Exercise in Geometry Constrained Feature Matching

## 1 The Eight Point Algorithm

Write a function Fest 8point which implements the eight point algorithm for fundamental matrix estimation.

1. Confirm that q1 and q2 are consistent with Ftrue.

P\_1 = A \* [R1 T1]; %3 by 4, 5 dof

P\_2 = A \* [R2 T2];

q1 = P\_1 \* Q;

q2 = P\_2 \* Q;

q1(1,:)=q1(1,:)./q1(3,:);

q1(2,:)=q1(2,:)./q1(3,:);

q1(3,:)=q1(3,:)./q1(3,:);

q2(1,:)=q2(1,:)./q2(3,:);

q2(2,:)=q2(2,:)./q2(3,:);

q2(3,:)=q2(3,:)./q2(3,:);

CrossOp =@(T2) [0 -T2(3) T2(2); T2(3) 0 -T2(1); -T2(2) T2(1) 0];

T2\_cross = CrossOp(T2);

F = pinv(A)' \* T2\_cross \* R2 \* A^-1 %Ftrue

F1= Fest\_8point(q1,q2); % P.55

% fundamental matrix can be used as a constraint on point correspondences

2. From q1 and q2 construct the linear equations constraining the fundamental matrix, and confirm

that they correspond to the elements of Ftrue.

3. Set up the functionality to estimate a fundamental matrix from these linear equations. Confirm

that this fundamental matrix is equal to Ftrue – up to numerical error.

4. Normalize the data, as described in ”Exercise in Estimating View Geometry”. The estimated

fundamental matrix estimated from these normalized points should be modified in the following

manner to correspond to the original points:

F=T2’\*F\*T1; % Denormalise

5. Confirm that this estimate is equal to Ftrue – up to numerical error.

b=F(:);

a=F1(:);

diff=a'\*b/(norm(a)\*norm(b))

F =

1.0e-03 \*

0.0000 -0.0000 0.0001

-0.0000 -0.0000 0.0329

0.0004 -0.0322 -0.2120

F1 =

0.0000 -0.0000 0.0000

-0.0000 -0.0000 0.0055

0.0001 -0.0054 -0.0353

diff = 1 the two F are the same.

6. Wrap the code in a function, e.g. named function F=Fest 8point(q1,q2).

[F] = Fest\_8point(varargin)

[x1, x2, npts] = checkargs(varargin(:));

% Normalise each set of points so that the origin

% is at centroid and mean distance from origin is sqrt(2).

% normalise2dpts also ensures the scale parameter is 1.

[x1, T1] = normalise2dpts(x1); % P.56 Normalization of Points

[x2, T2] = normalise2dpts(x2);

% Build the constraint matrix

A = [x2(1,:)'.\*x1(1,:)' x2(1,:)'.\*x1(2,:)' x2(1,:)' ...

x2(2,:)'.\*x1(1,:)' x2(2,:)'.\*x1(2,:)' x2(2,:)' ...

x1(1,:)' x1(2,:)' ones(npts,1) ];

[U,D,V] = svd(A,0);

% Extract fundamental matrix from the column of V corresponding to

% smallest singular value.

F = reshape(V(:,9),3,3)';

% Enforce constraint that fundamental matrix has rank 2 by performing

% a svd and then reconstructing with the two largest singular values.

[U,D,V] = svd(F,0);

F = U\*diag([D(1,1) D(2,2) 0])\*V';

% Denormalise

F = T2'\*F\*T1;

# 2 Feature Matching

Repeat part of the exercise ”Exercise in Feature Matching”, by matching the two images from TwoImageData.I.e. load the appropriate images and initialize vl setup, and execute the code:

[fa, da] = vl sift(single(im1));

