 -1 -1;
       3  5  5  5  3;
      -1 -1 -1 -1 -1;
       0 -1 -1 -1  0];  % the filter

% part a
ima = conv2(rsqr,h);
subplot(221)
imagesc(ima); colormap gray; colorbar
title('part a')
% part b
rsqrpad = zeros([62 62]);
hpad = zeros([62 62]);
rsqrpad(3:60,3:60) = rsqr;
hpad(30:34,30:34) = h;
imb = real(fftshift(ifft2(fft2(fftshift(hpad))...
    .*fft2(fftshift(rsqrpad)))));
subplot(222)
imagesc(imb); colormap gray; colorbar
sseb = sum((ima(:)-imb(:)).^2);
str = sprintf('part b, SSE(a-b) = %0.4g',sseb);
title(str)

% part d
z1 = zeros([57 1]);
h1 = [h(:,1);z1;h(:,2);z1;h(:,3);z1;h(:,4);z1;h(:,5)];
rsqrpad1 = rsqrpad(:);
% remove unnecessary zeros
excessz = 62*2+2;
r1 = rsqrpad1(excessz+1:end-excessz);
%1d conv
y1 = conv(h1,r1);
imd = reshape(y1,62,62);
subplot(223)
imagesc(imd); colormap gray; colorbar
ssed = sum((ima(:)-imd(:)).^2);
str = sprintf('part d, SSE(a-d) = %0.4g',ssed);
title(str)

lh1 = length(h1);
lr1 = length(r1);
ly2 = lh1+lr1-1;
r1pad = zeros([ly2 1]);
h1pad = r1pad;
r1pad(ly2/2+1-lr1/2:ly2/2+lr1/2) = r1;
h1pad(ly2/2+1-(lh1-1)/2:ly2/2+1+(lh1-1)/2) = h1;
y2 = real(fftshift(ifft(fft(fftshift(h1pad))...
    .*fft(fftshift(r1pad)))));
ime = reshape(y2,62,62);
subplot(224)
imagesc(ime); colormap gray; colorbar
ssee = sum((ima(:)-ime(:)).^2);
str = sprintf('part e, SSE(a-e) = %0.4g',ssee);
title(str)
```