Estimate of force coupling between an optic with a uniform surface charge and the metal frame on the optic cage.

B. Lantz, Oct 1, 2014, G1401179

We estimate the motion of the test mass caused by a moving metal frame (i.e. an image plane) moving near a charged optic. We show this will be a problem if the surface charge on the optic is roughly sigma >= 2e6 C/m^2. This is a Rough Estimate. Electrostatic FEA could be used to make a Better Estimate.

We do a simple matlab estimate of the spring rate (dForce/dL) caused by a uniform charge density (sigma) on the front surface of the test mass interacting with its image charge in the metal frame nearby. We assume a fixed surface charge and and moving metal frame.

This complement's Rai's calculation of a 'hopping' charge distribution and a fixed gap between the optic and reaction mass. We assume a fixed charge and moving gap; Rai uses a moving charge and fixed gap. Reality is probably that everything is charged, changing, and moving.

Summary (all forces are in the optical beam direction, L, normal to the face of the optic):
For sigma = 2e6 C/m^2:
F_0 = 1.6e-4 N * (sigma/2e6 C/m^2)^2
dF/dL = 8e-3 N/m * (sigma/2e6 C/m^2)^2
BSC ISI motion req. at 10 Hz = 2e-12 m/rtHz
Resulting force ASD on optic F_{optic} = 1.6e-14 N/rtHz
driven optic motion at 10 Hz, $x_{optic} = F_{optic}/(40\ kg\ (2\pi*10\ Hz)^2)$ = 1e-19 m/rtHz

Coupling mechanism

Basic coupling mechanism :

1) Charge on the optic face induces image charge in the nearby grounded metal frame. The force on the optic changes as the frame moves. The frame motion at10 Hz is ~1e7 times larger than the optic motion, so small couplings can be important.

2) The image charge distribution is a function of :
   a. the nominal frame and optic geometry (fixed),
   b. the distribution of charge on the optic, sigma (assumed to be uniform and fixed), and
   c. the position of the optic relative to the frame (intentionally varied along the beam direction for the calculation).

3) The charge on the optic feels a force from the image charge, F_0.

4) The force varies as a function of the optic position (the image charge changes, and the gap to the image charge also changes). This is K_{image} = dF/dL. Use +/- 1e-6 m displacement to recalculate things.

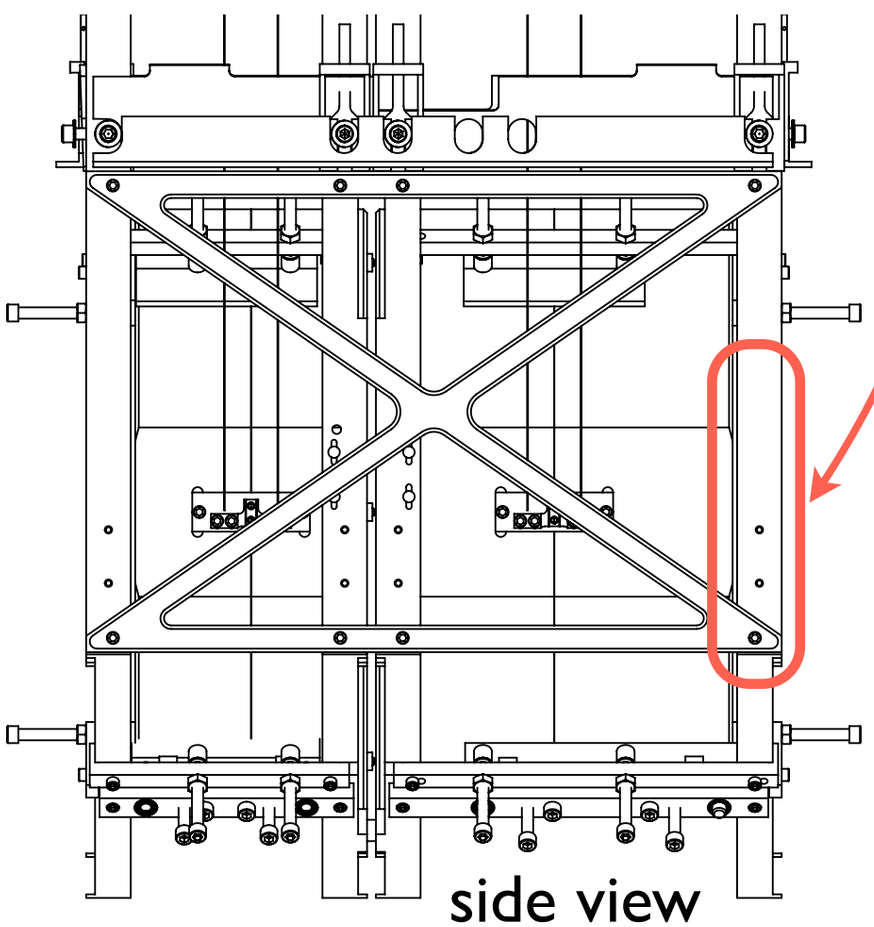
5) In reality, the frame is the piece which is moving, since it is attached to stage 2 of the BSC-ISI.

6) Given K_{image} and the motion of stage-2 optical table (x_{stg2}),
   - calculate the force on the optic $F_{optic} = K_{image} * x_{stg2}$, and
   - calculate the resulting motion of the optic, $x_{optic} = F_{optic}/ (40kg * w^2)$.

