Analysis of B4 overlapping swaths

Ramon Arrowsmith

ramon.arrowsmith@asu.edu

August 30, 2009

Like most LiDAR topography scanning projects, the B4 effort (http://www.earthsciences.osu.edu/b4/Site/Welcome.html; data are available from http://www.opentopography.org/) involved multiple overlapping passes to achieve the return densities that were desired. This strategy introduces a type of "corduroy" artifact that results from the relative mislocation of the different swaths at the decimeter level. Mike Bevis (http://www.earthsciences.osu.edu/faculty_bios.php?id=79) once explained to me that this was largely due to an error source in the GPS positioning that had a wavelength measured in minutes and amplitude of a decimeter or so. The plane flies ~50 km long swaths and returns, so the juxtaposition of the different swaths occurs at times separated by minutes to tens of minutes—just about right to see the effect of the GPS error source. I have talked with numerous people, including Mike Bevis about looking more into this problem by doing some swath to swath comparisons. In addition, the swath mismatch problem can be a prototype of actual change detection, by assuming that the first swath's different from subsequent ones is not error, but actual displacement.

I finally had some time to look into this problem. I wrote some MATLAB code which is slow (see appendices), but it takes any point cloud output from www.opentopography.org and reads each line, parsing the x,y,z positions into separate flightline groups. The program determines how many flightlines there are and analyzes each one separately. This includes gridding up the data to produce a DEM, visualizing it as a hillshade, and subtracting one flightline's DEM from the next. This makes the analysis a bit simpler because we know that the grid nodes of the DEMS are in the same places, so it effectively reduces the problem to 2 dimensions and looks at the vertical difference between the flightlines, but really, it is a 3D displacement as well as possible rotation problem. But doing a point cloud comparison (cloud match) is not something I know how do to at the moment.

In my example, I pulled a dataset of 571,318 points in the area of Calimesa near San Bernardino California. See this link http://lidar.asu.edu/KnowledgeBase/124875615909578.kmz for a more extensive KMZ file which can be browsed in GoogleEarth. There were 4 flight lines: '"Str_187"' '"Str_188"'   '"Str_189"'   '"Str_190"'. The data were separated and gridded into a ~1.25 m DEM, and figure 1 shows the resulting hillshades. The study area is 397 m wide and 355 m tall. The overall average shot density is 4 shots per square meter.

I subtracted each successive grid from the prior to produce the difference plots in figures 2, 3, and 4. What I did not expect was the range of differences. There are cases of a few tens to a few hundred meters of difference, while most of the differences are in the -0.5 to 0.5 m range. The large differences are actually the result of real changes in the scans, such as different illuminations of structures such as bridges, low altitude atmospheric changes (dust or exhaust), and movements such as vehicles. The finer differences are probably the result of GPS-induced error. While these finer errors are not skewed much, they are biased several cm in each case (figures 2-4).

For a detailed example, I took the points from Str_188 and Str_188 from the area around the I-10 overpass and selected them interactively and plotted them in 3D (figures 5 and 6). The clear aspect variation in elevation difference is a result mostly of aircraft position relative to the target. For example, the south side of the bridge looked higher in the first flight (negative means that Str_188 grid is higher than the Str_189 grid). That can be explained if the plane was relatively farther south in the earlier flight (makes sense if you look at the hillshades in Figure 1; the plane was moving north in each swath). In the earlier flight, the laser could see further under the bridge. The opposite is true for the north side. In addition, the elongate blue and red blobs on I-10 are vehicles that were there in the Str_188 flight if they were blue and the Str_189 flight if they are red, but not there for both flights.

In summary, I did not easily find the GPS-induced error for which I was looking except for a hint of it in the bias between the finer differences of scans. I did find actual change in the study area due do variation in scan illumination direction and in feature movement.

