Instabilités de Saffman-Taylor en milieu granulaire : Matériels et méthodes

Jean-Baptiste Bouhiron, Jean-Baptiste Boutin et Angelo Couto

Promotion 135

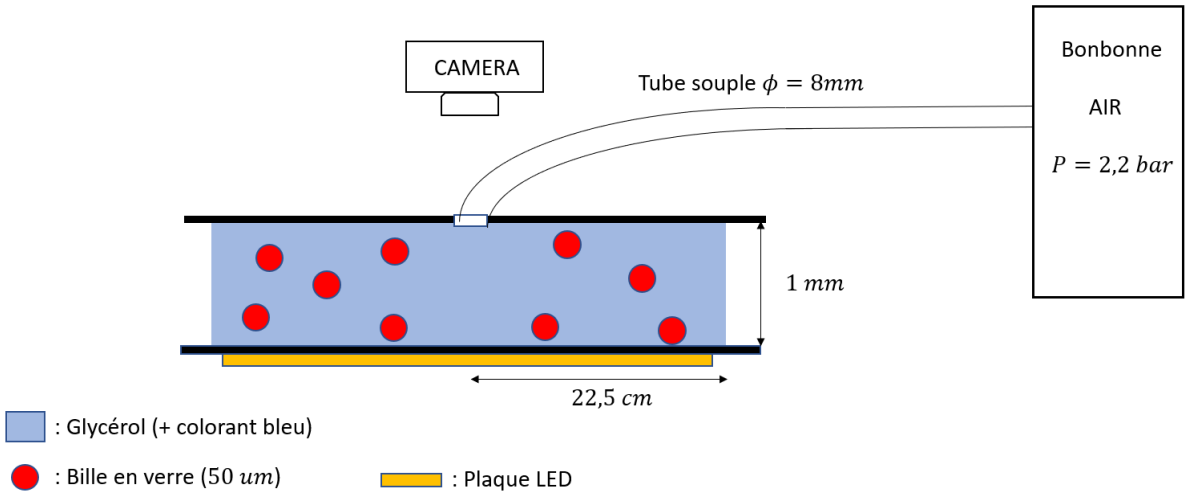
PSE, ESPCI Paris

Matériels

- glycérol
- billes de verre (d ~ 50 µm)
- colorant noir
- balance
- petites cales en plastique (épaisseur = 0,8 mm)
- cristallisoirs
- spatule
- 2 plaques de plexiglas (d = 45 cm ; e = 1 cm) dont une avec un trou en son centre (d_{trou} = 4 mm)
- bouteille d'air comprimé
- tuyau en plastique ($d_{extérieur}$ = 4 mm ; l = 2 m)
- caméra ultra rapide
- objectif
- plaque lumineuse
- alimentation continue
- niveau à bulle

- 3 supports élévateurs
- 4 pinces
- un ordinateur
- ImageJ, Matlab et PylonViewer

Schéma de l'expérience



Méthodes

Réglage du montage : La caméra est réglée de façon à ne filmer que la zone éclairée du montage. Elle enregistre 139 images.s^{-1} avec un temps d'exposition de 350 µs. La plaque lumineuse est alimentée par une alimentation continue (I = 0,89 A et U = 3,8 V). On positionne la première plaque de plexiglas horizontalement sur les supports élévateurs. On ajuste alors la hauteur des supports élévateurs pour régler l'horizontalité à l'aide du niveau à bulle. La pression à la sortie de la bouteille d'air comprimé est réglée à 2,2 bars. On place la caméra munie de son objectif à 1 m des plaques de plexiglas à l'aide d'une structure en métal.

Protocole : A l'aide de la balance, préparer plusieurs mélanges, glycérol/billes de verre, de fractions massiques en billes différentes où la masse de glycérol reste constante ($m_{glycérol}$ = 150 g ; x_{bille} = 0%, 5%, 16%, 30%, 45%, 65 %). Ajouter dans chaque préparation 6 gouttes de colorant noir. Verser la préparation entre les 2 plaques de plexiglas. Ajuster les pinces pour bloquer les 2 plaques et enfoncer le tuyau dans le trou de la plaque du dessus afin de relier la cellule de Hele-Shaw ainsi formée avec la bombonne d'air comprimé. Lancer l'acquisition de la caméra et ouvrir la bouteille 2 secondes. Arrêter l'acquisition. On recommence avec les autres préparations après avoir lavé les plaques.