Part for calculation

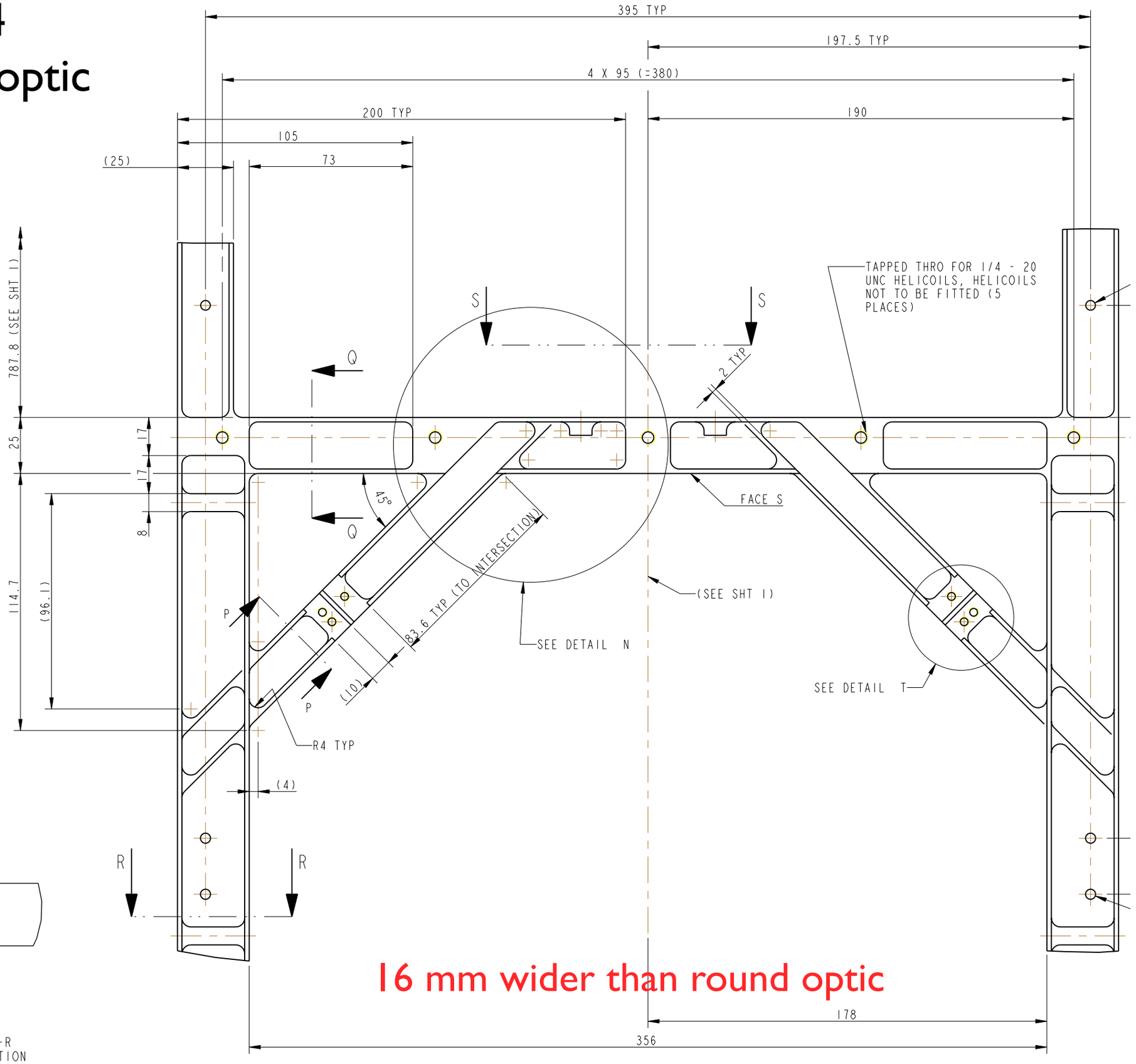
From D0901346-v12

calculate for this edge
(then x2)

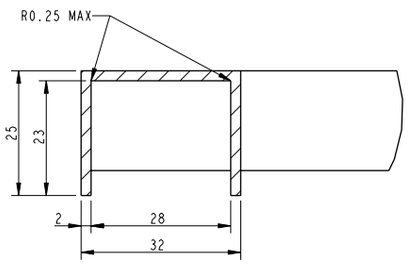


side view

Detail of D060434
Frame geometry near optic



25 mm high



PARTIAL SECTION R-R
SHOWING DEPTH OF SECTION
SCALE 2:1

R 1/4 - 20 UNC HELICOIL X 14 DEEP
IL NOT TO BE FITTED
CES - ONE ON OTHER SIDE OF PLATE)

16 mm wider than round optic

DETAIL M
(SEE SHT 1)
SCALE 1:1

Details about calculation

Details:

1) Assume sigma = 1e-6 C/m^2 for the calculations. The image charge will scale as sigma, whereas the force and spring rate (K_{image}) will scale as sigma^2. It is likely true that the front surface charge distribution is not uniform, because the charge affects in the 4 quadrants of the electrostatic drive are quite different.

2) The closest large piece of metal is the frame at the front side of the optic. The piece jutting out in front of the optic seems to have the greatest effect. The forces from the piece moving out to the side are much smaller, and so ignored for now.

3) These calculations assume that the image charge in a small piece of metal will be the same as for the image charge on that same piece of metal in an infinite ground plane. This is clearly not true at the edges. The edges are the part closest to the optic in this calculation, and so the whole calculation is quite suspect.

4) If an accurate answer is desired, a more sophisticated model will be required.

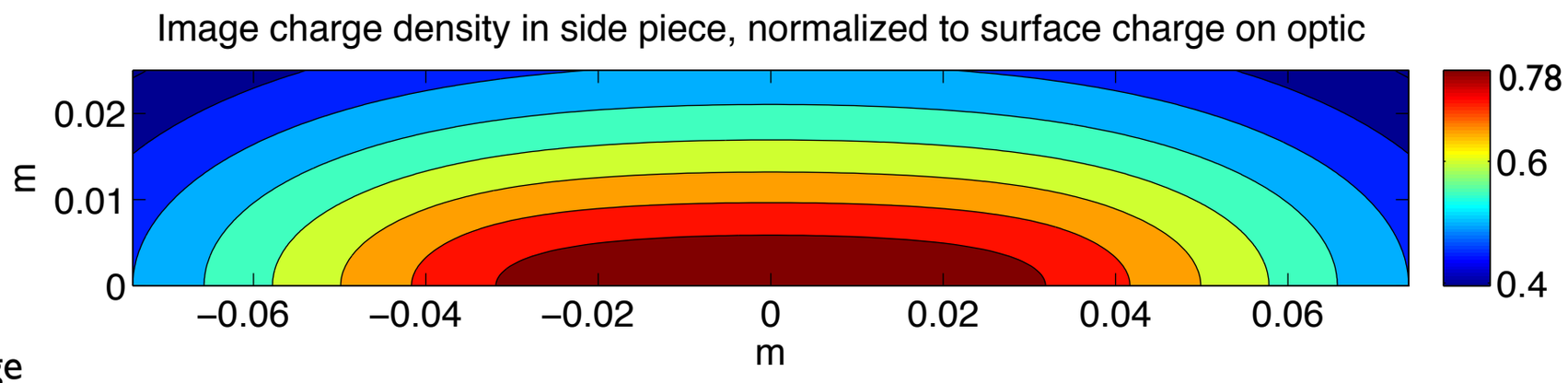
5) Since the sides are the closest, we just use the force from the 2 sides (i.e. twice the force from 1 side).

6) Done in Matlab. image plane is 100 x 100 grid. optic is grid with 2.25 mm spacing, first 2 rows are skipped to mimic the 6.75 mm cut-off to make the flat where the ears go. Careful 'fence post' analysis has not been done, see (4) above.

