EECE5646 Midterm Exam

with Solutions

Prof. Charles A. DiMarzio
Department of Electrical and Computer Engineering
Northeastern University

Fall Semester 2012
11 October 2012

Overall Observations: Many students typeset the whole exam in Word or LaTeX, which made it easy to follow. At least one student submitted an excellent hand–written exam that was as easy to read as the typeset ones. That student placed the computer work at the back, but included a table of contents at the front. These exams were the easiest to grade, particularly when I sought to understand the reasoning in order to assign partial credit.

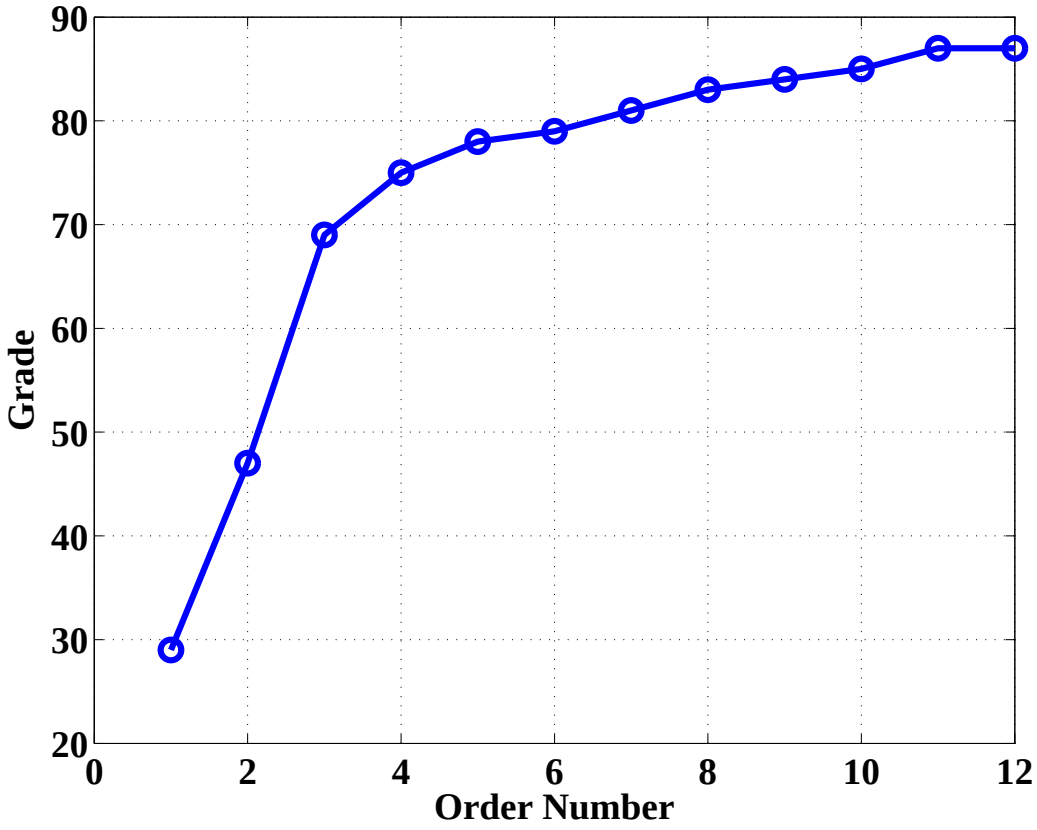
Some students did not complete the simulated image, which was an important part of the first problem.

Some students seem to have been confused by the three–lens afocal in Problem 2. The afocal telescopes we discussed in class had only two lenses, but an an afocal can consist of any number of lenses. All that is required is that the lower–left matrix element be zero.

In problem 3, there was a wide range of answers. In some cases I was able to figure out what went wrong, and in others I was not. See my solutions below.

I graded problems 1 and 3 going from the top to bottom of the alphabetical list of papers. I graded 2 and 4 going from bottom to top. That way, any changing bias I might have had in grading should cancel. I returned a brief text file with some comments. A plus before a part number indicates that almost everything was correct, and a minus or double minus indicated some deficiency. An X indicated that most was wrong.

Scores are shown on a cumulative histogram below.



1 Snell's Window

The goal here is to construct a synthetic image of the view from a swimming pool, looking up, to illustrate Snell's Window. At angles below the critical angle, we will assume that no reflection occurs, and thus solve only the refraction problem. After Chapter 6, we could modify the result to account for the actual amount of light reflected and refracted at each angle, but we will not do so here.

The pool is square, with a width of 12 meters and a depth of 3 meters. The room in which it is located is also square, with walls separated by 18 meters and a ceiling 4 meters above the water surface.

1.1 Simple Camera

Design a camera (Specify the focal length of the lens and the distance to the image plane) such that the entrance window will be a 4–meter square on the surface when the camera is placed in the center of the pool at a depth of 1.5 meters. The design of this camera is not critical to the problem, so you may assume water on both sides of the lens. In my solution, I assumed 401 pixels in each direction, to keep the computation time short.

For your choice of focal length, what would be the focal length of this lens in air?