Exploitation des images : On binarise toutes les images à l'aide de la fonction Brightness (Image → Adjust → Brigthness) en noir et blanc du logiciel Image J. A l'aide du programme 1(cf annexes), on vérifie que les expériences sont répétables en mesurant

la quantité d'air injectée au cours du temps (Aire de l'air). On vérifie que le débit d'air injecté est constant et identique pour chaque fraction en billes. Ensuite on utilise le programme 2 (cf annexe), sur la 12 ième image de chaque expérience, afin d'obtenir la distribution de la largeur de chacun des doigts du pattern. On calcule alors la largeur moyenne des doigts que l'on trace en fonction de la fraction massique en grain.

Annexes

Programme 1 :

```
[filename,pathname]=uigetfile('C:\Users\Jean-
Baptiste\Documents\MATLAB\PSE_PLS\phi30exp23\*','multiselect','on');
longueur=length(filename);
pas=1000/139; %donne le pas de temps, ie l'écart temporel entre deux
photos, en ms.
echelle = 0.1*246/1409;%1pixel=echelle cm
Liste_aires=ones(1,longueur);%tableau contenant les aires

for f=1:1:longueur
    number=filename{f};
    s=strcat(pathname,number);
    A=imread(s);
    Abin=A(:,:,1);
    c=0; %compteur
    [Nlmax,Ncmax]=size(A); % Donne la taille de l'image
    for i=1:1:Nlmax
        for j=1:1:Ncmax
           if (A(i,j)>0)
                   c=c+1;
           end
        end
    end
    Liste_aires(1,f)=c;
    c=0;

end
temps8=[0:1:longueur-1]*pas;%pour laxe des abscisses converti en temps (en
ms)
Liste_aires=Liste_aires.*(echelle^2); %pour convertir l'aire en cm^2
exp8=(Liste_aires);
% plot(temps,(Liste_aires),'b'); xlabel('temps [ms]'); ylabel('surface
[cm^2]'); grid on;grid minor;%première image en origine des temps
% title('aire occupée par les bras en fonction du temps')
```