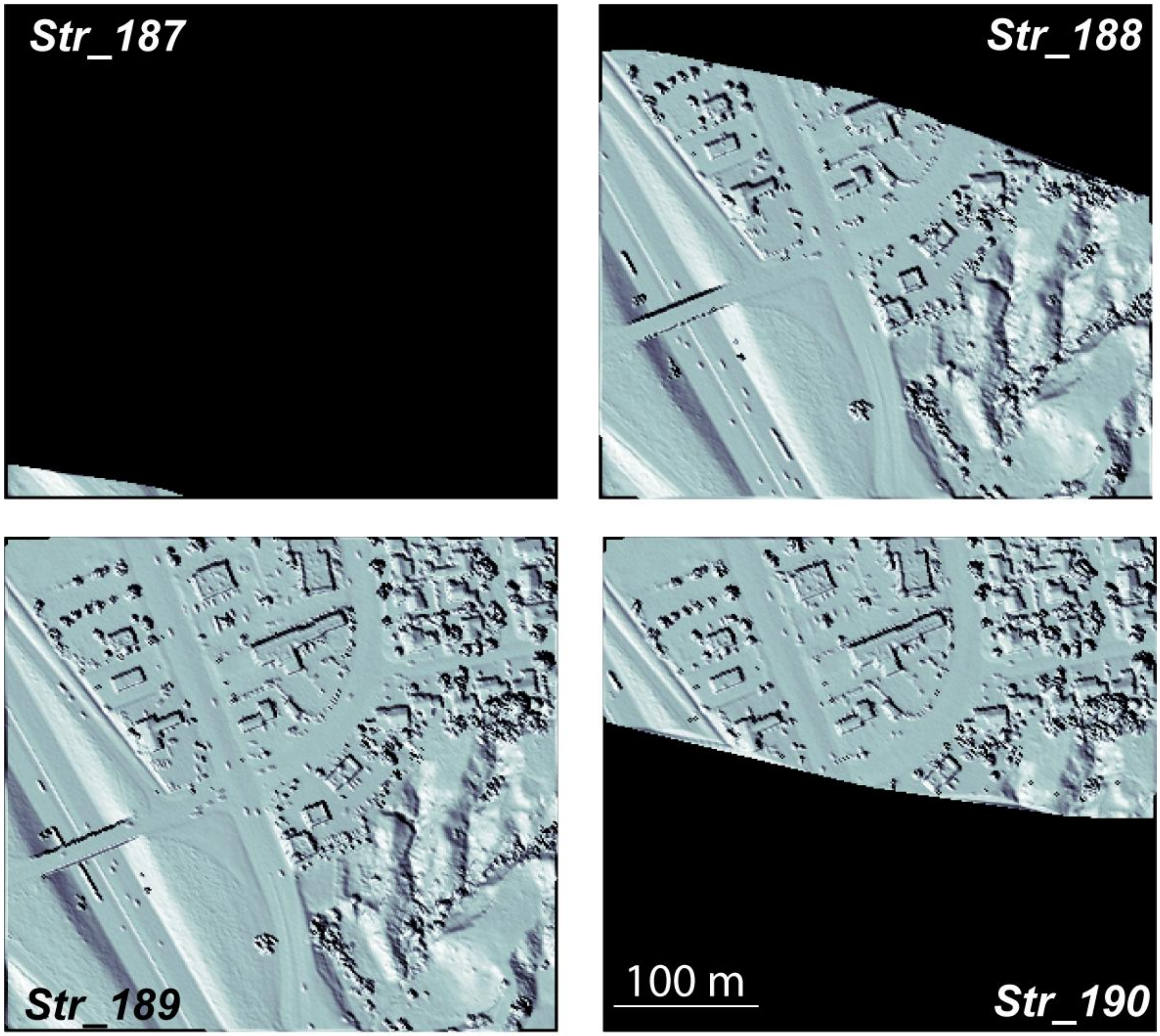

Figure 1. 4 DEMS shown as hillshades from 4 successive B4 overflights. Note that actual changes such as vehicle movements are evident from the flightlines whose acquisitions were separated by 10s of minutes.