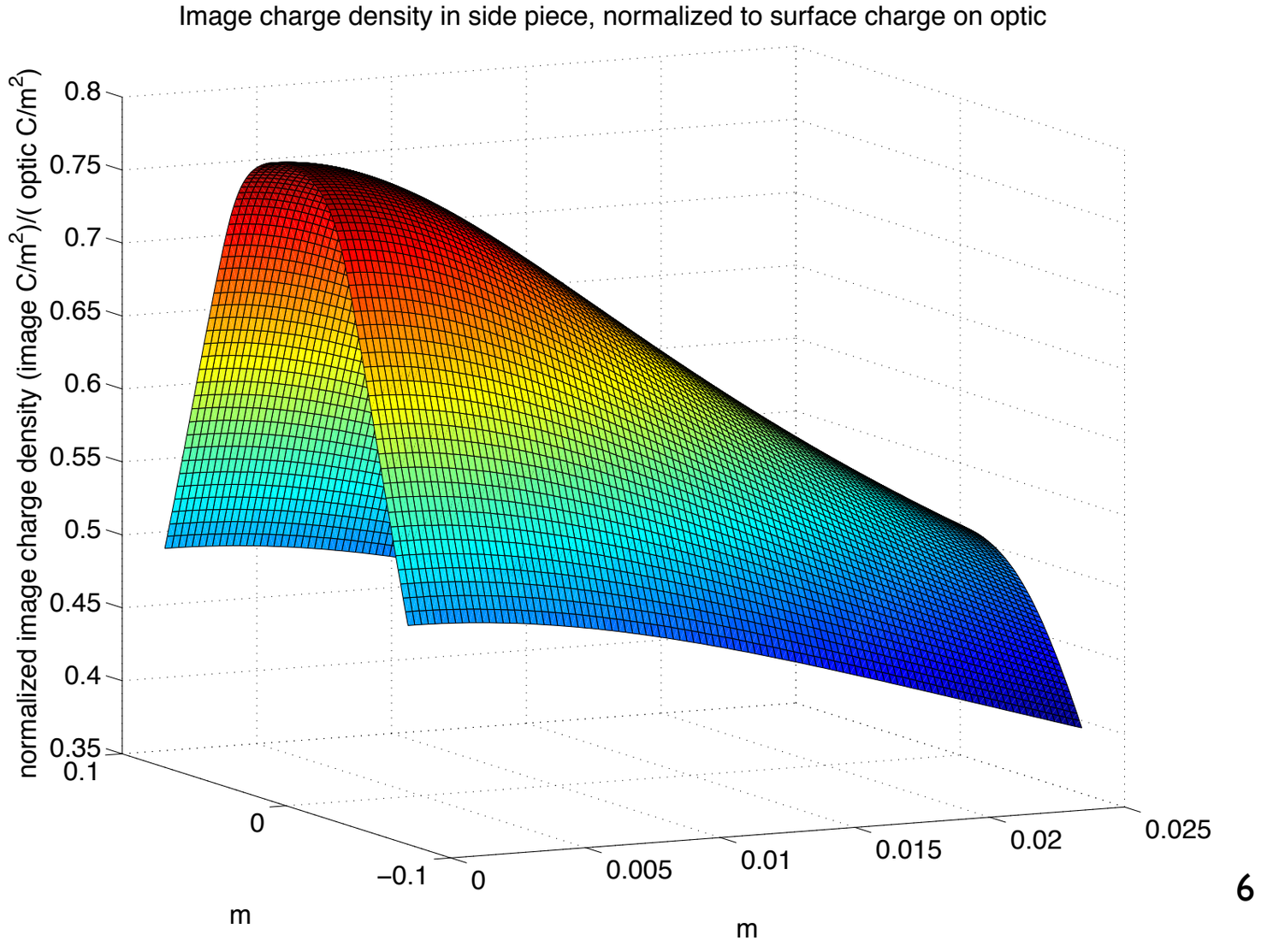
7) Only calculate for 1/2 the optic. Forces drop off pretty quickly.

8) Calculations done with script *charged_optic_face.m* and functions *image_charge_rectangle.m* and *image_charge_force.m*.

Image charge



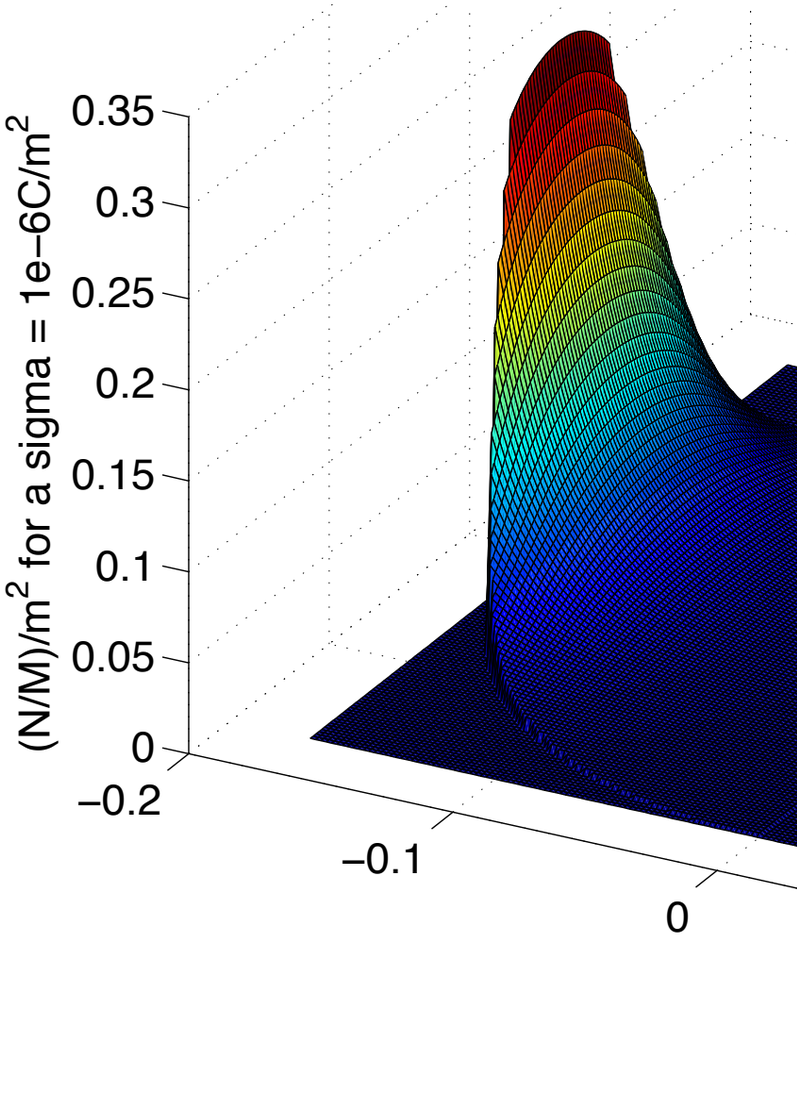
These are 2 views of the map of the image charge density which appears on the side plate. It has been normalized by the surface charge density on the optic. The max is 0.78 of the surface charge density, and appears next to the close edge.

