Varun Aggarwal
CE 597 – Lab 4

LAB 4: LINEAR APPROACH FOR PHOTOGRAMMETRIC INTERSECTION

Introduction

The objective of this lab is to implement Linear two-light intersection and Linear Multi-light Intersection to get the ground co-ordinates of the ground points. For this lab, 4 images are given. For each image, the following information is also available:

1. Exterior Orientation Parameters
2. Interior Orientation Parameters
3. Image coordinates of the ground points in each of the image

The implementation is done in MATLAB. Only points which are common in all the four images are considered for this project. Those points are: 5, 7, 8, 9, 10, 12, 13, 14, 15, 17, 18, 19, 20.

Linear two-light Intersection

Directly solving for ground co-ordinates would require solving system of non-linear equations. A work around to avoid solving non-linear equation is by calculating the scale values, λ & μ.

$$\begin{bmatrix} X_{o_r} - X_{o_l} \\ Y_{o_r} - Y_{o_l} \\ Z_{o_r} - Z_{o_l} \end{bmatrix} = \lambda \, R_{(\omega_l,\, \phi_l,\, \kappa_l)} \begin{bmatrix} x_l - x_p - dist_x \\ y_l - y_p - dist_y \\ -c \end{bmatrix} - \mu \, R_{(\omega_r,\, \phi_r,\, \kappa_r)} \begin{bmatrix} x_r - x_p - dist_x \\ y_r - y_p - dist_y \\ -c \end{bmatrix}$$

The 3 equations have 2 unknowns. Over-determined systems can be solved for using Least Square Adjustments.

Following this, the ground coordinates are calculated by weighted average of the result of each scale

$$\begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix}_l = \begin{bmatrix} X_{o_l} \\ Y_{o_l} \\ Z_{o_l} \end{bmatrix} + \hat{\lambda} \, R_{(\omega_l,\phi_l,\, \kappa_l)} \begin{bmatrix} x_l - x_p - dist_x \\ y_l - y_p - dist_y \\ -c \end{bmatrix}$$

$$\begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix}_r = \begin{bmatrix} X_{o_r} \\ Y_{o_r} \\ Z_{o_r} \end{bmatrix} + \hat{\mu} \, R_{(\omega_r,\phi_r,\, \kappa_r)} \begin{bmatrix} x_r - x_p - dist_x \\ y_r - y_p - dist_y \\ -c \end{bmatrix}$$

Results – Image pair 4 and 5

Unit (m)	XI	error BASC XI	YI	error BASC YI	ZI	error BASC ZI
Point 05	3.7271E+00	4.0175E-03	3.7781E+00	2.0321E-03	-6.4673E-02	6.1653E-03
Point 07	9.0258E-01	1.4920E-03	2.8177E+00	2.9734E-04	2.5854E-02	1.5411E-04
Point 08	1.8493E+00	1.9067E-03	2.8256E+00	1.8733E-04	6.4997E-03	5.7148E-04
Point 09	2.7993E+00	1.7449E-03	2.8334E+00	1.5724E-04	-2.0160E-02	3.1357E-03
Point 10	3.7431E+00	7.8965E-04	2.8390E+00	1.1151E-03	-4.4061E-02	5.1388E-04
Point 12	9.0742E-01	1.5102E-03	1.8701E+00	2.2540E-03	1.7017E-03	8.5066E-03
Point 13	1.8570E+00	1.8186E-03	1.8757E+00	1.4979E-04	7.4177E-03	2.5589E-04
Point 14	2.8034E+00	2.8366E-03	1.8834E+00	4.1523E-04	-6.0888E-03	3.5810E-03
Point 15	3.7531E+00	1.0501E-03	1.8926E+00	3.3926E-04	-2.1426E-02	5.1545E-04
Point 17	9.1419E-01	3.4874E-04	9.1890E-01	1.7560E-03	-4.3863E-03	8.2638E-03
Point 18	1.8597E+00	3.6567E-05	9.2751E-01	9.5603E-05	5.2343E-03	2.1853E-03
Point 19	2.8046E+00	3.2091E-03	9.3655E-01	1.0139E-03	5.0851E-03	8.1360E-03
Point 20	3.7607E+00	6.1937E-03	9.4204E-01	3.5349E-03	-1.6614E-02	6.8017E-03
Root Mean Square Error		**6.8608E-06**		**2.1175E-06**		**2.4577E-05**

The values obtained by this procedure give values very close to the values obtained from BASC. The root mean square error for XI is $6.8 \times 10^{-6}\, m$. Similarly, for YI and ZI, the RMSE is of the order of 10^{-6} and 10^{-5} respectively.