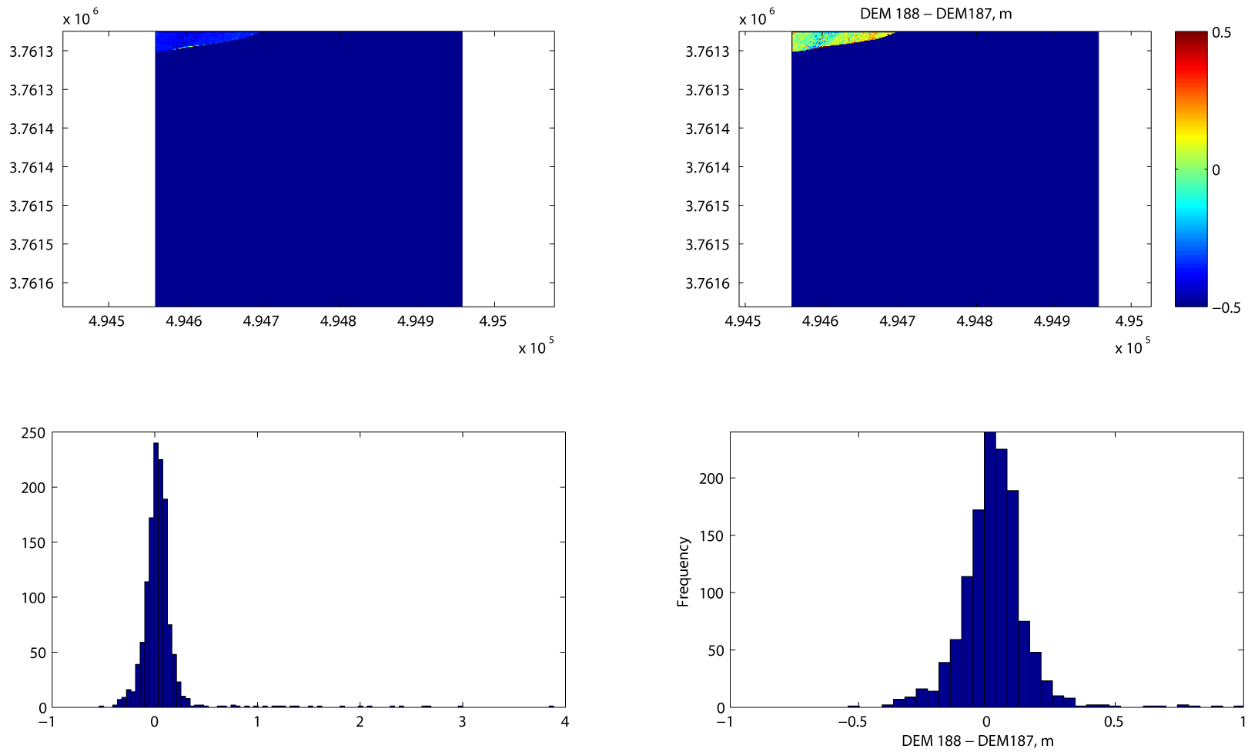

Figure 2. DEM subtraction flight 187 minus flight 188. Upper panels are the differences mapped and the lower panels are the histograms of the differences in meters. The left panels show the full range of differences while the right panels limit the differences to -0.5 to 0.5 m.

