UNIVERSITY OF CALIFORNIA,
IRVINE

APPENDIX
DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

GEOLOGICAL ENGINEER

in Civil and Environmental Engineering

by

Clara Salvador Roda

Dissertation Committee:
Professor Russell L. Detwiler, Chair
Professor Daniel Fernandez

2014
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF FIGURES</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>iv</td>
</tr>
<tr>
<td>0.1 Appendix 1</td>
<td>v</td>
</tr>
<tr>
<td>0.2 Matlab code</td>
<td>v</td>
</tr>
<tr>
<td>Bibliography</td>
<td>v</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Main common parameters in aperture 2.</td>
<td>v</td>
</tr>
<tr>
<td>2</td>
<td>Main common parameters in aperture 3.</td>
<td>vi</td>
</tr>
<tr>
<td>3</td>
<td>Main common parameters in aperture 4.</td>
<td>vii</td>
</tr>
<tr>
<td>4</td>
<td>Gray level adjustment. Gladjust.m script.</td>
<td>viii</td>
</tr>
<tr>
<td>5</td>
<td>Threshold function. Threshold.m script.</td>
<td>viii</td>
</tr>
<tr>
<td>6</td>
<td>Apply shift function. Applyshift.m script.</td>
<td>ix</td>
</tr>
<tr>
<td>7</td>
<td>Apply shift function. Applyshift.m script.</td>
<td>ix</td>
</tr>
<tr>
<td>8</td>
<td>Full pixel shift function. Full pixel shift new.m script.</td>
<td>x</td>
</tr>
<tr>
<td>9</td>
<td>Full pixel shift function. Full pixel shift new.m script.</td>
<td>x</td>
</tr>
<tr>
<td>10</td>
<td>Full pixel shift function. Full pixel shift new.m script.</td>
<td>xi</td>
</tr>
<tr>
<td>11</td>
<td>Averaging images after a image processing treatment.</td>
<td>xii</td>
</tr>
<tr>
<td>12</td>
<td>Averaging images after a image processing treatment.</td>
<td>xiii</td>
</tr>
<tr>
<td>13</td>
<td>General and main function ( part 1).</td>
<td>xiv</td>
</tr>
<tr>
<td>14</td>
<td>General and main function ( part 2).</td>
<td>xiv</td>
</tr>
<tr>
<td>15</td>
<td>General and main function ( part 3).</td>
<td>xiv</td>
</tr>
<tr>
<td>16</td>
<td>General and main function where all system will be displayed (part 1).</td>
<td>xv</td>
</tr>
<tr>
<td>17</td>
<td>General and main function where all system will be displayed (part 2).</td>
<td>xv</td>
</tr>
<tr>
<td>18</td>
<td>General and main function where all system will be displayed (part 3).</td>
<td>xvi</td>
</tr>
</tbody>
</table>
LIST OF TABLES
0.1 Appendix 1

0.2 Matlab code

```matlab
clear all
%Script made for defining all the common parameters that experiments 1to10
%have, then they belong to APERTURE 2.
%%
%We will save all the values as a matrix of values format .mat
output_file = 'parameters_ap2.mat';

%% Where are saved the roots of our work
root_dir = ['/Volumes/LaCie/Project/APERTURES/AP2/Images_and_scripts/'];
ref_dir = ['./Volumes/LaCie/Project/APERTURES/AP2/Images_and_scripts/'];
%% Defining where are the reference,dye, clear and aperture files.
ref_file = [ref_dir 'StrStr_AP2(Clear)_0000'];
dye_file = [ref_dir 'ap2_dye.fits'];
c1_file = [ref_dir 'ap2_clean.fits'];
cp_file = [ref_dir 'AP2.fits'];
%% Parameter values :
%pixel dimensions and size ( cm)
L=0.01571*ones(920,1907); %Where dx=dy=0.01571cm;
% ROI values taken from the reference image.
L = 63;
R = 1969;
T = 658;
D = 1577;
%Cell dimensions(cm):
L=30.4325;
B=14.90;
mean_ap = 0.02214;
% Total volume
Vt=b*h*mean_ap;

%ROIs used for calculating gray level adjustments
%Each ROI is defined by [Left Right Top Bottom]

gl_adj = [1560, 1580, 56, 180]; %Rightmost
1492, 1534, 60, 183; %Second Rightmost
1432, 1473, 60, 176;
1368, 1412, 60, 177;
1305, 1347, 60, 173;
1249, 1289, 60, 172];
% check registration parameters
i0= 190; % test point in y-direction (second value in impixelinfo)
j0=1588; % test point in x-direction (first value in impixelinfo)
size_tmp=20; % template size for shift checking
nshift=10 % amount of shift to check during registration

%Define your Aperature ROI here: [Left Right Top Bottom]
cut_roi = [L R T B];

save(output_file);
par=load(output_file);
```

