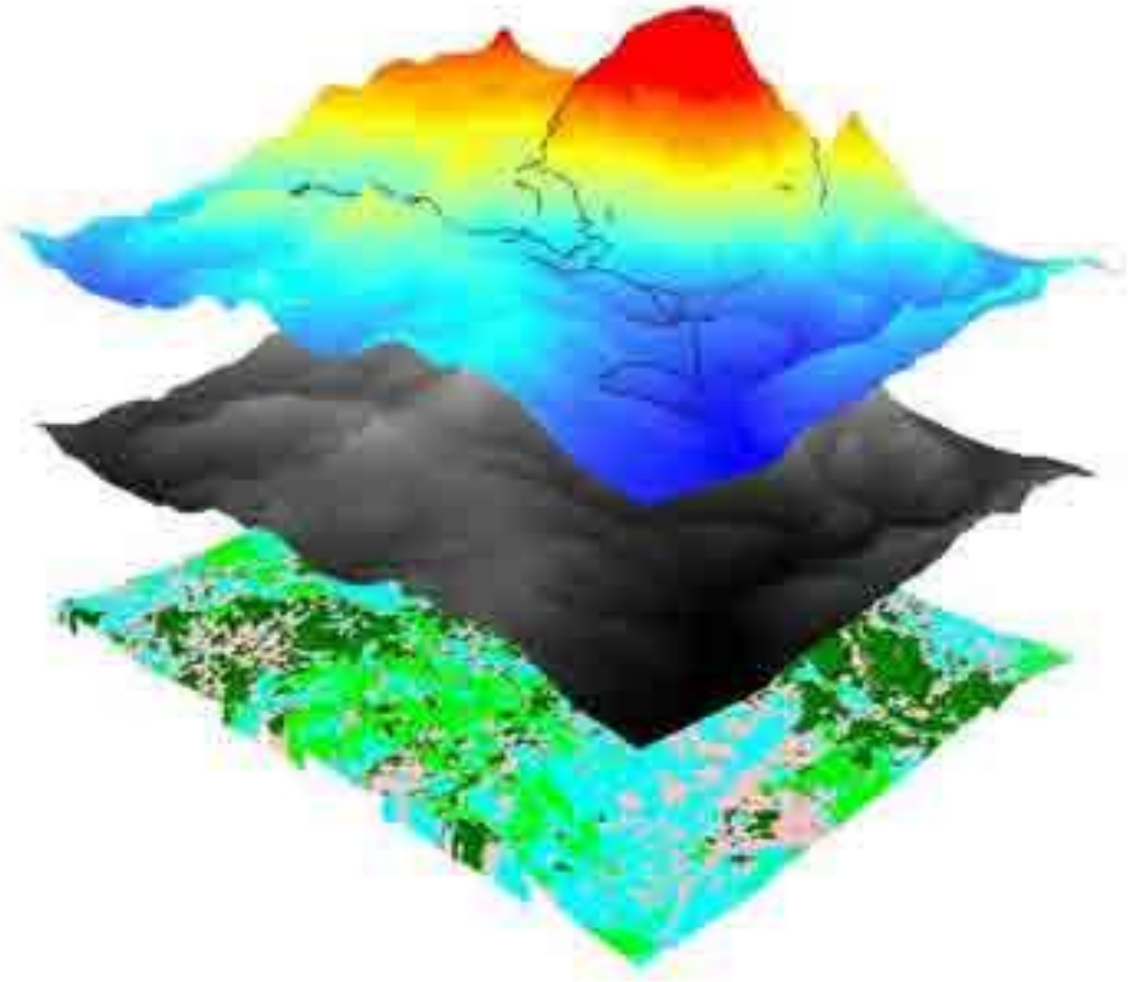

PHOTOGRAMMETRY II B

RELATIVE ORIENTATION IN DIGITAL PHOTOGRAMMETRY

THE BUNDLE ADJUSTMENT PROCEDURE

F19/1722/2013 | **F19/36677/2013** | FGE 442 | October 19, 2016

UNIVERSITY OF NAIROBI

SCOPE

The bundle adjustment process of aerial triangulation is a two stage process involving the formation of the bundle through the application of the collinearity condition equation. The second phase of this process is the execution of absolute orientation.

