FORTH-ICS / TR-417 May 2011

Capillaroscope †

Polykarpos Karamaounas and Xenophon Zabulis

Polykarpos Karamaounas and Xenophon Zabulis

Institute of Computer Science
Foundation for Research and Technology — Hellas (FORTH)
N. Plastira 100, Vassilika Vouton
Heraklion, Crete, 700 13 Greece

Web: `http://www.ics.forth.gr/`
E-mail: {karam | zabulis}@ics.forth.gr
Tel: +30 2810 391600, Fax: +30 2810 391601

Technical Report FORTH-ICS / TR-417— May 2011

Abstract

A software application is presented, for the detection and measurement of capillaries in images of capillaroscopy. This software is comprised by a measurement module and a Graphical User Interface module. The measurement module segments capillaries in the input image. The user interface provides methods for the automatic and interactive measurement of capillaries at regions of interest. In addition, it provides methods for editing the obtained capillary representation, in order to recover from segmentation errors. These modules are integrated into a medical application, which increases the automation and detail of the measurement process and supports diagnosis.

1 Introduction

Capillaroscopy is a non-invasive imaging technique that is used for in vivo assessment of the microcirculation. In clinical practice, the skin capillaries are generally observed through an incident light microscope. Morphologic evaluation of skin capillaries is generally performed at the nail fold because that area is easily accessible for examination and because the major axis of the capillaries is parallel to the skin surface, whereas in other areas it appears in a perpendicular status. The introduction of the videomicroscopy equipment, with optical contact probes, gave a significant boost in the study of microcirculation.

Capillaroscope is a piece of software for the measurement of capillaries in images of human nails. The software increases the automation of the data extraction process and measurement processing, by automatically detecting capillaries, and measuring their number throughout the image extent. Furthermore, *Capillaroscope* contains graphically interactive functionalities for specifying regions of interest, in order to automate targeted measurements, such as:

- editing of measurements and restriction of measurements in image regions of interest.
- export and saving measurements to files.

while, at the same time, provides an ergonomic graphical user interface. Capillaroscope also provides functionalities for editing and exporting results. Fig. 1 provides a brief overview of system functionalities.

2 Image processing

The input images are acquired by the VideoCap [1] imaging apparatus. and were of 640x480 pixel resolution. The output result is the detection, enumeration, shape and size estimation of the imaged capillaries.

In Fig 2, shown is a typical image from the dataset. The capillaries appear in the images with a weak contrast, while the occurrence of bubbles occludes other capillaries. A blob detection approach is employed to detect the imaged capillaries, but which however produces spurious detections of capillaries near the bubbles, as it detects blobs associated with them. Thereby, an auxiliary bubble detection method is employed to exclude image regions where bubbles appear, from being searched for capillaries. Overall the images exhibited significant noise artifacts while also their contrast was rather low.

2.1 Blob detection

The blob detection method operates on a monochromatic image. The “green” image channel is selected as input to it, as in the given datasets it exhibited the greatest

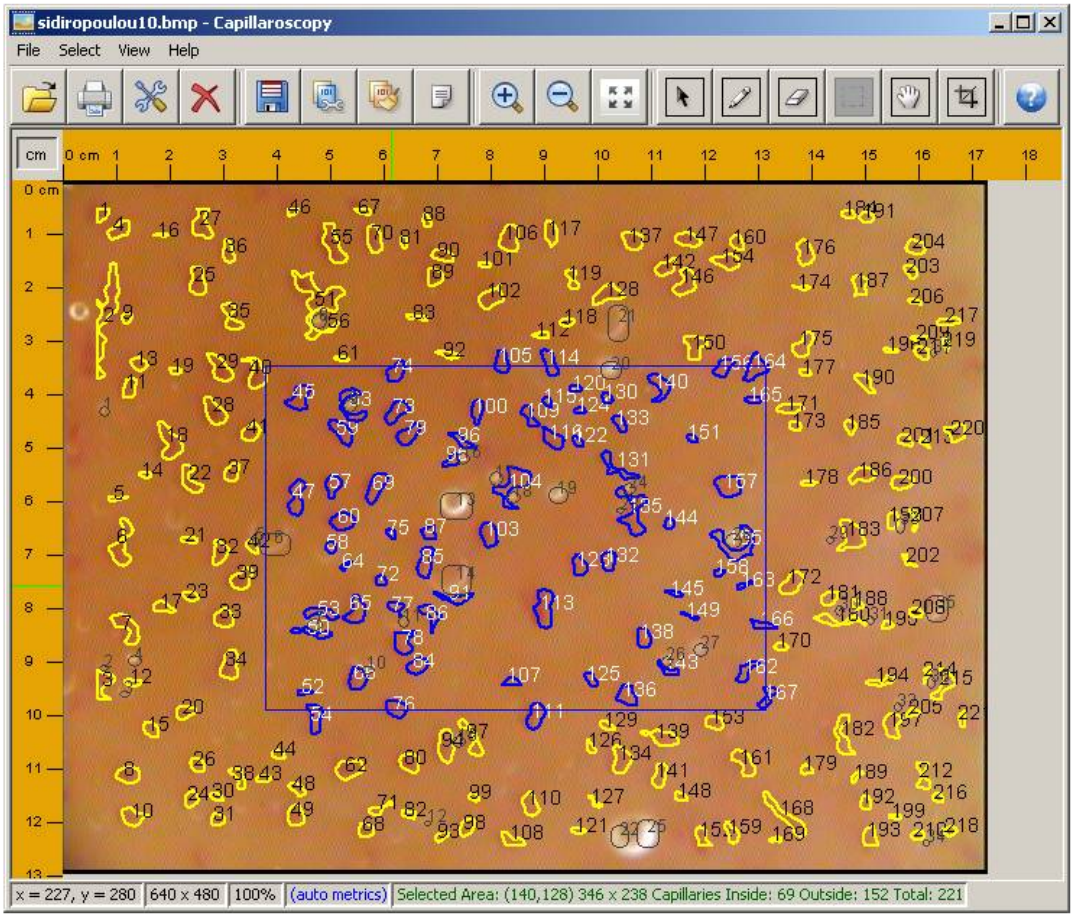


Figure 1: Main user interface UI panel. An image has been loaded and analyzed. The user has defined an area of interest (blue rectangle area).The automatically detected capillaries are marked with yellow (in case of being outside the area of interest) and with blue (in case of being inside the area of interest). The automatically detected bubbles are marked with grey. On the upper part of the figure shown is a toolbar that provides the functionalities in a user-friendly manner. The system incorporates image analysis methods within the UI in order to provide high-level image analysis tools, in a point and click fashion.