In a medium of index, n,

$$\frac{n}{f} = \frac{n}{f'} = P = (n_\ell - n)\left(\frac{1}{r_1} - \frac{1}{r_2}\right).$$

Thus the focal length is proportional to

$$\frac{n}{n_\ell - n}$$

Thus

$$f_{air} = \frac{1}{1.5 - 1} \times \frac{1.5 - 1.33}{1.33} \times f_{water} = 0.26 \times f_{water}.$$

As expected, the focal length is much shorter (*i.e.* the lens is more powerful) in air than in water. The power, P, is larger by a factor of about 4.

1.2 Reflection

We have discussed the refraction equation in vector form. Derive an equation that can be used similarly for reflection.

The output vector has the same component in the direction transverse to the normal after reflection that it had before, but its component parallel to the normal is exactly reversed. Therefore

$$\mathbf{V}_1 = \mathbf{V}_0 - 2(\mathbf{V}_0 \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}$$

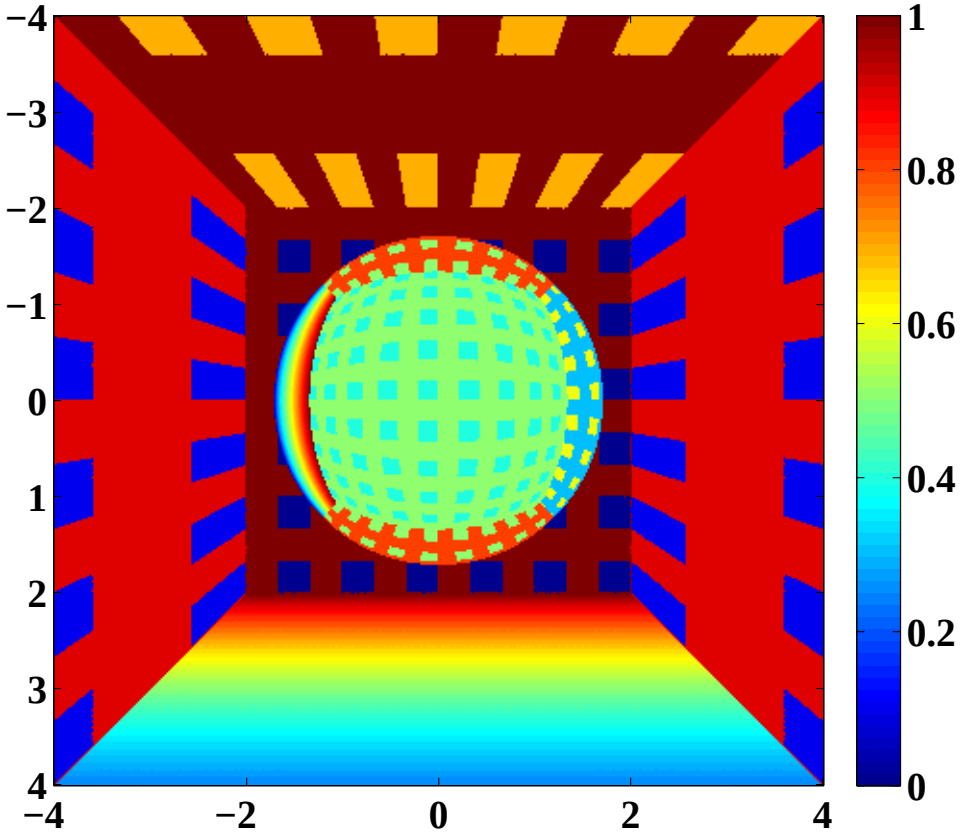
1.3 Synthetic Image

In this case, it's hard to trace rays forward and predict which pixel (if any) they will hit. It is easier to trace backward and "close" the ray to find where it originated. Trace rays from the camera pupil to each pixel in the entrance window (on the surface), and then consider refraction or reflection

as appropriate, and compute the intersection with the surfaces that comprise the sides and bottom of the pool, the sides of the room, and the ceiling. Put some pattern on the wall, so that you can interpret the synthetic image. I simply truncated each of the coordinates (x,y,z) of the final point to the nearest meter, and determined if the product was odd or even to generate a 1 for a bright panel or 0 for a dark one. The calculation was relatively easy, and the result was easy to interpret.

Hint: Start with the pool bottom and ceiling. Once those are working, start work on one side, because there is one additional complication on the sides that does not occur on the horizontal surfaces. *Another hint:* Calculate the unit vector for refraction before trying reflection. If the refraction calculation fails in some way, it tells you that you are beyond the critical angle. *Yet another hint:* You will have multiple solutions to "close" each ray. Give some thought to the easiest way to pick the correct solution.