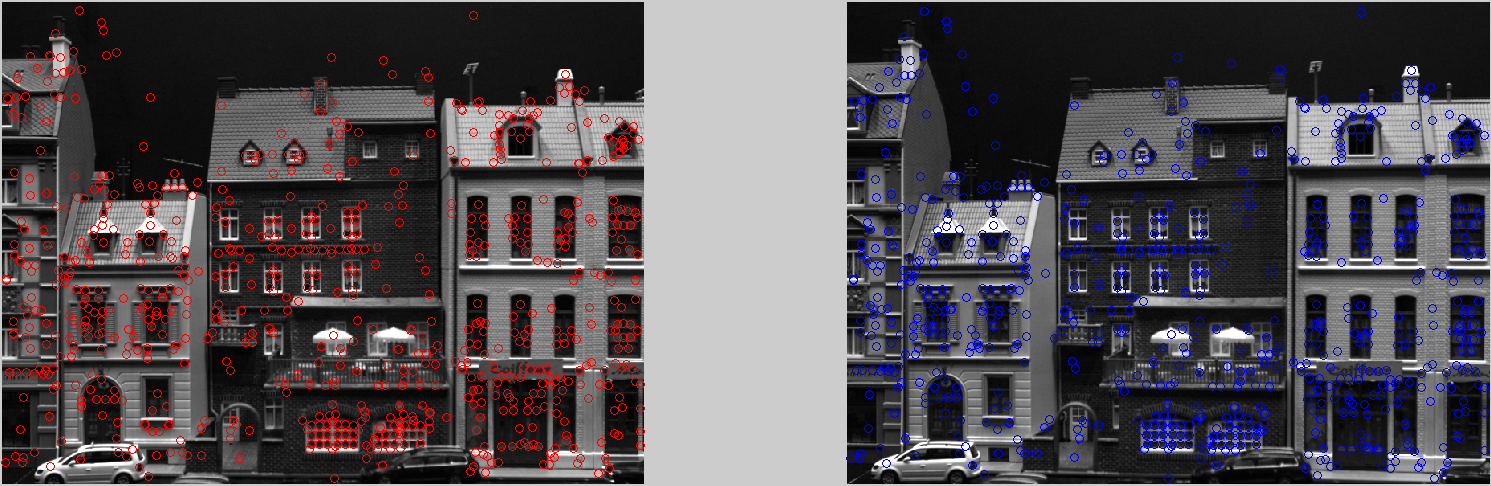
[fb, db] = vl sift(single(im2));

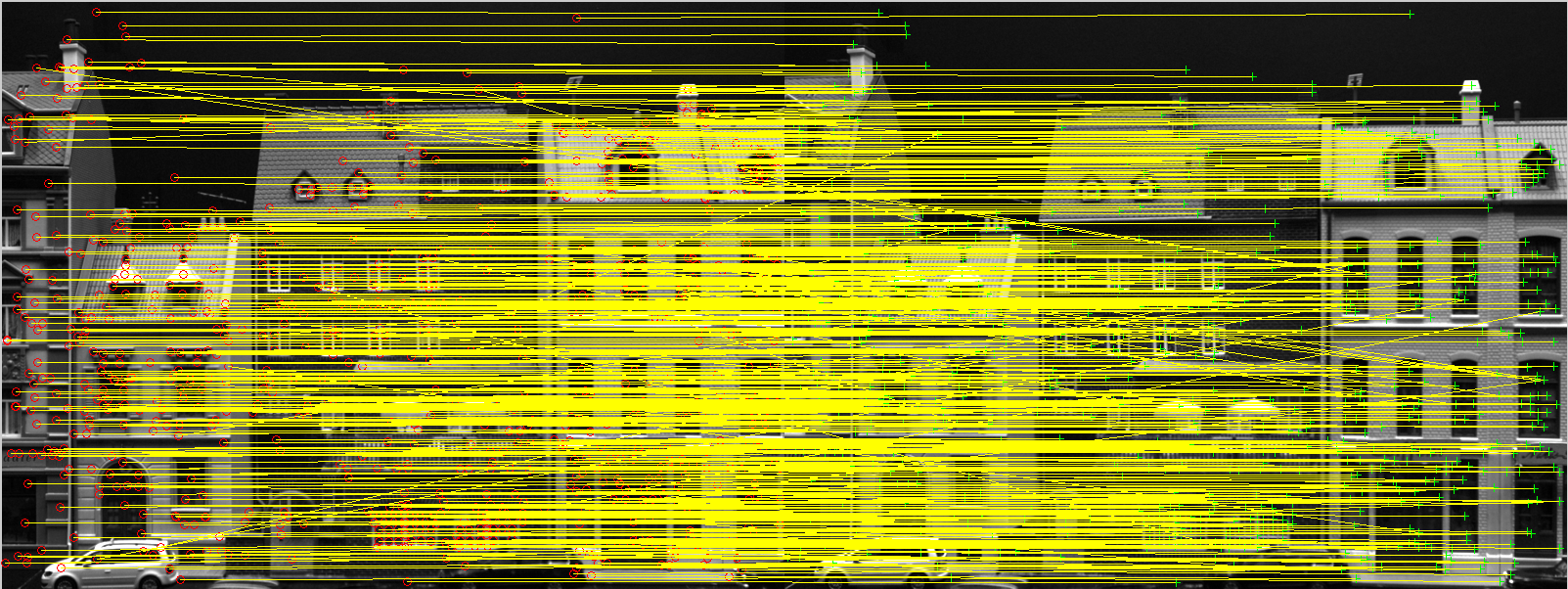
[matches, scores] = vl ubcmatch(da, db);