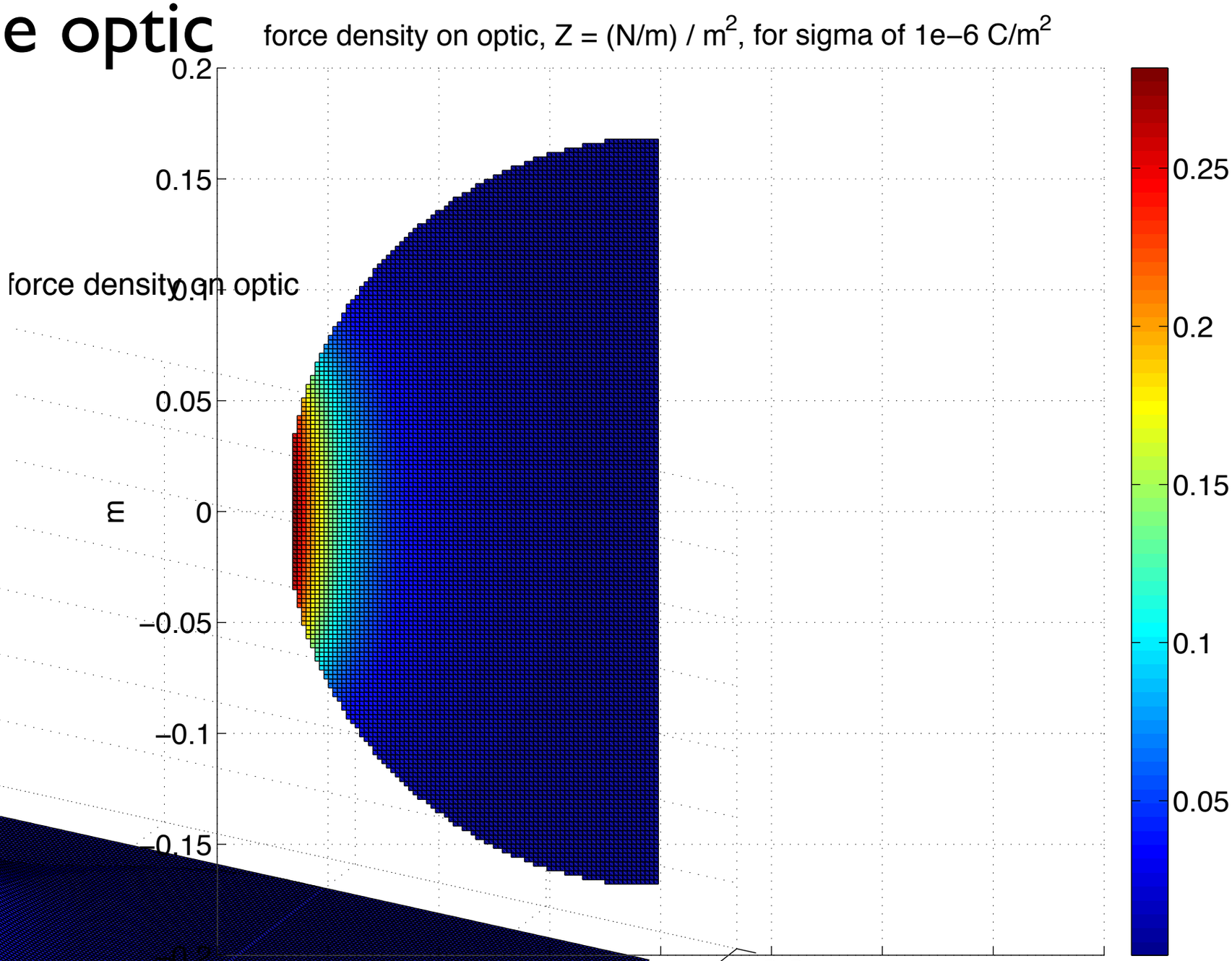


dForce/dL on the optic

These are 2 views of the map of the dF/dL distribution on the optic. The electro-static spring is dominated by the part of the optic closest to the image plane, as expected. Units are dF/dL per m^2 for optic surface charge of 1e-6 C/m^2. These are the grid elements used for the calculation.

When the grid elements are summed up, dF/dL = 2.3e-3 N/m for one side plate. L is normal to the optic face. Lateral forces are also calculated, but ignored.