I chose to make one wall in each case (in the water and in the air) have a continuously varying signal, proportional to height, and the rest to have a checkerboard pattern.




```
%%% m11700mt1.m  Midterm Problem 1
function [signal,xax,yax]=m11700mt1;
xax=[-4:0.02:4];yax=[-4:0.02:4]; % Axes of coordinates
```

```
signal=zeros(length(yax),length(xax));
xstore=zeros(length(xax),length(yax),3,3);
vstore=zeros(length(xax),length(yax),3-1,3);
norm=[0,0,1]';
index=1.33/1;
for n=1:length(xax);
  x=xax(n);
  for m=1:length(yax);
    y=yax(m);
    x0=[0,0,-1.5]';
    v0=[x,y,1.5]';v0=v0/sqrt(v0'*v0);
    xstore(m,n,1,:)=x0;
    vstore(m,n,1,:)=v0;
    ell=-x0(3)/v0(3);
    x1=x0+ell*v0;
%    norm=[0,0,1]'; % Executed outside loop above
%     index=1.33/1;
     argsqrt=1-index^2*(1-(v0'*norm).^2);
     if(argsqrt)>0; % Going into air
       v1=index*v0+...
          (sqrt(argsqrt)-index*v0'*norm)*norm;
      [signal(m,n),x2]=airclose(x1,v1);
    else; % TIR
      v1=v0-2*(v0'*norm)*norm;
      [signal(m,n),x2]=waterclose(x1,v1);
    end;
      xstore(m,n,2,:)=x1;
      vstore(m,n,2,:)=v1;
      xstore(m,n,3,:)=x2;
  end;
end;

function [waterclose,x2]=waterclose(x1,v1);
[a(1),ellt(1)]=waterclosezb(x1,v1); % a is the signal, ellt is the test ell
[a(2),ellt(2)]=waterclosexa(x1,v1);
[a(3),ellt(3)]=waterclosexb(x1,v1);
[a(4),ellt(4)]=watercloseya(x1,v1);
[a(5),ellt(5)]=watercloseyb(x1,v1);
ell=min(ellt(find(ellt>0)));  % ell is minimum positive ellt
waterclose=a(find(ell==ellt,1));
x2=x1+ell*v1;

function [waterclosezb,ell]=waterclosezb(x1,v1); % Bottom of pool
% (x1 + ell*v1)'*[0,0,1] = -3; (3 meters deep)
ell=(-3-x1(3))/v1(3);
```

```
x2=x1+ell*v1;
test=floor(x2(1))*floor(x2(2));
if((floor(test/2)*2==test));
  waterclosezb=1;
else
  waterclosezb=0;
end;

function [waterclosexa,ell]=waterclosexa(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = -6;
if(v1(1)~=0);
ell=(-6-x1(1))/v1(1);
x2=x1+ell*v1;
test=floor(x2(2))*floor(x2(3));
if((floor(test/2)*2==test));
  waterclosexa=0.9;
else
  waterclosexa=0.1;
end;
else; % Handle the case of no intersection
  ell=-1;
  waterclosexa=-1;
end;

function [waterclosexb,ell]=waterclosexb(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = 6;
if(v1(1)~=0);
ell=(6-x1(1))/v1(1);
x2=x1+ell*v1;
test=floor(x2(2))*floor(x2(3));
if((floor(test/2)*2==test));
  waterclosexb=0.9;
else
  waterclosexb=0.1;
end;
else; % Handle the case of no intersection
  ell=-1;
  waterclosexb=-1;
end;

function [watercloseya,ell]=watercloseya(x1,v1); % Side
% (x1 + ell*v1)'*[0,1,0] = -6;
if(v1(2)~=0);
ell=(-6-x1(2))/v1(2);
x2=x1+ell*v1;
```

```
test=floor(x2(1))*floor(x2(3));
if((floor(test/2)*2==test));
  watercloseya=1;
else
  watercloseya=0.7;
end;
else; % Handle the case of no intersection
  ell=-1;
  watercloseya=-1;
end;

function [watercloseyb,ell]=watercloseyb(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = 6;
if(v1(2)~=0);
ell=(6-x1(2))/v1(2);
x2=x1+ell*v1;
test=floor(x2(1))*floor(x2(3));
%if((floor(test/2)*2==test));
%  watercloseyb=1;
%else
%  watercloseyb=0.7;
%end;
watercloseyb=-x2(3)/3;
else; % Handle the case of no intersection
  ell=-1;
  watercloseyb=-1;
end;


function [airclose,x2]=airclose(x1,v1);
[a(1),ellt(1)]=aircloseza(x1,v1); % a is the signal, ellt is the test ell
[a(2),ellt(2)]=airclosexa(x1,v1);
[a(3),ellt(3)]=airclosexb(x1,v1);
[a(4),ellt(4)]=aircloseya(x1,v1);
[a(5),ellt(5)]=aircloseyb(x1,v1);
ell=min(ellt(find(ellt>0)));  % ell is minimum positive ellt
airclose=a(find(ell==ellt,1));
x2=x1+ell*v1;


function [aircloseza,ell]=aircloseza(x1,v1); % Ceiling
% (x1 + ell*v1)'*[0,0,1] = 4; (4 meter ceiling)
ell=(4-x1(3))/v1(3);
x2=x1+ell*v1;
test=floor(x2(1))*floor(x2(2));
```

```
if((floor(test/2)*2==test));
  aircloseza=0.5;
else
  aircloseza=0.4;
end;


function [airclosexa,ell]=airclosexa(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = -9;
if(v1(1)~=0);
ell=(-9-x1(1))/v1(1);
x2=x1+ell*v1;
test=floor(x2(2))*floor(x2(3));
%if((floor(test/2)*2==test));
%  airclosexa=0.3;
%else
%  airclosexa=0.6;
%end;
airclosexa=x2(3)/4;
else; % Handle the case of no intersection
  ell=-1;
  airclosexa=-1;
end;

function [airclosexb,ell]=airclosexb(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = 9;
if(v1(1)~=0);
ell=(9-x1(1))/v1(1);
x2=x1+ell*v1;
test=floor(x2(2))*floor(x2(3));
if((floor(test/2)*2==test));
  airclosexb=0.3;
else
  airclosexb=0.6;
end;
else; % Handle the case of no intersection
  ell=-1;
  airclosexb=-1;
end;

function [aircloseya,ell]=aircloseya(x1,v1); % Side
% (x1 + ell*v1)'*[0,1,0] = -9;
if(v1(2)~=0);
ell=(-9-x1(2))/v1(2);
x2=x1+ell*v1;
```

```
test=floor(x2(1))*floor(x2(3));
if((floor(test/2)*2==test));
  aircloseya=0.8;
else
  aircloseya=0.5;
end;
else; % Handle the case of no intersection
  ell=-1;
  aircloseya=-1;
end;

function [aircloseyb,ell]=aircloseyb(x1,v1); % Side
% (x1 + ell*v1)'*[1,0,0] = 9;
if(v1(2)~=0);
ell=(9-x1(2))/v1(2);
x2=x1+ell*v1;
test=floor(x2(1))*floor(x2(3));
if((floor(test/2)*2==test));
  aircloseyb=0.8;
else
  aircloseyb=0.5;
end;
else; % Handle the case of no intersection
  ell=-1;
  aircloseyb=-1;
end;

figure;imagesc(yax,xax,signal);axis image;colorbar;
print -depsc 11700mt1a.eps
```