nMatch=size(matches,2);

Illustrate the result and confirm that it looks reasonable compared to your expectations.

Original Matching points = 879 MANY skewed lines.





# 3 F-estimation via Ransac

Modify your Ransac algorithm from ”Exercise in Robust Model Fitting” to fit fundamental matrices

instead of lines. Hints:

iter=100;

for i=1:iter

idx=randperm(nMatch);

idx=idx(1:8);

F{i} = Fest\_8point(p1(:,idx),p2(:,idx)); %takes homo points

inlier = 0;

for cM=1:nMatch,

if(FSampDist(F{i},X1(:,cM),X2(:,cM))<3.84\*3^ 2) %takes non-homo points

inlier = inlier + 1;

tr1(:,cM) = [X1(:,cM)];

tr2(:,cM) = [X2(:,cM)];

end

inlierT(i) = inlier;

end

tt1{i} = mat2cell(tr1);

tt2{i} = mat2cell(tr2);

end

Ftrue =

1.0e-03 \*

0.0000 -0.0000 0.0001

-0.0000 -0.0000 0.0329

0.0004 -0.0322 -0.2120

F =

0.0000 0.0000 -0.0014

-0.0000 0.0000 0.0047

0.0014 -0.0046 -0.0147

diff =

-0.9997

The two F have negative correlation.

inliers = 850

# 4 Refine F Estimate

Find all the inliers w.r.t. your estimated fundamental matrix. Refine this fundamental matrix estimate

by running Fest 8point, on all the inliers. Compare estimated/best fundamental matrix with

Ftrue, again. Comment.

Illustrate the matched inliers, and compare with the result in Section 2, i.e. without use of the

fundamental matrix.

