Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

Question 1

Taking the partial derivative of D w.r.t to disparity d:

$$D = \frac{fB}{d}$$

$$\frac{\partial D}{\partial d} = \frac{\partial \frac{fB}{d}}{\partial d} = \frac{(fB)\partial\left(\frac{1}{d}\right)}{\partial d} = \frac{\left(-\frac{fB}{d^2}\right)\partial d}{\partial d} = -\frac{fB}{d^2}$$

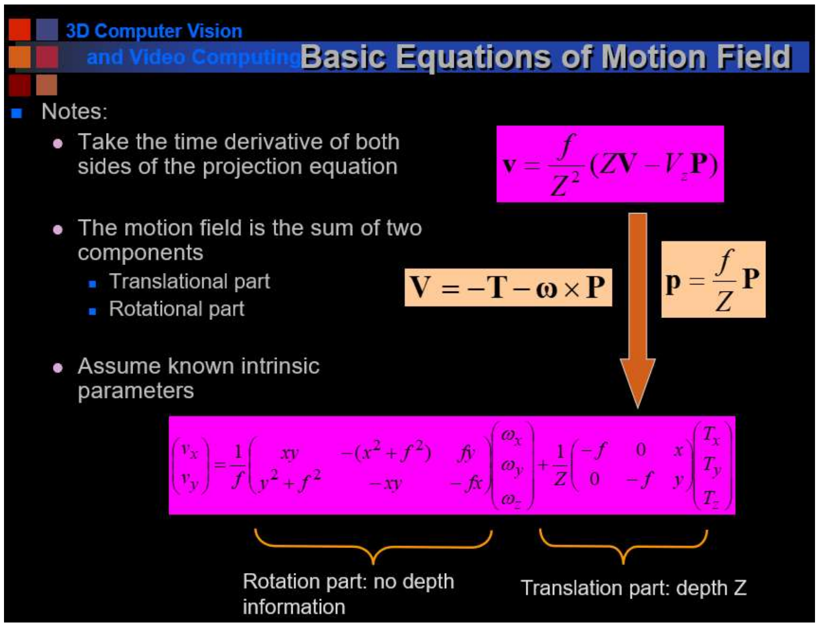
$$|\partial D| = \left|-\frac{fB}{d^2}\right|\partial d = \left|-(fB)\left(\frac{fB}{D}\right)^2\right|\partial d = \frac{D^2}{fB}\partial d$$

Conclusion:
The dependence of the error in-depth estimation of a 3D point as a function of:

- Baseline length increase means a decrease in the error (inverse relation)
- Focal length increase means decrease in error (inverse relation)
- Stereo matching error increase means increase in error (proportional relationship)
- 3D point depth increase means increase in error (proportional relationship)

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

Question 2



3D Computer Vision
and Video Computing

Basic Equations of Motion Field

- Notes:
  - Take the time derivative of both sides of the projection equation
  - The motion field is the sum of two components
    - Translational part
    - Rotational part
  - Assume known intrinsic parameters

$$\mathbf{v} = \frac{f}{Z^2}(Z\mathbf{V} - V_z\mathbf{P})$$

$$\mathbf{V} = -\mathbf{T} - \boldsymbol{\omega} \times \mathbf{P}$$

$$\mathbf{p} = \frac{f}{Z}\mathbf{P}$$

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \underbrace{\frac{1}{f}\begin{pmatrix} xy & -(x^2+f^2) & fy \\ y^2+f^2 & -xy & -fx \end{pmatrix}}_{\text{Rotation part: no depth information}}\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} + \underbrace{\frac{1}{Z}\begin{pmatrix} -f & 0 & x \\ 0 & -f & y \end{pmatrix}\begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix}}_{\text{Translation part: depth Z}}$$

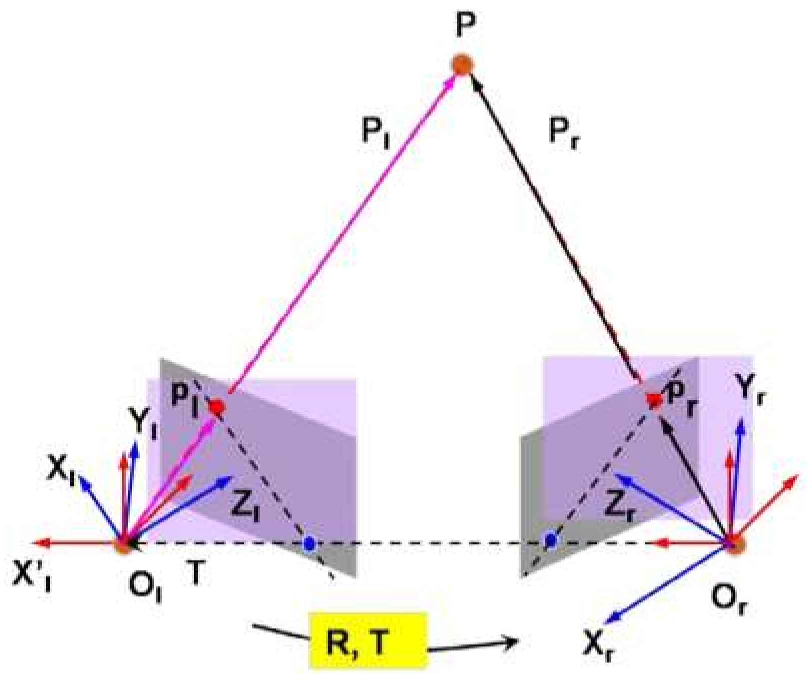
No, simply rotating about the optical center yields no depth information, it captures images of a 3D scene as if the scene were painted on a plane infinitely far away from the camera. We get no depth information since we don't have a translation between two views (i.e. we can't triangulate the multiple frames), so we don't have an 3D information. Depth cues (parallax) can only be recovered when T is nonzero.

