sample sequence of commands for analyzing a series of images scanned using a particular scanner. commands are run in MATLAB

1.
determine the correct path to the folder containing images from a run; in this example that path is 'D:\run_3-18-19\'
'SCAN_3-3_20190318_2056.tif' — a sample filename of the first image scanned
'D:\run_3-18-19\boardhint.mat' — the location where the data defining plate location in the images is stored

CreateBoardHint('D:\run_3-18-19\SCAN_3-3_20190318_2056.tif',6,'D:\run_3-18-19\boardhint.mat', 90)

2.
an image shows 6 plates. Create 6 folders, each containing sequential images of a particular plate
'D:\run_3-18-19\SCAN_3-3*.tif' - the sample image showcased here was generated by scanner 3-3; specify using images scanned by this scanner as input
{‘D:\run_3-18-19\out\AS149_1\1',
'D:\run_3-18-19\out\AS149_1\2','D:\run_3-18-19\out\AS149_1\3','D:\run_3-18-19\out\AS149_1\4','D:\run_3-18-19\out\AS149_1\5','D:\run_3-18-19\out\AS149_1\6'} — specify the 6 output folders. This assumes that every plate on a scanner was inoculated with the same strain, here AS149. Change as needed.
'D:\run_3-18-19\boardhint.mat' — specify the location of the board hint - the same as specified in the previous step;
[1;2;3;4;5;6] — specify the plates that correspond to the output folders

CropROI('D:\run_3-18-19\SCAN_3-3*.tif', {'D:\run_3-18-19\out\AS149_1\1',
'D:\run_3-18-19\out\AS149_1\2','D:\run_3-18-19\out\AS149_1\3','D:\run_3-18-19\out\AS149_1\4','D:\run_3-18-19\out\AS149_1\5','D:\run_3-18-19\out\AS149_1\6'},'D:\run_3-18-19\boardhint.mat',[1;2;3;4;5;6]);

3.
Specify the plate area to be used in detecting colonies. Tip: avoid edges and very small ‘colonies’

setMaskApp('D:\run_3-18-19\out\AS149_1\1')
setMaskApp('D:\run_3-18-19\out\AS149_1\2')
setMaskApp('D:\run_3-18-19\out\AS149_1\3')
setMaskApp('D:\run_3-18-19\out\AS149_1\4')
setMaskApp('D:\run_3-18-19\out\AS149_1\5')
setMaskApp('D:\run_3-18-19\out\AS149_1\6')

4.
calculate the increase in surface area over time for all the colonies on a given plate

ProcessPlates({'D:\run_3-18-19\out\AS149_1\1'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-1'});
ProcessPlates({'D:\run_3-18-19\out\AS149_1\2'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-2'});
ProcessPlates({'D:\run_3-18-19\out\AS149_1\3'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-3'});
ProcessPlates({'D:\run_3-18-19\out\AS149_1\4'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-4'});
ProcessPlates({'D:\run_3-18-19\out\AS149_1\5'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-5'});
ProcessPlates({'D:\run_3-18-19\out\AS149_1\6'},{'D:\run_3-18-19\run_SCAN20190318_2054.log'},{'AS149_1-6'});

5.
optionally, visualize the raw growth curves generated for each plate

PlateAnalyzer('D:\run_3-18-19\out\AS149_1\1')
PlateAnalyzer('D:\run_3-18-19\out\AS149_1\2')
PlateAnalyzer('D:\run_3-18-19\out\AS149_1\3')
PlateAnalyzer('D:\run_3-18-19\out\AS149_1\4')
PlateAnalyzer('D:\run_3-18-19\out\AS149_1\5')
PlateAnalyzer('D:\run_3-18-19\out\AS149_1\6')

6.
specify the path names to each folder containing image data for a particular plate. All plates correspond to the same strain

names149 = {D:\run_3-18-19\out\AS149_1\1',
'D:\run_3-18-19\out\AS149_1\2','D:\run_3-18-19\out\AS149_1\3','D:\run_3-18-19\out\AS149_1\4','D:\run_3-18-19\out\AS149_1\5','D:\run_3-18-19\out\AS149_1\6'};