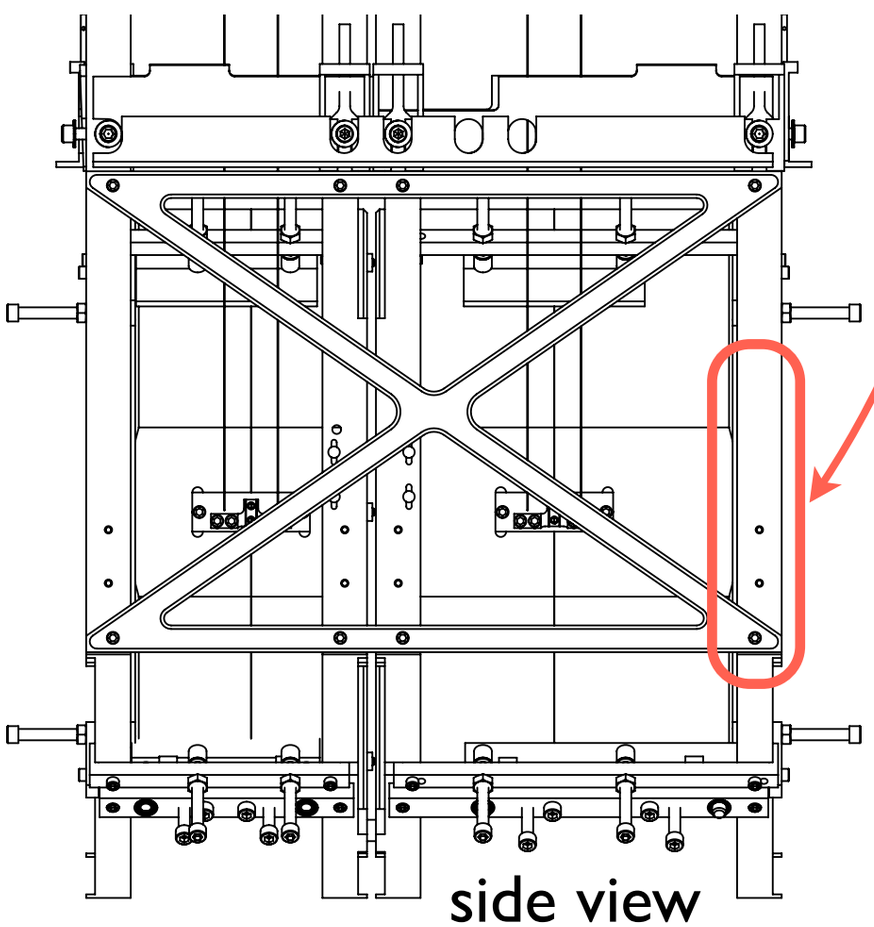




Drawing with force map overlay

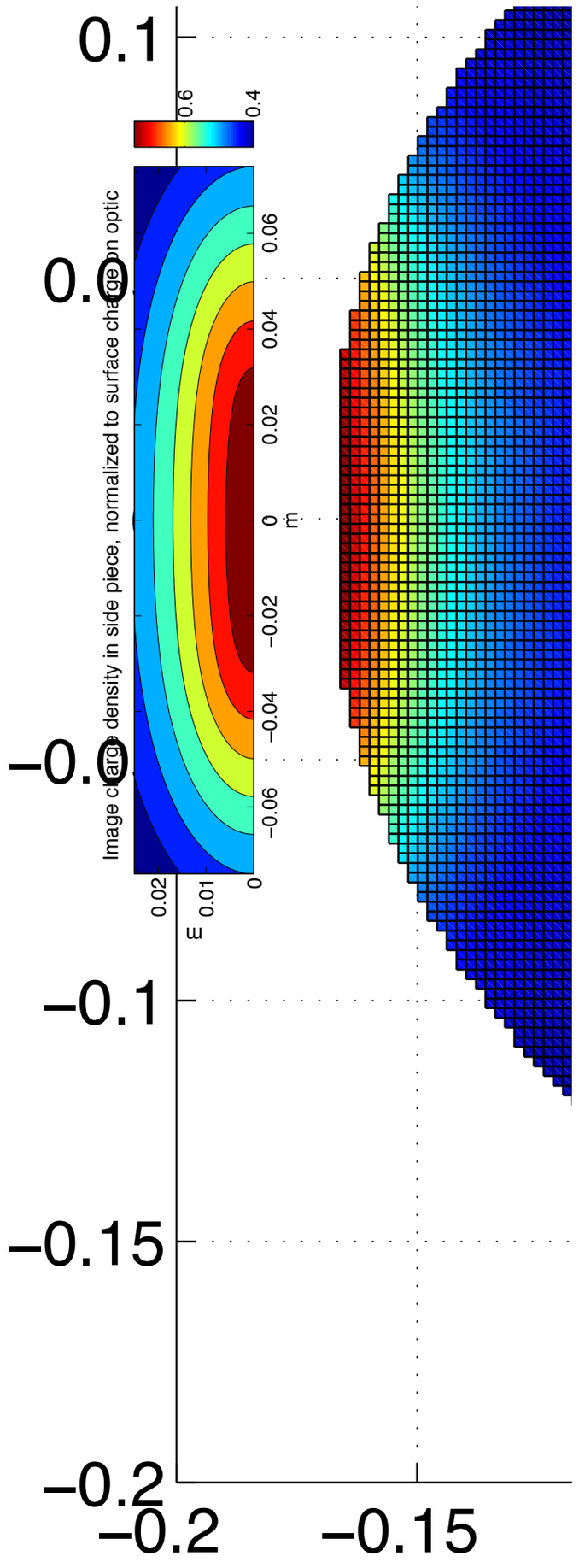
From D0901346-v12

calculate for this edge
(then x2)



side view

The frame is 8 mm from the edge of a round optic. About 6.75 mm of the round optic has been removed from each side to create a flat place to mount the ear. On the right, I've put the two images (force on optic and charge on image plane in approximately the correct relative location and scale. Note - the close edge of the image plane is in the correct place, the rest of it should be coming up, out of the page.



Trick: Forget about the actual problem; we're going to study a *completely different* situation for a moment. This new problem consists of *two* point charges, $+q$ at $(0, 0, d)$ and $-q$ at $(0, 0, -d)$, and *no* conducting plane (Fig. 3.11). For this configuration I can easily write down the potential:

$$V(x, y, z) = \frac{1}{4\pi\epsilon_0}\left[\frac{q}{\sqrt{x^2 + y^2 + (z - d)^2}} - \frac{q}{\sqrt{x^2 + y^2 + (z + d)^2}}\right] \tag{3.8}$$

(The denominators represent the distances from (x, y, z) to the charges $+q$ and $-q$, respectively.) Notice that

1. $V = 0$ when $z = 0$
2. $V \to 0$ for $x^2 + y^2 + z^2 \gg d^2$

and the only charge in the region $z > 0$ is the point charge $+q$ at $(0, 0, d)$. But these are precisely the conditions of our original problem! Evidently the "upper half" ($z \geq 0$) of the second configuration happens to have exactly the same potential as the first configuration. (The "lower half," $z < 0$, is completely different, but who cares? The upper half is all we need, and it fits our specifications.) *Conclusion*: The potential of a point charge above an infinite grounded conductor is given by equation (3.8), with $z \geq 0$.

Notice the crucial role played by the uniqueness theorem in this argument: without it, no one would believe this solution, since it was, after all, obtained for a completely different charge distribution. Yet the uniqueness theorem assures us that if it satisfies Poisson's equation in the region of interest and assumes the correct value at the boundaries, then it must be right.

3.2.2 The Induced Surface Charge

Now that we know the potential, it is a straightforward matter to compute the surface charge σ induced on the conductor. According to equation (2.42),

$$\sigma = -\epsilon_0 \frac{\partial V}{\partial n}$$

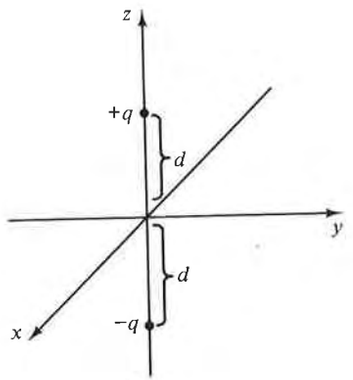


Figure 3.11

where $\partial V/\partial n$ is the normal derivative of V at the surface. In this instance, the normal direction is the z-direction, so

$$\sigma = -\epsilon_0 \frac{\partial V}{\partial z}\bigg|_{z=0}$$

From equation (3.8),

$$\frac{\partial V}{\partial z} = \frac{1}{4\pi\epsilon_0}\left\{\frac{-q(z - d)}{[x^2 + y^2 + (z - d)^2]^{3/2}} + \frac{q(z + d)}{[x^2 + y^2 + (z + d)^2]^{3/2}}\right\}$$

so

$$\sigma(x, y) = \frac{-qd}{2\pi(x^2 + y^2 + d^2)^{3/2}} \tag{3.9}$$

As expected, the induced charge is negative (assuming q is positive) and greatest at $x = y = 0$.