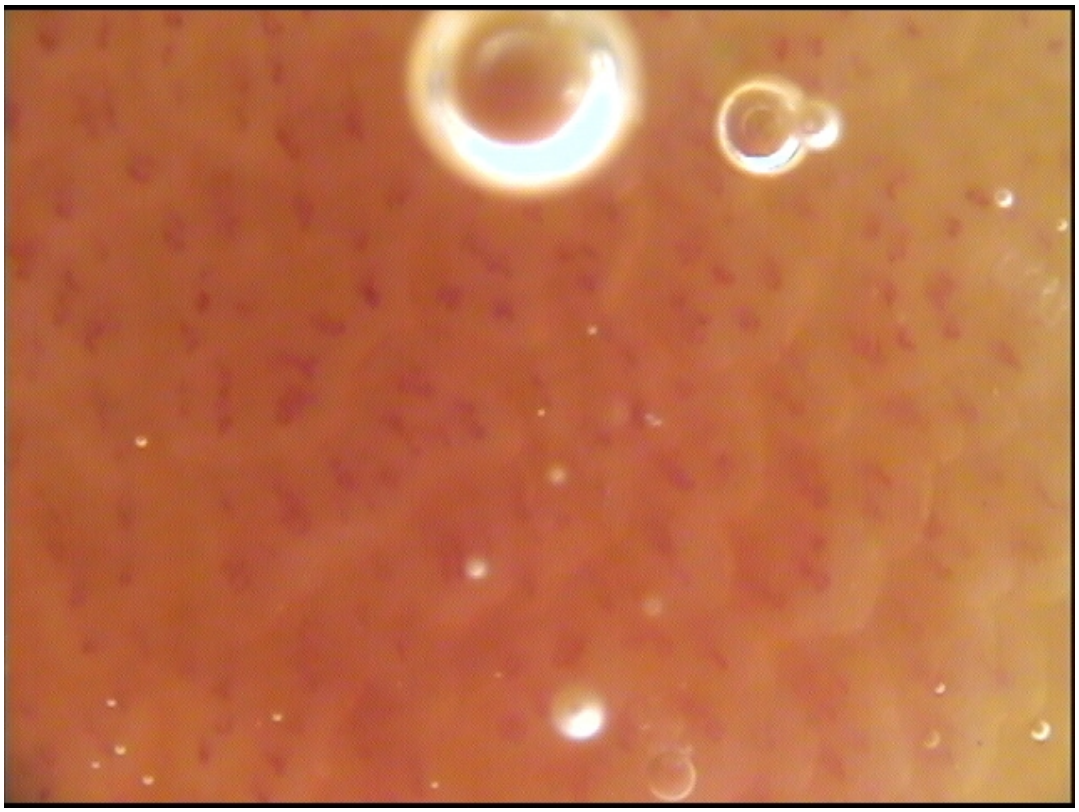


Figure 2: Examination image. Along with capillaries, bubbles also appear in the image, potentially occluding part of the capillaries intended to be photographed.

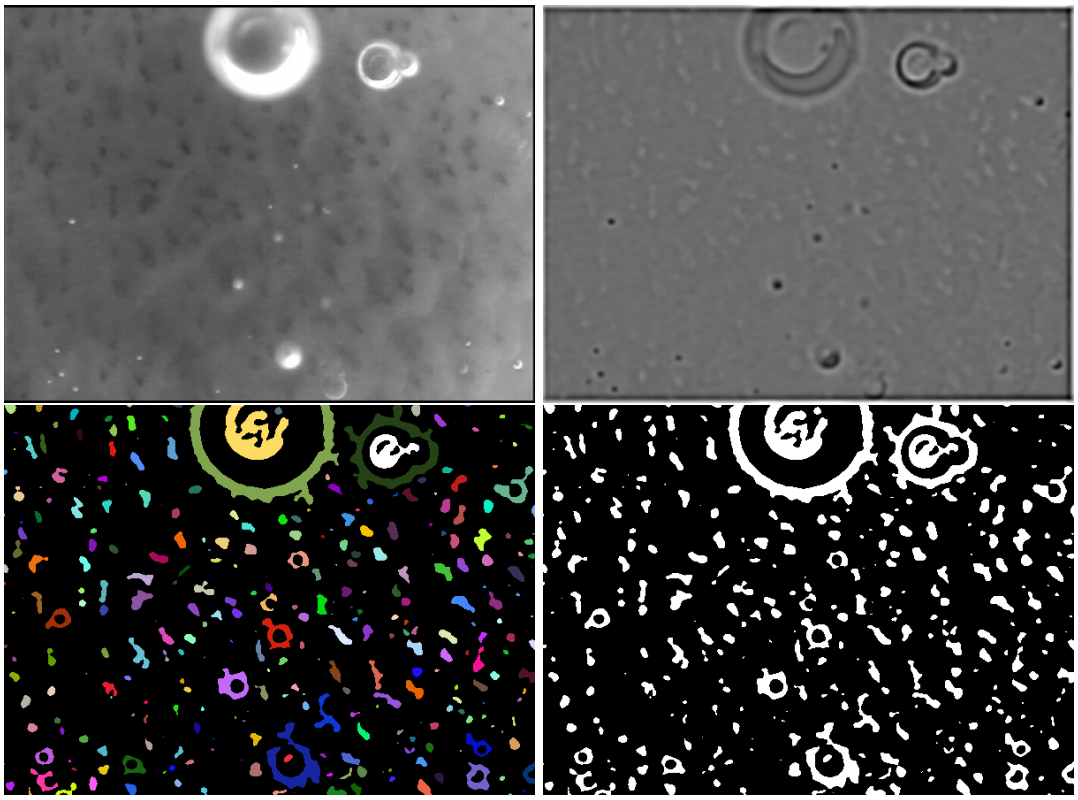


Figure 3: Blob detection. Clockwise from top left: (a) The "green" channel of the input image, (b) Laplacian response L (c) result of pixel classification into blob and non-blob pixels, and (d) the result of connected components labeling, indicating by superimposing the detected blob outlines on the original image; red outlines indicate detected blobs, while blue outline indicate blobs which have been rejected due to their relatively small size.

contrast and signal to noise ratio. The method classifies image pixels into blob and non-blob pixels. Upon this classification, a Connected Component labeling method outputs image blobs. Then, using conventional contour tracing techniques the outline, area, centroid, and bounding box of the blobs are calculated.

