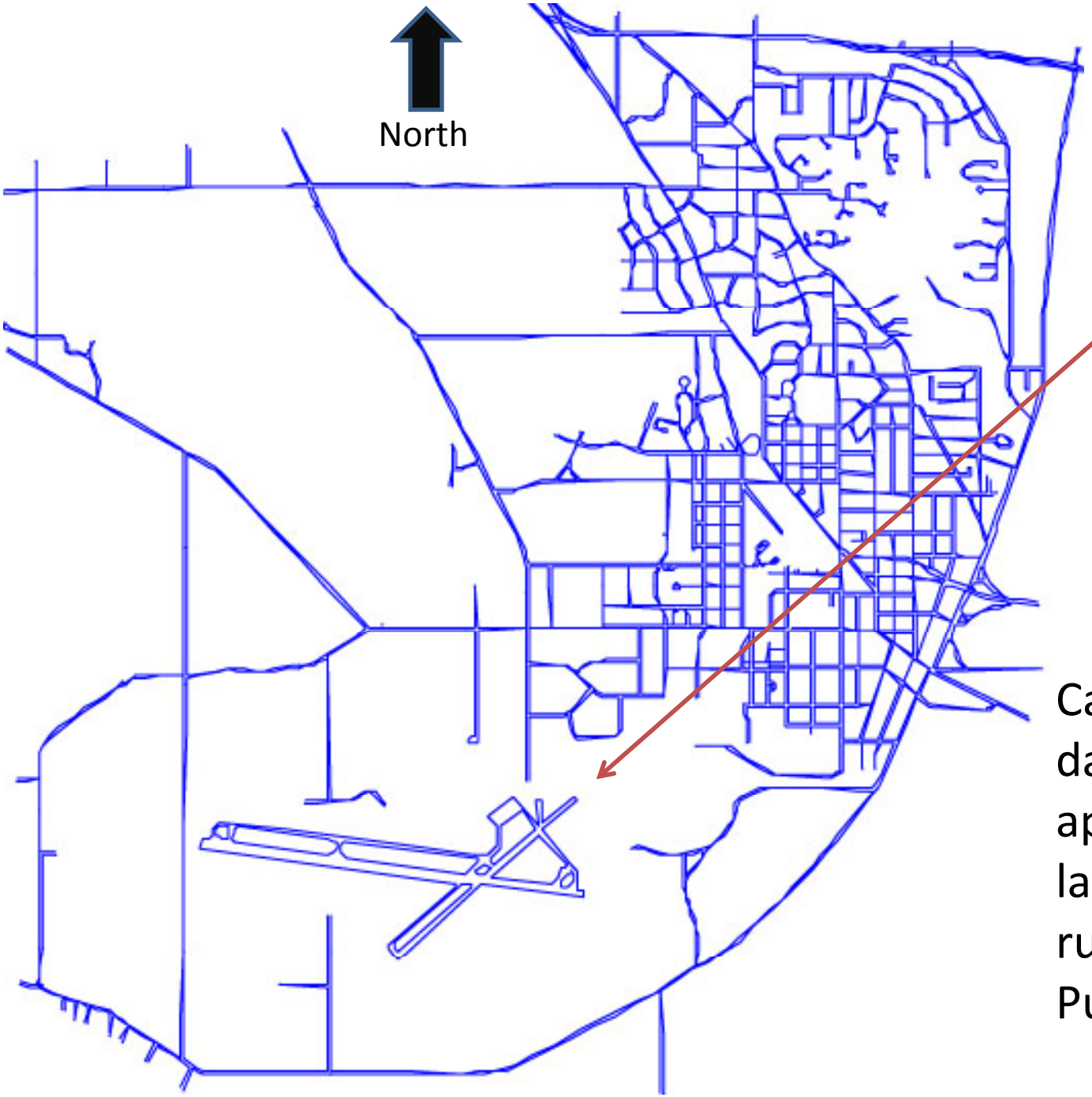
Photo-1 Homework #2

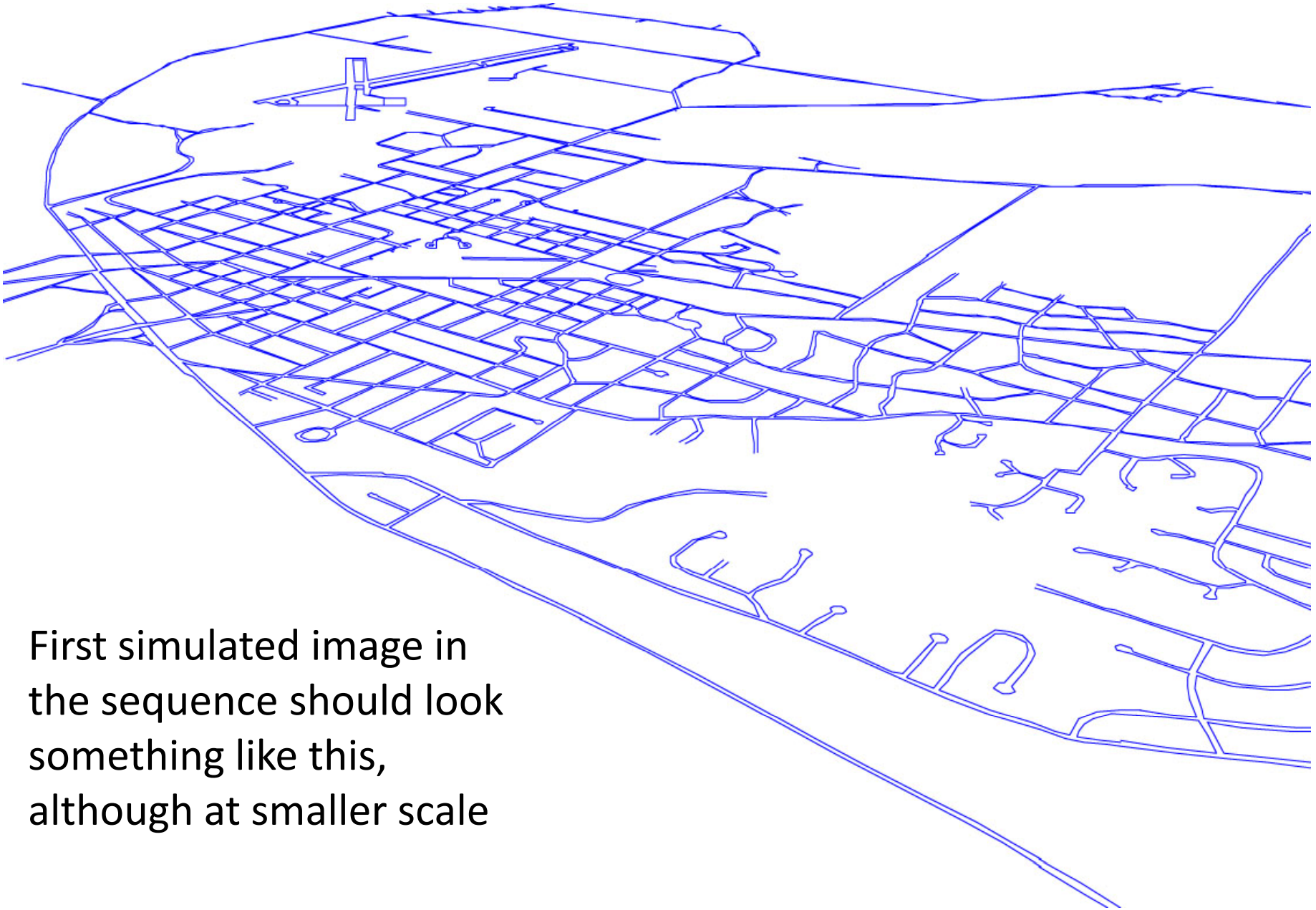
1. Retrieve the file “photo1_10_hw2.zip and unpack to find a camera trajectory file, a “shape” file of road vectors, and a “template” matlab script file which shows how to create an animation from a sequence of simulated frame images.
2. Add photogrammetry math as needed (whenever you see a *** you need to add something) to project (G2I) the road vectors into image space. You will need to create a rotation matrix once for each camera station, and you will need to evaluate the collinearity equations for each vertex point.
3. We fix all ground point elevations to 184.7 m, focal length, f=100, x0=y0=0, units of ground points (X,Y) are meters, units of angles are radians. We let matlab handle all necessary clipping.
4. There are some “if” statements in the template code to exclude points that are “behind” the camera
5. Recommend “Cinepak” for compression, see help avifile

6. When you are done, zip the .avi file, should be about 2.5 MB, and your m-file, matlab script, and email them to me.
7. Due on Tuesday, 28 Sept.



Road vectors for West Lafayette, IN found in file, roads.shp, 2D. Coordinates are UTM zone 16, meters

Camera trajectory data simulates approach and landing on runway 23 at Purdue Airport



First simulated image in the sequence should look something like this, although at smaller scale

```
% template_code.m 21-sep-10
% template for creating animation from sequence of simulated frame↙
photographs

% load the camera trajectory from 'traj.mat'
% variables are OM,PH,KP,XL,YL,ZL
% each a vector with 201 elements