While we're at it, let's compute the *total* induced charge,

$$Q = \int \sigma \, da$$

This integral, over the xy plane, could be done in Cartesian coordinates, with $da = dx\, dy$, but it's a little easier to use plane polar coordinates (r, θ), with $r^2 = x^2 + y^2$ and $da = r\, dr\, d\theta$. Then,

$$\sigma(r) = \frac{-qd}{2\pi(r^2 + d^2)^{3/2}}$$

and

$$Q = \int_0^{2\pi}\int_0^{\infty} \frac{-qd}{2\pi(r^2 + d^2)^{3/2}} r\, dr\, d\theta = -qd\left(\frac{-1}{\sqrt{r^2 + d^2}}\right)\bigg|_0^{\infty} = -q \tag{3.10}$$

So the total charge induced on the plane is $-q$ as, with benefit of hindsight, you can perhaps convince yourself it *had* to be.

3.2.3 Force and Energy

The charge q is attracted toward the plane, because of the negative induced charge. Let's calculate the force of attraction. Since the potential in the vicinity of q is the same as in the analog problem (the one with $+q$ and $-q$ but no conductor), so also is the field and, therefore, the force as well:

$$\mathbf{F} = -\frac{1}{4\pi\epsilon_0}\frac{q^2}{(2d)^2}\hat{k} \tag{3.11}$$

Beware: It is easy to get careless and assume that *everything* is the same in the two problems; the energy, however, is *not* the same. With the two point charges and no conductor the energy is, according to equation (2.36),