$$v = Bw, Z \text{ is not involved}$$

Yes, since we're translating along the optical axis's (i.e. along the Baseline from O_l to O_r). We have (translation depth Z):

$$v = \frac{Tz}{Z}(x - x_0)$$

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu



$X_l' = T, \quad Y_l' = X_l' x Z_l, \qquad Z'_l = X_l' x Y_l'$

Figure 10. Stereo rectification

Then with both rotation and translation (R, T) we net 3D information among the two image planes in reference to the point P.

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

Question 3

Human:

1. Parallax and stereo motion in our eyes. For example, if we observe the same object in two different scenarios, our brains know to calibrate a weighted combination of information. The object in question would appear “large” In a small room and “small” a large room.
2. We can then relate object size to the size of nearby reference objects, like a brick wall, ignoring perceived sizes SL and SR when making their match.
3. In a static environment, performance depends on the distance of comparison cubes alone.
4. We can infer a lot of information of a blurry image sequence based on motion alone.
5. Triangulation of the human eyes, a view of profundity arises from a unique combination of a given 3D point in our left and right retinal pictures (for example when we try to directly look at our nose and “cross” our eyes).

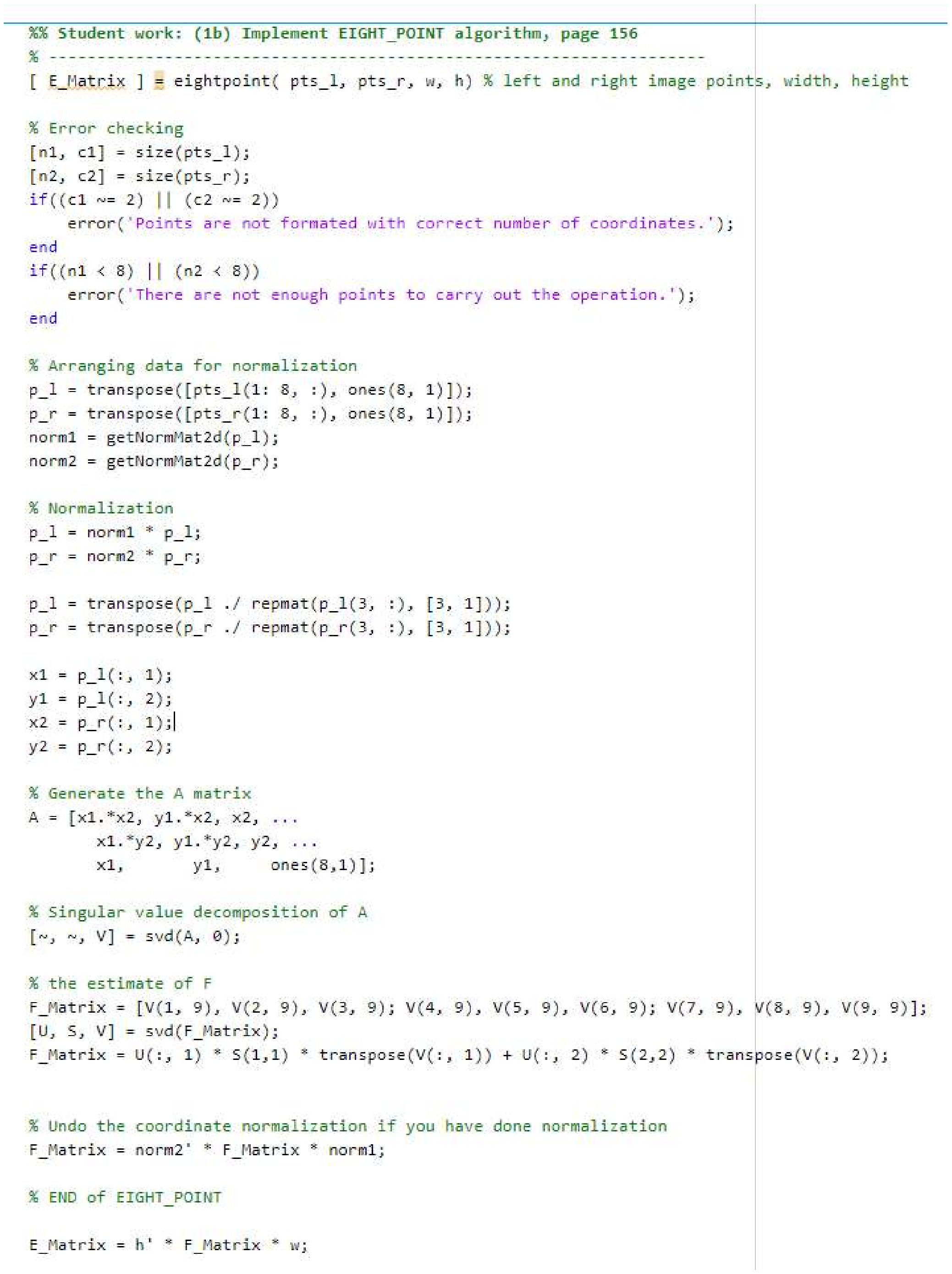
Machine:

1. Video Coding and Compression & Video View interpolation
2. Robot navigation (robot vision)
3. Laser Scanning models
4. Recovering 3D models (Computer Vision) & Image-based Rendering
5. Anaglyph from Mars Rover

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

Question 4 (attempt)

Fundamental Matrix:



```
%% Student work: (1b) Implement EIGHT_POINT algorithm, page 156
% ------------------------------------------------------------------
[ E_Matrix ] = eightpoint( pts_l, pts_r, w, h) % left and right image points, width, height

% Error checking
[n1, c1] = size(pts_l);
[n2, c2] = size(pts_r);
if((c1 ~= 2) || (c2 ~= 2))
    error('Points are not formated with correct number of coordinates.');
end
if((n1 < 8) || (n2 < 8))
    error('There are not enough points to carry out the operation.');
end

% Arranging data for normalization
p_l = transpose([pts_l(1: 8, :), ones(8, 1)]);
p_r = transpose([pts_r(1: 8, :), ones(8, 1)]);
norm1 = getNormMat2d(p_l);
norm2 = getNormMat2d(p_r);

% Normalization
p_l = norm1 * p_l;
p_r = norm2 * p_r;

p_l = transpose(p_l ./ repmat(p_l(3, :), [3, 1]));
p_r = transpose(p_r ./ repmat(p_r(3, :), [3, 1]));

x1 = p_l(:, 1);
y1 = p_l(:, 2);
x2 = p_r(:, 1);
y2 = p_r(:, 2);

% Generate the A matrix
A = [x1.*x2, y1.*x2, x2, ...
       x1.*y2, y1.*y2, y2, ...
       x1,       y1,     ones(8,1)];

% Singular value decomposition of A
[~, ~, V] = svd(A, 0);

% the estimate of F
F_Matrix = [V(1, 9), V(2, 9), V(3, 9); V(4, 9), V(5, 9), V(6, 9); V(7, 9), V(8, 9), V(9, 9)];
[U, S, V] = svd(F_Matrix);
F_Matrix = U(:, 1) * S(1,1) * transpose(V(:, 1)) + U(:, 2) * S(2,2) * transpose(V(:, 2));


% Undo the coordinate normalization if you have done normalization
F_Matrix = norm2' * F_Matrix * norm1;

% END of EIGHT_POINT

E_Matrix = h' * F_Matrix * w;
```

Fig 1. Code Eight_Point

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

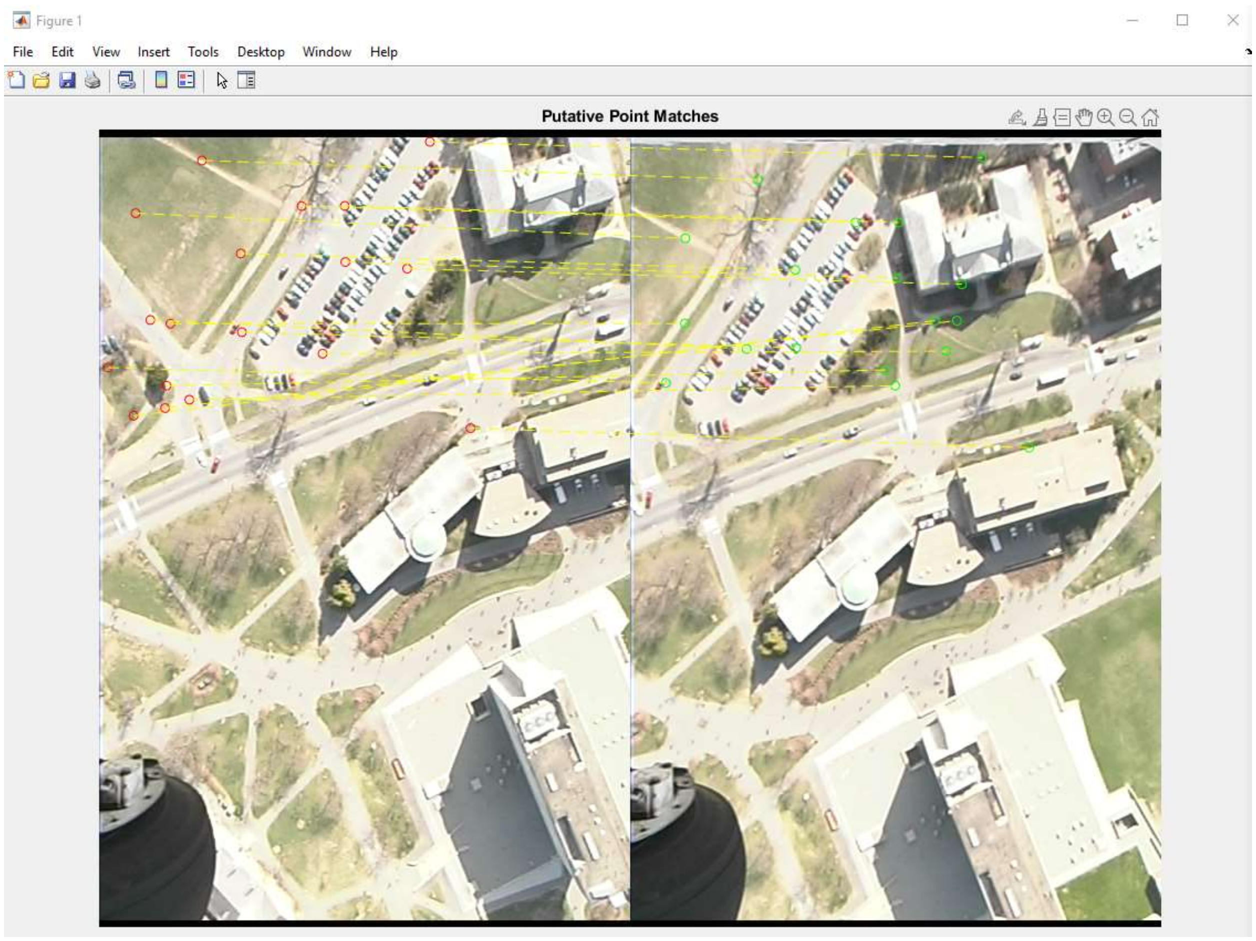


Fig 2. Epipole attempt 1 (using in-built)

I tried implementing the mouse click feature but I could only manage the red star output from two clicks.

