Leidenfrost effect on a liquid surface : Materials and Methods

Delory Alexandre[1], Ollier Valentin[1], and Ruscher Mael[1]

[1]PSE, ESPCI Paris
{alexandre.delory,valentin.ollier,mael.ruscher}@espci.fr

May 2018

Abstract

This paper describes the methods and materials used for the study of the Leidenfrost effect on liquid surfaces. The method for the deposition and formation of the Leidenfrost drop is explained as well as the method of images analyzing with MatLab.

Keywords : Leidenfrost effect, isopropanol drop, oil pool, MatLab coding

Materials

- ☐ Crystallizer (bath) ϕ20cm
- ☐ Plexiglass with squared Hole (bath) $a = 4.8$cm
- ☐ Crystallizer (pool) ϕ3cm
- ☐ Heating Plate, associated with a PID controller
- ☐ Thermocouple
- ☐ P200 syringe
- ☐ Peanut oil
- ☐ isopropanol
- ☐ Camera
- ☐ Semi-refracting Blade
- ☐ metallic structure in order to hold every equipment and be in the dark

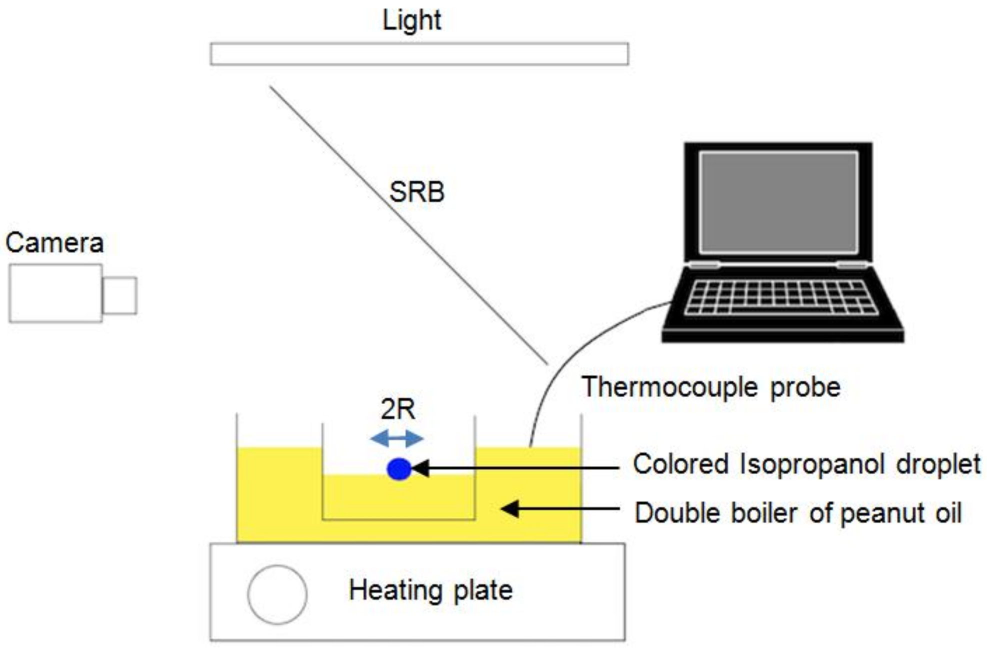

Figure 1: Schema illustrating the setup

Methods

The experimental setup is the following : an isopropanol drop of a radius R is released at $\approx$ 1cm above a hot pool of peanut oil. Using a P200 syringe, the droplet is gently deposited on the hot pool. This one is maintained at a temperature T_p using another peanut oil bath to homogenize the temperature around the container, a heating plate and a PID controller to stabilize the temperature. An additional thermocouple is plunged in the hot peanut pool to precisely note the pool temperature T_p. However, the temperature remains uncertain to $\pm 0.5\,°$C. Denoting $T_{sat} = 82.5\,°$C the boiling point of isopropanol at atmospheric pressure, the difference between the pool temperature and the boiling point is denoted $\Delta T = T_p - T_{sat}$ and will remain positive throughout this paper. Our experiments were conducted with isopropanol initially at ambient temperature, and droplets released at this temperature. Peanut oil allows heating the pool up to $150\,°$C and is practically nonvolatile.