Figure 1: Main common parameters in aperture 2.
Figure 2: Main common parameters in aperture 3.

```matlab
%Script made for defining all the common parameters that experiments 11 to 15
%have, then they belong to APERTURE 2.
%Clara Salvador Roda 30/10/2014
%%
% We will save all the values as a matrix of values format .mat
output_file = 'parameters_ap3.mat';
root_dir = ['/Volumes/LeCie/Project/APERTURES/AP3/Images and scripts/'];
% Defining where are the reference, dye, clear and aperture file:
ref_file = [root_dir 'StrStr_AP3(Clear)_0000\'];
dye_file = [root_dir 'Ap3_avg_dye.fts'];
c1_file = [root_dir 'Ap3_avg_clear.fts'];
ap_file = [root_dir 'AP3.fts'];
% Parameter values:
% ROI values taken from the reference image.
L = 52;
R = 1971.5;
T = 631;
B = 1567;
% Cell dimensions (cm):
b=30.4325;
h=14.90;
mean_ap = 0.02214;
% Pixel size
A=(0.01571*0.01571).*ones(937,1920); % Where dx=dy=0.01571 cm
% Total volume
Vt=b*h*mean_ap;

% Looking at the reference image, define ROI’s. Put all ROI’S in Matrix ‘R’
% Each ROI is defined by [Left Right Top Bottom]

\[
\begin{bmatrix}
1547, 1586, 36, 181; \text{ Rightmost} \\
1402, 1542, 36, 181; \text{ Second Rightmost} \\
1421, 1477, 36, 181; \\
1358, 1414, 36, 181; \\
1296, 1352, 36, 181; \\
1236, 1287, 36, 181;
\end{bmatrix}
\]

io = 197; % test point in x-direction (second value in impixelinfo)
j0=1593; % test point in y-direction (first value in impixelinfo)
size_tmp=20; % template size for shift checking
nshift=10; % amount of shift to check during registration

% Define your Aperature ROI here: [Left Right Top Bottom]
out_roi = [L R T B];
% Define a test point for any necessary shift adjustment (upcorr % function)

save(output_file);
par=load(output_file);
```
clear all
%Script made for defining all the common parameters that experiments 11 to 15
%have, then they belong to APERTURE 2.
%Clara Salvador Roda 30/10/2014
%%
%We will save all the values as a matrix of values format .mat
output_file = 'parameters_ap4.mat';
%Defining where are the reference, dye, clear and aperture file:
root_dir = ['/Volumes/LaCie/Project/'];
ref_dir = ['/Volumes/LaCie/Project/APERTURBS/AP4/Images_and_scripts/'];
%Reference Image
ref_file = [ref_dir 'StrStr_AP4(Clear)_0000'];
dye_file = [ref_dir 'Ap4_avg_dye.fts'];
c1_file = [ref_dir 'Ap4_avg_clear.fts'];
ap_file = [ref_dir 'AP4.fts'];
A=(0.01571*0.01571).*ones(933,1920);
% Parameter values:

%Call dimensions (Cm):
b=30.4325;
R=14.90;
mean_ap = 0.02214;
%Total volume
Vt=b*h*mean_ap;
% ROI values taken from the reference image.
%function [L,R,T,b] = parameters_ap3()
L = 53;
R = 1972;
T = 631;
B = 1563;

%Looking at the reference image/ StrStr3(Clear), define ROI's. Put all ROI's in Matrix 'B'
%Each ROI is defined by [Left Right Top Bottom]

% NEW VALUES;
    gl_edj = [1547, 1587, 34, 173; %Rightmostq
      1486, 1542, 34, 173;  %Second Rightmost
      1422, 1478, 34, 173;
      1360, 1417, 34, 173;
      1296, 1350, 34, 173;
      1236, 1286,24, 173;];

    io = 197;  % test point in x-direction (second value in impixelinfo)
    jo = 1593; % test point in y-direction (first value in impixelinfo)
    size_tmp=20; % template size for shift checking
    nshift=10;  % amount of shift to check during registration