F\_refined =

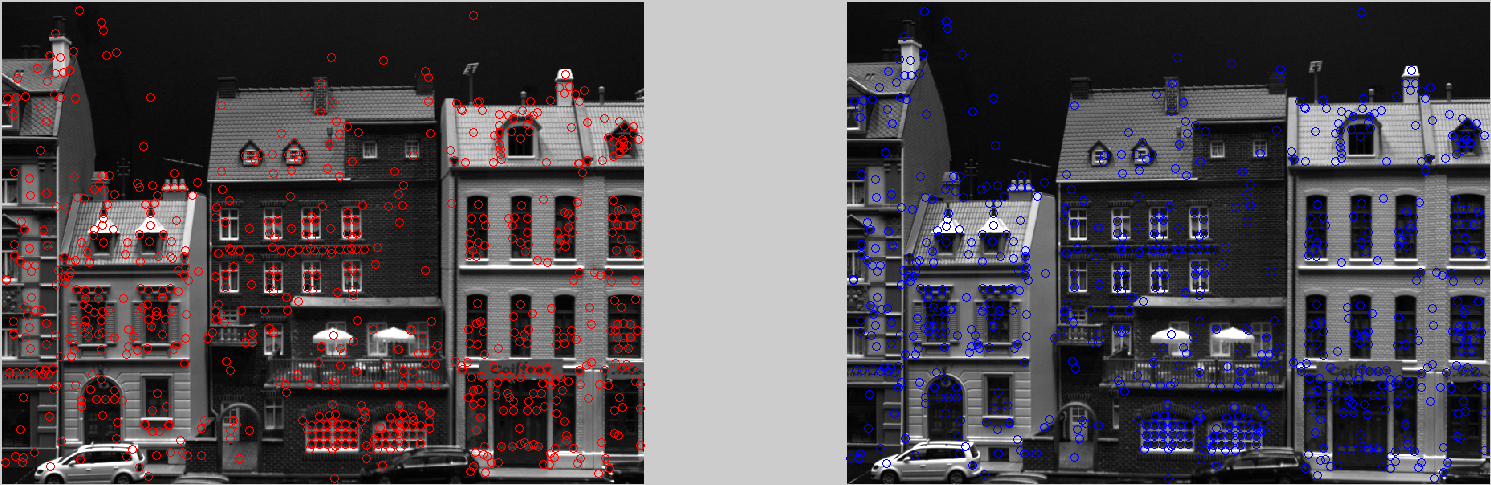
0.0000 0.0000 -0.0015

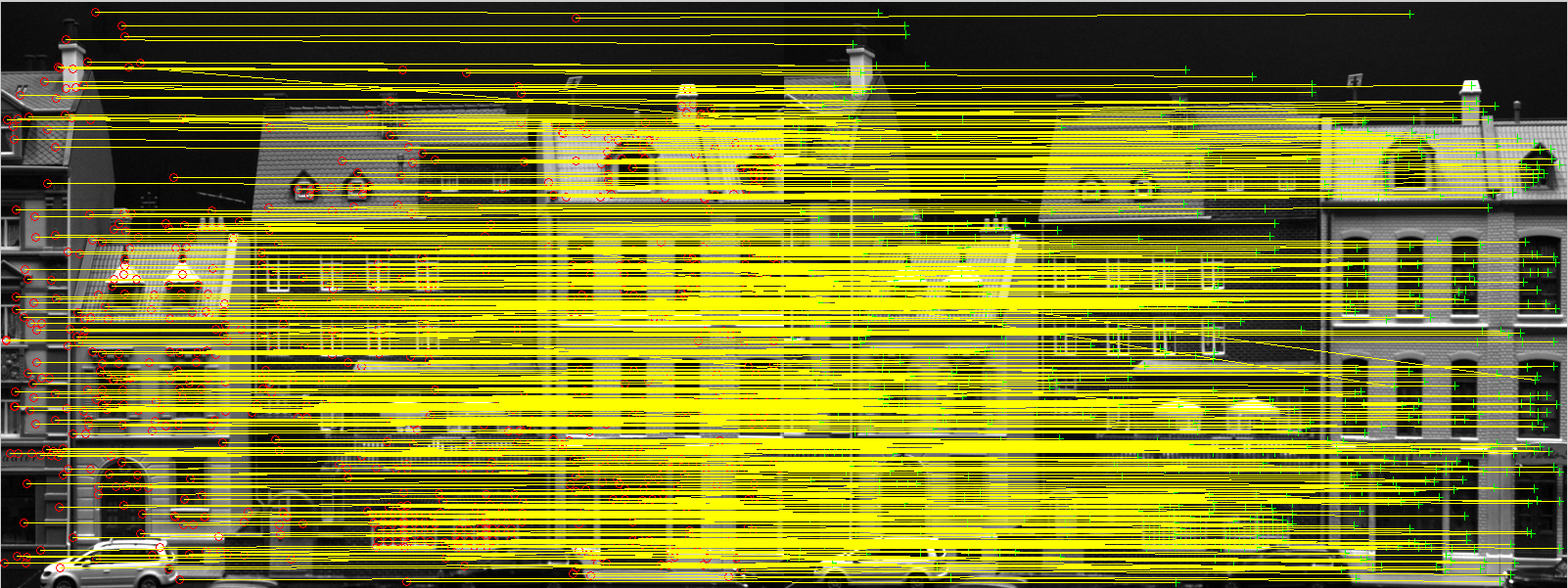
-0.0000 0.0000 0.0049

0.0015 -0.0049 -0.0111

dist = 0.9292

The two F have very positive correlation. VERY FEW skewed lines, compare to SITF descriptors, using F increase the matching accuracy.