Thanks to a camera and a semi-reflecting blade we were able to record the droplet from above. In fact, the semi-reflecting blade is used to enlighten from above and an horizontally oriented camera that records as shown in the following illustration and which is fixed to the metallic structure. The movies have to be taken in the dark with an homogenous illuminance from above in order to avoid shadows that prevent the code from detecting the droplet. In deed, if the shadows were too big the code will confuse the droplet and its shadow, leading to a false estimation of the radius and the position. At the beginning we tried to process the images thanks to imageJ, but it was too long. Therefore we designed matlab codes to process the sequences of images. Those are available in Annex. Thus, we were able to measure the Radius R, the position (X,Y) of the droplet in the bath and its speed (V_x,V_y). An other code was used to convert the data and represents many useful curve.

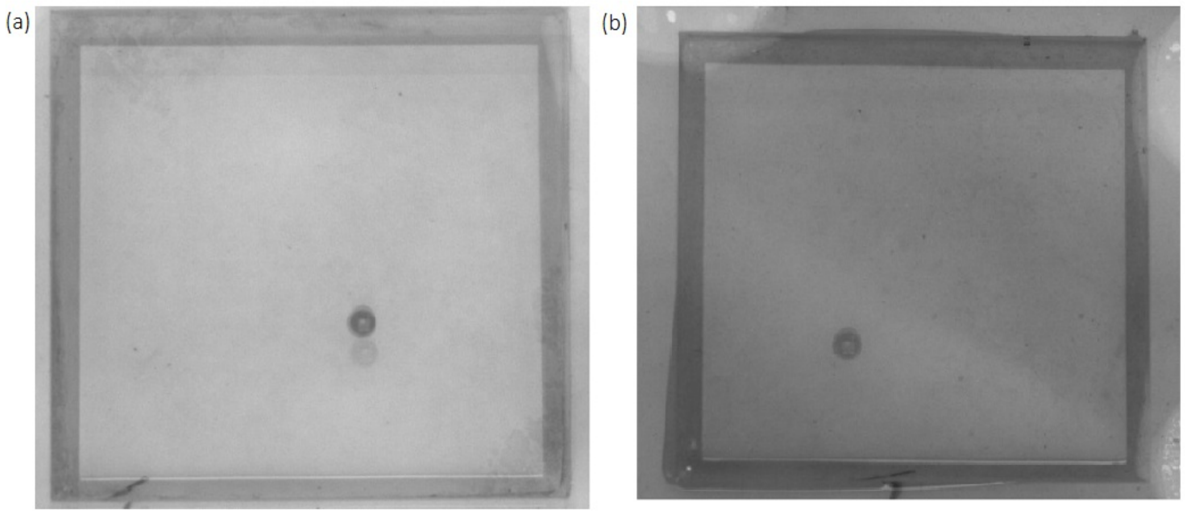

Figure 2: Exemple of a droplet with the wrong lighting because of the shadow (a) and with the right enlightening (in a squared bath made with Plexiglass)

Experiment 1

The first experiment was carried out in circular container (crystalliser $\phi = 3$cm). Its main goal was to characterize the evolution of the Radius as a function of time. We used the methods and the codes which are described above (code 1 3) to characterize our droplets. As we discovered the existence of two regime we decided to study both.

Experiment 2

We tried in this experiment to characterize the droplet when it moves. The temperature of the bath has to be above the Leidenfrost temperature (Leidenfrost effect is possible) and below 111 °C. In order to understand better how the droplet behave on the bath, we change our pool to squared one made out of Plexiglass. In deed, we crafted a 4.8cm square into a slab of Plexiglass in order to understand how the droplet will bounce against the wall. We used the same imaging methods as before and the program was modified too (code 2 4).