Results – Image pair 11 and 12

Unit (m)	XI	error BASC XI	YI	error BASC YI	ZI	error BASC ZI
Point 05	3.7300E+00	1.1273E-03	3.7791E+00	1.0198E-03	-6.7625E-02	3.2136E-03
Point 07	9.0089E-01	1.9157E-04	2.8178E+00	2.1952E-04	2.4319E-02	1.6894E-03
Point 08	1.8514E+00	2.6236E-04	2.8267E+00	9.3598E-04	2.4914E-03	3.4369E-03
Point 09	2.8022E+00	1.2152E-03	2.8338E+00	5.6219E-04	-2.6835E-02	3.5397E-03
Point 10	3.7458E+00	3.4640E-03	2.8386E+00	7.6201E-04	-4.6814E-02	2.2391E-03
Point 12	9.1125E-01	2.3268E-03	1.8683E+00	4.2771E-04	2.0834E-02	1.0626E-02
Point 13	1.8586E+00	2.6870E-04	1.8772E+00	1.2839E-03	9.0529E-03	1.8911E-03
Point 14	2.8086E+00	2.3550E-03	1.8835E+00	3.0779E-04	-1.4684E-02	5.0139E-03
Point 15	3.7557E+00	3.6072E-03	1.8915E+00	7.9403E-04	-2.6051E-02	5.1409E-03
Point 17	9.1444E-01	9.6639E-05	9.2021E-01	4.5103E-04	7.1993E-03	3.3217E-03
Point 18	1.8594E+00	2.9525E-04	9.2689E-01	7.1931E-04	2.0876E-03	9.6150E-04
Point 19	2.8105E+00	2.7345E-03	9.3253E-01	3.0105E-03	-1.1176E-02	8.1255E-03
Point 20	3.7422E+00	1.2251E-02	9.5151E-01	5.9403E-03	3.4129E-03	1.3226E-02
Root Mean Square Error		**1.5119E-05**		**3.8837E-06**		**3.5653E-05**

Similar to image pair 4-5, the ground coordinates for this stereo pair is very close to the results obtained from BASC. The RMSE for XI, YI and ZI is of the order of 10^{-5}, 10^{-6} and 10^{-5} respectively.

Linear Multi Light Ray Intersection

This procedure is used when a single tie point or ground point appears in multiple images. The ground coordinates of the said point is calculated by linearizing a system of non-linear equations followed by least squares adjustment.

From each image where a point appears, we get 2 equations. Hence for calculating, 3 unknown ground coordinates, we require the point to appear in at least 2 images. In this example, each point appears in 4 images. Hence, we have an over-determined system with 8 equations and 3 unknowns. The two equations from each of the images take the following form.

$$w_i^j X_I - u_i^j Z_I = w_i^j X_o^j - u_i^j Z_o^j$$
$$w_i^j Y_I - v_i^j Z_I = w_i^j Y_o^j - v_i^j Z_o^j$$

In this equation, $X_I, Y_I and\ Z_I$ are the ground coordinates which must be calculated, X_0, Y_0, Z_0 are the exterior orientation parameters while u, v, w are parameters which be obtain after linearization and are constant for each of the image point.