2 Zoom Lens

We can make a zoom lens (one that has a variable focal length) for a camera, starting with a camera lens of fixed focal length, f_4, by placing an afocal telescope before the lens. If the afocal telescope has an adjustable magnification, m, the focal length of the combination will change. We would like to do this in such a way that we don't have to move the image plane (focus the camera) when we adjust the focus (zoom).

2.1 Zoom Concept

The most general matrix for an afocal telescope in air is

$$\mathcal{A} = \begin{pmatrix} m & q \\ 0 & 1/m \end{pmatrix},$$

where m is the magnification, and q can have any value, depending on the implementation of the telescope.

If such a telescope is place in front of a camera lens having a focal length, f_4, what is the focal length of the combination?

Hint: Matrix optics is probably the easiest approach here.

$$\mathcal{Z} = \mathcal{L}_4\mathcal{A}$$

$$\begin{pmatrix} 1 & 0 \\ -1/f_4 & 1 \end{pmatrix}\begin{pmatrix} m & q \\ 0 & 1/m \end{pmatrix} = \begin{pmatrix} m & q \\ -m/f_4 & 1/m - q/f_4 \end{pmatrix}$$

The focal length is

$$f = f_4/m.$$

Assuming thin lenses and a telescope with a total length, z_t, where are the principal planes relative to the microscope lens? Write your answer as a function of f_4, m, q, and z_t.

$$h = \frac{1 - m_{22}}{m_{21}} = \frac{q-1}{m} f_4$$

Then h is measured from the front of the telescope, so the location of the front principal plane is

$$\frac{q-1}{m} f_4 + z_t$$

in front of the lens.

$$h' = \frac{1 - m_{11}}{m_{21}} = \frac{m-1}{m} f_4$$

For an object at infinity, where is the image?

$$s' = f \qquad w' = h' + s' = \frac{m-1}{m} f_4 + \frac{f_4}{m} = f_4.$$

Explain why this answer is so simple. *Hint:* If it wasn't simple, you might want to go back and look at your equations again. How would the result be more complicated if the object distance were finite?

Parallel rays entering the afocal telescope exit as parallel rays. Thus, the lens, f_4 sees the image from the telescope as an object at infinity. The image is at its back focal point.

For a finite object distance, the image distance can be calculated using the lens equation. However, for the same original object distance, the image through the telescope will be located at different distances for different magnifications. Therefore, the image distance will change. We must adjust the focus when we adjust the zoom.

2.2 Example

Consider a telescope consisting of three lenses, f_1, f_2, f_3 mounted in front of a camera lens, f_4;

$$f_1 = f_3 = 70 \text{ mm} \qquad f_2 = -22.5 \text{ mm} \qquad f_4 = 100 \text{ mm}.$$

The distance between the last telescope lens and the camera lens is

$$z_{34} = 10 \text{ mm}.$$

Write the matrices, and find the principal planes, focal length, and focal points for

$$z_{12} = 25 \text{ mm} \qquad z_{23} = 25 \text{ mm},$$
$$z_{12} = 10 \text{ mm} \qquad z_{23} = 34 \text{ mm},$$

and

$$z_{12} = 34 \text{ mm} \qquad z_{23} = 10 \text{ mm}.$$

Write the matrix for the telescope alone. Use the condition that the telescope is afocal to determine z_{23} and the telescope length, $z_{23} + z_{12}$ as functions of z_{12}. Compare your results to the three fixed points above, and comment on the useful range of focal lengths with the zoom lens in this problem.