I resorted to using in-built functions to net the epipoles in this attempt as yellow dashed lines. However, they don’t quite line up.

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

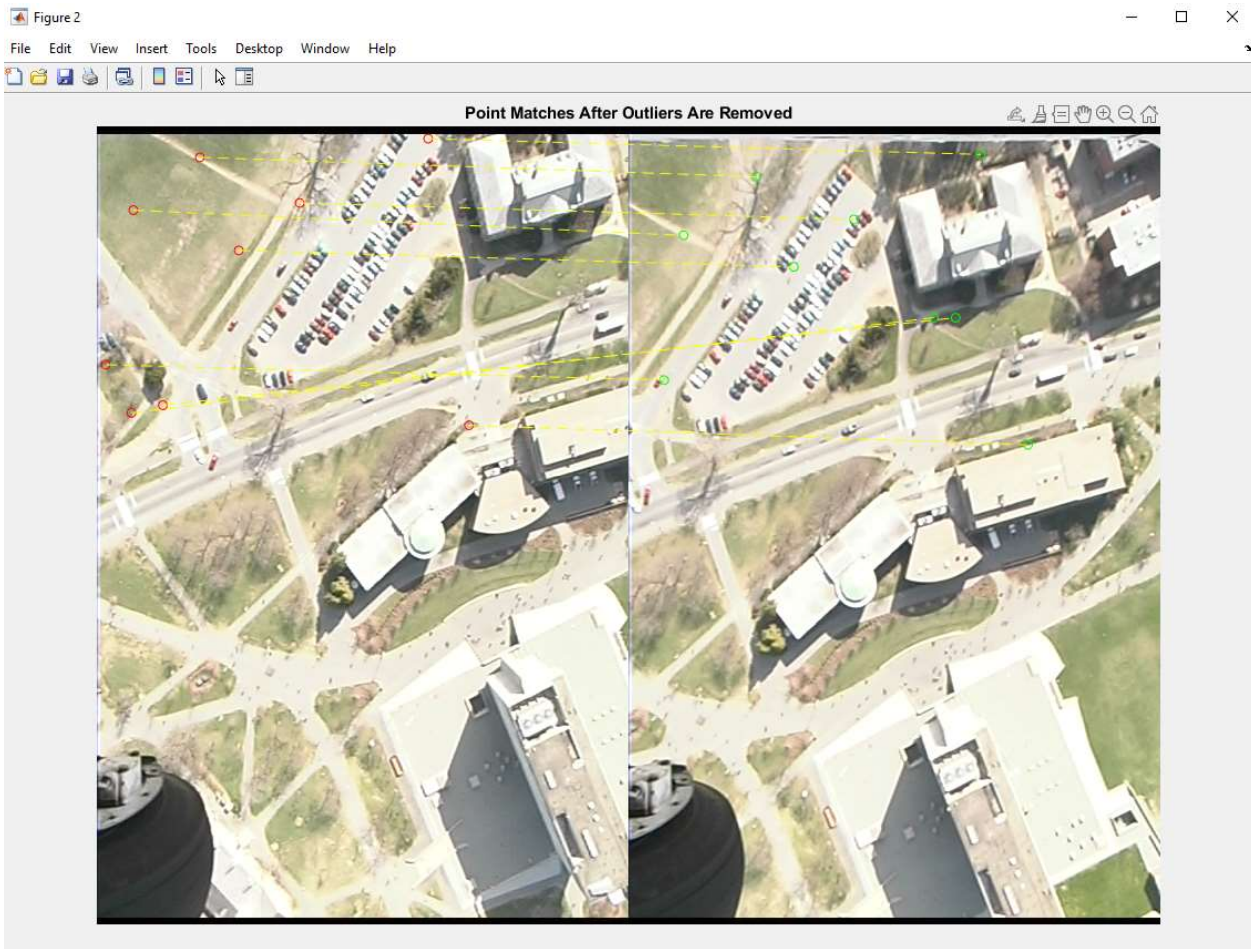


Figure 2
File Edit View Insert Tools Desktop Window Help
Point Matches After Outliers Are Removed

Fig 3. Attempting to correct the outliers.