Results for images 4, 5, 11 and 12

Unit (m)	XI	error BASC XI	YI	error BASC YI	ZI	error BASC ZI
Point 05	3.7300E+00	1.1141E-03	3.7796E+00	5.7627E-04	-6.7947E-02	2.8917E-03
Point 07	9.0192E-01	8.3415E-04	2.8177E+00	3.1833E-04	2.5678E-02	3.3054E-04
Point 08	1.8504E+00	7.5742E-04	2.8264E+00	6.1279E-04	3.6212E-03	2.3071E-03
Point 09	2.8012E+00	1.9838E-04	2.8339E+00	6.5504E-04	-2.4339E-02	1.0439E-03
Point 10	3.7451E+00	2.7684E-03	2.8391E+00	1.2979E-03	-4.6393E-02	1.8183E-03
Point 12	9.0885E-01	7.7661E-05	1.8689E+00	9.9069E-04	9.6011E-03	6.0713E-04
Point 13	1.8579E+00	9.3776E-04	1.8763E+00	4.7272E-04	7.1150E-03	4.6808E-05
Point 14	2.8067E+00	4.5999E-04	1.8835E+00	3.5587E-04	-1.1664E-02	1.9939E-03
Point 15	3.7549E+00	2.8769E-03	1.8923E+00	6.5864E-05	-2.4447E-02	3.5368E-03
Point 17	9.1419E-01	3.4862E-04	9.1920E-01	1.4625E-03	1.1640E-03	2.7136E-03
Point 18	1.8594E+00	2.5122E-04	9.2727E-01	3.3200E-04	3.8189E-03	7.6981E-04
Point 19	2.8077E+00	7.4908E-05	9.3461E-01	9.3293E-04	-3.1107E-03	5.9775E-05
Point 20	3.7500E+00	4.5017E-03	9.4741E-01	1.8431E-03	-5.1198E-03	4.6929E-03
Root Mean Square Error		**3.0803E-06**		**8.2886E-07**		**5.0022E-06**

With multi-light ray intersection, the ground co-ordinates for each of the point is very close to the value obtained from BASC. The RMSE value for XI, YI and ZI is of the order of 10^{-6}, 10^{-7} and 10^{-6} respectively.

Compared to two-light ray intersection, this procedure gives more accurate results. This can be attributed to more data available to the model for calculating the ground co-ordinates. The more images a point is visible in, the higher accuracy would be obtained.

Conclusion

The two procedures for calculating the ground co-ordinates of points {5, 7, 8, 9, 10, 12, 13, 14, 15, 17, 18, 19, 20} were successfully implemented. Least Squares Adjustment is used for over-determined systems. The multi-light ray intersection gave results closer to BASC output compared to two light intersection.

Code

Following is the code for the two implementations. Each sub-heading in this section is the name of the function.

Main.m

```
% Author: Varun Aggarwal
% images: 4,5 and 11,12
% visible: 5,7-10,12-15,17-20
clc; clear all;
%% Linear Intersection - Images 4,5
[IOP, dist, EOP_4, xa_4, ya_4] = data(4);
[~,~,EOP_5, xa_5, ya_5] = data(5);
[XI_1,YI_1,ZI_1] =
linear_intersection(EOP_4,EOP_5,IOP,dist,xa_4,ya_4,xa_5,ya_5);

%% Linear Intersection - Images 11,12
[IOP, dist, EOP_11, xa_11, ya_11] = data(11);
[~,~,EOP_12, xa_12, ya_12] = data(12);
[XI_2,YI_2,ZI_2] =
linear_intersection(EOP_11,EOP_12,IOP,dist,xa_11,ya_11,xa_12,ya_12);

%% multi-ray intersection - Images, 4,5,11,12
[XI_3,YI_3,ZI_3] =
multi_intersection(EOP_4,EOP_5,EOP_11,EOP_12,IOP,dist,xa_4,ya_4,xa_5,y
a_5,xa_11,ya_11,xa_12,ya_12);

%% results
[table1,table2,table3] =
compile_results(XI_1,YI_1,ZI_1,XI_2,YI_2,ZI_2,XI_3,YI_3,ZI_3);
writetable(table1,"table.xls","Sheet",1,'WriteRowNames',true);
writetable(table2,"table.xls","Sheet",2,'WriteRowNames',true);
writetable(table3,"table.xls","Sheet",3,'WriteRowNames',true);
```

