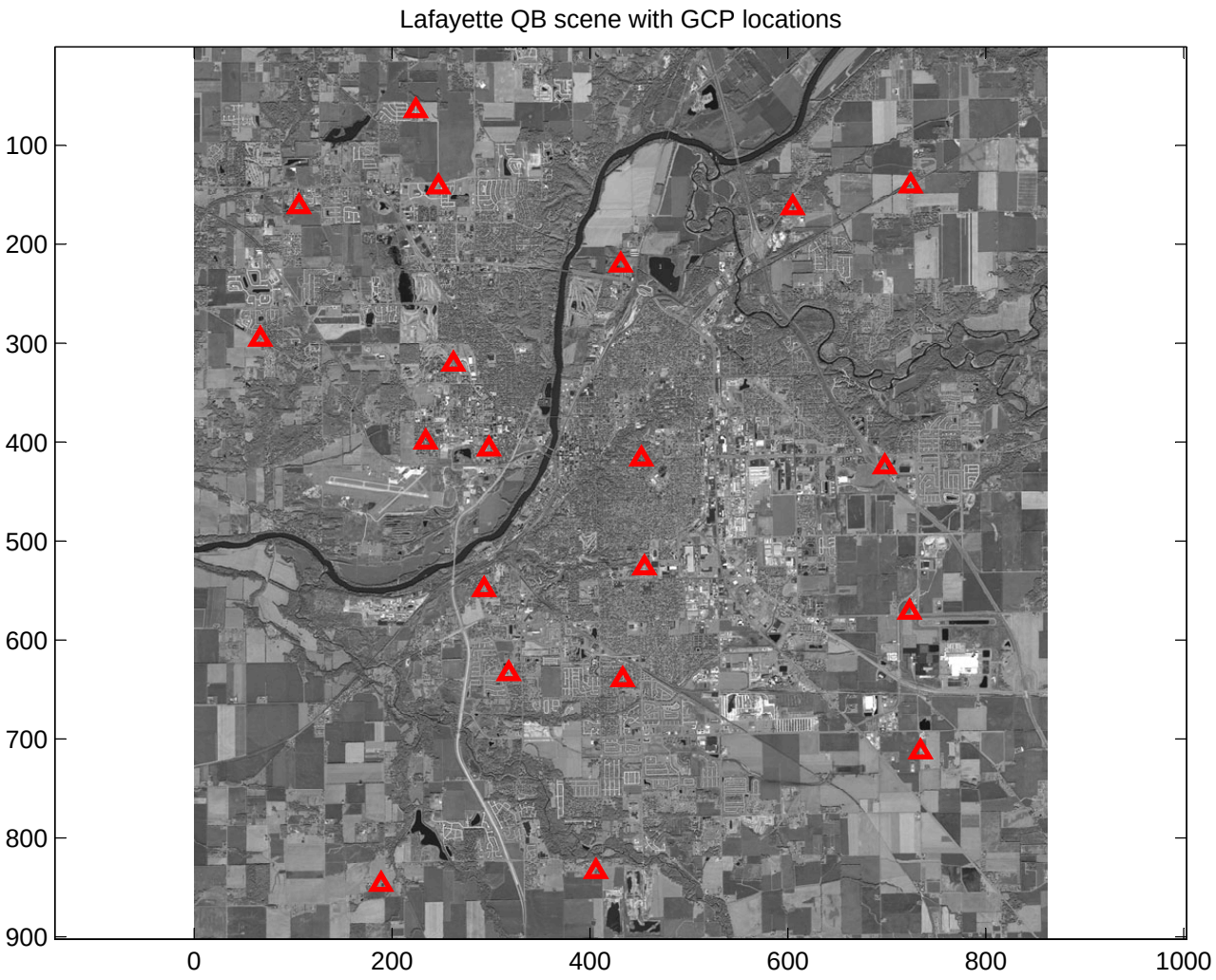
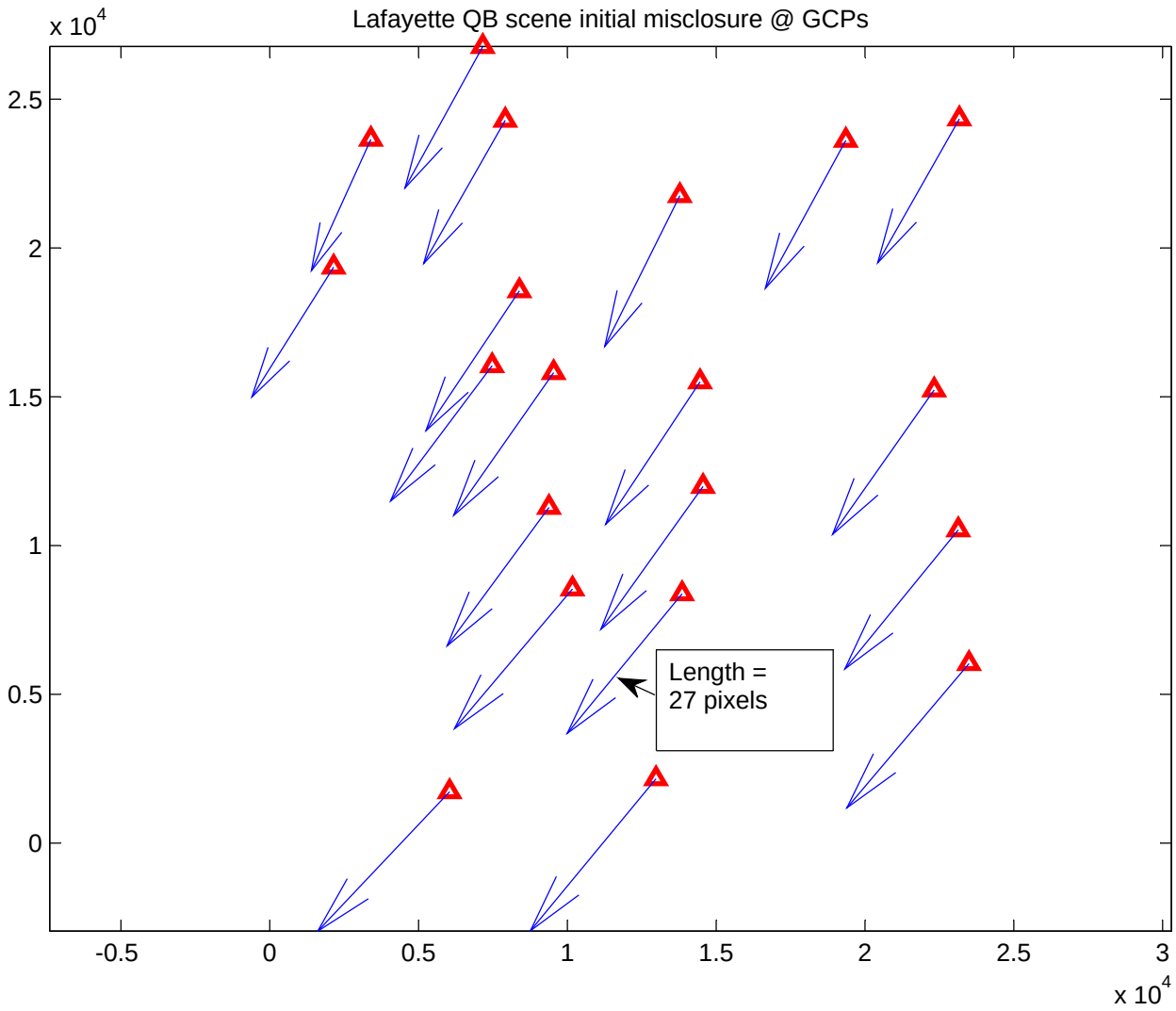
columns are sequence number, line , sample, misclosure X, misclosure Y

these produced by program qb_misc.m and associated function qb.m

misclosures computed as in lecture 13

1	20464	13855	21.264	-17.5
2	20302	10174	21.287	-17.984
3	12797	7473	20.635	-15.474
4	17561	9382	21.098	-15.512
5	16854	14559	21.79	-15.557
6	13341	14461	21.705	-14.377
7	13032	9541.2	21.751	-15.236
8	10279	8391.8	21.307	-14.257
9	9482.5	2150.3	19.803	-12.469
10	5189.9	3400.9	20.02	-9.0777
11	4551.4	7916.6	21.824	-12.443
12	7085.6	13781	22.967	-11.45
13	2071	7158.4	21.548	-11.842
14	18315	23138	21.199	-17.298
15	13613	22327	21.994	-15.49
16	4504.8	23170	21.94	-12.446
17	5224.3	19358	22.562	-12.269
18	27113	6044.9	21.234	-20.104
19	26678	12978	23.142	-19.129
20	22805	23495	21.981	-18.599


Lafayette QB scene with GCP locations
100
200
300
400
500
600
700
800
900
0
200
400
600
800
1000


Lafayette QB scene initial misclosure @ GCPs
x 10^4
2.5
2
1.5
1
0.5
0
-0.5
0
0.5
1
1.5
2
2.5
3
x 10^4
Length =
27 pixels

```
% qb_misc.m  1-april-08
% quickbird misclosure computation from file of gcp's
% call function F=QB(line,sample,phi,lam,ht,eph,att,par)

% read gcps and image obs
[id,pd,pm,ps,ld,lm,ls,h,l,s]=textread('gcp.txt','%s %f %f %f %f %f %f %f↙
%f %f');
[m,n]=size(pd);
npts=m;
% read ephemeris data
eph=dlmread('eph_cdf.txt');
att=dlmread('att_cdf.txt');
% or is it csvread() ???