I tried to align the lines again to no avail.

```
    0.0000   -0.0004    0.0348
    0.0004    0.0000   -0.0937
   -0.0426    0.0993    0.9892
```

Fundamental Matrix output

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

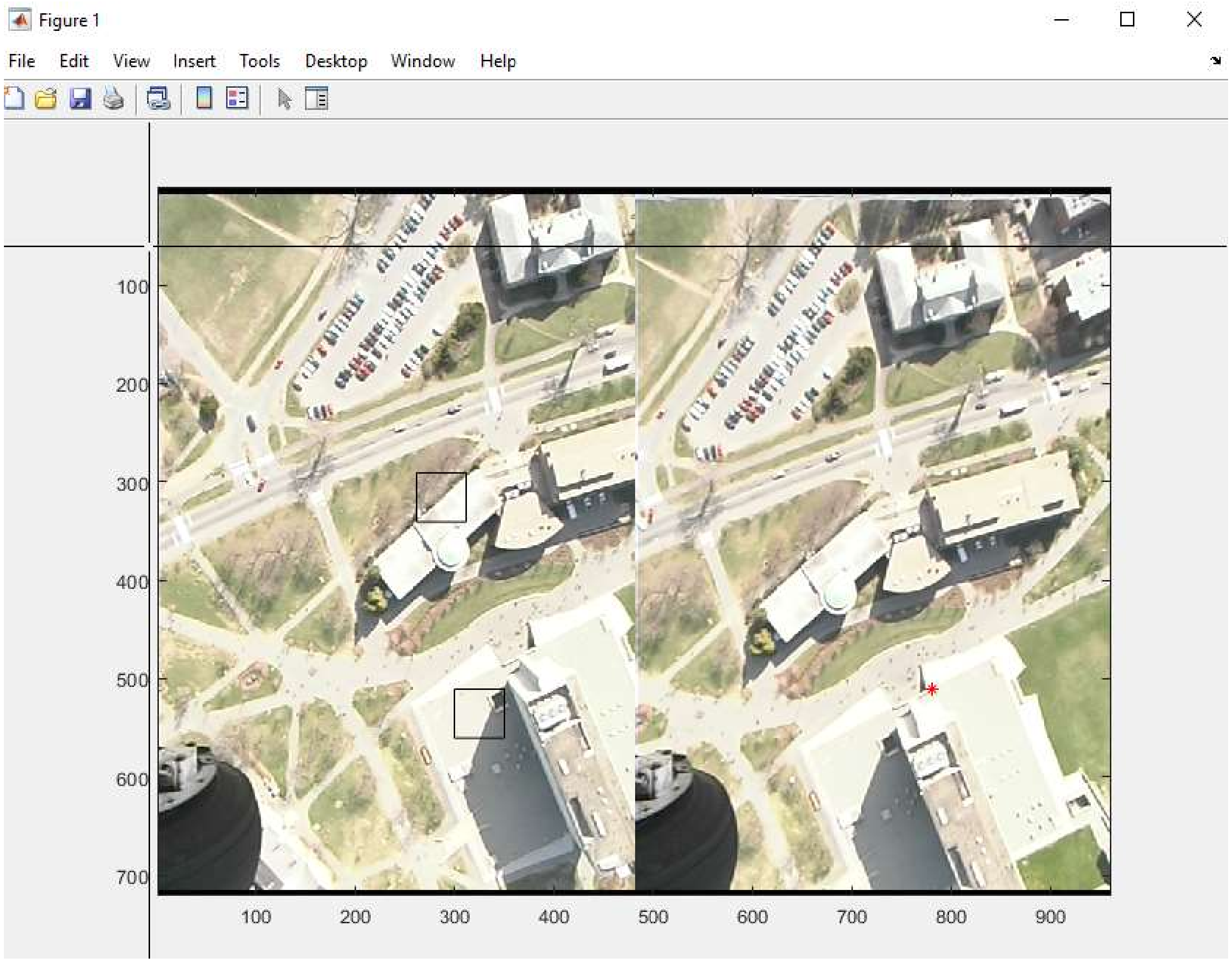
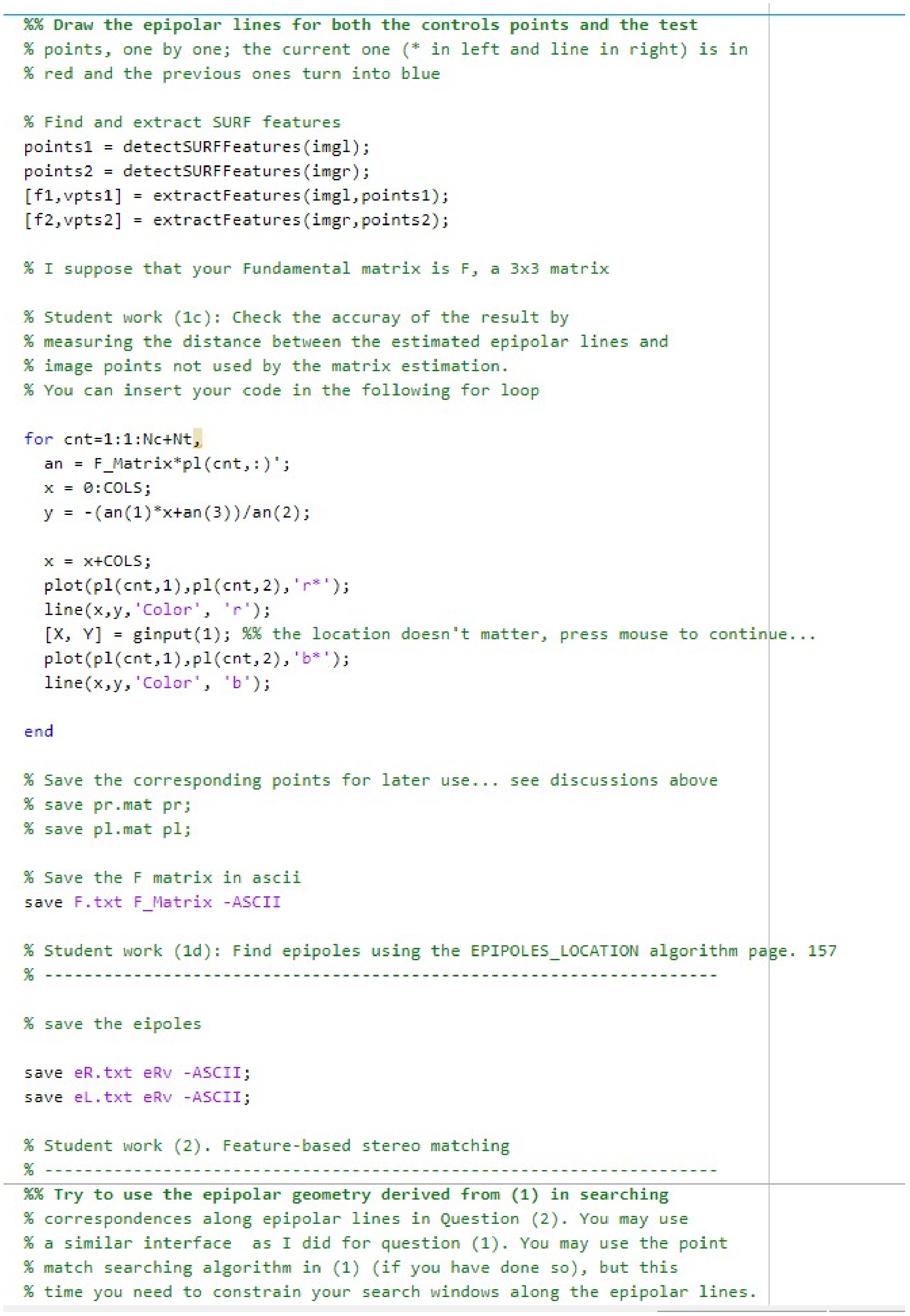


