LIF data evaluation – image processing algorithms

M. Isoz

Institute of Chemical Technology, Prague

1 Introduction

Measurements of the liquid films and related forms of flow are often performed via an optical experimental technique[1, 2]. One of such methods is so called Light Induced Fluorescence (LIF).

It is based on the principle of adding a marker to the measured liquid, illumination of the liquid by a monochromatic light and measurement of the intensities of light emitted by the marked liquid.

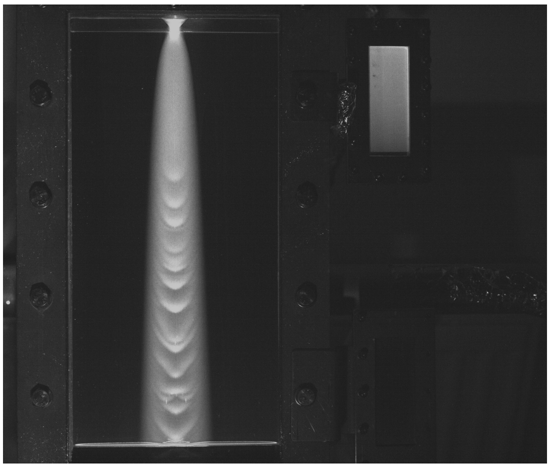

Figure 1: Example of experimental data obtained during LIF based measurement of gas–liquid interface of rivulet falling down on an inclined plate

Example of LIF based experimental method data output can be seen in Figure 1. Crucial parts of image are the inclined plate on which the studied rivulet is positioned and the calibration cell in upper right corner of the image. Calibration cell serves as a scale for conversion of measured light intensities in local film thicknesses.

For automatization of the data evaluation process, locating these two objects in images obtained during experiments is of key importance.

Moreover, as the measurements are quite easy to perform and quick, usually more than 40 images with the same position of these two elements were available. This fact can be used to refine the found coordinates through simple statistics.

Each experimental image corresponds to a matrix A of type (m, n), where m and n are the vertical and horizontal resolutions of the image. Elements of matrix A, (a_{ij}) are the pixels of processed image, $(a_{ij}) \in \langle 0; 1 \rangle$. Case of $a_{ij} = 0$ corresponds to a black pixel and $a_{ij} = 1$ to the white one.

Both described algorithms work with black and white images. The transformation of grayscale image to black and white is done through comparison of matrix elements to a preset threshold.

Result of this transformation is matrix $\tilde{A}$ with

$$\tilde{a}_{ij} = \begin{cases} 1 & \text{if} \quad a_{ij} < \text{threshold} \\ 0 & \text{if} \quad a_{ij} \geq \text{threshold}\,. \end{cases} , \quad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{1}$$

2 Calibration cell finding algorithm

The calibration cell is a very distinct object. Hence the algorithm does not have to be very complex. Besides, as the program was implemented in MATLAB, most of the steps have been performed via functions available in Image Processing Toolbox of this environment.

However, the most crucial step of the algorithm, selection of the calibration cell from all the candidate elements that passed through basic size based filtering, had to be developped. The proposed selection algorithm is based on comparing values of custom objective function.

From Figure 1, it is clear that the calibration cell is almost perfectly rectangular object. Additionally, it is always placed vertically. So the custom objective function penalizes the elements for not being rectangular and for not having edges parallel to image borders. This penalization is done through sum of two almost independent terms.

The first term of objective function penalizes checked object for not being rectangular and for not being oriented in the above described way.

Let us denote the term as $\Delta_R A$ and define it by relation,

$$\Delta_R A = \frac{\delta x\, \delta y - \int_S dS}{\delta x\, \delta y}\,, \tag{2}$$

where $\int_S dS$ stands for actual area of the element calculated directly from the number of pixels of which it consists. Other terms of the equation, δx and δy are the maximal distances between pixels in horizontal (x) and vertical (y) direction, respectively.

The product $\delta x\, \delta y$ always stands for the maximal possible area of tested element. Only for exactly vertically or horizontally placed rectangular elements, $\delta x\, \delta y = \int_S dS$, hence dividing their difference by $\delta x\, \delta y$ ensures

$$\Delta_R A \in \langle 0, 1)\,. \tag{3}$$

Other custom objective function term is penalizing the object for not being rectangular. It is based on the sum of dot products of the direction vectors of elements sides. Main idea is that the tangent vectors defined in the clockwise and counterclockwise directions in the top left and bottom right corner of the tested element should be perpendicular to each other. The second term of the objective function is denoted by DP and defined as follows,

$$DP = \sum_{i=1}^{2} \frac{\vec{u}_i}{\|\vec{u}_i\|} \cdot \frac{\vec{v}_i}{\|\vec{v}_i\|}\,, \tag{4}$$

where $\vec{u}_i$, $i = 1, 2$ are the tangent vectors defined in the clockwise direction in opposite corners and $\vec{v}_i$, $i = 1, 2$ are the vectors defined in the counterclockwise direction.

Value of the objective function for the i-th tested element is then calculated as sum of the above defined terms,

$$S_i = \Delta_R A_i + DP_i\,. \tag{5}$$

Algorithm can malfunction for the case of multiple distinct, roughly rectangular elements present in the image. However, testing proved it to be very dependable.