To classify pixels we utilize the response of convolving the image with the Laplacian, but employed in a range of scales similarly to [2]. In our case, we compute the convolutions for each scale and average responses across scales, weighting scales that the Laplacian is attuned to more [3]. This results in an image, L, where capillaries appear as bright blobs against a dark background. Image L is then thresholded and a connected components labeling clustering is applied. For each component its contour, centroid and outline and bounding box are extracted. These components are filtered according to size and very small blobs are considered as spurious, and excluded from the result. In Fig. 3, the described image processing pipeline is demonstrated.

Figure 4: Bubble detection on the monochromatic image, shown in Fig.4. Left to right: (a) intensity thresholding of input image, (b) thresholding of image L with a high threshold value, and (c) logical intersection of the two intermediate, (b) and (c) results.

2.2 Bubble detection

Bubble detection is based on the observation that bubbles in the acquired datasets occur as circular blobs and very bright in intensity. Thereby the "green" image channel is thresholded. In addition, image L (see Sec. 2.1) is also thresholded for very high values. This is because the Laplacian being circular produces a particularly high response for circular blobs. The logical intersection of the two thresholding procedures is implemented by a pixelwise logical AND operation. The detected bubbles are extracted as blobs using Connected Component Labeling, as in Sec. 2.1. Fig. 4 illustrates the process.

2.3 Capillary detection

Capillaries are detected by combining the results of blob and bubble detection. The blob detector of Sec. 2.1 is applied to the entire image. Then the image regions where bubbles appear are detected, as in Sec. 2.2. Finally, blobs which overlap with bubbles are excluded from the result.

Fig. 5, demonstrates the final output of the proposed image processing methods.

3 Integrated application

The functionalities provided by the software are integrated within a Graphical User Interface (GUI). Beside the conventional logistic operations (i.e. select file or interest, export measurement results to file, etc), the main goals of this GUI are:

- to support interactive measurements of medical professionals on regions and vessels of interest in an ergonomic and to
- facilitate recovery from shortcomings of the automatic image processing algorithms, i.e. in problematic imaging conditions, and allow the expert user to modify their outcome.

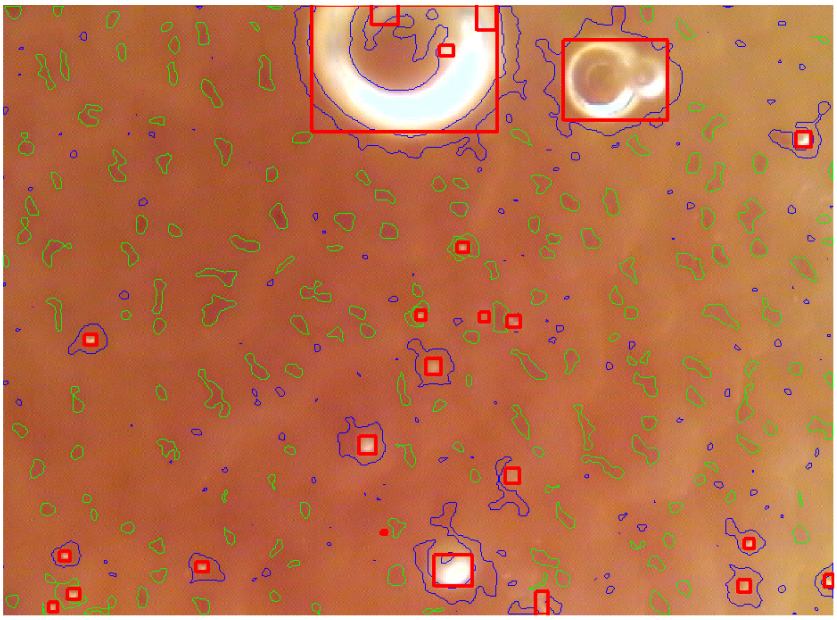


Figure 5: Detection of capillaries in an image. Detected capillaries are outlined and enumerated in green. Very small blobs or blobs that are rejected because they occur in the vicinity of bubbles are outlined in blue. Image regions identified to occur within bubbles are also marked with a bounding box.

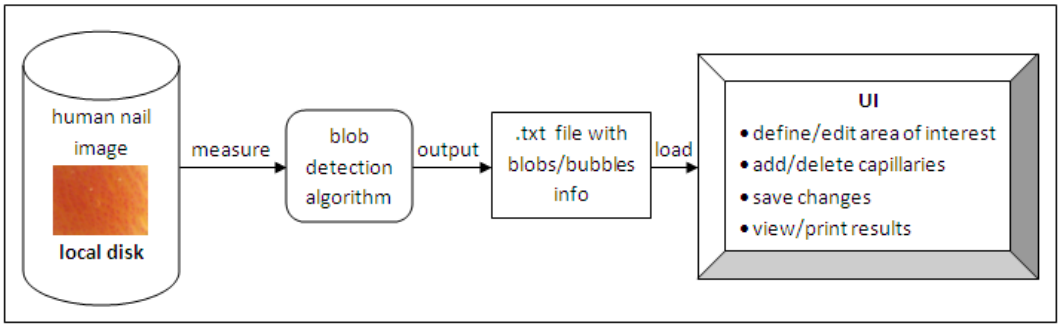


Figure 6: typical scenario for the enumeration of the capillaries in a human nail image, loading and editing automatic results.