Fig 4. Two mouse clicks to generate a red point

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu



```
%% Draw the epipolar lines for both the controls points and the test
% points, one by one; the current one (* in left and line in right) is in
% red and the previous ones turn into blue

% Find and extract SURF features
points1 = detectSURFFeatures(imgl);
points2 = detectSURFFeatures(imgr);
[f1,vpts1] = extractFeatures(imgl,points1);
[f2,vpts2] = extractFeatures(imgr,points2);

% I suppose that your Fundamental matrix is F, a 3x3 matrix

% Student work (1c): Check the accuray of the result by
% measuring the distance between the estimated epipolar lines and
% image points not used by the matrix estimation.
% You can insert your code in the following for loop

for cnt=1:1:Nc+Nt,
  an = F_Matrix*pl(cnt,:)';
  x = 0:COLS;
  y = -(an(1)*x+an(3))/an(2);

  x = x+COLS;
  plot(pl(cnt,1),pl(cnt,2),'r*');
  line(x,y,'Color', 'r');
  [X, Y] = ginput(1); %% the location doesn't matter, press mouse to continue...
  plot(pl(cnt,1),pl(cnt,2),'b*');
  line(x,y,'Color', 'b');

end

% Save the corresponding points for later use... see discussions above
% save pr.mat pr;
% save pl.mat pl;

% Save the F matrix in ascii
save F.txt F_Matrix -ASCII

% Student work (1d): Find epipoles using the EPIPOLES_LOCATION algorithm page. 157
% --------------------------------------------------------------------

% save the eipoles

save eR.txt eRv -ASCII;
save eL.txt eRv -ASCII;

% Student work (2). Feature-based stereo matching
% --------------------------------------------------------------------
%% Try to use the epipolar geometry derived from (1) in searching
% correspondences along epipolar lines in Question (2). You may use
% a similar interface  as I did for question (1). You may use the point
% match searching algorithm in (1) (if you have done so), but this
% time you need to constrain your search windows along the epipolar lines.
```