7.
determine the pixel range that results in the least percent standard deviation. View details by exploring the created as149Summary and as149Struct variables. Save the struct into a mat file for future use, to prevent rerunning the command, which has a long runtime, in future

```
[as149Summary, as149Struct] = optimal_pixel_range(names789);
save('D:\mat_data\summary_structs\AS149.mat', '-struct', 'as149Struct');



# 8.
# after doing steps 1-7 for each strain, generate a boxplot showing the doubling time for
each strain


dists = [transpose(as149Struct.growth*24) transpose(as821Struct.growth*24)
transpose(as789Struct.growth*24) transpose(as790Struct.growth*24)
transpose(as791Struct.growth*24) transpose(as792Struct.growth*24)
transpose(as794Struct.growth*24) transpose(as798Struct.growth*24)
transpose(as799Struct.growth*24) transpose(as800Struct.growth*24)
transpose(as802Struct.growth*24) transpose(as803Struct.growth*24)
transpose(as804Struct.growth*24)];
grp = [ones(1,length(as149Struct.growth)), 2*ones(1,length(as821Struct.growth)),
3*ones(1,length(as789Struct.growth)), 4*ones(1,length(as790Struct.growth)),
5*ones(1,length(as791Struct.growth)), 6*ones(1,length(as792Struct.growth)),
7*ones(1,length(as794Struct.growth)), 8*ones(1,length(as798Struct.growth)),
9*ones(1,length(as799Struct.growth)), 10*ones(1,length(as800Struct.growth)),
11*ones(1,length(as802Struct.growth)), 12*ones(1,length(as803Struct.growth)),
13*ones(1,length(as804Struct.growth))];
labels = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12'}
boxplot(dists, grp, 'Notch', 'on', 'Labels', labels)
set(gca,'FontSize',28)
xlabel('Strain');
ylabel('Doubling time (hrs)');
set(findobj(gcf, 'type', 'line', 'Tag', 'Median'), 'Color', 'r');
boxes = findobj(gcf, 'type', 'line', 'Tag', 'Box');
set(boxes, 'linew', 2)
set(boxes(1:3), 'Color', [0.1333 0.1412   0.4314]);
set(boxes(4:6), 'Color', [0.3569 0.5020 0.1882]);
set(boxes(7:9), 'Color', [0.8706 0.7098 0.1451]);
set(boxes(10:12), 'Color', [0.8784 0.1804 0.2824]);
set(boxes(13), 'Color', [0 0 0]);



# 9.
# optionally, view a bar graph showing standard error


std_devs = [as149Struct.minStd as821Struct.minStd as789Struct.minStd
as790Struct.minStd as791Struct.minStd as792Struct.minStd as794Struct.minStd
as798Struct.minStd as799Struct.minStd as800Struct.minStd as802Struct.minStd
```

```
as803Struct.minStd as804Struct.minStd];
lengths = [length(as149Struct.growth) length(as821Struct.growth)
length(as789Struct.growth) length(as790Struct.growth) length(as791Struct.growth)
length(as792Struct.growth) length(as794Struct.growth) length(as798Struct.growth)
length(as799Struct.growth) length(as800Struct.growth) length(as802Struct.growth)
length(as803Struct.growth) length(as804Struct.growth)];
sqr_len = arrayfun(@(x) x^.5, lengths);
err = std_devs./sqr_len;
growths = [as149Struct.growthMean as821Struct.growthMean as789Struct.growthMean
as790Struct.growthMean as791Struct.growthMean as792Struct.growthMean
as794Struct.growthMean as798Struct.growthMean as799Struct.growthMean
as800Struct.growthMean as802Struct.growthMean as803Struct.growthMean
as804Struct.growthMean];
strains = {'AS149', 'AS821', 'AS789', 'AS790', 'AS791', 'AS792', 'AS794', 'AS798',
'AS799', 'AS800', 'AS802', 'AS803', 'AS804'};
x = categorical(strains);
bar(x, growths)
errorbar(1, growths(1), err(1), 'b', 'LineWidth', 2)
hold on
errorbar(2:4, growths(2:4), err(2:4), 'color',[ 0.9098 0.6824 0.7176], 'LineWidth', 2)
errorbar(5:7, growths(5:7), err(5:7), 'color',[0.7216 0.8824 1.0000], 'LineWidth', 2)
errorbar(8:10, growths(8:10), err(8:10), 'color',[0.6627 1.0000 0.9686], 'LineWidth', 2)
errorbar(11:13, growths(11:13), err(11:13), 'color',[0.5804 0.9843    0.6706], 'LineWidth',
2)
xlabel('Strain');
ylabel('Doubling time (hrs)');
hold off


# 10.
# assess statistical significance of any differences observed using ANOVA
grps = [repelem("0", length(as149Struct.growth)), repelem("1",
length(as821Struct.growth)), repelem("2", length(as789Struct.growth)), repelem("3",
length(as790Struct.growth)), repelem("4", length(as791Struct.growth)), repelem("5",
length(as792Struct.growth)), repelem("6", length(as794Struct.growth)), repelem("7",
length(as798Struct.growth)), repelem("8", length(as799Struct.growth)), repelem("9",
length(as800Struct.growth)), repelem("10", length(as802Struct.growth)), repelem("11",
length(as803Struct.growth)), repelem("12", length(as804Struct.growth))];
[p,tbl,stats] = anova1(dists, grps)
multcompare(stats)
set(gca,'FontSize',28)
ylabel('Strain');
xlabel('Doubling time (hrs)');
```