% document some constants here, they are hardwired inside the function
% (dg.geo)
%  PD = 8836.202;
% flollowing is in order qi,qj,qk,qs different from textbook
%  qcs1 =    -0.0037369299279122;
%  qcs2 =    -0.0046602381421084;
%  qcs3 =     0.0017171366422757;
%  qcs4 =     0.9999806843019142;
%  detOriginX =     9.5468399999999995;
%  detOriginY =   164.0277299999999900;
%  detRotAngle =     0.0000000000000000;
%  detPitch =     0.0119139600000000;
% (dg.imd)
% numRows = 28840;
% numColumns = 27552;
% firstLineTime = 2006-10-07T16:57:27.463116Z;
% avgLineRate = 6900.00;
% exposureDuration = 0.00188406;
% TDILevel = 13;
% (dg.eph) - same for att
% startTime = 2006-10-07T16:57:21.425796Z;
% numPoints = 814;
% timeInterval = 0.020;
% x0= -detOriginX/detPitch;
% y0= -detOriginY/detPitch;
% foc= PD/detPitch;

% par=[dx0,dx1,dx2,dy0,dy1,dy2,dz0,dz1,dz2,dw0,dw1,dw2,dp0,dp1,dp2,dk0,↙
dk1,dk2]
```

```
disp('misclosures');
par=zeros(18,1);
for i=1:npts
  phi=(pd(i)+pm(i)/60+ps(i)/3600)*(pi/180);
  lam=(ld(i)+lm(i)/60+ls(i)/3600)*(pi/180);
  lam=-lam;
  ht=h(i);
  line=l(i);
  sample=s(i);
  F=qb(line,sample,phi,lam,ht,eph,att,par);
  [i line sample F(1) F(2)]
  end
```

```
% qb.m  1-april-2008
% quickbird misclosure function
function result=qb(line,sample,phi,lam,ht,eph,att,par)

% put parameters into names we recognize
dx0=par(1); dx1=par(2); dx2=par(3);
dy0=par(4); dy1=par(5); dy2=par(6);
dz0=par(7); dz1=par(8); dz2=par(9);
dw0=par(10); dw1=par(11); dw2=par(12);
dp0=par(13); dp1=par(14); dp2=par(15);
dk0=par(16); dk1=par(17); dk2=par(18);

PD = 8836.202;
% flollowing is in order qi,qj,qk,qs different from textbook
qcs1 =    -0.0037369299279122;
qcs2 =    -0.0046602381421084;
qcs3 =     0.0017171366422757;
qcs4 =     0.9999806843019142;
detOriginX =     9.5468399999999995;
detOriginY =   164.0277299999999900;
detPitch =     0.0119139600000000;
% firstLineTime = 2006-10-07T16:57:27.463116Z;
% avgLineRate = 6900.00;
% exposureDuration = 0.00188406;
% TDILevel = 13;
% startTime = 2006-10-07T16:57:21.425796Z; eph & att
% timeInterval = 0.020;
x0= -detOriginX/detPitch;
y0= -detOriginY/detPitch;
foc= PD/detPitch;


XLe=eph(:,2);
YLe=eph(:,3);
ZLe=eph(:,4);
qie=att(:,2);
qje=att(:,3);
qke=att(:,4);
qse=att(:,5);

tes=16*3600+57*60+21.425796; % time in seconds, first ephemeris point
dte=0.02;
```

```
dt=1/6900.0;
% assume line measured with line #1 = 0
t0=16*3600+57*60+27.463116; % time in seconds, first line
tline=t0 + line*dt;