$$W = -\frac{1}{4\pi\epsilon_0}\frac{q^2}{(2d)} \tag{3.12}$$

```
%% calculate the effect of cage motion on the optic

clear
close all
disp(['  running ',mfilename,' on ',date])

%%
optic_rad = 0.34/2;
sigma = 1e-6; % avg charge density, C/m^2;

[optic.x, optic.y] = circle(optic_rad,200);

% make the optic dims x and y for transverse dims.
% there are several small plates around the optic.
% we will calculate them each one at a time,
% then add them up.

%% 8 plates are parallel with optic surface, and next to it.
% they come in pairs, only calc. each type one time

%pfake is to set up the calcs. the dims are not real
% pside is the side piece going out from front of optic,
% use it for also for the piece on the side going out to the side.
% ie the first calc.

pside.width = 0.148;
pside.height = 0.025;
pside.gap_to_optic = 0.008; % true if optic really round. actally 6.75mm shaved off side, see y_start
pside.z = 0.001;

pcap.width = 10*optic_rad;
pcap.height = 10*optic_rad;
pcap.gap_to_optic = -6*optic_rad;
pcap.z   = 5*optic_rad;

step_size = 0.002; % this is an estimate
x_steps = ceil(2*optic_rad/step_size);% this one is real
dx = 2*optic_rad/(x_steps-1);

y_steps = x_steps;
dy = 2*optic_rad/(y_steps-1);

%% put the rectangle in the co-ords of the optic
% put the first 4 'above' the optic,
%pcalc = pfake;
pcalc = pside;

pcalc.x1 = -pcalc.width/2;
pcalc.x2 =  pcalc.width/2;
pcalc.y1 = optic_rad + pcalc.gap_to_optic;
pcalc.y2 = pcalc.y1 + pcalc.height;

rect_x = [pcalc.x1, pcalc.x2, pcalc.x2, pcalc.x1, pcalc.x1];
rect_y = [pcalc.y1, pcalc.y1, pcalc.y2, pcalc.y2, pcalc.y1];
%%
figure(1)
plot(optic.x, optic.y)
hold on
plot(rect_x, rect_y);
%%
delta = 1e-6;% size of offset step for variation calcs.

Ftotal  = 0;
dFtotal = 0;
Qtotal  = 0;
dQtotal = 0;


Fmap    = zeros(x_steps, y_steps);
dFmap   = zeros(x_steps, y_steps);
dFmap_Z = zeros(x_steps, y_steps);
Qonface = 0;

% NOTE  charge_map is the charge on the image plane section
% in this calc, we are scaling it by the surface charge on the
% optic, so it is the total actual charge map
% However, it ignores other image plane areas, and the edges are probably
% wrong.
start_y = round(0.00675/step_size);
```

```
disp(['skip the first ',num2str(start_y),' steps for y'])

%% first - calc the total charge on the grounded rectangle
tic
map_size = 100;
charge_map      = zeros([map_size, map_size]);
charge_map_plus  = zeros([map_size, map_size]);
charge_map_minus = zeros([map_size, map_size]);

disp('calculate charge in image frame rect')
for yy = start_y:ceil(y_steps/2);
   y =  optic_rad - dy*(yy - 1); % count down from the top - most of the optic doesn't matter
   for xx = 1:x_steps;
      x =  -optic_rad + dx*(xx - 1);
      if (x^2 + y^2) > optic_rad^2
         %do nothing
      else
         qq = sigma * dx * dy;  % charge in this grid point;
         Qonface = Qonface + qq;
         [Qnom, this_Qmap_nom ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y ,pcalc.z+0, map_size);
         [Qplus, this_Qmap_plus ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y, pcalc.z+delta, map_size);
         [Qminus, this_Qmap_minus ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y, pcalc.z-delta, map_size);

         Qtotal  = Qtotal + qq * Qnom;
         dQtotal = dQtotal + qq * (Qplus - Qminus)/(2*delta);
         charge_map      = charge_map      + qq * this_Qmap_nom;
         charge_map_plus  = charge_map_plus  + qq * this_Qmap_plus;
         charge_map_minus = charge_map_minus + qq * this_Qmap_minus;
      end
   end
   if round(yy/10) == (yy/10)
      disp(['finish y loop ',num2str(yy), ' of ',num2str(y_steps)])
   end
end
disp(['elapsed time is ',num2str(toc),' secs'])
%%
figure
surf(charge_map)
title('charge in the image plate')

%% then go through again and calculate the forces!

tic
disp('calculate force to them image frame rect')
for yy = start_y:ceil(y_steps/2);
   y =  optic_rad - dy*(yy - 1); % count down from the top - most of the optic doesn't matter
   for xx = 1:x_steps;
      x =  -optic_rad + dx*(xx - 1);
      if (x^2 + y^2) > optic_rad^2
         %do nothing
      else
         qq = sigma * dx * dy;  % charge in this grid point;
         Qonface = Qonface + qq;
         [Fnom] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y ,...
            pcalc.z+0,charge_map);
         [Fplus ] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y, ...
            pcalc.z+delta, charge_map_plus);
         [Fminus ] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, pcalc.y1-y, pcalc.y2-y, ...
            pcalc.z-delta, charge_map_minus);

         Ftotal  = Ftotal + qq * Fnom;
         thisdF = qq * (Fplus - Fminus)/(2*delta);
         dFtotal = dFtotal + thisdF;

         Fmap(xx,yy)    = sqrt(sum(Fnom.^2));
         dFmap(xx,yy)   = sqrt(sum(thisdF.^2));
         dFmap_Z(xx,yy) = thisdF(3);
      end
   end
   if round(yy/10) == (yy/10)
      disp(['finish y loop ',num2str(yy), ' of ',num2str(y_steps)])
   end
end
```

```
disp(['elapsed time is ',num2str(toc),' secs'])



%%
figure
surf(Fmap)
set(gca,'View',[0 90])
title('top view of total Force')

figure
surf(dFmap_Z)
set(gca,'View',[0 90])
title('dF/dZ')


%%

piece(1).Ftotal    = Ftotal;
piece(1).dFtotal   = dFtotal;
piece(1).Qtotal    = Qtotal;
piece(1).dQtotal   = dQtotal;
piece(1).Fmap      = Fmap;
piece(1).dFmap     = dFmap;
piece(1).dFmap_Z   = dFmap_Z;
piece(1).charge_map = charge_map;



%% and one going up/down


% so the x dim scales the same way,
% y goes from 0 to height
% Z is the gap + (optic_rad - y)
Ftotal  = 0;
dFtotal = 0;
Qtotal  = 0;
dQtotal = 0;


Fmap    = zeros(x_steps, y_steps);
dFmap   = zeros(x_steps, y_steps);
dFmap_Y = zeros(x_steps, y_steps);
Qonface = 0;


%% calc #3 , calc the total charge on the upright grounded rectangle
tic
map_size = 100;
charge_map      = zeros([map_size, map_size]);
charge_map_plus  = zeros([map_size, map_size]);
charge_map_minus = zeros([map_size, map_size]);

disp('calculate charge in image frame rect')
for yy = start_y:ceil(y_steps/2);
   y =  optic_rad - dy*(yy - 1); % count down from the top - most of the optic doesn't matter
   for xx = 1:x_steps;
      x =  -optic_rad + dx*(xx - 1);
      if (x^2 + y^2) > optic_rad^2
         %do nothing
      else
         z = pcalc.gap_to_optic + (optic_rad - y);
         qq = sigma * dx * dy;  % charge in this grid point;
         Qonface = Qonface + qq;
         [Qnom, this_Qmap_nom,dA_chg,dx_chg, dy_chg ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, 0, pcalc.height ,z, map_size);
         [Qplus, this_Qmap_plus ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, 0+delta, pcalc.height+delta, z, map_size);
         [Qminus, this_Qmap_minus ] = ...
            image_charge_rectangle(pcalc.x1-x, pcalc.x2-x, 0-delta, pcalc.height-delta, z, map_size);

         Qtotal  = Qtotal + qq * Qnom;
         dQtotal = dQtotal + qq * (Qplus - Qminus)/(2*delta);
         charge_map      = charge_map      + qq * this_Qmap_nom;
         charge_map_plus  = charge_map_plus  + qq * this_Qmap_plus;
         charge_map_minus = charge_map_minus + qq * this_Qmap_minus;
```

```
      end
   end
   if round(yy/10) == (yy/10)
      disp(['finish y loop ',num2str(yy), ' of ',num2str(y_steps)])
   end
end
disp(['elapsed time is ',num2str(toc),' secs'])
%%
figure
surf(-1e8*charge_map)
title('charge in the image plate')

%% then go through again and calculate the forces!

tic
disp('calculate force to them image frame rect')
for yy = start_y:ceil(y_steps/2);
   y =  optic_rad - dy*(yy - 1); % count down from the top - most of the optic doesn't matter
   for xx = 1:x_steps;
      x =  -optic_rad + dx*(xx - 1);
      if (x^2 + y^2) > optic_rad^2
         %do nothing
      else
         z = pcalc.gap_to_optic + (optic_rad - y);
         qq = sigma * dx * dy;  % charge in this grid point;
         Qonface = Qonface + qq;
         [Fnom] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, 0, pcalc.height, z, charge_map);