linear_intersection.m

```
%% function to calculate linear intersection
function [XI,YI,ZI] =
linear_intersection(EOP_L,EOP_R,IOP,dist,xal,yal,xar,yar)

% initialize varibles
num_GCP = length(xal);
lambda = zeros(num_GCP,1);
mew = zeros(num_GCP,1);
XI = zeros(num_GCP,1);
YI = zeros(num_GCP,1);
ZI = zeros(num_GCP,1);

% unpack IOP and EOP
[xp,yp,c]=assign_IOP(IOP);
[X0l,Y0l,Z0l,omegal,phil,kappal] = assign_EOP(EOP_L);
[X0r,Y0r,Z0r,omegar,phir,kappar] = assign_EOP(EOP_R);

% remove image distortion
[xal,yal] = remove_dist(xal,yal,dist,xp,yp);
[xar,yar] = remove_dist(xar,yar,dist,xp,yp);

% get rotation matrices
rot_l = rotation(omegal,phil,kappal);
rot_r = rotation(omegar,phir,kappar);

% LHS of equation
LHS = [X0r-X0l;Y0r-Y0l;Z0r-Z0l];

% for each point calculate lambda and mew
for i=1:num_GCP
    % RHS of equation
    RHS = [rot_l*[xal(i),yal(i),-c]',-1*rot_r*[xar(i),yar(i),-c]'];
    temp = inv(RHS'*RHS)*RHS'*LHS;
    lambda(i) = temp(1);
    mew(i) = temp(2);
end

% XI , YI, ZI
for i=1:num_GCP
    RHS1 = rot_l*[xal(i),yal(i),-c]';
    RHS2 = rot_r*[xar(i),yar(i),-c]';
    XI(i) = ( (X0l + lambda(i)*RHS1(1)) + (X0r + mew(i)*RHS2(1)) )/2;
    YI(i) = ( (Y0l + lambda(i)*RHS1(2)) + (Y0r + mew(i)*RHS2(2)) )/2;
    ZI(i) = ( (Z0l + lambda(i)*RHS1(3)) + (Z0r + mew(i)*RHS2(3)) )/2;
end

end
```

multi_intersection.m

```
%% function to calculate multi intersection
function [XI,YI,ZI] =
multi_intersection(EOP1,EOP2,EOP3,EOP4,IOP,dist,xa1,ya1,xa2,ya2,xa3,ya
3,xa4,ya4)

% initialize varibles
num_GCP = length(xa1);
XI = zeros(num_GCP,1);
YI = zeros(num_GCP,1);
ZI = zeros(num_GCP,1);

% unpack IOP and EOP
[xp,yp,c]=assign_IOP(IOP);
[X01,Y01,Z01,omega1,phi1,kappa1] = assign_EOP(EOP1);
[X02,Y02,Z02,omega2,phi2,kappa2] = assign_EOP(EOP2);
[X03,Y03,Z03,omega3,phi3,kappa3] = assign_EOP(EOP3);
[X04,Y04,Z04,omega4,phi4,kappa4] = assign_EOP(EOP4);

% remove image distortion
[xa1,ya1] = remove_dist(xa1,ya1,dist,xp,yp);
[xa2,ya2] = remove_dist(xa2,ya2,dist,xp,yp);
[xa3,ya3] = remove_dist(xa3,ya3,dist,xp,yp);
[xa4,ya4] = remove_dist(xa4,ya4,dist,xp,yp);

% get rotation matrices
rot1 = rotation(omega1,phi1,kappa1);
rot2 = rotation(omega2,phi2,kappa2);
rot3 = rotation(omega3,phi3,kappa3);
rot4 = rotation(omega4,phi4,kappa4);

% XI , YI, ZI
for i=1:num_GCP
    RHS1 = rot1*[xa1(i),ya1(i),-c]';
    RHS2 = rot2*[xa2(i),ya2(i),-c]';
    RHS3 = rot3*[xa3(i),ya3(i),-c]';
    RHS4 = rot4*[xa4(i),ya4(i),-c]';
    y = [RHS1(3)*X01-RHS1(1)*Z01;RHS1(3)*Y01-RHS1(2)*Z01;RHS2(3)*X02-
RHS2(1)*Z02;RHS2(3)*Y02-RHS2(2)*Z02;RHS3(3)*X03-
RHS3(1)*Z03;RHS3(3)*Y03-RHS3(2)*Z03;RHS4(3)*X04-
RHS4(1)*Z04;RHS4(3)*Y04-RHS4(2)*Z04];
    A = [RHS1(3),0,-RHS1(1);0,RHS1(3),-RHS1(2);RHS2(3),0,-
RHS2(1);0,RHS2(3),-RHS2(2);RHS3(3),0,-RHS3(1);0,RHS3(3),-
RHS3(2);RHS4(3),0,-RHS4(1);0,RHS4(3),-RHS4(2)];
    temp = inv(A'*A)*A'*y;
    XI(i) = temp(1);
    YI(i) = temp(2);
    ZI(i) = temp(3);
end

end
```