%Define your Aperature ROI here: [Left Right Top Bottom]
out_roi = [L R T B];
%Define a test point for any necessary shift adjustment (cpcorr
%function)
test_points = [io jo];
save(output_file);
par=load(output_file);

Figure 3: Main common parameters in aperture 4.
function [adjusted m R]= gladjust(ref_image, meas_image, roi)
%Gray level adjustment, where from ROI intensity values from measured and 
%reference images a assumption is stablished and there is when the function 
%will be created :
% Clara Salvador Roda 10/13;
gl_adj=roi;
size(meas_image);
size(ref_image);
%calculating mean intensities of ref image in each specific ROI
for i = 1:size(gl_adj,1);
  refroi = ref_image(gl_adj(i,3):gl_adj(i,4),gl_adj(i,1):gl_adj(i,2));
  measroi = (meas_image(gl_adj(i,3):gl_adj(i,4),gl_adj(i,1):gl_adj(i,2)));
  refmean(i) = mean(refroi(:));
  measmean(i) = mean(measroi(:));
end
if (size(roi,1)>1)
  p=polyfit(refmean,measmean,1);
  R=corrcoef(refmean,measmean);
  R = R(1,2); %finding R^2 value
  m = p(1,1); %loco
else
  m=measmean/refmean;
end
adjusted=meas_image/m; %adjusted measured image

-----------------------------------------------------------------------

function [bin L] = threshold(x)
%Threshold function created Clara Salvador 02/2014;
x=mat2gray(x,[-2 2]);
%graythresh() computes a global threshold(level) that can be used to 
%convert an intensity image to a binary image with im2bw, is a normalized 
%intensity value that lies in a range(0,1).
% The graythresh function uses Otsu’s method, which chooses the threshold 
% to minimize the intraclass variance of the black and white pixels.
L=graythresh(x);

L=min(L);
bin = double(im2bw(x,L));

bin_size=0.005;
edges=0:bin_size:1;
% Histic Function : counts the number of values in vector x that
% fall between the edges vector (0,1)
figure(1); plot(edges,histc(x(:),edges));
SUBROUTINE: SHIFT AN INPUT IMAGE TO ALIGN WITH A REFERENCE IMAGE

OVERVIEW:
This code will use two or three inputs. If given only two inputs, (1) image requiring shift, (2) amount of shift needed, the code will shift the entire image and cut off the edges. If given three inputs, (1) image requiring shift, (2) amount of shift needed, (3) region of interest (roi), the code will only shift the input roi.

INPUTS:
- in = input image; input image that has shifted from a reference image
- shift = shift in i and j direction (get from: <<full_pixel_shift>>)  
- [roi] = region of interest, [..] refers to a secondary argument for this code, i.e. this code can shift either that whole image or just a specified roi.

OUTPUTS:
- imS = image shifted; the image after it is shifted to the reference image location. This image is now "registered" to the reference image.

DEVELOPED BY:
Ricardo Medina and Trevor Jones (9/17/2013)

FIXED BUG IN ORIGINAL CODE ('apply_shift_image')

function [imS] = apply_shift(im,shift)

    [ni nj] = size(im);
    dt_i = shift(1);
    dt_j = shift(2);
    A = zeros(size(im));

    % Define the range for i and j of the current image (im)
    i_int = 1;
    i_fin = ni;
    j_int = 1;
    j_fin = nj;

    A.i_int = i_int-(heaviside(-dt_i)*(dt_i));
    A.i_fin = i_fin-(heaviside(dt_i)*(dt_i));
    A.j_int = j_int-(heaviside(-dt_j)*(dt_j));
    A.j_fin = j_fin-(heaviside(dt_j)*(dt_j));

    im.i_int = i_int+heaviside(dt_i)*(dt_i);
    im.i_fin = i_fin-heaviside(-dt_i)*(dt_i);
    im.j_int = j_int+heaviside(dt_j)*(dt_j);
    im.j_fin = j_fin-heaviside(-dt_j)*(dt_j);

    A(A.i_int:A.i_fin,A.j_int:A.j_fin)=im(im.i_int:im.i_fin,im.j_int:im.j_fin);
    imS=A;

end

Figure 6: Apply shift function. Applyshift.m script.