# 5 Repeat on the **House** images

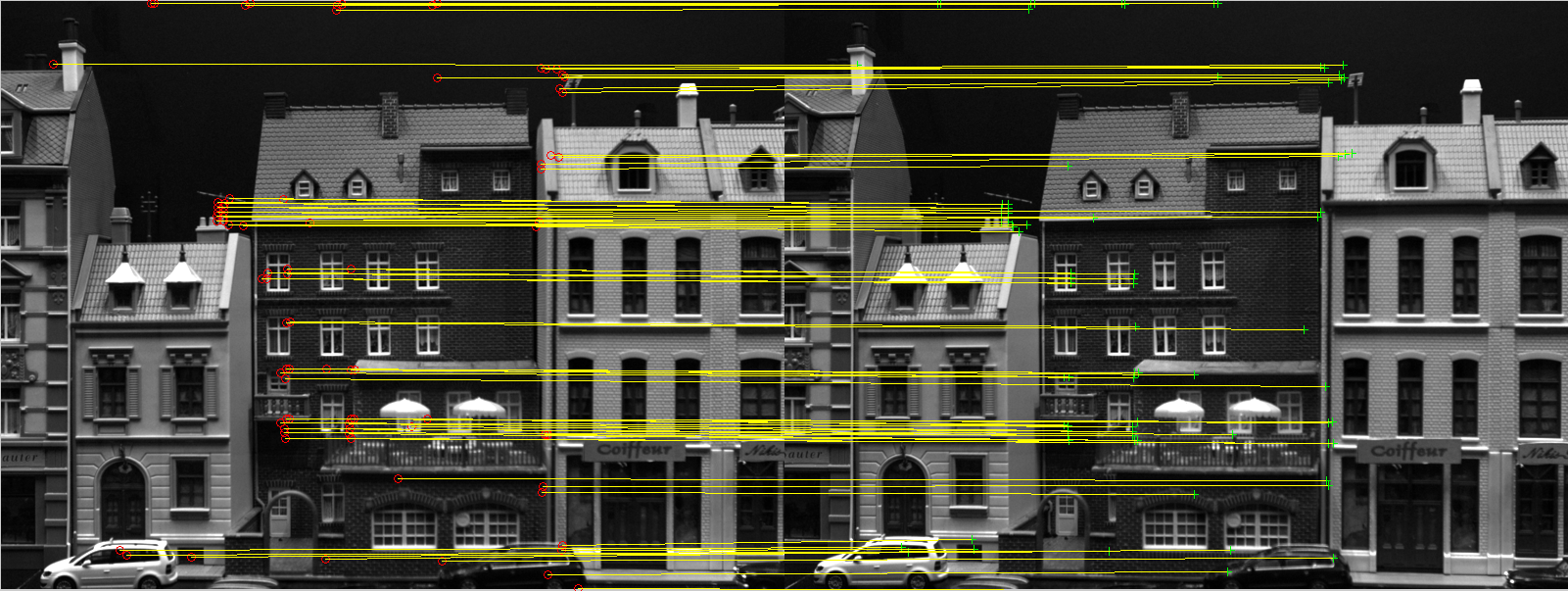
Apply your geometry based image matching algorithm on the House1.bmp and House2.bmp

images, where the fundamental matrix is unknown. Comment on the result, including the effect of

applying the fundamental matrix.

Sampson distance checks the correspondent points consistency.

Find corners as descriptors.





t\_cross = CrossOp(T2);

F = pinv(A)' \* t\_cross \* R2 \* A^-1;

d = sampsonFun(F,p1,p2)

thr = quantile(d,10)

sigma=3;

thresh = 5.99 \* sigma^2;

for i= 1:size(d,2)

if d(i) >= thresh

p1(:,i)=0;

p2(:,i)=0;

else

pf1(:,i) = p1(:,i);

pf2(:,i) = p2(:,i);

end

end

np1 = nonzeros(pf1);

np2 = nonzeros(pf2);

s = size(np1,1)/2

ppp1 = reshape(np1,2,s);

ppp2 = reshape(np2,2,s);