         [Fplus ] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, 0+delta, pcalc.height+delta, z, charge_map_plus);
         [Fminus ] = ...
            image_charge_force(pcalc.x1-x, pcalc.x2-x, 0-delta, pcalc.height-delta, z, charge_map_minus);

         Ftotal  = Ftotal + qq * Fnom;
         thisdF = qq * (Fplus - Fminus)/(2*delta);
         dFtotal = dFtotal + thisdF;

         Fmap(xx,yy)    = sqrt(sum(Fnom.^2));
         dFmap(xx,yy)   = sqrt(sum(thisdF.^2));
         dFmap_Y(xx,yy) = thisdF(2);
      end
   end
   if round(yy/10) == (yy/10)
      disp(['finish y loop ',num2str(yy), ' of ',num2str(y_steps)])
   end
end
disp(['elapsed time is ',num2str(toc),' secs'])
%%
x_vect = -optic_rad + dx*((1:x_steps) - 1);
y_vect =  optic_rad - dy*((1:y_steps) - 1);
[XX,YY] = meshgrid(x_vect, y_vect); % funny defn for 3d plots.

x_vect_chg = pcalc.x1 + dx_chg * ((1:map_size)-1);
y_vect_chg = 0 + dy_chg * ((1:map_size)-1);

% flip the X and Y axes so that the plot is aligned with the real part
% ie the long axis is up/down. like the part in the real system.
[XXchg, YYchg] = meshgrid(y_vect_chg, x_vect_chg);
%% store the stuff for this particular piece into the big data struct
% can be used to enable loops or multiple pieces later.

piece(2).Ftotal     = Ftotal;
piece(2).dFtotal    = dFtotal;
piece(2).Qtotal     = Qtotal;
piece(2).dQtotal    = dQtotal;
piece(2).Fmap       = Fmap;
piece(2).dFmap      = dFmap;
piece(2).dFmap_Z    = dFmap_Z;
piece(2).dF_beamdir = dFtotal(2);
piece(2).dFmap_beamdir = dFmap_Y;
piece(2).charge_map = charge_map;
piece(2).dx_chg = dx_chg;
piece(2).dy_chg = dy_chg;
piece(2).dA_chg = dA_chg;
piece(2).x_vect_chg = x_vect_chg;
piece(2).y_vect_chg = y_vect_chg;
piece(2).XXchg = XXchg;
piece(2).YYchg = YYchg;
%%
```

```
img_chg_iso_fig = figure;

surf(piece(2).XXchg,piece(2).YYchg,...
   (-1/sigma)*(piece(2).charge_map)/piece(2).dA_chg)
view([-26,10])
xlabel('m'); ylabel('m')
zlabel('normalized image charge density (image C/m^2)/(optic C/m^2)')
title('Image charge density in side piece, normalized to surface charge on optic   ')
FillPage('w')

%%
img_chg_top_fig = figure;

contourf(piece(2).YYchg.',piece(2).XXchg.',...
   (-1/sigma)*(piece(2).charge_map.')/piece(2).dA_chg)
view([0,90])
colorbar
xlabel('m')
ylabel('m')
axis image
title(' Image charge density in side piece, normalized to surface charge on optic   ')
FillPage('w')


peak_sigma = min(min((1/sigma)*(piece(2).charge_map)/piece(2).dA_chg));
%%

disp(['Peak induced image charge is ',num2str(peak_sigma),' of the sigma on the optic face'])
disp(['Force is ',num2str(Ftotal(2),'%1.1e'),' Newtons for optic sigma of ',num2str(sigma),' C/m^2'])
disp(' ')
disp('in the beam direction (L),');
disp(['dF/dL is ',num2str(dFtotal(2),'%1.1e'),' Newtons/meter for sigma of ',num2str(sigma),' C/m^2'])

%%


% maybe this helps the plotting? 0 plots, but NaN does not plot.

piece(2).dFmap_beamdir_clean = piece(2).dFmap_beamdir;

for ii = 1:size(piece(2).dFmap_beamdir_clean,1)
   for jj = 1:size(piece(2).dFmap_beamdir_clean,2)
      if piece(2).dFmap_beamdir_clean(ii,jj) == 0;
         piece(2).dFmap_beamdir_clean(ii,jj) = NaN;
      end
   end
end

%%
dA = dx * dy;

force_den_iso_fig = figure;
surf(XX,YY,-piece(2).dFmap_beamdir/dA)
xlabel('m')
ylabel('m')
zlabel('(N/M)/m^2 for a sigma = 1e-6C/m^2')
title('force density on optic')
view([28,38])
FillPage('w')


%%

force_den_top_fig = figure;
surf(XX,YY,-piece(2).dFmap_beamdir_clean/dA)
xlabel('m')
ylabel('m')
zlabel('(N/M)/m^2 for a sigma = 1e-6C/m^2')
title(' force density on optic, Z = (N/m) / m^2, for sigma of 1e-6 C/m^2  ')

view([0,90])
axis square
colorbar
FillPage('w')


%%
```

```
break
%%
figure(img_chg_iso_fig)
print -dpdf fig_img_chg_iso.pdf

figure(img_chg_top_fig)
print -dpdf fig_img_chg_top.pdf

figure(force_den_top_fig)
print -dpdf fig_force_den_top.pdf

figure(force_den_iso_fig)
print -dpdf fig_force_den_iso.pdf
```

```
function [total_charge, charge_map ,dA, dx, dy ] = image_charge_rectangle(x1,x2,y1,y2,h, calc_points)
%charge_rectangle Calculates the image charge on a grounded rectangle from a point charge
% [total_charge, charge_map, dA ] = image_charge_rectangle(x1,x2,y1,y2,z,calc_points);
%  geometry is:
%    point charge at [x,y,z] = [0,0,h] and
%    grounded rectangle at [x,y,z] = [{x1 to x2}, {y1 to y2}, 0]
%
% distances in Meters, charge is coulombs.
% calc_points specifies how many points along the side to use (defaults to
%       100, for a 100 x 100 point grid)
% put unit charge (1 coulomb) at charge point,
% calculate:
% total_charge = the total charge accumulated on the plate per
%             coulomb at the point charge (should be <= 1).
% charge_map = map of the inducted charge.
%
% Units are (coulombs/coulomb)/ grid_point. so to get the actual charge at each
% grid point, you need to multiply by the source charge.
% to get the acutal charge density, you need to multiply by the
% source charge and divide by the area of each grid point
%
% dA is the area of each grid point (in the units of x1, x2, etc.)
% dx and dy are the spacing of the points along x and y.
%
% expressions from Griffiths, 2nd ed. sect 3.2 (method of images)
% eqn. 3.9
%
% Use image_charge_force to calculate the forces between the source and
% this image plane section.
%
% BTL, sept 2014
%

if nargin == 5
   points_on_image_plane = 100;
else
   points_on_image_plane = calc_points;
end

dx = (x2-x1)/(points_on_image_plane-1);
dy = (y2-y1)/(points_on_image_plane-1);
dA = dx * dy;

sigma = zeros(points_on_image_plane);
charge_map = zeros(size(sigma));

% sigma(xx,yy) is the charge density at index point (xx,yy) on the plane

for xx = 1:points_on_image_plane;
   x = x1 + dx*(xx - 1);
   for yy = 1:points_on_image_plane;
      y = y1 + dy * (yy-1);
      D = sqrt(x^2 + y^2 + h^2);
      sigma(xx,yy) = -h/(2*pi*(D^3));
      charge_map(xx,yy) = sigma(xx,yy) * dx * dy;
   end
end

total_charge = sum(sum(charge_map));
end
```

```
function [force_vector] = image_charge_force(x1,x2,y1,y2,h, charge_map)
%image_charge_force Calculates the force from a pre-calculated image_charge_rectangle
% [force_vector ] = image_charge_force(x1,x2,y1,y2,z,charge_map);
%  geometry is:
%    point charge at [x,y,z] = [0,0,h] and
%    grounded rectangle at [x,y,z] = [{x1 to x2}, {y1 to y2}, 0]
%
% distances in Meters, force is Newtons/(Q^2), charge is coulombs.
%
% charge_map is calculated by image_charge_rectangle
%
% put unit charge (1 coulomb) at charge point,
% calculate:
% total_charge = the total charge accumulated on the plate per
%              coulomb at the point charge (should be <= 1).
% force_vector = force between the plate and the point.
%   the force is a vector [Fx, Fy, Fz] for the forces in the x, y, z
%   direction.
%
% for a real charge of Q, then induced charge needs to be scaled by Q
% and the force by Q^2 (one for source, one for image)
% expressions from Griffiths, 2nd ed. sect 3.2 (method of images)
% eqn. 3.9
%
% BTL, sept 2014
%
 ep0 = 8.85e-12; % colm^2/N-m^2

[points_on_image_plane, check] = size(charge_map);

if points_on_image_plane ~= check
   error('charge_map has different X and Y dims, use image_charge_rectangle to make the map');
end

dx = (x2-x1)/(points_on_image_plane-1);
dy = (y2-y1)/(points_on_image_plane-1);

force_x  = zeros(size(charge_map));
force_y  = zeros(size(charge_map));
force_z  = zeros(size(charge_map));

force_vector = [0,0,0];

for xx = 1:points_on_image_plane;
   x = x1 + dx*(xx - 1);
   for yy = 1:points_on_image_plane;
      y = y1 + dy * (yy-1);
      D = sqrt(x^2 + y^2 + h^2);

      force_mag    = (1/(4*pi*ep0)) * (charge_map(xx,yy) * 1)/(D^2);
      force_x(xx,yy)  = force_mag * (x/D);
      force_y(xx,yy)  = force_mag * (y/D);
      force_z(xx,yy)  = force_mag * (h/D);
      force_vector(1)  = force_vector(1) + force_x(xx,yy);
      force_vector(2)  = force_vector(2) + force_y(xx,yy);
      force_vector(3)  = force_vector(3) + force_z(xx,yy);
   end
end


end
```