Figure 7: Apply shift function. Applyshift.m script
function shift = full_pixel_shift_new(reference, test, io, jo, size_tmp, nshift)

% FULL_PIXEL_SHIFT measure amount of shift between pair of images
% INPUTS:
% REFERENCE - 2D reference image.
% TEST - 2D test image (should be same size as reference)
% IO, JO - i and j index of the center of the region to test for
% alignment
% SIZE_TMP - size of the template to use for checking (equivalent to
% box_size in old version)
% NSHIFT - number of pixels to shift through to check alignment (the
% following must be true: NSHIFT < SIZE_TMP)
% OUTPUTS:
% SHIFT -
% Note: IO-SIZE_TMP and JO-SIZE_TMP must be > 0
% IO+SIZE_TMP and JO+SIZE_TMP must be <= size(REFERENCE)

% Developed by Russ 7/2013

figure 8: Full pixel shift function. Full pixel shift new.m script

function shifting = full_pixel_shift_new(reference, test, io, jo, size_tmp, nshift)

% Test that region to be checked doesn't extend beyond reference image
[nx ny] = size(reference);
i_begin = io-nshift;
i_end = io+nshift;
j_begin = jo-nshift;
j_end = jo+nshift;
if (i_begin <= 0) || (i_end > nx) || (j_begin <= 0) || (j_end > ny),
    error('you asked me to check regions that fall outside of image boundaries')
end

% Define ROI in reference image
reference_roi = reference(i_begin:i_end,j_begin:j_end);

% Test that region to be checked doesn't extend beyond test image
[nx ny] = size(test);
i_begin = io-size_tmp;
i_end = io+size_tmp;
j_begin = jo-size_tmp;
j_end = jo+size_tmp;
if (i_begin <= 0) || (i_end > nx) || (j_begin <= 0) || (j_end > ny),
    error('you asked me to check regions that fall outside of image boundaries')
end

% Define ROI in test image
test_roi = test(i_begin:i_end,j_begin:j_end);

figure 9: Full pixel shift function. Full pixel shift new.m script
% Calculate cross-correlation matrix
C = normxcorr2(reference_roi, test_roi);
% cimshow(C)
% Define location corresponding to zero shift in cross-correlation matrix
delta = size(tmp) + nshift + 1;

% Find location of peak - actual shift
[max_c, imax] = max(abs(C(:, :)));
[ippeak, jpeak] = ind2sub(size(C), imax(1));

% this crashes on occasion - not sure why; perhaps two locations
% with max value? but the above version seems more robust
%(xpeak ypeak) = find(C == max(abs(C(:))));
shift = [(ipeak + delta) (jpeak + delta)];
end
Figure 11: Averaging images after a image processing treatment.
```matlab
fl = textrread(file_list,'%s');
num_imgs = length(fl);
ref_image = imread(Par.ref_file); % fitsread = imread
ref_roi = ref_image(Par.T:Par.B,Par.L:Par.R);
sum = zeros(size(ref_roi));

for m=1:num_imgs
    file = char(fl{m});
    im = imread(file);
    % Grey level adjust if necessary
    im = gladjust(ref_image,im,Par.gl_adj);
    % Check registration and adjust if necessary
    shift = full_pixel_shift_new(ref_image,im,Par.io,Par.jo,Par.size_tmp,Par.nshift);
    im = apply_shift(im,shift);
    % Extract ROI from image and increment sum
    im_roi = im(Par.T:Par.B,Par.L:Par.R);
    sum = sum + im_roi;
end

cvg_im = sum/num_imgs;
```

Figure 12: Averaging images after a image processing treatment.

```matlab
% Main function animate_bin new where the image processing and data is
% tested, and where is calling other functions as 
% gladjust, full_pixel_shift_new, apply_shift). From that file we will obtain
% 2 animations where: 1- the image is binarized and 2- histogram where is a
% thresholding function (graythresh).
% In addition, volume and saturation values will be given as a result.
% Clara Salvador Roda 11/13-2/14

function [L,V,S]=animate_bin_new(Par,Par_bin,file_list)

    f_list = textrread(file_list,'%s');
    nimg=length(f_list);

    ref = imread(Par.ref_file);
    dry = imread(Par_bin.dry_file);
    dry = gladjust(ref,dry,Par.gl_adj);
    shift = full_pixel_shift_new(ref,dry,Par.io,Par.jo,Par.size_tmp,Par.nshift);
    dry = apply_shift(dry,shift);
    dry = dry(Par.T:Par.B,Par.L:Par.R);

    sat = imread(Par.cl_file);
    ap = imread(Par.ap_file);

    % open an avi file object for writing frames to
display(Par_bin.avi_file)
aviobj=avifile(Par_bin.avi_file);
% set parameters of the aviobj (see 'avifile' help for options)
aviobj.fps=16;
aviobj.quality=100;
```