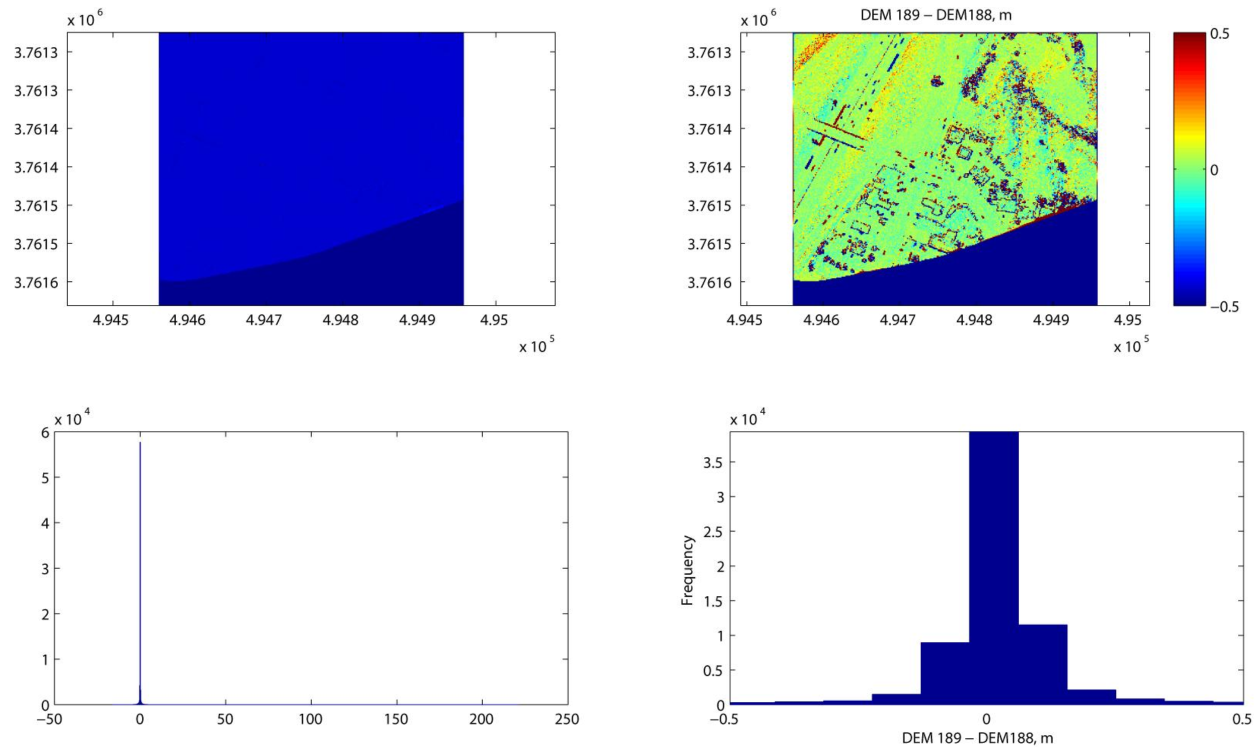

Figure 3. DEM subtraction flight 188 minus flight 189. Upper panels are the differences mapped and the lower panels are the histograms of the differences in meters. The left panels show the full range of differences while the right panels limit the differences to -0.5 to 0.5 m.

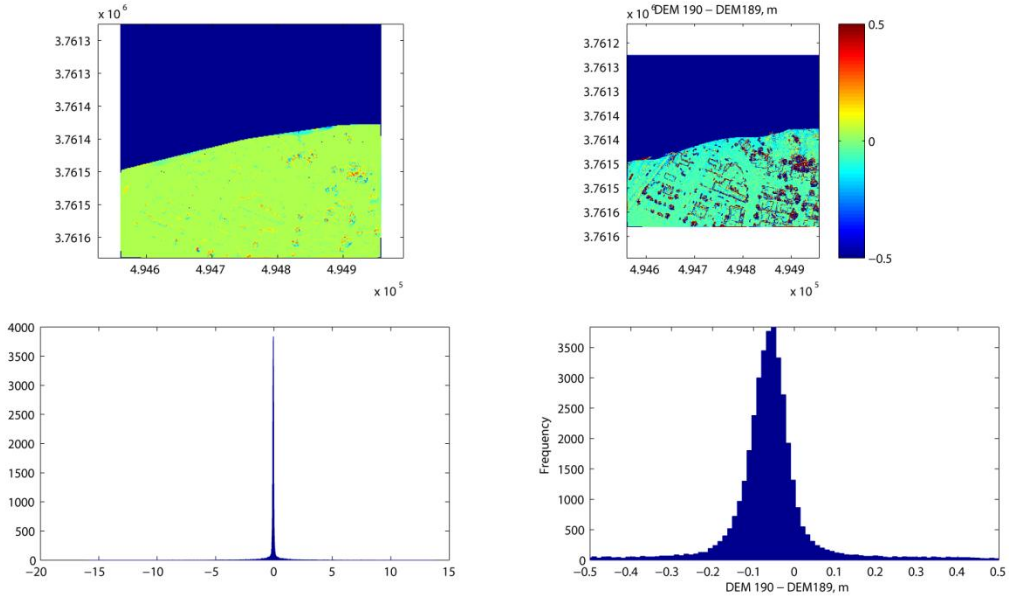

Figure 4. DEM subtraction flight 190 minus flight 189. Upper panels are the differences mapped and the lower panels are the histograms of the differences in meters. The left panels show the full range of differences while the right panels limit the differences to -0.5 to 0.5 m.