remove_dis.m

```
function [xa,ya] = remove_dist(xa,ya,dist,xp,yp)

% unpack distortion parameters
k1 = dist(1);
k2 = dist(2);
k3 = dist(3);
p1 = dist(4);
p2 = dist(5);
p3 = dist(6);

% calculate x_bar, y_bar, r
xa = xa - xp;
ya = ya - yp;
r = sqrt(xa.^2 + ya.^2);

% calculate delta_xr, delta_yr - radial distortion
delta_xr =  xa .* (k1*r.^2 + k2*r.^4 + k3*r.^6);
delta_yr =  ya .* (k1*r.^2 + k2*r.^4 + k3*r.^6);

% calculate delta_xd, delta_yd - de-centering lens distortion
delta_xd = (1+p3*r.^2).*(p1*(r.^2 + 2*xa.^2) + 2*p2*xa.*ya);
delta_yd = (1+p3*r.^2).*(2*p1*xa.*ya + p2*(r.^2 + 2*ya.^2));

% remove distortions
xa = xa - delta_xr - delta_xd;
ya = ya - delta_yr - delta_yd;

end
```

assign_IOP.m

```
function [xp,yp,c]=assign_IOP(IOP)
    xp = IOP(1);
    yp = IOP(2);
     c = IOP(3);
end
```

assign_EOP.m

```
function [X0,Y0,Z0,omega,phi,kappa]=assign_EOP(EOP)
       X0 = EOP(1);
       Y0 = EOP(2);
       Z0 = EOP(3);
    omega = EOP(4);
      phi = EOP(5);
    kappa = EOP(6);
end
```

rotation.m

```
function [rot]=rotation(omega, phi, kappa)
rot_o = [1 0 0; 0 cosd(omega) -sind(omega); 0 sind(omega)
cosd(omega)];
rot_p = [cosd(phi) 0 sind(phi); 0 1 0; -sind(phi) 0 cosd(phi)];
rot_k = [cosd(kappa) -sind(kappa) 0; sind(kappa) cosd(kappa) 0; 0 0
1];
rot = rot_o * rot_p * rot_k;
end
```

compile_results.m

```
function [tableXI,tableYI,tableZI] =
compile_results(XI1,YI1,ZI1,XI2,YI2,ZI2,XI3,YI3,ZI3)
    XIbasc =
[3.731135328;0.90108606018;1.8511589932;2.8010339963;3.7422959221;0.90
892725171;1.8588605461;2.8062554595;3.7520655984;0.91454137562;1.85966
74595;2.8078118825;3.7544680979];
    YIbasc =
[3.7801397656;2.8180218203;2.8257925207;2.8332144474;2.8378518827;1.86
78901457;1.8758730363;1.8838128654;1.8922797004;0.92066047041;0.927605
53219;0.93553829291;0.94557055946];
    ZIbasc = [-0.070838477818;0.026008316122;0.0059282538012;-
0.023295585379;-0.044574905144;0.010208246675;0.0071618170061;-
0.0096698096332;-0.020910207448;0.003877584251;0.003049069041;-
0.0030509411619;-0.0098127723745];
    out = [XI1,abs(XI1-XIbasc),YI1,abs(YI1-YIbasc),ZI1,abs(ZI1-
ZIbasc)];
    tableXI = array2table(out,'RowNames',["Image 05","Image 07","Image
08","Image 09","Image 10","Image 12","Image 13","Image 14","Image
15","Image 17","Image 18","Image 19","Image 20"] ...
                               ,'VariableNames',["XI","error BASC
XI","YI","error BASC YI","ZI","error BASC ZI"]);
    out = [XI2,abs(XI2-XIbasc),YI2,abs(YI2-YIbasc),ZI2,abs(ZI2-
ZIbasc)];
    tableYI = array2table(out,'RowNames',["Image 05","Image 07","Image
08","Image 09","Image 10","Image 12","Image 13","Image 14","Image
15","Image 17","Image 18","Image 19","Image 20"] ...
                               ,'VariableNames',["XI","error BASC
XI","YI","error BASC YI","ZI","error BASC ZI"]);
    out = [XI3,abs(XI3-XIbasc),YI3,abs(YI3-YIbasc),ZI3,abs(ZI3-
ZIbasc)];
```

```
    tableZI = array2table(out,'RowNames',["Image 05","Image 07","Image 
08","Image 09","Image 10","Image 12","Image 13","Image 14","Image 
15","Image 17","Image 18","Image 19","Image 20"] ...
                                   ,'VariableNames',["XI","error BASC 
XI","YI","error BASC YI","ZI","error BASC ZI"]);
end
```