In the second stage, the models are transformed into the rightful ground coordinate system (GCS) through scaling, translation and orientation into parallelism with the ground system.

This is a report on the first stage of the bundle adjustment. Procedures followed are as per the class instructions in the Digital photogrammetry unit FGE 441, 2016 and the custom Matlab code used is referenced and attached.

OBJECTIVES

The objective of this procedure was the formation of a model from data (coordinates) of points in a pair of overlapping aerial photographs. The deliverables are model coordinates for all the conjugate points given as well as the five relative orientation parameters of the stereo pair.

Specific objectives were outlined as:

i. Use all the data points to compute the five relative orientation parameters
   These consist of:
   - Base components: BY and BZ,
   - The attitude / orientation parameters of photo 2 relative to photo 1: omega2 (w2), phi2 (p2) and kappa2 (k2)

ii. Determine the X, Y and Z model point coordinates of all the six points. Coordinates are defined as XI, YI and ZI.

METHODOLOGY AND RESULTS

DATA

Data used in the exercise are tabled as below. For each data point, image point coordinates in photo 1 and in photo 2 are given.

	Photo 1		Photo 2	
Point	x	y	x	y
1	0.966	-88.738	-91.627	-86.419
2	-0.798	1.403	-89.994	4.162
3	-2.511	92.055	-88.824	95.641
4	92.337	-88.145	-1.022	-89.392
5	96.602	3.491	0.818	2.564
6	85.156	90.647	2.595	90.518

NOTATIONS

These are the notations used in the Matlab scripting and in this report:

- x, y - photo coordinates of a point as given in the data. These are read as matrices x1, y1 and x2, y2 into the program.
- f – focal length of the camera
- XI, YI, ZI – initial values of the model coordinates of each conjugate point pair, 1 through 6.
- w2, p2, k2 – initial values of the orientation / rotational angles of photo 2 relative to photo 1. These are angles in radians.
- R – This is the rotational Eulerian matrix whose elements are functions of the rotational elements *omega, phi* and *kappa*.
- *drw, drp* and *drk* – Matrices containing respective first derivatives of the functions of rotational elements, i.e. required derivatives of the elements of R.

Full source code is available at https://github.com/jodom/Relative_Orientation

INITIAL PARAMETER DETERMINATION

The determination of the initial parameters were either given or computed manually and fed into the process. The results from the computations are as follows:

BX given as **200** mm.

F given as **152** mm.

	Initial model coordinates (mm)		
Point	XI	YI	ZI
1	0	0	-330.43
2	200	0	-330.43
3	0	187.5	-330.43
4	200	187.5	-330.43
5	0	-187.5	-330.43
6	200	-187.5	-330.43

These are scaled values of the model coordinates computed from a standard pair of aerial photographs taken at a 60% forward overlap and a side overlap of 25%. In the standard computation, BX is 92 mm and so the scaling factor used is 200/92.

The values omega phi and kappa denoted by **w2** and **p2** and **k2** respectively can be justified to be zero, initially.

LEAST SQUARE ITERATIVE SOLUTION

Given the data points are more than the minimum required (5), for a solution, the least squares iterative solution was preferred. In this case, data for point 6 represents the excess for determination of the R.O parameters using a least squares solution.

As the formulation goes, -L = A*dx + V

The values of the unknown changes, dx, is the immediate result of the functions:

Below is the Matlab script to form components of the linearized collinearity condition equations resulting in an **A** matrix of coefficients:

```
function A = getA(x1, y1, x2, y2, BX, BY, BZ, XI, YI, ZI, R, drw, drp, drk, f, n)
    % compute the matrix of coefficients of the unknowns, the A matrix.
    format short

    A = zeros(n*4,8+3*(n-1));  %dynamically calculate the size of A matrix.
    %in this case A will be of size (24 by 23) ==> 24 by 8+3*5
    p=6; %initialize column count for model coords

    for m = 1:n

        %index navigations
        a = m*4;

        % 8 unknowns for the first pair of conjugates
    %%elements from differentials of F1
        A(a-3,1) = 0;
        A(a-3,2) = 0;
        A(a-3,3) = 0;
        A(a-3,4) = 0;
        A(a-3,5) = 0;
        A(a-3,p) = f;
        A(a-3,p+1) = 0;
        A(a-3,p+2) = x1(m,1);

        %%elements from differentials of F2
        A(a-2,1) = 0;
        A(a-2,2) = 0;
        A(a-2,3) = 0;
        A(a-2,4) = 0;
        A(a-2,5) = 0;
        A(a-2,p) = 0;
        A(a-2,p+1) = f;
        A(a-2,p+2) = y1(m,1);
```

```
    %%elements from differentials of F3
        A(a-1,1) = ( drw(1,1)*x2(m,1) + drw(1,2)*y2(m,1) - drw(1,3)*f )*( ZI(m,1) - BZ ) - ( drw(3,1)*x2(m,1) + drw(3,2)*y2(m,1) - drw(3,3)*f )*(

        A(a-1,2) = ( drp(1,1)*x2(m,1) + drp(1,2)*y2(m,1) - drp(1,3)*f )*( ZI(m,1) - BZ ) - ( drp(3,1)*x2(m,1) + drp(3,2)*y2(m,1) - drp(3,3)*f )*(

        A(a-1,3) = ( drk(1,1)*x2(m,1) + drk(1,2)*y2(m,1) - drk(1,3)*f )*( ZI(m,1) - BZ ) - ( drk(3,1)*x2(m,1) + drk(3,2)*y2(m,1) - drk(3,3)*f )*(

    A(a-1,4) = 0;
        A(a-1,5) = -( R(1,1)*x2(m,1) + R(1,2)*y2(m,1) - R(1,3)*f );

        A(a-1,p) = -( R(3,1)*x2(m,1) + R(3,2)*y2(m,1) - R(3,3)*f );
        A(a-1,p+1) = 0;
        A(a-1,p+2) = R(1,1)*x2(m,1) + R(1,2)*y2(m,1) - R(1,3)*f;

    %%elements from differentials of F4
        A(a,1) = ( drw(2,1)*x2(m,1) + drw(2,2)*y2(m,1) - drw(2,3)*f )*( ZI(m,1) - BZ ) - ( drw(3,1)*x2(m,1) + drw(3,2)*y2(m,1) - drw(3,3)*f )*( Y

        A(a,2) = ( drp(2,1)*x2(m,1) + drp(2,2)*y2(m,1) - drp(2,3)*f )*( ZI(m,1) - BZ ) - ( drp(3,1)*x2(m,1) + drp(3,2)*y2(m,1) - drp(3,3)*f )*( Y

        A(a,3) = ( drk(2,1)*x2(m,1) + drk(2,2)*y2(m,1) - drk(2,3)*f )*( ZI(m,1) - BZ ) - ( drk(3,1)*x2(m,1) + drk(3,2)*y2(m,1) - drk(3,3)*f )*( Y

    A(a,4) = R(3,1)*x2(m,1) + R(3,2)*y2(m,1) - R(3,3)*f;
        A(a,5) = -( R(2,1)*x2(m,1) + R(2,2)*y2(m,1) - R(2,3)*f );

        A(a,p) = 0;
        A(a,p+1) = -( R(3,1)*x2(m,1) + R(3,2)*y2(m,1) - R(3,3)*f );
        A(a,p+2) = R(2,1)*x2(m,1) + R(2,2)*y2(m,1) - R(2,3)*f;

        p=p+3;  %increment column positioning for model coordinate coefficients in A

    end
end
```

In order to execute, the script uses the data x1, y1 and x2, y2 and f given as well as initial parameters BX, BY, BX, XI, YI, ZI.

The elements of the A matrix represent coefficients to the unknown parameters in the dx matrix. These are *changes* in the order of : w2, p2, k2, BY, BZ, (XI, YI, ZI)- model coordinates for each point pair.