Programme 2 :

```
%% Ouverture images

clear all
close all
clc

[filename,pathname]=uigetfile('C:\Users\Angelo\Documents\MATLAB\PSE_26_JANV
IER\expérience8\*','multiselect','on');

d = dir(pathname);
longueur_fichier = length(find([d.isdir]==0));
```

```
number0=filename{longueur_fichier};
s0=strcat(pathname,number0);
image0=imread(s0);                    % image à traiter
figure; colormap gray; axis image;
imagesc(image0);

number0=filename{1};
s0=strcat(pathname,number0);
image0=imread(s0);                    % image à traiter
figure; colormap gray; axis image;
imagesc(image0);

size_image = size(image0);
i_max = size_image(1); j_max = size_image(2);        % i_max, j_max =
hauteur, largeur de l'image en pixels

%% Paramètres %%

x_centre = 619;
y_centre = 672;                          % coordonnées du centre de la
figure
seuil = 52;                              % seuil
valeur_minimale = 4;                     % valeur minimale en pixel de la
largeur d'un bras
R_min = 43;
R_max = 500;
delta_R =1;
distance_pixel_reference = 1230;


facteur_echelle_spatiale = 252 / distance_pixel_reference;               %
coef conversion pixels --> mm
facteur_echelle_temporelle = 1000 / 125;                                 %
coef conversion numero image --> ms
N_R = floor((R_max-R_min)/delta_R)+1;    % balayage de la figure depuis
R_min jusqu'à R_max avec un incrément de delta_R.  N_R = nombre de rayons
(=nombre d'incréments)

%%  Calcul points

matrice_nombre_bras = zeros(N_R, longueur_fichier);
vecteur_aire = zeros(1, longueur_fichier);
cellule_longueurs_bras = cell(longueur_fichier);

[X_cellule,Y_cellule, N1] = determineCoordonneesPolaire(N_R,R_min, delta_R,
x_centre, y_centre);

%%   Calculs bras

pourcentage = 10;

for numero_image=1:longueur_fichier

    if (numero_image / longueur_fichier)*100 > pourcentage
        pourcentage
        pourcentage = pourcentage + 10;
```

```
    end

    number=filename{numero_image};
    s=strcat(pathname,number);
    image=imread(s);

    [test_seuil, aire] = determineTestSeuil (image, X_cellule, Y_cellule,
seuil, N_R, N1);
    nombre_bras = determineNombreBras(test_seuil, N_R, N1);
    [indices_debut,indices_fin] = determineBeginingEnd(nombre_bras,
test_seuil, N_R, N1);
    [longueurs_bras,nombre_bras_rectifie] =
determineLongueurBras(nombre_bras, indices_debut,indices_fin,
X_cellule,Y_cellule,valeur_minimale, N_R, R_min, delta_R, x_centre,
y_centre);

    matrice_nombre_bras (:,numero_image) = nombre_bras_rectifie;
    vecteur_aire(numero_image) = aire;
    cellule_longueurs_bras{numero_image} = longueurs_bras;

end

%%   Largeur bras 1 image

numero_image = 3;

longueurs_bras = cellule_longueurs_bras{numero_image};
longueurs_bras(longueurs_bras < valeur_minimale) = [];
longueurs_bras(longueurs_bras > 200) = [];
figure; a = histogram(longueurs_bras*facteur_echelle_spatiale, 50,
'Normalization', 'probability');xlabel('Largeur des bras
(mm)');ylabel('Proportion');title('Distribution de la largeur des bras ');
esperance = linspace(a.BinLimits(1), a.BinLimits(2), a.NumBins) *
a.Values';
hold on;plot([esperance esperance], [0 1.1*max(a.Values)],'LineWidth',2);



function [test_seuil, aire] = determineTestSeuil (image, X_cellule,
Y_cellule, seuil, N_R, N1)
    %Détermine la matrice des seuils
    test_seuil = cell(1, N_R);   % test_seuil{n_R} = liste de 0 et de 1
suivant que le niveau de gris de l'image aux points d'indices n1 est en
dessous ou au dessus du seuil
    aire = 0;   % aire en pixel

    for n_R = 1:N_R % itération sur les cercles
    test_seuil{n_R} = zeros(1, N1(n_R));  % initialisation
        for n1 = 1:N1(n_R)
            if image(Y_cellule{n_R}(n1), X_cellule{n_R}(n1)) > seuil  % = 1
si >seuil ,  =0 si <=seuil
                test_seuil{n_R}(n1) = 1;
                aire = aire + 1;
            end
        end
    end
end

function [longueurs_bras,nombre_bras_rectifie] =
determineLongueurBras(nombre_bras, indices_debut,indices_fin,
```

```
X_cellule,Y_cellule,valeur_minimale, N_R, R_min, delta_R, x_centre,
y_centre)
%Détermine la longueur des bras

    longueurs_bras = zeros(1, sum(nombre_bras));
    rectification = zeros(1, N_R);
    increment = 1;
    for n_R = 1:N_R
        for numero_bras = 1:nombre_bras(n_R)
            R = R_min + n_R * delta_R;
            xA = X_cellule{n_R}(indices_debut{n_R}(numero_bras)); yA =
Y_cellule{n_R}(indices_debut{n_R}(numero_bras));
            xB = X_cellule{n_R}(indices_fin{n_R}(numero_bras)); yB =
Y_cellule{n_R}(indices_fin{n_R}(numero_bras));
            longueur = ((xA-xB)^2 + (yA-yB)^2)^0.5;
            if longueur < valeur_minimale