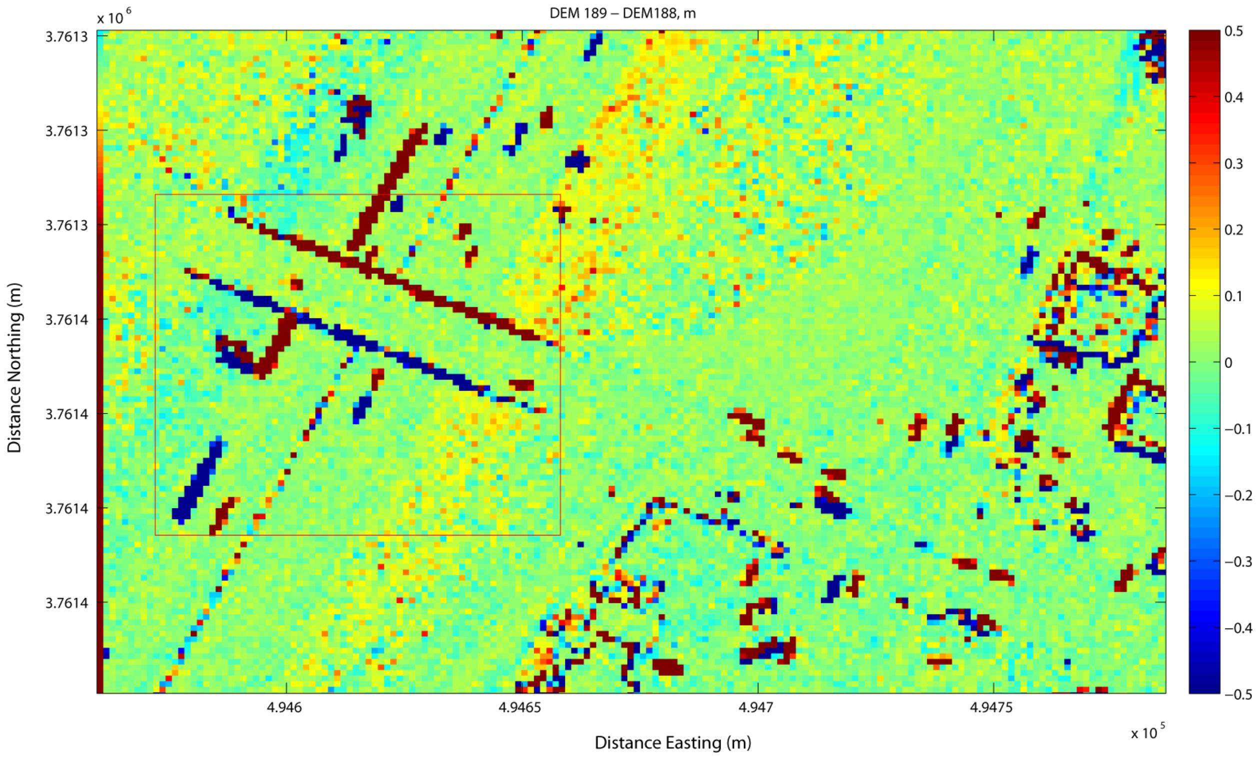

Figure 5. DEM subtraction flight 189 minus flight 188 zoomed to the overpass of I-10. The difference range is limited here to -0.5 to 0.5. The clear aspect variation in elevation difference is a result mostly of aircraft position relative to the target. The red box shows the selection area for the points in figure 6.