data.m

```
% images: 4,5 and 11,12
% visible: 55,7-10,12-15,17-20
% data: IOP, dist, EOPs(xyz,angle), image co-ordinates
function [IOP,dist,EOP,xa,ya]=data(image_no)
    IOP = [6.7451660984e-2,-1.1709829919e-1,8.1671200690];
    dist = [-2.9350008918e-4, 9.2190322166e-6,-2.2562559450e-
7,6.1878890685e-5,-7.2907688047e-5,0];

    switch image_no
        case 4
            EOP = [1.45699767,2.64424976,3.03050972,-15.4008911,-
9.57443972,1.47863766];
            xa = [4.9602257,-2.9072616,-0.1185862,2.3693304,4.55226,-
2.8170852,-0.2523924,2.0639685,4.1357188,-2.7435201,-
0.377666,1.7892301,3.7434257];
            ya = 
[4.8371958,2.8457077,2.6435584,2.4535729,2.2749642,0.1145881,0.0522464
,0.0108191,-0.0225489,-2.1970896,-2.1551888,-2.1014338,-2.0315526];
        case 5
            EOP = [1.2837936,1.2641037,3.70830313,7.55600777,-
12.1892602,1.18991695];
            xa = [2.9054506,-2.5036402,-
0.4372496,1.4070151,3.0541146,-2.5632522,-
0.4249594,1.4804908,3.205601,-2.6161877,-
0.4229522,1.5473036,3.3306421];
            ya = 
[3.4328731,2.2674354,2.1077771,1.9560619,1.8169063,0.1949646,0.1512282
,0.1093482,0.0757197,-2.0096401,-1.9341135,-1.8615865,-1.7965655];
        case 11
            EOP = [1.01405585,2.15717101,3.04105652,-0.265578564,-
16.4035252,92.7137826];
            xa = [3.4408691,2.1209013,1.8429667,1.6039644,1.4025225,-
0.5840479,-0.6140985,-0.643868,-0.6658373,-3.2440187,-3.049448,-
2.8851356,-2.740538];
            ya = [-4.0409373,2.5226965,-0.0616827,-2.2054623,-
4.0054995,2.6192178,0.0357842,-2.1219981,-
3.9517332,2.72723,0.1558043,-2.0154152,-3.8645389];
        case 12
            EOP = [2.25034272,2.22607133,3.76196308,-
0.80352916,3.90459588,92.4978077];
```

```
            xa = [3.4328675,1.5432244,1.4960436,1.4360757,1.3713602,-
0.4810031,-0.5612962,-0.6418672,-0.7224963,-2.4757077,-2.6001547,-
2.7110572,-2.805411];
            ya = [-4.0845237,2.1554622,0.1297945,-1.9445646,-
4.0438733,2.2103863,0.2024928,-1.8679638,-
3.9819054,2.2669273,0.2853745,-1.7759561,-3.8695091];
    end
end
```