                rectification(n_R) = rectification(n_R) +1;
            end
            longueurs_bras(increment) = longueur;
            increment = increment +1;
        end
    end
    nombre_bras_rectifie = nombre_bras - rectification;
end


function [X_cellule,Y_cellule, N1] = determineCoordonneesPolaire(N_R,R_min,
delta_R, x_centre, y_centre)
    %Détermine la liste des coordonnées polaires

    X0_cellule = cell(1, N_R);Y0_cellule = cell(1, N_R);   % Initialisation
de la liste des listes des positions (i,j) (ou (y, x)) pour un rayon R
    X_cellule = cell(1, N_R);Y_cellule = cell(1, N_R);     % idem
    N1 = zeros(1, N_R);                                    % initialisation
du nombre de pixels parcourus pour chaque rayon

    for n_R = 1:N_R  % itération sur les cercles numéro n_R
        R = R_min + n_R * delta_R;     % rayon du cercle n_R
        delta_theta = 1 / R;  N = floor(2*pi / delta_theta);   %calcul de
l'incrément de l'angle theta pour que tous les pixels appartenant au cercle
n_R soient pris en compte. N = nombre de valeur de l'angle theta

        X0_cellule{n_R} = zeros(1,N); Y0_cellule{n_R} = zeros(1,N);
        X0_cellule{n_R}(1) = x_centre + R; Y0_cellule{n_R}(1) = y_centre;
% initialisation de la liste des coordionnées x, y du cercle

        for n = 2:N
            X0_cellule{n_R}(n) = floor(x_centre + R * cos(n *
delta_theta)); Y0_cellule{n_R}(n) = floor(y_centre + R * sin(n *
delta_theta));  %calcul des coordonnées x, y du cercle n_R en fonction de
l'angle theta
        end



        X_cellule{n_R} = zeros(1, N); Y_cellule{n_R} = zeros(1, N);
        X_cellule{n_R}(1) = x_centre + R; Y_cellule{n_R}(1) = y_centre;  %
initialisation de la liste des coordionnées x, y du cercle

        n1 = 2;  % indice de X_cellule{n_R} et Y_cellule{n_R}
```

```
        for n = 2:N  % il se peut que la liste des pixels [X0_cellule{n_R},
Y0_cellule{n_R}] contiennent des doublons, la liste
[X_cellule{n_R},Y_cellule{n_R}] est la liste sans les doublons
            if X0_cellule{n_R}(n) ~= X0_cellule{n_R}(n-1) |
Y0_cellule{n_R}(n) ~= Y0_cellule{n_R}(n-1)  % si deux points consécutifs
sont différents, alors on l'ajoute à X, Y
                X_cellule{n_R}(n1) = X0_cellule{n_R}(n) ;
Y_cellule{n_R}(n1) = Y0_cellule{n_R}(n);
                n1 = n1 + 1;
            end
        end

        X_cellule{n_R}(X_cellule{n_R}==0) = [];
Y_cellule{n_R}(Y_cellule{n_R}==0)= []; %supprime les élements nuls
        N1(n_R) = length(X_cellule{n_R});  % nombre de pixels parcourus par
le cercle n_R de rayon R
    end
end
function [indices_debut,indices_fin] = determineBeginingEnd(nombre_bras,
test_seuil, N_R, N1)
%Détermine le début et la fin des bras
    indices_debut = cell(1, N_R);indices_fin= cell(1, N_R);  % repèrtorie
l'indice des pixels correspondant au début des bras et à la fin des bras

    for n_R = 1:N_R
        indices_debut{n_R} = zeros(1, nombre_bras(n_R)); indices_fin{n_R} =
zeros(1, nombre_bras(n_R));
        ind_d = 1; ind_f = 1;
        if test_seuil{n_R}(1) == 1
            for n1 = 1:N1(n_R)-1
                if test_seuil{n_R}(n1)==1 && test_seuil{n_R}(n1+1)==0
                    indices_fin{n_R}(ind_f) = n1;
                        ind_f = ind_f + 1;
                end
                if test_seuil{n_R}(n1)==0 && test_seuil{n_R}(n1+1)==1
                    indices_debut{n_R}(ind_d) = n1+1;
                        ind_d = ind_d + 1;
                end
            end
            if test_seuil{n_R}(N1(n_R)) == 0
                indices_debut{n_R}(nombre_bras(n_R)) = 1;
            end
            indices_fin{n_R} = circshift(indices_fin{n_R}, -1);
        end
        if test_seuil{n_R}(1) == 0
            for n1 = 2:N1(n_R)
                if test_seuil{n_R}(n1)==1 && test_seuil{n_R}(n1-1)==0
                    indices_debut{n_R}(ind_d) = n1;
                        ind_d = ind_d + 1;
                end
                if test_seuil{n_R}(n1)==0 && test_seuil{n_R}(n1-1)==1
                    indices_fin{n_R}(ind_f) = n1-1;
                        ind_f = ind_f + 1;
                end
            end
            if test_seuil{n_R}(N1(n_R))==1
                indices_fin{n_R}(nombre_bras(n_R))= N1(n_R);
            end
        end
        %test%
        %if ind_f~=nombre_bras(n_R) | ind_d~=nombre_bras(n_R)
```

```
        %    [n_R ind_d ind_f nombre_bras(n_R)]
        %end
        for indice = indices_debut{n_R}
            if indice == 0
                [1 n_R]
            end
        end
        for indice = indices_fin{n_R}
            if indice == 0
                [2 n_R]
            end
        end
    end
end
```