Fig 5. SURF features attempt 2

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

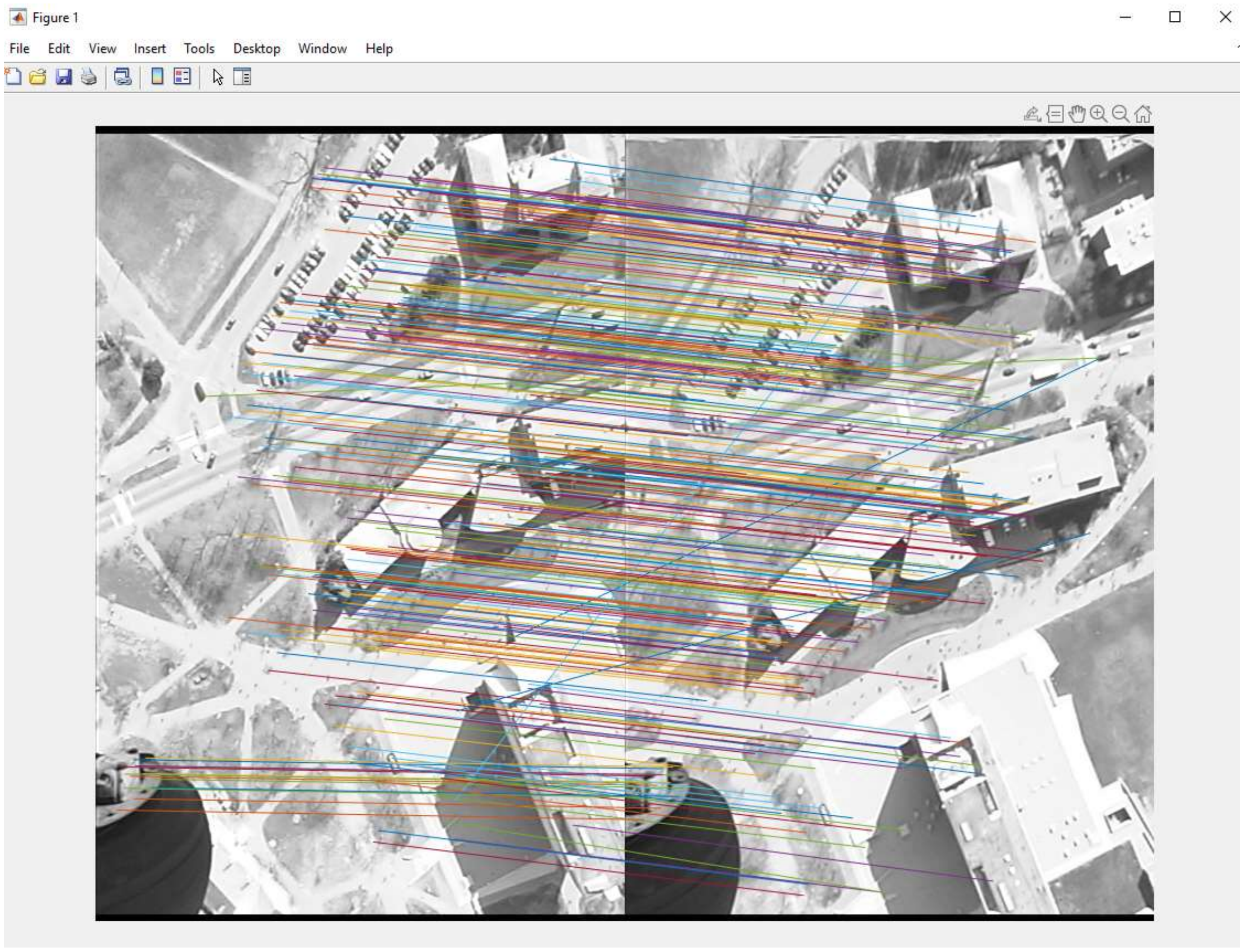


Fig 6. Epipoles matching the correct features but not interactive

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

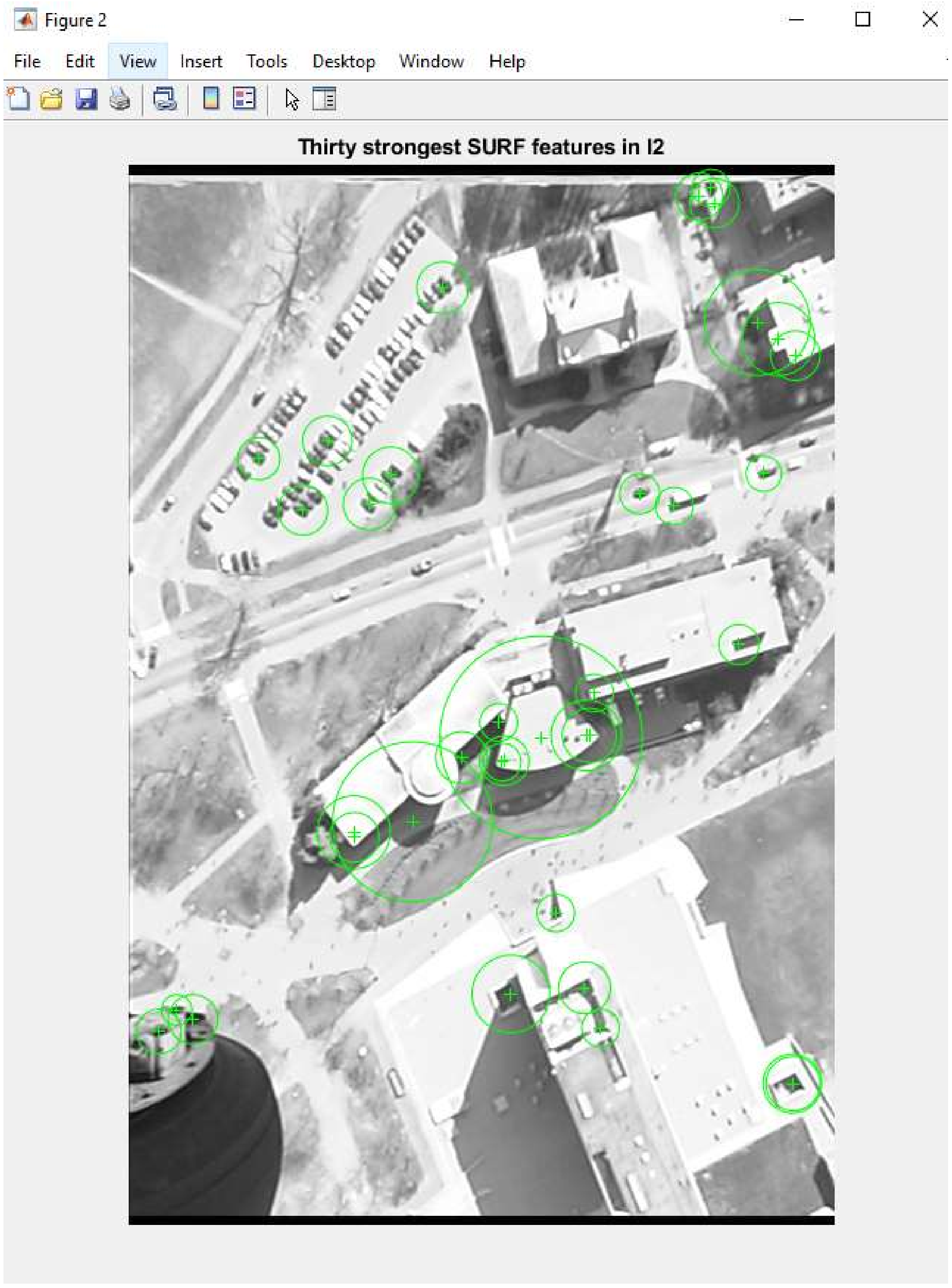


Fig 7. Surf features

SURF features highlighted the most prominent features of the images (green circles) so I could set up for triangulation in 3D.

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

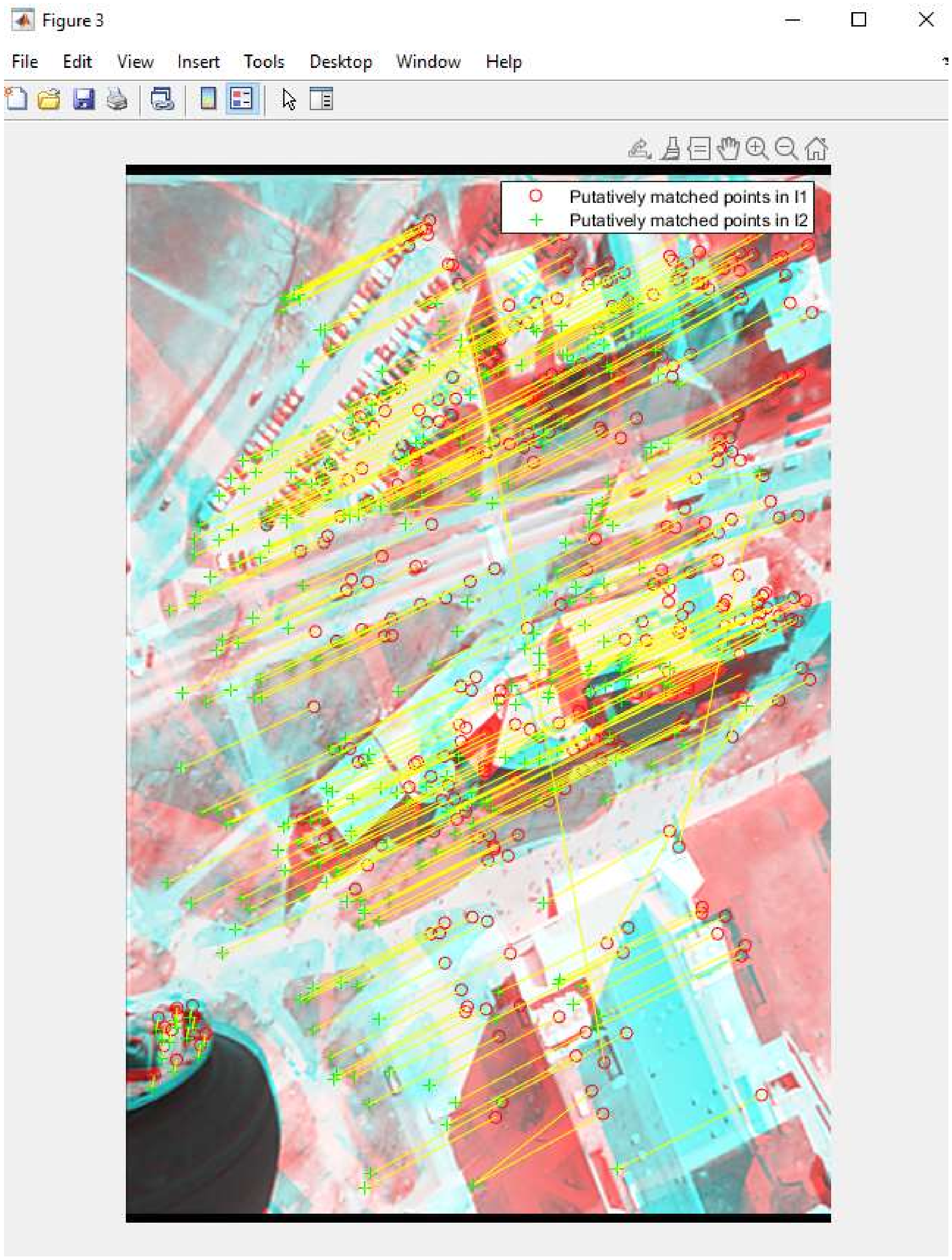


Figure 3
File
Edit
View
Insert
Tools
Desktop
Window
Help
Putatively matched points in I1
Putatively matched points in I2

Fig 8. 3D construction

I tried aligned the feature matches together to net a 3D estimation. Not interactive though.

Ismail Akram
8834
CSc 471 Computer Vision
Prof. Zhu

Discussion: The images above have corners, edges, smooth regions (roof and parking lot), textured regions (grass), and occluded regions (behind the stadium).

My vision system was partially successful in finding point matches (mainly in the second surf features attempt) yet fails to correctly highlight 8 correct prominent matches. It would also fail to fully detect occluded features due to the nature of 2D imagery and the computer not knowing the 3D structure like we humans would. We can “fill in” occluded space in our mind through a fairly accurate estimation. Machines don’t have that luxury.