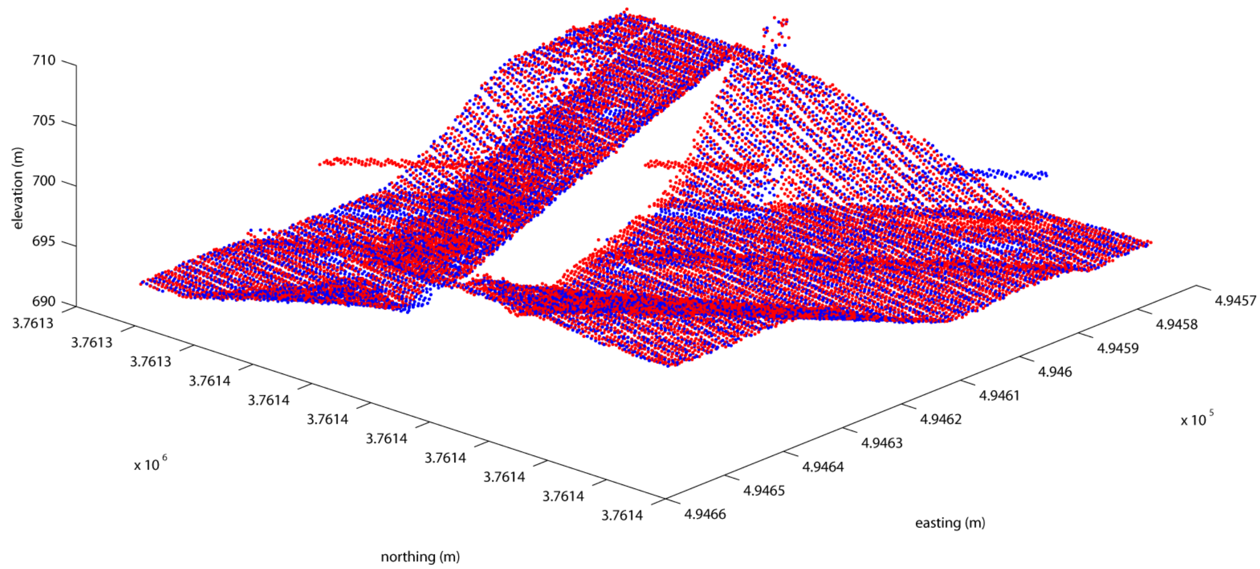

Figure 6. Point cloud comparison from selection area indicated by red box in figure 5. 3D view looking southeast. Red points are from Str_189 and blue points are from Str_188. This view also shows the better illumination from the north in Str_189 (note the red points further under the bridge). “Red” and”blue” vehicles are also apparent, as well as some exhaust (?) in the right center.

Appendix A. Main MATLAB script for analysis

```
%script to analyze different flightlines in a LiDAR point cloud
%JRA 7/27/09
%clear all
close all
clear all
%load all the points and concatenate them
input_fid=fopen('1248760015973169933689 9.txt');
% file should look like:
% 494836.83,3761290.97,693.73,"493022.181687",49,"Str_188"
% 494629.24,3761355.02,700.07,"493825.723392",48,"Str_189"
xcol=1;
ycol=2;
zcol=3;
delimiter=',';
flightcol=6;
data_line=0;
line_number = 0;
line_counter = 0;
num_lines = 1000;
xmin=inf;
ymin=inf;
zmin=inf;
xmax=-inf;
ymax=-inf;
zmax=-inf;


%This is the main while loop
fgetl(input_fid)
x_vector=[];
y_vector=[];
z_vector=[];
flightvector=[];
i=1;
tic
while data_line~=-1
    data_line = fgetl(input_fid);

    [x, y, z, flight, line_number] = pull_XYZClass(data_line, xcol, ycol,
zcol, flightcol, line_number, delimiter);
    if data_line~=-1
    x_vector=[x_vector x];
    y_vector=[y_vector y];
    z_vector=[z_vector z];
    flightvector=[flightvector flight];

    if x<xmin
        xmin=x;
    end
    if y<ymin
        ymin=y;
    end
    if z<zmin
        zmin=z;
```

```
    end
    if x>xmax
        xmax=x;
    end
    if y>ymax
        ymax=y;
    end
    if z>zmax
        zmax=z;
    end


    if line_number==1
        flightname=flight;
    else
        tf = strcmp(flightname,flight);
        tf = sum(tf);
        if tf==0
           flightname=[flightname flight];
        end
    end


    line_counter=line_counter+1;
    if line_counter==num_lines
    fprintf('I have analyzed %d lines\n', line_number)
    line_counter=0;
    end
    end
end
fclose(input_fid);
t=toc;
fprintf('\n')
fprintf('%6.2f seconds to process %d lines = %6.2f lines per second\n',t,
line_number, line_number./t)

size(flightname);
num_flights=ans(2)
fprintf('There are %d flights:\n', num_flights)
flightname

figure(1)
clf
hold on
figure(2)
clf
hold on
figure(3)
clf
hold on

Azimuth=315;
Zenith=45;
Zfact=1;
```

```
num_steps=300;
dx = (xmax-xmin)/num_steps;
xnodes = xmin:dx:xmax;
dy = (ymax-ymin)/num_steps;
ynodes = ymin:dy:ymax;

for i=1:num_flights
    tf=strcmp(flightname(i),flightvector);
    x_flight=x_vector(tf);
    y_flight=y_vector(tf);
    z_flight=z_vector(tf);

    figure(1)
    subplot(1,num_flights,i);
    plot(x_flight,y_flight,'k.')
    axis equal
    axis([xmin xmax ymin ymax])

    [XI,YI] = meshgrid(xnodes,ynodes);
    ElevArray{i} = griddata(x_flight,y_flight,z_flight,XI,YI);

    figure(2)
    subplot(1,num_flights,i);
    plot(xmin,ymin,'k.')
    imagesc(xnodes, ynodes, ElevArray{i}, [zmin zmax])
    axis equal
    axis([xmin xmax ymin ymax])

    figure(3)
    subplot(1,num_flights,i);
    [dElev_dx, dElev_dy]   = gradient(ElevArray{i},dx,dy);
    Hillshds=hillshademe(Azimuth, Zenith, Zfact, dElev_dy, dElev_dx);
    plothillshade(xnodes, ynodes, Hillshds)
    axis([xmin xmax ymin ymax])




end

figure(4)
clf
n=length(ElevArray)-1;
for i=1:n
    difference=ElevArray{i+1}-ElevArray{i};
    subplot(1,n,i)
    imagesc(xnodes,ynodes,difference)
    axis equal
end
colormap jet
colorbar
```