In the very first execution, the value of ***A*** matrix of size 24 by 23 was computed:

```
A =

   1.0e+04 *

  Columns 1 through 16

         0         0         0         0         0    0.0152         0    0.0001         0         0         0         0         0         0         0         0
         0         0         0         0         0         0    0.0152   -0.0089         0         0         0         0         0         0         0         0
    1.7284   -6.8551    2.8556         0    0.0092    0.0152         0   -0.0092         0         0         0         0         0         0         0         0
    5.0226         0   -3.0277   -0.0152    0.0086         0    0.0152   -0.0086         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0    0.0152         0   -0.0001         0         0         0         0         0
         0         0         0         0         0         0         0         0         0    0.0152    0.0001         0         0         0         0         0
         0   -5.0226   -0.1375         0    0.0090         0         0         0    0.0152         0   -0.0090         0         0         0         0         0
    5.0226         0   -2.9737   -0.0152   -0.0004         0         0         0         0    0.0152    0.0004         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0    0.0152         0   -0.0003         0         0
         0         0         0         0         0         0         0         0         0         0         0         0    0.0152    0.0092         0         0
   -1.9128   -6.7991   -3.1603         0    0.0089         0         0         0         0         0         0    0.0152         0   -0.0089         0         0
    6.8159    1.6655   -2.9351   -0.0152   -0.0096         0         0         0         0         0         0         0    0.0152    0.0096         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.0152         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.0152
         0   -5.0226    2.9538         0    0.0001         0         0         0         0         0         0         0         0         0    0.0152         0
    3.3465    0.0192   -0.0338   -0.0152    0.0089         0         0         0         0         0         0         0         0         0         0    0.0152
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
   -0.0513   -5.0062   -0.0847         0   -0.0001         0         0         0         0         0         0         0         0         0         0         0
    4.9745    0.0153    0.0270   -0.0152   -0.0003         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0   -5.0226   -2.9910         0   -0.0003         0         0         0         0         0         0         0         0         0         0         0
    3.3254    0.0487    0.0857   -0.0152   -0.0091         0         0         0         0         0         0         0         0         0         0         0
```

```
  Columns 17 through 23

         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
         0         0         0         0         0         0         0
    0.0092         0         0         0         0         0         0
   -0.0088         0         0         0         0         0         0
   -0.0001         0         0         0         0         0         0
   -0.0089         0         0         0         0         0         0
         0    0.0152         0    0.0097         0         0         0
         0         0    0.0152    0.0003         0         0         0
         0    0.0152         0    0.0001         0         0         0
         0         0    0.0152    0.0003         0         0         0
         0         0         0         0    0.0152         0    0.0085
         0         0         0         0         0    0.0152    0.0091
         0         0         0         0    0.0152         0    0.0003
         0         0         0         0         0    0.0152    0.0091
```

Subsequently, the L matrix is made from a call to the function below:

```
function L = getL(x1, y1, x2, y2, BX, BY, BZ, XI, YI, ZI, R, f, n)
    % compute the L matrix
    L = zeros(n*4,1);

    for m=1:n

        a= m*4; %matrix index navigations

        L(a-3,1) = x1(m,1)*ZI(m,1) + f*XI(m,1);
        L(a-2,1) = y1(m,1)*ZI(m,1) + f*YI(m,1);
        L(a-1,1) = (R(1,1)*x2(m,1) + R(1,2)*y2(m,1) - R(1,3)*f )*( ZI(m,1) - BZ ) - (R(3,1)*x2(m,1) + R(3,2)*y2(m,1) - R(3,3)*f )*( XI(m,1) - BX );
        L(a,1)   = (R(2,1)*x2(m,1) + R(2,2)*y2(m,1) - R(2,3)*f )*( ZI(m,1) - BZ ) - (R(3,1)*x2(m,1) + R(3,2)*y2(m,1) - R(3,3)*f )*( YI(m,1) - BY );

    end
end
```

