```
% hw4_campus_rect_bl.m  7-nov-2010
% rectify oblique purdue (pictometry) image bilinear interpolation

% read in the input, source image
A=imread('campus_obliq.jpg');
image(A);
axis equal
[inprow,inpcol,nband]=size(A);

om=0.859278455;
ph=0.064982673;
kp=0.051001973;
XL= 506499.554;
YL=4472522.661;
ZL=   1521.522;

Xmin=504840;
Xmax=507640;
Ymin=4473350;
Ymax=4475640;
gsd=0.5;

l0=1624;
s0=2436;
f=5524.28;
Z=155.0;
ncol=fix((Xmax-Xmin)/gsd);
nrow=fix((Ymax-Ymin)/gsd);

% create blank image to fill by rectifying and resampling A

B=zeros(nrow,ncol,3,'uint8');
%whos

% step exhaustively through the destination image
for i=1:nrow
  if (mod(i,50) == 0)
    i
    end
  for j=1:ncol
    % transform from destination to source
    XG=Xmin + (j-1)*gsd;
    YG=Ymax - (i-1)*gsd;
    ZG=Z;
    GV=[XG-XL; YG-YL; ZG-ZL];
    m=m3(kp)*m2(ph)*m1(om);
    UVW=m*GV;
    x=-f*UVW(1)/UVW(3);
    y=-f*UVW(2)/UVW(3);
    row=-y + l0;
    col=x + s0;

    % change from photoshop (0 - n-1) to matlab (1 - n)
    row=row+1;
    col=col+1;

    % resample the source image (bilinear here)
    i1=floor(row);
    i2=i1 + 1;
```

```
    j1=floor(col);
    j2=j1+1;
    % if the transformed pixel lies inside the source image
    % then resample, if outside then code as uniform gray
    if((i1 > 1) & (i2 < inprow) & (j1 > 1) & (j2 < inpcol))
    ar=[double(A(i1,j1,1)) double(A(i1,j2,1));
        double(A(i2,j1,1)) double(A(i2,j2,1))];
    ag=[double(A(i1,j1,2)) double(A(i1,j2,2));
        double(A(i2,j1,2)) double(A(i2,j2,2))];
    ab=[double(A(i1,j1,3)) double(A(i1,j2,3));
        double(A(i2,j1,3)) double(A(i2,j2,3))];
    frow=[fl(i1-row); fl(i2-row)];
    fcol=[fl(j1-col); fl(j2-col)];
    rd=frow'*ar*fcol;
    gr=frow'*ag*fcol;
    bl=frow'*ab*fcol;
    if(rd < 0)
      rd=0;
      end
    if(rd > 255)
      rd=255;
      end
    if(gr < 0)
      gr=0;
      end
    if(gr > 255)
      gr=255;
      end
    if(bl < 0)
      bl=0;
      end
    if(bl > 255)
      bl=255;
      end
    ird=uint8(rd);
    igr=uint8(gr);
    ibl=uint8(bl);

    B(i,j,1)=ird; % R
    B(i,j,2)=igr; % G
    B(i,j,3)=ibl; % B
  else
    B(i,j,1)=128; % mid-level gray tone
    B(i,j,2)=128;
    B(i,j,3)=128;
    end
  end
end
figure(2);
image(B);
axis equal
% write to ourput file
imwrite(B,'hw4_campus_rect_bl.jpg','JPEG');
```

```
% fl.m  28-oct-09
% linear interpolation kernel
function result=fl(x)
xp=abs(x);
if((xp >= 0) && (xp <= 1))
  result= -1*xp + 1;
else
  result=0;
  end
```