The first goal is served by using an online representation of the detected vessels. In this way, the user can indicate vessels and vessel segments of interest with ease.

At the same time, the GUI provides a visualization of this representation indicating to the medical professional, the detected vessels on the original image. Using this visualization as feedback, the GUI supports the second goal by providing an editor where inaccurate vessel detection results can be corrected.

3.1 Enumeration of capillaries

A typical operation during the inspection of the image by the medical professional is the enumeration of the capillaries in the image (see Fig. 6). User starts with measuring a human nail image which is given as input to the blob detection algorithm described above. The output of the algorithm is a text file with the full information of the detected blobs and bubbles. This info is loaded to suitable data structures of the application. Results are displayed in the graphical user interface of the application. User is able to define/edit (move-resize) a rectangular area of interest, add/delete capillaries, save changes, view/print results in textual or graphical format.

3.2 Semi-automatic preprocessing

Using the user interface the operator of the software can edit the output of capillaries detection, by adding and deleting capillaries in the binary segmentation image. The purpose of implementing this functionality is twofold:

1. Improve measurements by correcting for shortcomings of segmentation. That is, to provide the user a way to correct for remaining errors in the segmentation image and update measurement results accordingly.

2. Acquire ground truth results. Using the interface a medical professional can have a basis to correct the result of segmentation and provide ground truth, regarding segmentation in human nails images.

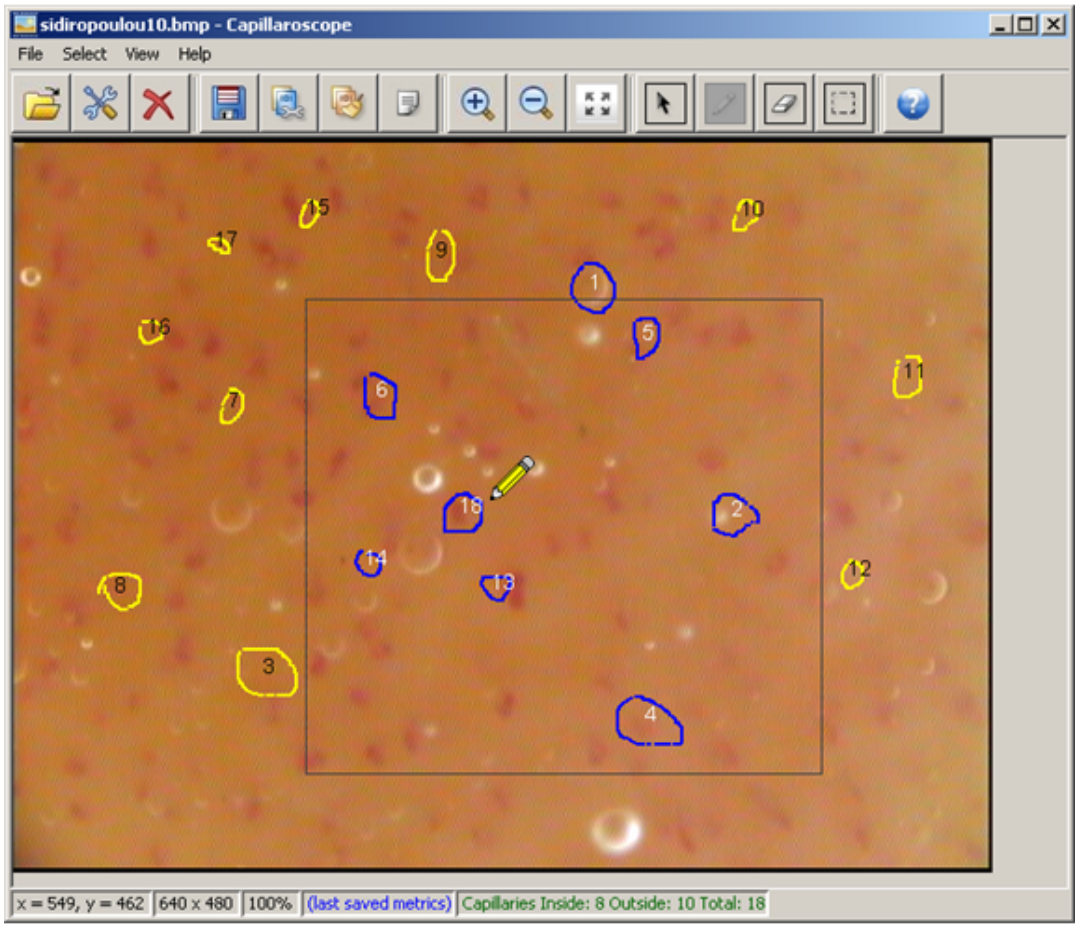


Figure 7: Adding cappilaries.

Increasing the automation in processes 1 and 2 above is the topic of current and future work. The edited segmentations and corresponding results are stored in separate files.

The application graphical user interface facilitates the process. The examples below illustrate cases of use.

3.2.1 Example 1. Adding cappilaries

 Pencil tool: user can add a new capillary.

Short description of the algorithm for adding new capillaries (see Fig. 7):

In case pencil tool () is currently selected:

- On mouse pressed in image, add a new empty capillary to the list of capillaries.
- On mouse dragged in image, with current point (x, y).
  - draw current point (x, y).
  - add current point (x, y) to the list of points of the last added capillary.
- On mouse released in image
  - in case last added capillary is empty (user just clicked the pencil without dragging), remove it from the list of capillaries
  - otherwise, calculate capillarys middle point (used for displaying the capillarys number)
  - redraw image capillaries

3.2.2 Example 2. Deleting cappilaries

 Eraser tool: user can delete a capillary.

Short description of the algorithm for deleting capillaries (see Fig. 8):

In case eraser tool () is currently selected:

- On mouse moved in image, display with a tool tip text the capillary for deletion
- On mouse pressed in image with current point (x, y)
  - get the capillary with the closest middle point to the current one
  - remove it from the list of capillaries
  - redraw image capillaries

3.2.3 Example 3. Selecting area of interest

 Select area tool: user can drag the mouse to define a rectangular area of interest. After this, the image measurements are limited to this area.