% so we have a unique position and attitude for each simulated image
% we genrate 201 frames for the animation

load traj

[m,n]=size(OM); % it should be [1,201]
S=shaperead('roads');
Z=184.7; % fixed ground elevation
f=100.0; % focal length
aviobj=avifile('fly.avi','compression','Cinepak','fps',24);

% loop for each trajectory point
% when you are debugging you might want to make this n=3 or n=5
% so it goes quickly, then restore to n=201 for final run

for i=1:n
  figure(1);
  xlim([-100 100]);
  xlim('manual');
  ylim([-100 100]);
  ylim('manual');
  daspect([1 1 1]);
  axis off
  hold on

  % project and plot each of 461 polylines
  for j=1:461
    [rowdim,coldim]=size(S(j).X);
    % plot each segment in the polyline searately
    % form rotation matrix ***

    for k=1:coldim-1
      feature_in_front=1;
      % endpoints for this segment will be
      % S(j).X(k),S(j).Y(k),z   S(j).X(k+1),S(j).Y(k+1),Z
```

```
      % compute vector UVW for first endpoint ***
      if(UVW(3)>0.0)
        feature_in_front=0;
        end
      % compute xp1,yp1 using collinearity equations ***


      % compute vector UVW for second endpoint ***
      if(UVW(3)>0.0)
        feature_in_front=0;
        end
      % compute xp2,yp2 using collinearity equations ***

      if(feature_in_front == 1)
        plot([xp1 xp2],[yp1 yp2]);
        end
      end
    end
  axis off
  F=getframe(1);
  % warning - if you are multi-tasking, with other windows up while
  % matlab is running in the background, this command will grab whatever
  % graphics are on the screen!
  aviobj=addframe(aviobj,F);
  close(1);
  end
aviobj=close(aviobj);
```