Experiment 3

We characterized the droplets this time when it does not move. That is to say for a temperature above 111 °C. Once again the droplets were created the same way and we used the program 2 3 to collect and process our data.

Appendices

Code 1: Processing the images from a round bath

```
close all

cen=[0 0] ;
centers=[0 0 0] ;
radii=[0 0] ;
Rmin=10;
Rmax=30;
chemin='C:\Users\PSE\Desktop\Delory_Rusher_Ollier\2A\seance_4_3_novembre\112_gif\
Basler_acA1300-60gm__21537217__20171103_160034493_';
diametre=24; %% en mm
coeff=24/500;

image=imread([chemin num2str(0) num2str(0) num2str(0) num2str(1) '.tiff']);
[image1,coordonne]=imcrop(image);
ymin = floor(coordonne(1)) ;
xmin = floor(coordonne(2)) ;
ymax=ymin+floor(coordonne(3));
xmax=xmin+floor(coordonne(4));
invert_image=255-(image(xmin:xmax , ymin:ymax));
[center, radi] = imfindcircles(invert_image,[Rmin 15]) ;
cen = [cen ; center] ;
if length(radi)>0
    for j=1:length(radi)
        radii = [radii ; [radi(j) 1]] ;
        centers = [centers ; [center(j,1) center(j,2) 1]] ;
    end
end

for i=2:9
    image=imread([chemin num2str(0) num2str(0) num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
%     histeq(invert_image) ;
%     colormap gray
%     axis equal
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
for i=10:99
    image=imread([chemin num2str(0) num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
for i=100:999
    image=imread([chemin num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
```

Code 2: Processing the images from a squared bath

```
close all

cen=[0 0] ;
centers=[0 0 0] ;
radii=[0 0] ;
Rmin=10;
Rmax=30;
chemin='C:\Users\PSE\Desktop\Delory_Rusher_Ollier\2A\seance_26_janvier_pc_focus_20_fps\112_2\
Basler_acA1300-60gm__21537217__20180126_155853101_';
cote_carre=48; %% en mm
coeff=48/544;
deltaT=0.05; % fps 20 image par seconde

image=imread([chemin num2str(0) num2str(0) num2str(0) num2str(1) '.tiff']);
[image1,coordonne]=imcrop(image);
ymin = floor(coordonne(1)) ;
xmin = floor(coordonne(2)) ;
ymax=ymin+floor(coordonne(3));
xmax=xmin+floor(coordonne(4));


invert_image=255-(image(xmin:xmax , ymin:ymax));
[center, radi] = imfindcircles(invert_image,[Rmin 15]) ;
cen = [cen ; center] ;
if length(radi)>0
    for j=1:length(radi)
        radii = [radii ; [radi(j) 1]] ;
        centers = [centers ; [center(j,1) center(j,2) 1]] ;
    end
    disp(['le numero de la photo est: ' num2str(i)])
else
    p=0;
    while p==0
        i=2;
        image=imread([chemin num2str(0) num2str(0) num2str(0) num2str(i) '.tiff']);
        invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
        [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
        if length(radi)>0
            p=1;
            image1=image(xmin:xmax , ymin:ymax);
        else
            i=i+1;
        end
    end
    disp(['le numero de la photo est: ' num2str(i)])
end

close all

for i=2:9
    image=imread([chemin num2str(0) num2str(0) num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
for i=10:99
    image=imread([chemin num2str(0) num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
for i=100:999
    image=imread([chemin num2str(0) num2str(i) '.tiff']);
    invert_image=255-(image(xmin:xmax , ymin:ymax)) ;
    [center, radi] = imfindcircles(invert_image,[Rmin Rmax]) ;
    cen = [cen ; center] ;
    if length(radi)>0
        for j=1:length(radi)
            radii = [radii ; [radi(j) i]] ;
            centers = [centers ; [center(j,1) center(j,2) i]] ;
        end
    end
    i
end
```