Short description of the algorithm for selecting area of interest (see Fig. 9):

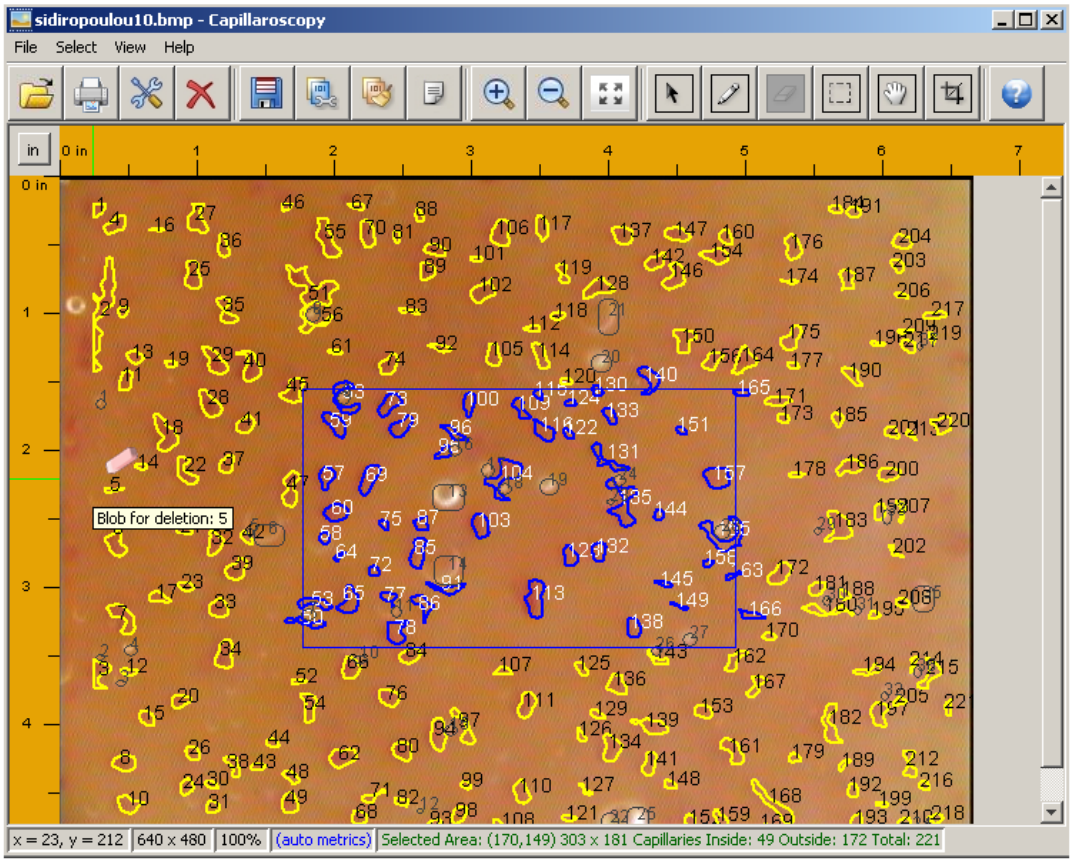


Figure 8: Deleting cappilaries.

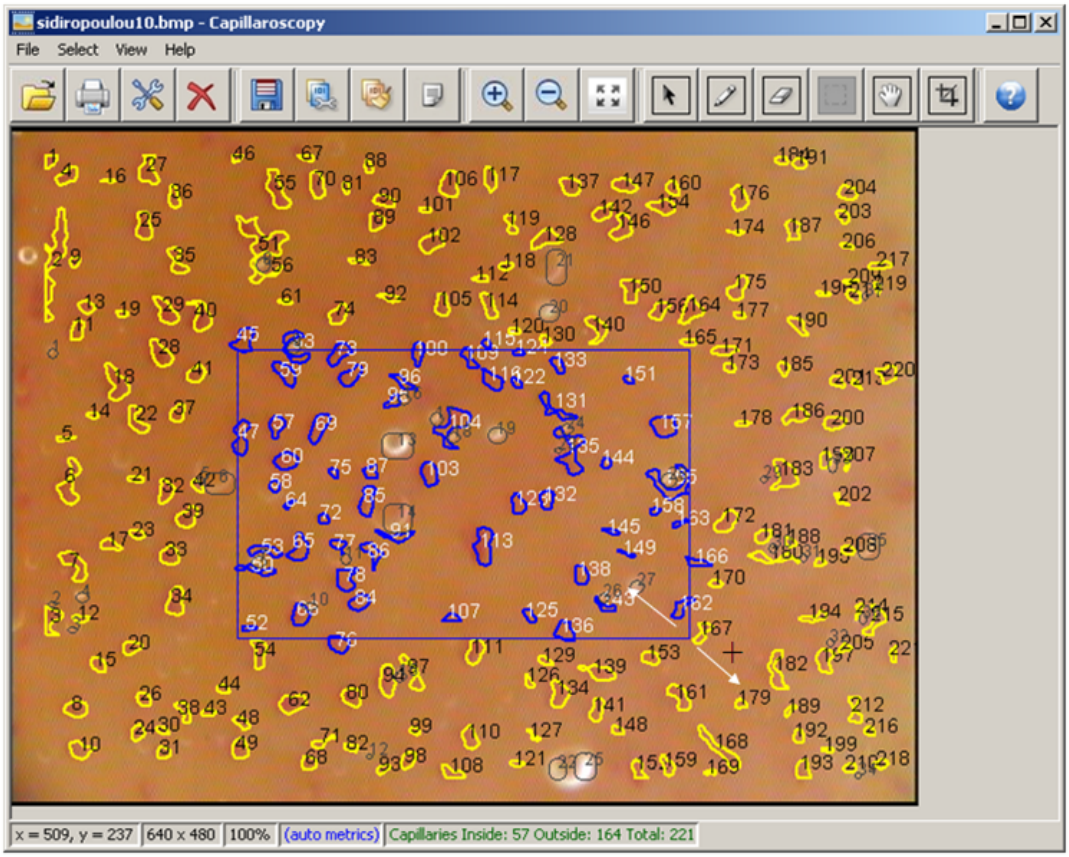


Figure 9: Selecting area of interest.