$$\mathcal{M} = \begin{pmatrix} 1 & 0 \\ -1/f_3 & 1 \end{pmatrix} \begin{pmatrix} 1 & z_{23} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -1/f_2 & 1 \end{pmatrix} \begin{pmatrix} 1 & z_{12} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -1/f_1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ -1/f_3 & 1 \end{pmatrix} \begin{pmatrix} 1 - z_{23}/f_2 & z_{23} \\ -1/f_2 & 1 \end{pmatrix} \begin{pmatrix} 1 - z_{12}/f_1 & z_{12} \\ -1/f_1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ -1/f_3 & 1 \end{pmatrix} \begin{pmatrix} 1 - z_{23}/f_2 - z_{12}/f_1 + z_{23}z_{12}/f_2f_1 - z_{23}/f_1 & z_{12} - z_{12}z_{23}/f_2 + z_{23} \\ -1/f_2 + z_{12}/(f_1f_2) - 1/f_1 & -z_{12}/f_2 + 1 \end{pmatrix}$$

$$\mathcal{M} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}$$

$$m_{11} = 1 - z_{23}/f_2 - z_{12}/f_1 + z_{23}z_{12}/f_2f_1 - z_{23}/f_1$$

$$m_{12} = z_{12} - z_{12}z_{23}/f_2 + z_{23}$$

$$m_{21} = -1/f_3 + z_{23}/(f_2f_3) + z_{12}/(f_1f_3) - z_{23}z_{12}/f_2f_1f_3 + z_{23}/(f_1f_3) - 1/f_2 + z_{12}/(f_1f_2) - 1/f_1$$

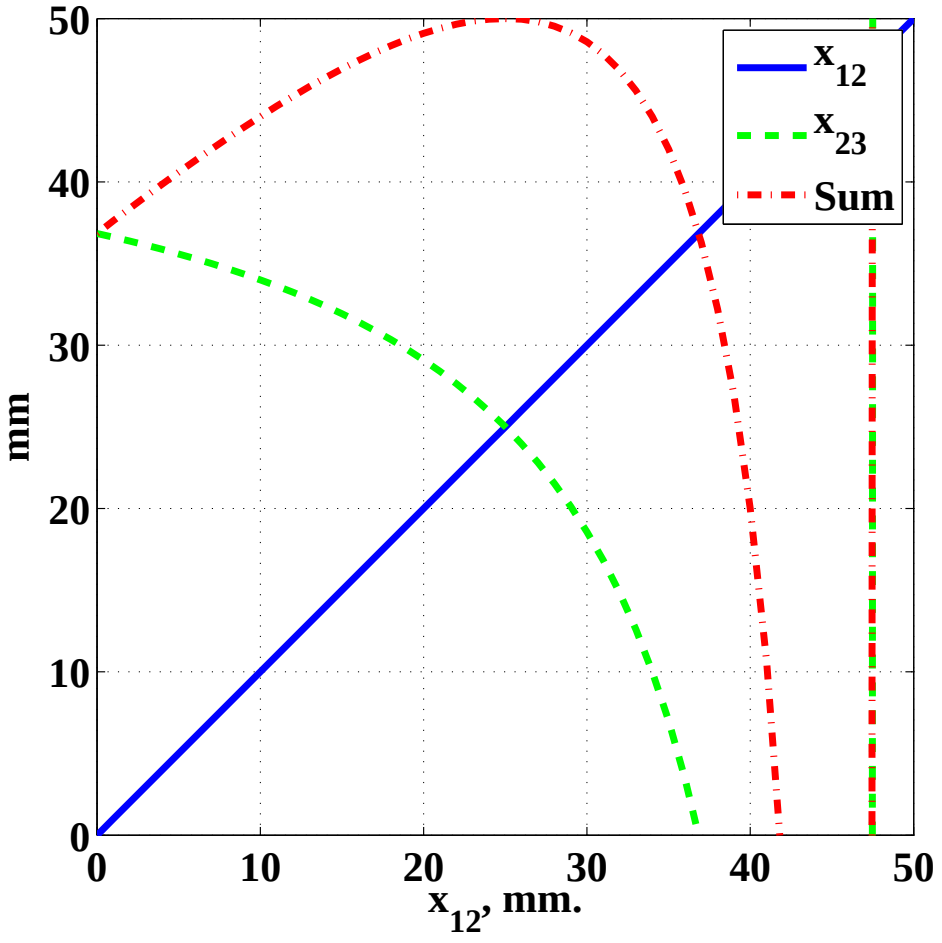
$$m_{22} = -z_{12}/f_3 + z_{12}z_{23}/(f_2f_3) - z_{23}/f_3 - z_{12}/f_2 + 1$$

Set $f_1 = f_3$ and solve for $m_{21} = 0$.

$$-1/f_1 + z_{23}/(f_2f_1) + z_{12}/(f_1^2) - z_{23}z_{12}/f_2f_1^2 + z_{23}/(f_1^2) - 1/f_2 + z_{12}/(f_1f_2) - 1/f_1 = 0$$

$$z_{23}/(f_2f_1) - z_{23}z_{12}/f_2f_1^2 + z_{23}/(f_1^2) = -z_{12}/(f_1^2) - z_{12}/(f_1f_2)n + 1/f_2 + 1/f_1 + 1/f_1$$

$$z_{23} = \frac{-z_{12}/f_1 - z_{12}/f_2 + f_1/f_2 + 2}{1/f_2 - z_{12}/f_2f_1 + 1/f_1}$$