Code 3: Representation of the Radius as a function of time

```
close all;

radii_mm=zeros(length(radii),2);
    for i=1:length(radii)
    radii_mm(i,1)=radii(i,1)*coeff;
    radii_mm(i,2)=radii(i,2)/10;
    end
    figure; plot(radii_mm(2:end-2,2),radii_mm(2:end-2,1))
        title('Evolution du rayon de la goutte en fonction du temps');
        xlabel('Temps (s)');
        ylabel('Rayon (mm)');
```

Code 4: Representation of the data on 4 different curves

- the first curve is the trajectory
- the second curve is the kinetic energy as a function of time
- the third curve is the radius as a function of time
- the fourth curve is the speed as a function of the radius

```
close all;
clear Vx Vy V X Y t;
figure

% definition parametre de conversion
coupe=65;
coeff=48/544;

% donnees a supprimer a la fin de la liste; l correspond au numero de la
% ligne a partir de laquelle on commence la suppression
% mettre l=-1 si aucune donnee a supprimer
l=351;

% creation vecteur vitesse et position pour la trajectoire
Vx=[];
Vy=[];
V=[];
X=[];
```

```
Y=[];
t=[0];
if centers(1,1)==0
    centers(1,:)=[];
    radii(1,:)=[];
end
taille=length(centers(:,1));

radii_mm122_2_carre=zeros(length(radii),2);
    for i=1:length(radii)
    radii_mm122_2_carre(i,1)=radii(i,1)*coeff;
    radii_mm122_2_carre(i,2)=radii(i,2)*deltaT;
    end

for i=1:taille-1
    if centers(i+1,3)-centers(i,3)==0
        i=1+i;
        temps=(centers(i+1,3)-centers(i,3))*deltaT;
    else
        temps=(centers(i+1,3)-centers(i,3))*deltaT;
    end
    temps=(centers(i+1,3)-centers(i,3))*deltaT;
    x2=centers(i+1,1);
    x1=centers(i,1);
    y2=centers(i+1,2);
    y1=centers(i,2);
    vx=(x2-x1)/temps;
    vy=(y2-y1)/temps;
    temporaire=temps+t(end);
    t=[t ; temporaire];
    v=sqrt(vx^2+vy^2);
    X=[X;x1];
    Y=[Y;y1];
    Vx=[Vx ;vx];
    Vy=[Vy ;vy];
    V=[V;v];

end
Y=ymax-Y;
% conversion
X=X*coeff;
Y=Y*coeff;
V=V*coeff;
Vx=Vx*coeff;
Vy=Vy*coeff;

% trace du graphique

subplot(2,2,1)

imagesc(image1)
colormap gray
```

```
axis equal
axis tight
X=linspace(0,length(image1(:,1))*48/544,length(image1(:,1)));
Y=linspace(0,length(image1(1,:))*48/544,length(image1(1,:)));
xlabel('X en mm');
ylabel('Y en mm');
title('Trajectoire de la goutte et vecteurs vitesse');

hold on
if centers(1,1)==0
    centers(1,:)=[];
end

plot(centers(2:end-coupe,1),centers(2:end-coupe,2))

subplot(2,2,2)
plot(t(1:end-1),V)
xlabel('temps (en s)');
ylabel('norme vitesse mm/s');
title('Evolution de la norme de la vitesse en fonction du temps');


if l==-1
else
    for i=l:length(radii_mm122_2_carre(:,1))
        radii_mm122_2_carre(l,:)=[];
        V(l-1,:)=[];
    end
end


subplot(2,2,3)
plot(radii_mm122_2_carre(2:end-2,2),smooth(radii_mm122_2_carre(2:end-2,1)))
title('Evolution du rayon de la goutte en fonction du temps');
xlabel('Temps (s)');
ylabel('Rayon (mm)');

subplot(2,2,4)
plot(radii_mm122_2_carre(1:end-1,1),smooth(V),'.')
title('Evolution de la vitesse de la goutte en fonction de son rayon');
xlabel('Rayon (mm)');
ylabel('vitesse (mm/s)');

clear y1 y2 x1 x2 i j l v vx vy  taille temporaire temps
```