In case selection tool () is currently selected:

- On mouse pressed in image with current point (x, y)
  - initialize with current point (x, y) a rectangle data structure $AreaOfInterest$ keeping the following information:
    - int $topLeftPointX$, $topLeftPointY$, $bottomRightPointX$, $bottomRightPointY$ (coordinates of the peak points of the rectangle)
    - int $startPointX$, $startPointY$ (coordinates of the 1st point pressed by user)
- On mouse dragged in image with current point (x, y), set
  - $topLeftPointX = \text{minimum}\{x, startPointX\}$
  - $topLeftPointY = \text{minimum}\{y, startPointY\}$
  - $bottomRightPointX = \text{maximum}\{x, startPointX\}$
  - $bottomRightPointY = \text{maximum}\ \{y, startPointY\}$
- On mouse released in image, in case current rectangle width ($bottomRightPointX - topLeftPointX$) or height ($bottomRightPointY - topLeftPointY$) is zero (0), cancel selected area of interest by selecting all image:
  - $topLeftPointX = 0$
  - $topLeftPointY = 0$
  - $bottomRightPointX =$ image width
  - $bottomRightPointY =$ image height
- redraw image capillaries and area of interest
  - capillaries inside the area of interest $\Leftrightarrow$ with at least one (1) point (x, y): $x \in [topLeftPointX, bottomRightPointX]$ and $y \in [topLeftPointY, bottomRightPointY]$, are displayed with blue colour.
  - capillaries outside the area of interest $\Leftrightarrow$ no points inside it, are displayed with yellow colour.

3.2.4 Example 4. Moving area of interest

 Move area tool: user can drag the mouse to move the predefined rectangular area of interest. After this, the image measurements are limited to this area.

Short description of the algorithm for moving area of interest (see Fig. 10):

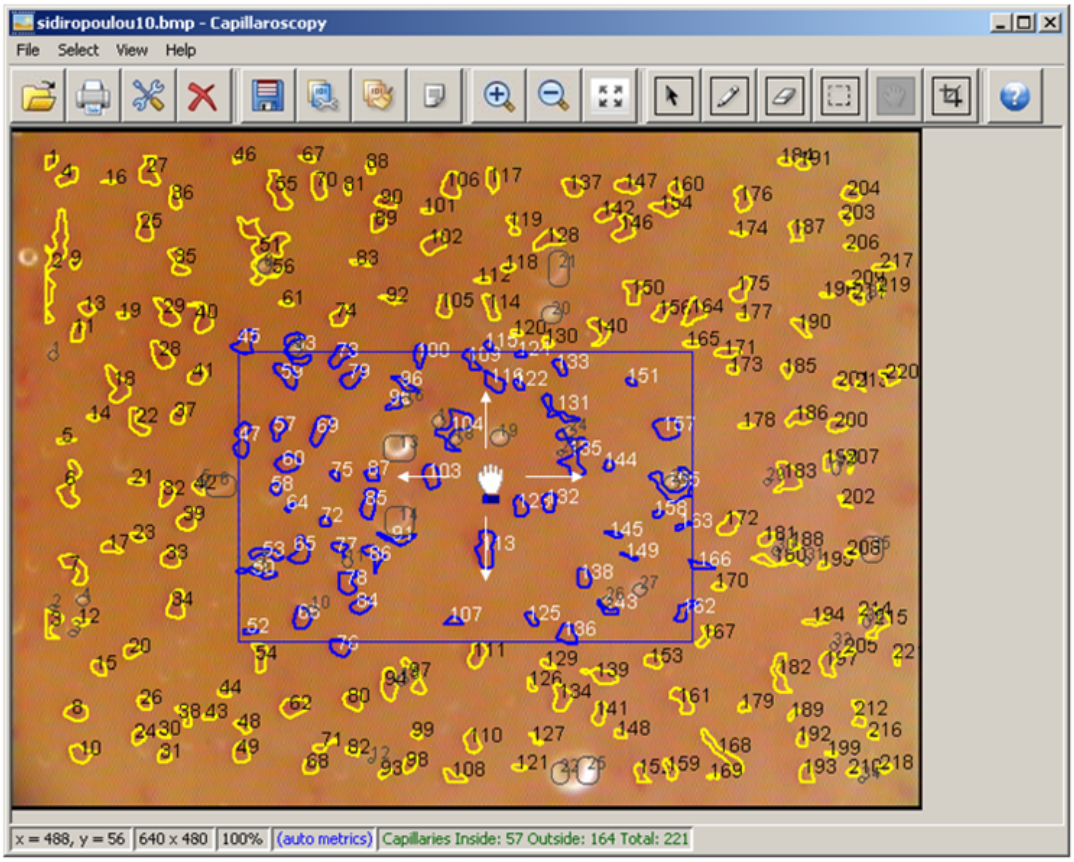


Figure 10: Moving area of interest.

In case move area tool () is currently selected:

- On mouse pressed in image with current point (x, y)
  - initialize with current point (x, y) a rectangle data structure $AreaOfInterest$ keeping the following information:
    - int $startPointX$, $startPointY$ (coordinates of the 1st point pressed by user)
- On mouse dragged in image with current point (x, y), set
  - $topLeftPointX = topLeftPointX + (x$ - $startPointX)$
  - $topLeftPointY = topLeftPointY + (y$ - $startPointY)$
  - $bottomRightPointX = bottomRightPointX + (x$ - $startPointX)$
  - $bottomRightPointY = bottomRightPointY + (y$ - $startPointY)$
  - $startPointX = x$ and $startPointY = y$
  - and redraw area of interest
- On mouse released in image, in case current rectangle width ($bottomRightPointX$ - $topLeftPointX$) or height ($bottomRightPointY$ - $topLeftPointY$) is zero (0), cancel selected area of interest by selecting all image:
  - $topLeftPointX = 0$
  - $topLeftPointY = 0$
  - $bottomRightPointX =$ image width
  - $bottomRightPointY =$ image height
- redraw image capillaries and area of interest

3.2.5 Example 5. Resizing area of interest

 Resize area tool: user can resize the predefined rectangular area of interest by dragging one of the eight (8) hot points of the area (4 angles and 4 middle points). After this, the image measurements are limited to this area.

