Contents

```
clc;
close all;
clear all;
```

extract images from videos

```
cd('/data2/nacere/Processing/XG05/Time4/Time4_data/');
a=VideoReader('mig_slp001_lrgcha_outwalnot_gentle_insi_1layergravel_exp4_inlet_20230818_.mov');
for img=1:a.numFrames
    filename=strcat('frame', num2str(img),'.png');
    b=read(a,img);
    imwrite(b,filename);

end
```

```
Error using cd
Unable to change current folder to '/data2/nacere/Processing/XG05/Time4/Time4_data' (Name is nonexistent or not a folder).

Error in Step1_From_extractionofimages_to_binarizationofmiddlechannel (line 5)
cd('/data2/nacere/Processing/XG05/Time4/Time4_data/');
```

reduce image number for processing

select every 10th images for lowest xanthan gum videos (0 to 0.3%) select every 50th images for highest xanthan gum videos (0.4to 0.5%)

```
folder='/data2/nacere/Processing/XG04/Time5/Time5_data/';
%folder = '/data1/Nacere/imageprocess/ImagesXG03/Videos/';
newfolder = '/data2/nacere/Processing/XG04/Time5/Time5_every50th/';

if ~exist(newfolder, 'dir')
    mkdir(newfolder);
end
files = natsortfiles(dir(fullfile(folder, '*.png')));
% files = sort({files1.name});
disp(['Total files found: ', num2str(length(files))]);
for i = 1:50:length(files)
    sourcefile = fullfile(files(i).folder, files(i).name);
    % sourcefile = sort({files.name});
    destfile = fullfile(newfolder, files(i).name);
    disp(['Copying: ', sourcefile, ' to ', destfile]);
    copyfile(sourcefile, destfile);
    disp('Copy operation completed.');
end
```

load piv : from here all the sess.mat files were processed individually: For instance sess3.mat was uploaded and binarized

```
%then further processing were done to remove unwanted dots and keep the
%middle channel
load sess3.mat;
```

velocity of grains

```
Velocity_x=u_original{1};
Velocity_y=v_original{1};
```

Magnitude

```
velo_mag=sqrt(Velocity_x.^2+Velocity_y.^2);
threshold =0.3;
```

```
ks=find(velo_mag <0.5);
kh=find(velo_mag >= 0.5);
Binar=zeros(size(velo_mag));
Binar(ks)=0;
Binar(kh)=1;

X=x{1};
Y=y{1};
figure;
quiver(X,Y,Velocity_x,Velocity_y,10)
figure;
contourf(X,Y,(velo_mag));
```

ffilter based on velocity mag

```
threshold=0.3;
above_threshold=velo_mag>=threshold;
binary_mask=double(above_threshold);
X=x{1};
Y=y{1};
% plot
figure;
quiver(X(above_threshold),Y(above_threshold), Velocity_x(above_threshold), Velocity_y(above_threshold), 1);
figure;
contourf(X,Y, binary_mask, 1);
colormap([1 1 1; 0 0 0]);
```

```
bin_velomag=imbinarize(velo_mag,threshold);
figure;imshow(bin_velomag);
```

```
conn=conndef(2,'minimal');
erode= imerode(bin_velomag,strel('disk', 1));
cl=imclose(bin_velomag, strel('disk',1)); %% to close gasp between pixel very important
windowSize = 2;
kernel = ones(windowSize) / windowSize^ 2;
blurryImage = convn(single(cl), kernel, 'same');
binaryImage = blurryImage > 0.1; % Rethreshold
figure; imshow(binaryImage);
```

Morpho

```
bmorph=bwmorph(cl,'close');
figure; imshow(bmorph);
```

code to remove unconnected dots if any

```
numberToExtract = 2;
biggestBlob = ExtractNLargestBlobs(bmorph, numberToExtract);
subplot(2, 2, 4);
figure;
imshow(biggestBlob, []);
% Make the number positive again.  We don't need it negative for smallest extraction anymore.
numberToExtract = abs(numberToExtract);
```

```
% Function to return the specified number of largest or smallest blobs in a binary image.
% If numberToExtract > 0 it returns the numberToExtract largest blobs.
% If numberToExtract < 0 it returns the numberToExtract smallest blobs.
% Example: return a binary image with only the largest blob:
%   binaryImage = ExtractNLargestBlobs(binaryImage, 1)
% Example: return a binary image with the 3 smallest blobs:
%   binaryImage = ExtractNLargestBlobs(binaryImage, -3)
try
        % Get all the blob properties.  Can only pass in originalImage in version R2008a and later.
        [labeledImage, numberOfBlobs] = bwlabel(bmorph);
        blobMeasurements = regionprops(labeledImage, 'area');
    numberToExtract=1;
        % Get all the areas
        allAreas = [blobMeasurements.Area];
        if numberToExtract > 0
                % For positive numbers, sort in order of largest to smallest.
                % Sort them.
                [sortedAreas, sortIndexes] = sort(allAreas, 'descend');
        elseif numberToExtract < 0
                % For negative numbers, sort in order of smallest to largest.
                % Sort them.
                [sortedAreas, sortIndexes] = sort(allAreas, 'ascend');
                % Need to negate numberToExtract so we can use it in sortIndexes later.
                numberToExtract = -numberToExtract;
        else
```

```
                % numberToExtract = 0.  Shouldn't happen.  Return no blobs.
```

```
                bmorph1 = false(size(bmorph1));
                return;
```

```
        end
        % Extract the "numberToExtract" largest blob(a)s using ismember().
        biggestchannel = ismember(labeledImage, sortIndexes(1:numberToExtract));
        % Convert from integer labeled image into binary (logical) image.
        bmorph1 = biggestchannel > 0;
catch ME
        errorMessage = sprintf('Error in function ExtractNLargestBlobs().\n\nError Message:\n%s', ME.message);
        fprintf(1, '%s\n', errorMessage);
        uiwait(warndlg(errorMessage));
end
```

```
figure; imshow(biggestchannel);
```

sving the binary images

```
imwrite(biggestchannel,'binary3.png');
```

Published with MATLAB® R2023b