T=tline-t0; % local time within image = independent variable for↙
polynomials

index1=fix((tline-tes)/dte) + 1;
index2=index1+1;
t1=tes + (index1-1)*dte;
t2=t1 + dte;
delt=tline-t1;
frac=delt/dte;

XL=frac*XLe(index2) + (1-frac)*XLe(index1);
YL=frac*YLe(index2) + (1-frac)*YLe(index1);
ZL=frac*ZLe(index2) + (1-frac)*ZLe(index1);
qi=frac*qie(index2) + (1-frac)*qie(index1);
qj=frac*qje(index2) + (1-frac)*qje(index1);
qk=frac*qke(index2) + (1-frac)*qke(index1);
qs=frac*qse(index2) + (1-frac)*qse(index1);
len=sqrt(qi^2 + qj^2 + qk^2 + qs^2);
qi=qi/len;
qj=qj/len;
qk=qk/len;
qs=qs/len;

qcami =  qcs1;
qcamj =  qcs2;
qcamk =  qcs3;
qcams =  qcs4;

M=q2m(qi,qj,qk,qs);
Mc=q2m(qcami,qcamj,qcamk,qcams);

dX=dx0 + dx1*T + dx2*T^2;
dY=dy0 + dy1*T + dy2*T^2;
dZ=dz0 + dz1*T + dz2*T^2;
dW=dw0 + dw1*T + dw2*T^2;
dP=dp0 + dp1*T + dp2*T^2;
dK=dk0 + dk1*T + dk2*T^2;
mw=m1(dW);
```

```
mp=m2(dP);
mk=m3(dK);
Ma=mk*mp*mw;

% WGS84 parameters
ae=6378137.0;%(m)
fe=1/298.257223563;
be=ae*(1-fe);
GMe=398600.5e+09;
ee=sqrt(2*fe - fe*fe);
we=0.000072921158553;
pare(1)=ae;
pare(2)=be;
pare(3)=fe;
pare(4)=GMe;
pare(5)=ee;
pare(6)=we;
XYZ=g2usr(phi,lam,ht,pare);
UVW=Ma*Mc'*M'*(XYZ - [XL+dX; YL+dY; ZL+dZ]);
U=UVW(1);
V=UVW(2);
W=UVW(3);
Fx= -x0 - foc*(U/W);
Fy= -sample - y0 - foc*(V/W);
result=[Fx;Fy];
```

```
% q2m.m 17-feb-05
% rotation matrix from quaternion

function m=q2m(qi,qj,qk,qs)
m=zeros(3,3);
m(1,1)=qs^2+qi^2-qj^2-qk^2;
m(1,2)=2*(qj*qi-qs*qk);
m(1,3)=2*(qi*qk+qs*qj);
m(2,1)=2*(qj*qi+qs*qk);
m(2,2)=qs^2-qi^2+qj^2-qk^2;
m(2,3)=2*(qj*qk-qs*qi);
m(3,1)=2*(qi*qk-qs*qj);
m(3,2)=2*(qj*qk+qs*qi);
m(3,3)=qs^2-qi^2-qj^2+qk^2;
```

```
function xyz = g2usr(phi,lambda,h,pare)
% gtousr.m 25-par-96
% compute geocentric xyz from lat/lon
% ae assumed to be in meters (although this routine
% does not care)
% ae=6378137.0;%(m)
% fe=1/298.257223563;
% be=ae*(1-fe);
% GMe=398600.5e+09;
% ee=sqrt(2*fe - fe*fe);
% we=0.000072921158553;
% pare(1)=ae;pare(2)=be;pare(3)=fe;pare(4)=GMe;pare(5)=ee;pare(6)=we;


ae=pare(1);
be=pare(2);
fe=pare(3);
GMe=pare(4);
ee=pare(5);
we=pare(6);

xyz=zeros(3,1);
tmp=sqrt(1 - ee*ee*sin(phi)*sin(phi));
N=ae/tmp;
xyz(1)=(N+h)*cos(phi)*cos(lambda);
xyz(2)=(N+h)*cos(phi)*sin(lambda);
xyz(3)=(N*(1 - ee*ee) + h)*sin(phi);
```

```
% m2.m
function m=m2(th)
m=[cos(th) 0 -sin(th);0 1 0;sin(th) 0 cos(th)];
```

```
% m1.m
function m=m1(th)
m=[1 0 0;0 cos(th) sin(th);0 -sin(th) cos(th)];
```

```
% m3.m
function m=m3(th)
m=[cos(th) sin(th) 0;-sin(th) cos(th) 0;0 0 1];
```