Short description of the algorithm for resizing area of interest (see Fig. 11):

In case resize area tool () is currently selected:

- On mouse moved in image with current point (x, y), in case it is over one of

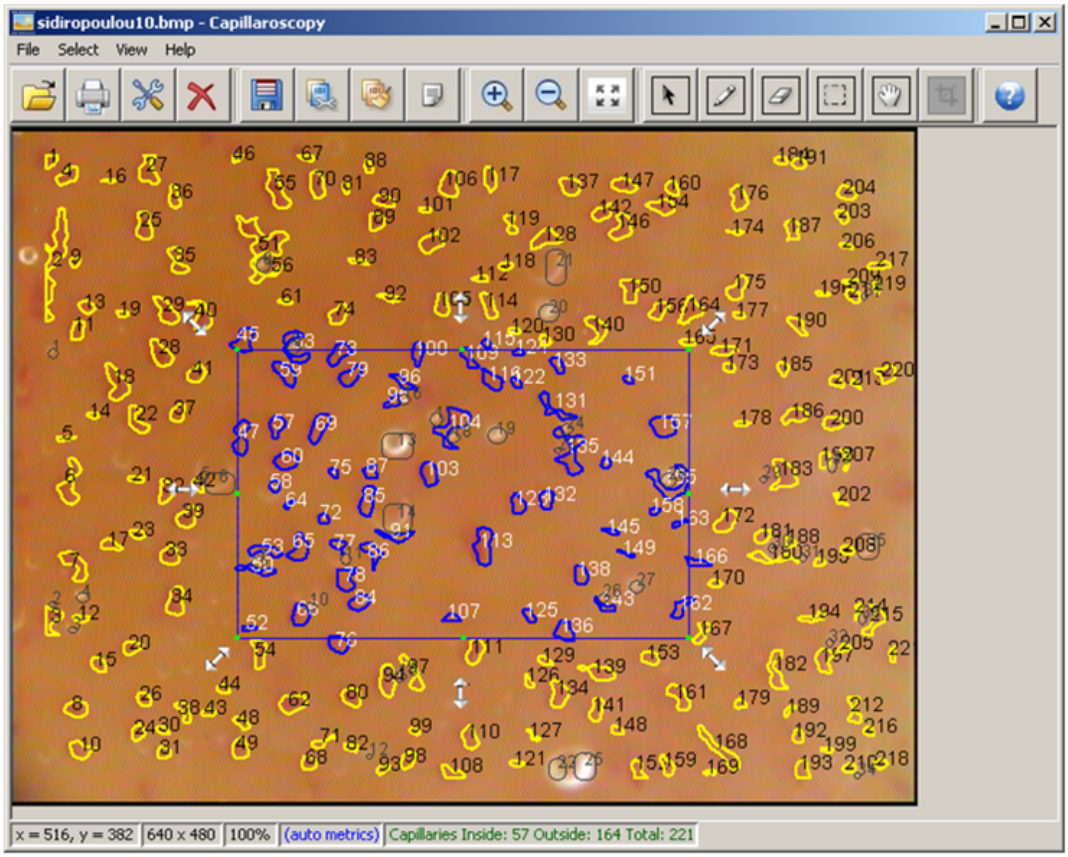
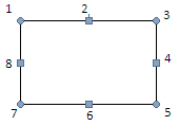


Figure 11: Resizing area of interest.

the eight (8) hot points of the area (4 angles and 4 middle points), set the corresponding cursor shape:  or  or  or 

- On mouse pressed in image with current point (x, y), in case it is over one of the eight (8) hot points of the area (4 angles and 4 middle points)
  - initialize int $currentResizeHotPointDragging$ with a value 1-8: 

- On mouse dragged in image with current point (x, y), for hot point:
  - (1)
    - $topLeftPointX = x$; if $(topLeftPointX > bottomRightPointX)$ $topLeftPointX = bottomRightPointX$
    - $topLeftPointY = y$; if $(topLeftPointY > bottomRightPointY)$ $topLeftPointY = bottomRightPointY$
  - (2)
    - $topLeftPointY = y$; if $(topLeftPointY > bottomRightPointY)$ $topLeftPointY = bottomRightPointY$
  - (3)
    - $topLeftPointY = y$; if $(topLeftPointY > bottomRightPointY)$ $topLeftPointY = bottomRightPointY$
    - $bottomRightPointX = x$; if $(bottomRightPointX < topLeftPointX)$ $bottomRightPointX = topLeftPointX$
  - (4)
    - $bottomRightPointX = x$; if $(bottomRightPointX < topLeftPointX)$ $bottomRightPointX = topLeftPointX$
  - (5)
    - $bottomRightPointX = x$; if $(bottomRightPointX < topLeftPointX)$ $bottomRightPointX = topLeftPointX$
    - $bottomRightPointY = y$; if $(bottomRightPointY < topLeftPointY)$ $bottomRightPointY = topLeftPointY$
  - (6)
    - $bottomRightPointY = y$; if $(bottomRightPointY < topLeftPointY)$ $bottomRightPointY = topLeftPointY$
  - (7)

    - $*$ $topLeftPointX = x$; if ($topLeftPointX > bottomRightPointX$) $topLeftPointX = bottomRightPointX$
    - $*$ $bottomRightPointY = y$; if ($bottomRightPointY < topLeftPointY$) $bottomRightPointY = topLeftPointY$
  - (8)
    - $*$ $topLeftPointX = x$; if ($topLeftPointX > bottomRightPointX$) $topLeftPointX = bottomRightPointX$
  - and redraw area of interest
- On mouse released in image, in case current rectangle width ($bottomRightPointX$ - $topLeftPointX$) or height ($bottomRightPointY$ - $topLeftPointY$) is zero (0), cancel selected area of interest by selecting all image:
  - $topLeftPointX = 0$
  - $topLeftPointY = 0$
  - $bottomRightPointX$ = image width
  - $bottomRightPointY$ = image height
- redraw image capillaries and area of interest

4 Application scenario and output

A characteristic scenario of use is presented in this section, indicating its context of use.