Appendix B. PullXYZClass function required for main MATLAB script

```
function [x, y, z, class, line_number] = pull_XYZClass(data_line, xcol, ycol,
zcol, classcol, line_number, delimiter)
%This function reads a line of a LiDAR point cloud file and returns the x,
%y, z coordinates as double precision numbers and the classification (or
flightline) as a
%string
%the line might look like this
%x,y,z,gpstime,intensity,classification,flight_line
%572184.64,4131383.47,626.35,"414262.936400",0,"2","1"
%
%This sequentially reads through the line, pulling the parts out depending
%on the comma delimeter



remain = data_line;
data_vector =[];
i=1;
%this came from the Matlab Help:
while true
    [str, remain] = strtok(remain, delimiter);
   if isempty(str),  break;  end
   data_vector{i}=str;
   i=i+1;
end

%pull the x, y, z, class out
if data_line~=-1
    line_number=line_number+1;
    x=str2double(data_vector(xcol));
    y=str2double(data_vector(ycol));
    z=str2double(data_vector(zcol));
    class=data_vector(classcol);
else
    x=inf; y=inf;z=inf; class=inf;
end
```

Appendix C: Function hillshademe necessary for main MATLAB script

```
function Hillshds=hillshademe(CurrentAzimuth, CurrentZenith, CurrentZ_fact, 
dElev_dy, dElev_dx)
      %Azimuth is 0 towards south and increases counterclockwise so 135 is 
      %NE and 225 is NW

            CurrentAzimuth = 360.0-CurrentAzimuth+90; %convert to mathematic 
unit
            CurrentAzimuth(CurrentAzimuth>=360)=CurrentAzimuth-360;
            CurrentAzimuth = CurrentAzimuth * (pi/180); %  convert to radians

            %lighting altitude
            CurrentZenith = (90-CurrentZenith) * (pi/180); % convert to 
zenith angle in radians


            [asp,grad] = cart2pol(dElev_dy,dElev_dx); % convert to carthesian 
coordinates
            grad       = atan(CurrentZ_fact*grad); %steepest slope
            % convert asp
            asp(asp<pi)=asp(asp<pi)+(pi/2);
            asp(asp<0)=asp(asp<0)+(2*pi);


            Hillshds =  255.0*( (cos(CurrentZenith).*cos(grad) ) + ( 
sin(CurrentZenith).*sin(grad).*cos(CurrentAzimuth-asp)) ); % ESRIs algorithm
            Hillshds(Hillshds<0)=0; % set hillshade values to min of 0.
```

Appendix D: Function hillshademe necessary for main MATLAB script

```
function plothillshade(Easting, Northing, Hillshds)

plot(Easting(1,1),Northing(1,1), 'b.');
hold on
imagesc(Easting,Northing,Hillshds);
colormap('bone');
%colormap('summer');
axis equal;
```

Appendix E. Post processing script for MATLAB.