And the results for the zeroth computation being:

```
L =

   1.0e+04 *

   -0.0319
    2.9322
   -0.0123
    2.8556
    3.0664
   -0.0464
    2.9737
   -0.1375
    0.0830
   -0.1918
   -0.1049
   -0.3103
   -0.0111
    5.7626
    0.0338
    5.8038
   -3.1921
   -2.9654
   -3.0670
   -2.9347
    0.2261
   -5.8453
   -0.0857
   -5.8410
```

The change values are then computed using the normal least squares formulation coded as:

```
function dx = getdx(A,L)
    %get the matrix of the unknowns
  d = ones(24,1);
    % W is the weight matrix : ones in the main diagonal, equal weight matrix
  W = diag(d);

    N = A' * W * A;

    % Matrix of coefficients Qxx = inv(N)
    qxx = inv(N);

    % The error matrix dl
    dl = A' * W * (-L);

    %the -L applied results in the formation of the least square solution

    % matrix of unknowns in the order
        %% dw2, dp2, dk2, dBY, dBZ, dXi, dYi, dZi
          %% these unknowns in the matrix dx are computed as change values

  dx = qxx * dl;

end
```

And the values found are below:

```
dx =

   -0.0037
    0.0080
   -0.0424
    0.6974
   -2.2818
    2.2275
 -204.4129
  -19.8152
 -201.8236
    3.1392
  -16.5362
   -5.6908
   21.1337
  -14.0558
    8.5813
 -386.6413
  -12.9194
  204.0951
  194.8705
    9.2981
    0.3615
  400.8140
  -27.2024
```

Applying these changes to the initial values produced the following results.

-ANGLES w2, p2, k2 : values are in radians

	Initial	Change	Resulting d m s	Standard deviations
w2	0	-0.0037	-1 19 57.32	0.0005
p2	0	0.008	2 52 52.58	0.0008
k2	0	-0.0424	-15 16 14.69	0.0003

-BASE Components BY, BZ: (mm)

	Initial	Change	Resulting	Standard deviations
BY	0	0.6974	0.6974	0.1247
BZ	0	-2.2818	-2.2818	0.0461

-X Coordinates (mm)

Point	Initial	Change	XI	Standard deviations
1	0	2.2275	2.2275	0.0306
2	200	-201.824	-1.824	0.0309
3	0	-5.6908	-5.6908	0.033
4	200	8.5813	208.5813	0.2938
5	0	204.0951	204.0951	0.276
6	200	0.3615	200.3615	0.2742

-Y Coordinates (mm)

Point	Initial	Change	YI	Standard deviations
1	0	-204.413	-204.413	0.4063
2	0	3.1392	3.1392	0.0289
3	187.5	21.1337	208.6337	0.3332
4	187.5	-386.641	-199.141	0.2946
5	-187.5	194.8705	7.3705	0.0352
6	-187.5	400.814	213.314	0.2972

-Z Coordinates (mm)

Point	Initial	Change	ZI	Standard deviations
1	-330.43	-19.8152	-350.245	0.6988
2	-330.43	-16.5362	-346.966	0.4511
2	-330.43	-14.0558	-344.486	0.5681
3	-330.43	-12.9194	-343.349	0.486
4	-330.43	9.2981	-321.132	0.4372
6	-330.43	-27.2024	-357.632	0.4926

DISCUSSIONS

Although the results above are not based on any iterations to convergence, it can be said that they are somewhat conformal with the expectations. The standard deviations computed for each value are reasonable therefore it is safe to assume that the results would be better when the computations are done iteratively.

-Model coordinates on one run

Point	X	Y	Z
1	2.2275	-204.413	-350.245
2	-1.824	3.1392	-346.966
2	-5.6908	208.6337	-344.486
3	208.5813	-199.141	-343.349
4	204.0951	7.3705	-321.132
6	200.3615	213.314	-357.632

The exercise of formation of a model from a stereo pair by the use of a mathematical formula (The collinearity condition equations) was a success!

While further iterations were not done due to constraints and challenges in formulating the convergence conditions, we are working to make improvements to this procedure and process.

References

1. Photogrammetry - Francis H. Moffitt, Edward M. Mikhail. Harper & Row, 1 Mar 1980
2. FGE 411, Photogrammetry II A. Relative Orientation by collinearity condition equations.