3 Plate finding algorithm

The second crucial element on the images from experiments is the inclined plate with measured rivulet itself. As it can be seen in Figure 1, the plate edges are much less distinct than the calibration cell. This resulted in need of much more sophisticated and computational time consuming algorithm.

First, a preprocessing had to be done, where the approximate position of the plate had to be specified manually, the rivulet itself was removed from the selected area and the contrasts in the resulting image were enhanced. The result was a matrix B, submatrix of A containing only the plate itself. Image represented by B was transformed in black and white using the relation (1). Finally the Hough transform was used to detect the line segments in $\tilde{B}$[3, 4].

The result of previously described procedure were line segments represented by coordinates of their starting and ending points. As it can be deduced from the Figure 1, only the vertical and horizontal lines near the borders of manually selected area were relevant for the plate position estimation.

All the other found line segments had to be excluded. Furthermore, the remaining lines had to be sorted with respect to the plate edge they were adjacent to. At last, the plate edges positions were estimated from the weighted mean of relevant line segments coordinates. Weights were based on the line segment length, with longer segments taken as more important.

The process is depicted in algorithm based on pseudo MATLAB syntax below.

```
% constants
Dx, Dy % maximal non−horizontality (non−verticality) of kept lines
Eh, Ev % maximal tolerated distance from the selection edges
Sh, Sv % horizontal and vertical sizes of selected area
for i = 1:number of found lines
% prepare the statements for processing the found segments
ishorizontal= abs(xStart−xEnd) < Dx; isvertical   = abs(yStart−yEnd) < Dy;
isleft       = abs(xStart−0)    < Eh; isright      = abs(xStart−Sh)   < Eh;
istop        = abs(yStart−0)    < Ev; isbottom     = abs(yStart−Sv)   < Ev;
% if the line is not vertical or horizontal, discard it
if ¬isvertical && ¬ishorizontal, discard tested segment; break; end
% sort lines with respect to selected area edges they are adjacent to
% discard the rest (lines in center of the selected area)
if isvertical && (isleft || isright)
        Vweight(i)     = abs(yStart−yEnd)/Sv; % calculate the weight of segment
    if isleft
        propLeft(i)    = mean([xStart xEnd]); % save its horizontal position
    else
        propLeft(i)    = mean([xStart xEnd]);
    end
elseif ishorizontal && (istop || isbottom)
    Hweight(i)         = abs(xStart−xEnd)/Sh; % calculate the weight of segment
    if istop
        propTop(i)     = mean([yStart yEnd]); % save its vertical position
    else
        propBottom(i) = mean([yStart yEnd]);
    end
else, discard tested segment;break;
end;end
```

After all the lines were tested, the plate edges position was calculated from proposed coordinates and their weights.

4 Statistical processing of algorithms results

Result of the above described algorithms is a matrix of found elements coordinates on experimental images, C. Each row of C corresponds to one processed image and each column to one found coordinate. As it was mentioned before, there are usually more than 40 images with the same experimental apparatus set up. This fact can be used to refine the found coordinates and to compensate for possible algorithms malfunctions.

Final position of the looked up elements is calculated as mean value of each column of C with previously excluded outliers.

Outliers exclusion is based on the data kurtosis[5]. From each column of matrix C are left out all the values not satisfying the equation

$$|(c_i)_j - \mu_j| \geq \alpha_j \sigma_j, \quad i = 1, \ldots, \text{number of images}\,. \tag{6}$$

In (6), μ_j is the mean value and σ_j is the standard deviation of the j-th column of C and the coefficient α_j is calculated from the columns kurtosis by the formula

$$\alpha_j = \frac{7}{\mathrm{Kurt}((c_i)_j)}, \quad i = 1, \ldots, \text{number of images}\,. \tag{7}$$

For the case of standard distribution, the advised value of numerator in (7) is 9[3]. However, as the distrubution of found coordinate should be closer to δ-function (the experimental set up was not tempered with), value 7 was chosen and proven by testing as more appropritate.

5 Conclusion

Modern, LIF based measurements are very fast and with improving quality of digital capturing devices also accurate experimental techniques. Unfortunately, the obtained data are only as good as it is the image processing method used for their evaluation. With the above explained algorithms, it is possible to automatically and precisely locate the most important elements on experimental images and improve the quality of measured data.

6 References

[1] S. V. Alekseenko, V. A. Antipin, A. V. Bobylev, D. M. Markovich: *Application of PIV to velocity measurements in a liquid film flowing down an inclined cylinder*, In: Exp. Fluids, 2008, *43* pp. 197 – 207.

[2] T. Hagemeier, M. Hartmann, M. Kühle, D. Thévenin, K. Zähringer: *Experimental characterization of thin films, droplets and rivulets using LED fluorescence*, In: Exp. Fluids, 2011, *52*, pp. 361–374.

[3] Mathworks, Inc. MATLAB Documentation. http://www.mathworks.com/help/ (accessed Dec 15, 2013).

[4] R. O. Duda, P. E. Hart: *Use of the Hough transformation to detect lines and curves in pictures*, In: Comm. ACM, 1971, *15*, pp. 11–15.

[5] J. R. M. Hosking: *Moments or L-Moments? An Example Comparing Two Measures of Distributional Shape*, In: The American Statistician, 1992, *46* (3), pp. 186–189.