I used this to process the results that had been loaded into memory after the main script completed.

```
%script to process the Calimesa B4 overlaps
clims = [-0.5 0.5];
i=1;
difference188minus187=ElevArray{i+1}-ElevArray{i};
figure(10+i)
clf
subplot(2,2,1)
imagesc(xnodes,ynodes,difference188minus187)
axis equal

subplot(2,2,2)
imagesc(xnodes,ynodes,difference188minus187, clims)
colorbar
axis equal
title('DEM 188 - DEM187, m')

B=reshape(difference188minus187,301.*301,1);
subplot(2,2,3)
hist(B,100)
subplot(2,2,4)
N = hist(B,100);
axis([-1 1 -inf inf])
xlabel('DEM 188 - DEM187, m')
ylabel('Frequency')


i=2;
difference189minus188=ElevArray{i+1}-ElevArray{i};
figure(10+i)
clf
subplot(2,2,1)
imagesc(xnodes,ynodes,difference189minus188)
axis equal

subplot(2,2,2)
imagesc(xnodes,ynodes,difference189minus188, clims)
colorbar
axis equal
title('DEM 189 - DEM188, m')

B=reshape(difference189minus188,301.*301,1);
subplot(2,2,3)
hist(B,1000)
subplot(2,2,4)
N = hist(B,2500);
axis([-0.5 0.5 -inf inf])
xlabel('DEM 189 - DEM188, m')
ylabel('Frequency')
mode189minus188=mode(N)

i=3;
difference190minus189=ElevArray{i+1}-ElevArray{i};
```

```
figure(10+i)
clf
subplot(2,2,1)
imagesc(xnodes,ynodes,difference190minus189)
axis equal

subplot(2,2,2)
imagesc(xnodes,ynodes,difference190minus189, clims)
colorbar
axis equal
title('DEM 190 - DEM189, m')

B=reshape(difference190minus189,301.*301,1);
subplot(2,2,3)
hist(B,2500)
subplot(2,2,4)
N = hist(B,2500);
axis([-0.5 0.5 -inf inf])
xlabel('DEM 190 - DEM189, m')
ylabel('Frequency')
mode190minus189=mode(N)

figure(20)
clf
imagesc(xnodes,ynodes,difference189minus188, clims)
colorbar
axis equal
title('DEM 189 - DEM188, m')

%get the upper left and lower right of the area in which we want to plot
%the points

% rangeulandlr = ginput;
% xmin=rangeulandlr(1,1);
% xmax=rangeulandlr(2,1);
% ymin=rangeulandlr(1,2);
% ymax=rangeulandlr(2,2);

figure(20)
hold on
plot([xmin xmax xmax xmin xmin], [ymin ymin ymax ymax ymin],'r-')
i=2;
tf=strcmp(flightname(i),flightvector);
x_flight188=x_vector(tf);
y_flight188=y_vector(tf);
z_flight188=z_vector(tf);
locs = find(y_flight188>ymin & y_flight188 <ymax& x_flight188 > xmin &
x_flight188<xmax);
epoints188=x_flight188(locs);
npoints188=y_flight188(locs);
zpoints188=z_flight188(locs);
%plot(epoints188, npoints188, 'b.')
i=3;
tf=strcmp(flightname(i),flightvector);
x_flight189=x_vector(tf);
y_flight189=y_vector(tf);
```

```
z_flight189=z_vector(tf);
locs = find(y_flight189>ymin & y_flight189 <ymax& x_flight189 > xmin &
x_flight189<xmax);
epoints189=x_flight189(locs);
npoints189=y_flight189(locs);
zpoints189=z_flight189(locs);
%plot(epoints189, npoints189, 'r.')

figure(21)
clf
plot3(epoints188, npoints188, zpoints188, 'b.')
hold  on
plot3(epoints189, npoints189, zpoints189, 'r.')
xlabel('easting (m)')
ylabel('northing (m)')
zlabel('elevation (m)')
view(-37,30)

% el=45;
%
% for az = 1:360
%   view(az,el)
%     axis equal
%   M(az) = getframe;
% end
% figure(22)
% clf
% mov=movie(M)
```