See code below for the calculations for the three points and the generation of this figure.

```
%%% m11700mt2.m  Midterm exam zoom lens problem;
```

```
% Lens focal lengths
f1=70e-3;f3=f1;f2=-22.5e-3;f4=100e-3;

% Three configurations for example in 2.2
for n=1:3
  z12=25e-3;z23=25e-3;z34=10e-3;
  if(n==2);z12=10e-3;z23=34e-3;z34=10e-3;end;
  if(n==3);z12=34e-3;z23=10e-3;z34=10e-3;end;

  disp('-----------------');
  z12
  z23

% Global coordinates.  Put v4 at the origin
zV4=0;
zV3=zV4-z34;
zV2=zV3-z23;
zV1=zV2-z12;

m1=[1,0;-1/f1,1];
m2=[1,0;-1/f2,1];
m3=[1,0;-1/f3,1];
m4=[1,0;-1/f4,1];

t12=[1,z12;0,1];
t23=[1,z23;0,1];
t34=[1,z34;0,1];

m=m4*t34*m3*t23*m2*t12*m1
mtel=m3*t23*m2*t12*m1

h=(1-m(2,2))/m(2,1); % n=nprime=1;
hprime=(1-m(1,1))/m(2,1);

f=-1/m(2,1)
zH=zV1-h
zHprime=zV4+hprime
zF=zH-f
zFprime=zHprime+f

end;

% Plot for part 3.
z12test=[0:50]*1e-3;
z23test=(-z12test/f1-z12test/f2+f1/f2+2)./...
```

```
    (1/f2-z12test/f1/f2+1/f1);
zsumtest=z12test+z23test;
figure;plot(z12test*1e3,z12test*1e3,'b-',...
    z12test*1e3,z23test*1e3,'g--',...
    z12test*1e3,zsumtest*1e3,'r-.');grid on;
axis equal;axis tight;
axis([0,50,0,50]);
xlabel('x_{12}, mm.');
ylabel('mm');
legend('x_{12}','x_{23}','Sum');
```

3 Stops

Here we discuss the finite lens diameters in the zoom lens from Problem 2. We'll first find the location of all the apertures in image space, and the amount by which they are magnified. Next we'll choose to place the aperture stop at the first lens. To make sure it functions as the aperture stop, we will determine the minimum diameter of the other lenses so that they do not become aperture stops. Then we want the diagonal of the camera to be the exit window, so we will decide on the minimum lens sizes required to make sure none of them limits the field of view more than that diagonal. The diameters of the lenses must be large enough to satisfy both conditions.

3.1 Image Space

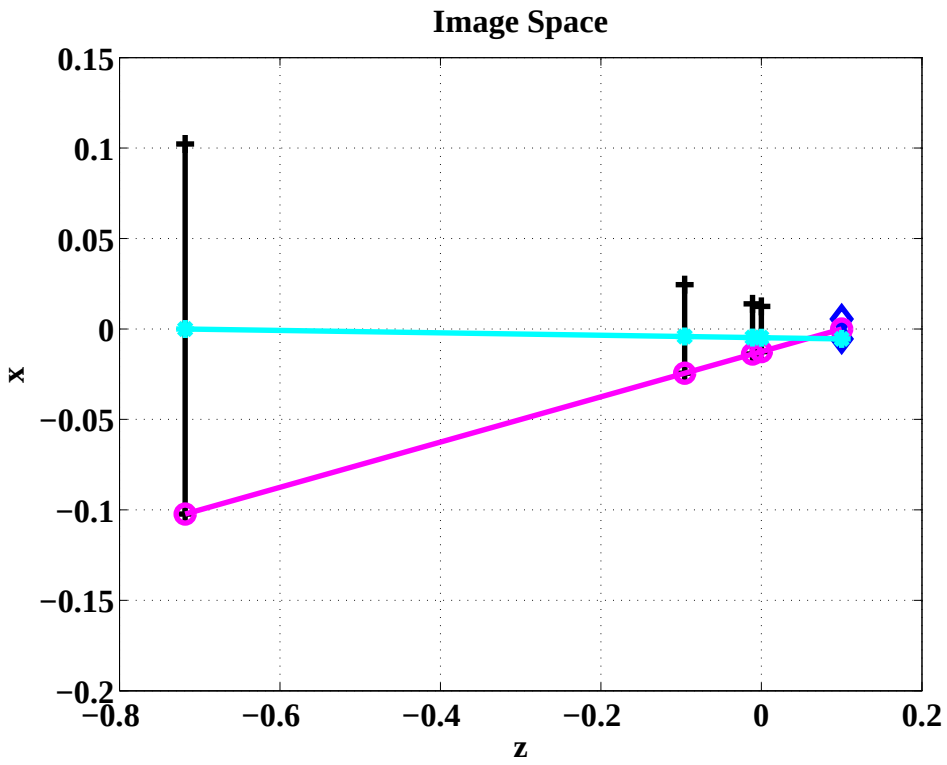
For the zoom lens in Problem 2, locate the images of all the lenses in image space. Use the mid–point configuration, where $z_{12} = 25$ mm. Identify each as real or virtual. Continue to assume thin lenses.