Figure 13: General and main function (part 1).
Figure 14: General and main function (part 2).

```matlab
avoibj2=avifile('histogram_test.avi');
avoibj2.fps=16;
avoibj2.quality=100;
Vt=Par.b*Par.h*Par.mean_sp;

avoibj3=avifile('final.avi');
avoibj3.fps=16;
avoibj3.quality=100;

for i=1:200

    file = char(f_list(i));
    im = imread(file);
    meas = double(imread(f_list(i))); % Read Image file, change to 'double'
    meas = double(imread(f_list(i)));

    % Conduct Gray Level Adjustment on Image file
    meas = gadijust(ref, meas, Par.gl_adj); % gray level adjust
    % Output the gray level image file and the the slope
    % Conduct Shift Adjustment on the new Gray Leveled Image file
    shift = full_pixel_shift_now(ref, meas, Par.lo, Par.l0, Par.size_tmp, Par.nshift);
    meas = apply_shift(meas, shift);
    meas = meas(Par.l:Par.B, Par.L:Par.R);
    x = (sat-meas)./(sat-dry);

    % Conduct Gray Level Adjustment on Image file
    meas = gadijust(ref, meas, Par.gl_adj); % gray level adjust
    % Output the gray level image file and the the slope
    % Conduct Shift Adjustment on the new Gray Leveled Image file
    shift = full_pixel_shift_now(ref, meas, Par.lo, Par.l0, Par.size_tmp, Par.nshift);
    meas = apply_shift(meas, shift);
    meas = meas(Par.l:Par.B, Par.L:Par.R);
    x = (sat-meas)./(sat-dry);

end
```

Figure 15: General and main function (part 3).
function [h]=animate_pressure(Par,Par_bin,file_list)
% Function where it will be displayed the experiment system (cell and
% manometer as an animation system)

f_list = tex.strip(file_list,'\$s\$');
nimg=length(f_list);

ref = imread(Par.ref_file);

dry = imread(Par_bin.dry_file);
dry = gladjust(ref,dry,Par.gl_adj);
shift = full_pixel_shift_new(ref,dry,Par.lo,Par.jo,Par.size_tmp,Par.nshift);
dry = apply_shift(dry,shift);
dry = dry(Par.Ti;Par.Bi;Par.L;Par.R);

sat = imread(Par.cl_file);
ap = imread(Par.ap_file);
writerObj = VideoWriter('exp_pre_test');
writerObj.FrameRate = 12;
open(writerObj);

for i=1:nimg   %Number of images are there in the experiment

% Image processing>
% %Read Image file, change to 'double'
meas = double(imread(f_list(i)));%

%Conduct Gray Level Adjustment on image file
meas = gladjust(ref,meas,Par.gl_adj); %gray level adjust
%Output the gray level image file and the teh slope
%Conduct Shift Adjustment on the new Gray Leveled Image file
shift = full_pixel_shift_new(ref,meas,Par.lo,Par.jo,Par.size_tmp,Par.nshift);

meas = apply_shift(meas,shift);
meas = meas(Par.Ti;Par.B, Par.L;Par.R);

x = (sat-meas)./(sat-dry);

%Thresholding:

[bin I] = threshold(x); % threshold is a function to be defined
%[g, NR, SI, TI]=regiongrow(f_list,x,0.1) % New function,
%segmentation. Is not working right now.
% bin = imrotate(bin,90);
bin = imrotate(bin,90);

%A-animate_bin_new(Par,Par_bin,file_list)
% Video manometer
% h(i)=[85.1-0.53301*i];% experiment 3, lst drainag
% 88.8-58.4=30.4. As we can see, the manometer scale will go from
% 0-30.4cm. And the air entry in image #8, makes to start at
% 30.4-4.4=26cm high
% h(i)=[30.4-0.53613*i] % First Drainage, exp.3
% h(i)=[0.565*i] % First wetting, exp3
% [h(i)=[30.4-0.51891*i] % Second Drainage, exp.3.
% h(i)=[0+0.54694*i] %Second wetting, exp3
%

%yy=[h(i) h(i)];

hfig=figure('Visible','off');

set(hfig,'Position',[50 50 1600 1200]);
hfig=subplot('Position',[0.4 0.2 0.2 0.88], 'Parent',hfig); hold(hfig,'on')

Figure 16: General and main function where all system will be displayed (part 1).

Figure 17: General and main function where all system will be displayed (part 2).
Figure 18: General and main function where all system will be displayed (part 3).