Initially the user can browse multiple files and select one or more human nails image files to load their metrics information to the system. Such files have been precomputed either as a direct result of the employed image processing methods, or be an edited version of these results stored by the user. Having precomputed the measurement results accelerates user interaction and decreases user waiting. Large file collection are handled in batch mode by a pertinent software utility. A text file for each image, with coordinates of capillaries and bubbles, is saved to disk.

Once the image is loaded, the user can select a specific area of interest and enumerate the capillaries inside it. Another online user operation is the ability to add or remove capillaries. In both cases, the collected measurements can be saved for reviewing or exported in a text file for future processing.

Several other utility functionalities are performed through the user interface, such as the saving of data file with measurements, as well as, the reviewing and editing of older measurement files. In the display the user, can view the image and the calculated measurements using magnification options as well as rulers that indicate metric spatial

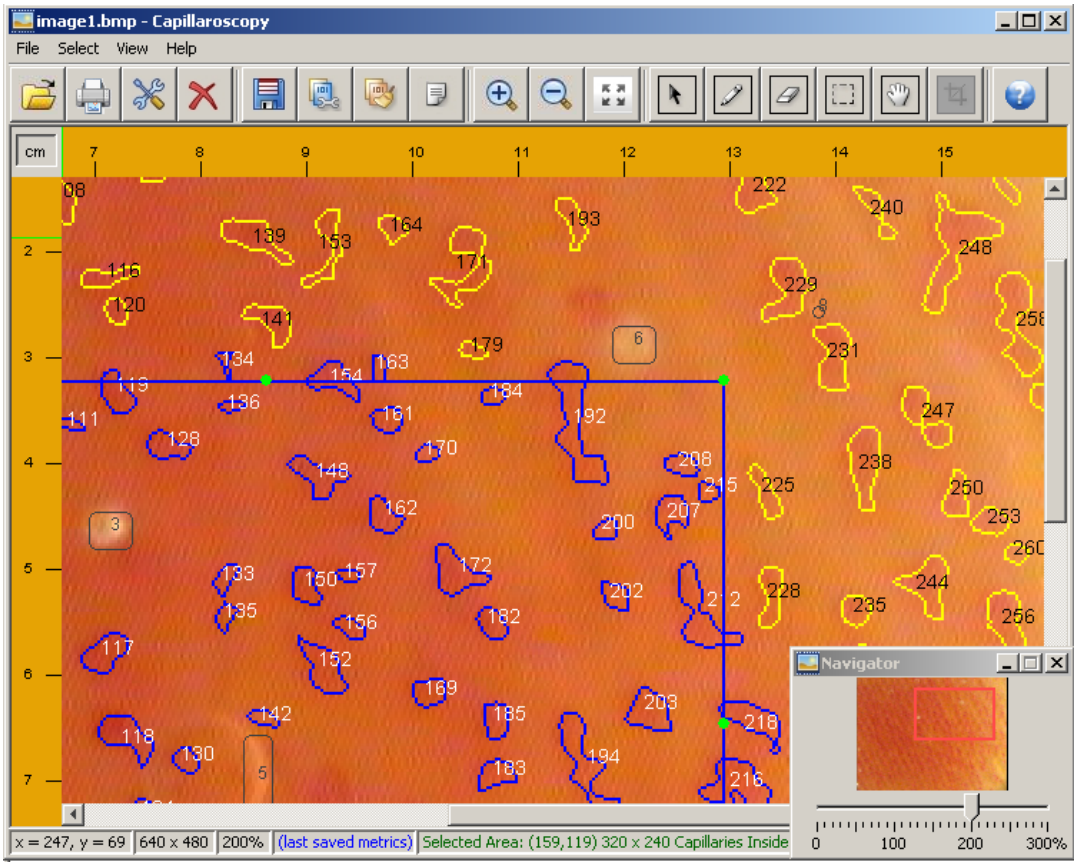


Figure 12: Image navigation utility panel of the user interface. Using this panel the user can display detailed views of the human nail image representation and inspect them in context to the original image.

Figure 13: User interface toolbar. A toolbar in the graphical user interface, provide direct access to the main functionalities of the system.

scales (see Fig. 12). When in magnification a GUI component indicates the currently visible image segment, with respect to the total image (see Fig. 12).

All of the above functionalities are available through a toolbar of GUI buttons (see Fig. 13).

5 Conclusion and future work

In this report, an application that increases the automation of capillary detection in capillaroscopy images is presented. This application employs an image segmentation

algorithm, which detects the appearing capillaries as blobs.

The above functionalities are integrated through a GUI, which assists the medical professional to perform measurements in an ergonomic fashion and apply targeted measurements according to medical protocols. This interface provides also functionalities that allow the medical professional to edit the detection result, in order to recover from segmentation errors.

Future work will be pursued along the improvement of the segmentation algorithm, in order to obtain more accurate measurement results, despite poor image quality.

Acknowledgements

This work has been supported by the FORTH-ICS internal RTD Programme 'Ambient Intelligence and Smart Environments'. We thank the Hypertension Unit, of the 2^{nd} Propedeutic Department of Internal Medicine, of the Hippokration Hospital, of the Aristotle University of Thessaloniki for posing the retinal vessel measurement problem as an application of interest, providing pertinent data, as well as, their insightful guidance into the requirements of medical professionals and contemporary diagnostic protocols. From this unit, we thank Areti Triantafyllou and Stella Douma for their contribution in the above collaboration.

References

[1] DS Medica S.r.l. Videocap. http://www.videocap.it/.

[2] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11:283–318, 1993.

[3] X. Zabulis, J. Sporring, and S. C. Orphanoudakis. Scale summarized and focused browsing of primitive visual content. In *Visual*, pages 269–278, 2000.