Lens 3, seen through Lens 4, with $f = f_4 = 100$ mm and $s = z_{34} = 10$ mm;

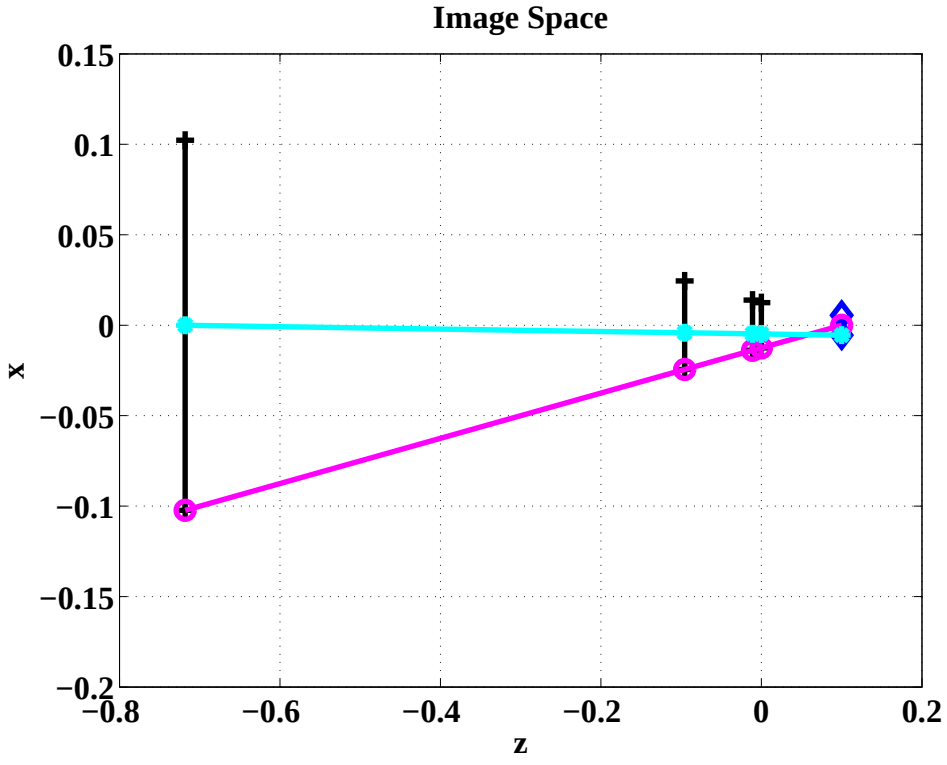
$$\frac{1}{s} + \frac{1}{s'} = \frac{1}{f} \qquad s' = -11 \text{ mm}$$

$L3''$ is 11 mm before L_4 and is virtual.

Below shows the layout in image space.



Below shows the physical layout. This is just a good check on the results.



See code below.

```
%%% m11700mt3.m  Midterm exam zoom lens stops problem (3);

f1=70e-3;f3=f1;f2=-22.5e-3;f4=100e-3;
z12=25e-3;z23=25e-3;z34=10e-3;

% Global coordinates of vertices
zV4=0;
zV3=zV4-z34;
zV2=zV3-z23;
zV1=zV2-z12;

m1=[1,0;-1/f1,1];
m2=[1,0;-1/f2,1];
m3=[1,0;-1/f3,1];
m4=[1,0;-1/f4,1];

t12=[1,z12;0,1];
t23=[1,z23;0,1];
t34=[1,z34;0,1];

% Fourth lens is last one sequentially so it is in image space
zSdoubleprime4=zV4
mlens4image=1 % Magnification from physical device to image space.
              % Note that we don't care about the sign here because we
              % are just trying to find the diameter.  However, it's
              % nice if we want to plot some rays as we do later.
% Image of third lens through fourth at -11mm.  See solution text.
s=z34;
sprime=1/(1/f4-1/s);
zSdoubleprime3=zV4+sprime
mlens3image=-sprime/s % Magnification from physical device to image space

% Image of second lens through third and fourth
m34=m4*t34*m3;
h34=(1-m34(2,2))/m34(2,1); % n=nprime=1;
hprime34=(1-m34(1,1))/m34(2,1);
f34=-1/m34(2,1);
zH34=zV3-h34;
zHprime34=zV4+hprime34;
s=zH34-zV2;
sprime=1/(1/f34-1/s);
zSdoubleprime2=zHprime34+sprime
mlens2image=-sprime/s % Magnification from physical device to image space
```

```
% Image of first lens through second, third, and fourth
m234=m4*t34*m3*t23*m2;
h234=(1-m234(2,2))/m234(2,1); % n=nprime=1;
hprime234=(1-m234(1,1))/m234(2,1);
f234=-1/m234(2,1);
zH234=zV2-h234;
zHprime234=zV4+hprime234;
s=zH234-zV1;
sprime=1/(1/f234-1/s);
zSdoubleprime1=zHprime234+sprime
mlens1image=-sprime/s % Magnification from physical device to image space

% Combined Lens
m=m4*t34*m3*t23*m2*t12*m1;
h=(1-m(2,2))/m(2,1); % n=nprime=1;
hprime=(1-m(1,1))/m(2,1);
f=-1/m(2,1)
zH=zV1-h;
zHprime=zV4+hprime;

% F/4 lens with d1 as aperture stop
disp('F/4 lens with d1 as aperture stop;  Minimum diameters');
zExitPupil=zSdoubleprime1; % Choose the exit pupil as image of Lens 1
tantheta=1/8; % half angle from base of image to exit pupil edge
sprime=f; % Location of image plane
zSprime=zV4+sprime; % Location of image (specific to this problem only)
ddoubleprime1a=(zSdoubleprime1-zSprime)*2*tantheta; % in image space
d1a=ddoubleprime1a/mlens1image  % the real thing
   % dxa means dimameter, dx of lens x, based on aperture requirment

ddoubleprime2a=(zSdoubleprime2-zSprime)*2*tantheta;
d2a=ddoubleprime2a/mlens2image

ddoubleprime3a=(zSdoubleprime3-zSprime)*2*tantheta;
d3a=ddoubleprime3a/mlens3image

d4a=(zV4-zSprime)*2*tantheta
ddoubleprime4a=d4a;  % It's the real thing!

% Field of view, assuming L1 image is the exit pupil
disp(['Field of view, assuming L1 image is the exit pupil; Minimum',...
    ' diameters'])
dExitWindow=11e-3;  % 11mm chip diagonal as exit window
rchip=dExitWindow/2;
zExitWindow=zSprime;
```

```
fovhalf=rchip/(zExitPupil-zExitWindow)

% dxw means diameter, dx of lens x, based on window constraint
ddoubleprime2w=(zSdoubleprime2-zExitPupil)*2*fovhalf;
d2w=ddoubleprime2w/mlens2image

ddoubleprime3w=(zSdoubleprime3-zExitPupil)*2*fovhalf;
d3w=ddoubleprime3w/mlens3image

ddoubleprime4w=(zV4-zExitPupil)*2*fovhalf;
d4w=ddoubleprime4w/mlens4image

% Diameters must equal or exceed all requirements
% Make them all positive in image space
ddoubleprime1=abs(ddoubleprime1a);
ddoubleprime2=max(abs(ddoubleprime2a),abs(ddoubleprime2w));
ddoubleprime3=max(abs(ddoubleprime3a),abs(ddoubleprime3w));
ddoubleprime4=max(abs(ddoubleprime4a),abs(ddoubleprime4w));
% Keep the sign in real life, just for plots
d1=ddoubleprime1/mlens1image;
d2=ddoubleprime2/mlens2image;
d3=ddoubleprime3/mlens3image;
d4=ddoubleprime4/mlens4image;


figure;
% Black lines are lenses, blue is image
% magenta is boundary of NA

plot(zSdoubleprime1*[1,1],ddoubleprime1*[-1,1]/2,'k-+',...
     zSdoubleprime2*[1,1],ddoubleprime2*[-1,1]/2,'k-+',...
     zSdoubleprime3*[1,1],ddoubleprime3*[-1,1]/2,'k-+',...
     zSdoubleprime4*[1,1],ddoubleprime4*[-1,1]/2,'k-+',...
     zExitWindow*[1,1],dExitWindow*[-1,1]/2,'b-d',...
     [zExitWindow,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
 zSdoubleprime1],...
     ([zExitWindow,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
 zSdoubleprime1]-zExitWindow)*tantheta,'m-o',...
           [zExitWindow,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
 zSdoubleprime1],...
     ([zExitWindow,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
 zSdoubleprime1]-zExitPupil)*fovhalf,'c-*');grid on;
xlabel('z'),ylabel('x');title('Image Space');
%print -depsc 11700mt3a.eps
```

```
figure;
plot(zV1*[1,1],d1*[-1,1]/2,'k-+',...
     zV2*[1,1],d2*[-1,1]/2,'k-+',...
     zV3*[1,1],d3*[-1,1]/2,'k-+',...
     zV4*[1,1],d4*[-1,1]/2,'k-+',...
     zExitWindow*[1,1],dExitWindow*[-1,1]/2,'b-d');grid on;
 xlabel('z'),ylabel('x');title('Physical Layout');

plot(zV1*[1,1],d1*[-1,1]/2,'k-+',...
     zV2*[1,1],d2*[-1,1]/2,'k-+',...
     zV3*[1,1],d3*[-1,1]/2,'k-+',...
     zV4*[1,1],d4*[-1,1]/2,'k-+',...
     zExitWindow*[1,1],dExitWindow*[-1,1]/2,'b-d',....
     [zSprime,zV4,zV3,zV2,zV1],...
     ([zSprime,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
 zSdoubleprime1]-zSprime)*tantheta./[1,mlens4image,mlens3image,...
                                     mlens2image,mlens1image],'m-o',...
            [zSprime,zV4,zV3,zV2,zV1],...
     ([zSprime,zSdoubleprime4,zSdoubleprime3,zSdoubleprime2,...
       zSdoubleprime1]-zExitPupil)*fovhalf./[1,mlens4image,mlens3image,...
                                     mlens2image,mlens1image],'c-*');grid on;
 xlabel('z'),ylabel('x');title('Physical Layout');
%print -depsc 11700mt3b.eps
```

3.2 Aperture Stop

We want Lens 1 to be the aperture stop, so we want its image in image space to be the exit pupil. Find the diameter of the exit pupil to make the lens f/4, and then find the actual diameter of the aperture stop (Lens 1). Find the minimum diameters of the other lenses to make sure they do not become aperture stops.

3.3 Field Stop

We want the diagonal of the imaging chip, 11 mm to limit the field of view. Every circular aperture should subtend an angle from the exit pupil that is larger than the angle subtended by the chip diagonal. Compute the minimum diameters of the lenses. The final design must have diameters that satisfy this condition and the one you obtained in Part 3.2.

4 Thick Lenses

Finally, let's put this zoom lens together using thick lenses. Assume that you have a collection of lenses with the desired focal lengths that are bi–convex, bi–concave, plano–convex, and plano–concave. Pick the best choice for each lens, and compute the vertex–to–vertex distance. Assume the lenses have vertex thicknesses of 5 mm.