Extracting liquid container boundaries in an image using separation from background (fully automatic mode).

1. Extracting the boundaries/contour of vessel in image by separation from of vessel from background

Extracting the borders and boundaries of symmetric vessel/object in image by separation of vessel from background. This mostly focus on transparent glass vessel for chemistry lab use but could probably be used for other symmetric object.

The script to use is in

Directory_Extract_from_background.m

This script scans all 'jpg' image in given directory in *dirname* (specify in line 4 of the script) and find the boundaries of the vessels in these images.
The script writes the boundaries/contour of the vessel found in the image **x.jpg** in in binary image **x_BORDERS.tif** in the same directory as the original image (*dirname*).
During the scan the size of the image is changed. The image **x.jpg** in its resize form and in greyscale is written in **x_SYSTEM.tif** file in the directory of *dirname*. The boundary of the object in the image **x.jpg** is written in **x_BORDERS.tif** in binary image where the borders marked 1(white). This image same size of the resize system image in **x_SYSTEM.tif.** In other words the borders marked in output file **x_BORDERS.tif** match the vessel in output image **x_SYSTEM.tif.** In addition the image of the object with the boundary of the vessel marked in white on the image is written as output in **x_MARKED.tif** in the same directory.

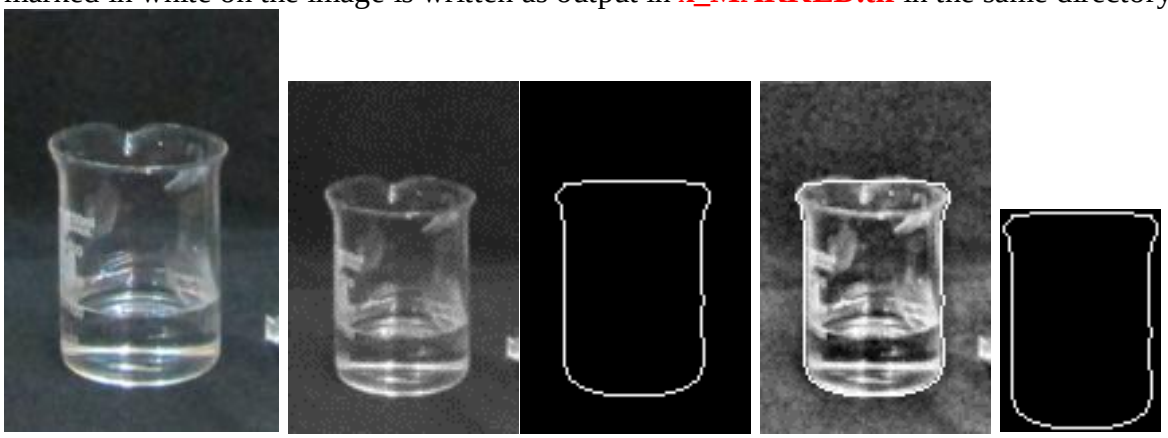


Figure 1 From left to right: The original image (x.jpg) used as input. The resized input image in greyscale (x_SYSTEM.tif) received as output. The boundary of the vessel in the image as binary image (x_BORDERS.tif) received as output (this image size is the same as x_SYSTEM.tif. The image with the found boundary marked upon it (x_MARKED.tif) again as output. Binary image with template in of the object (x_TEMPLATE.tif) which could be used to trace the object in other images

The binary image with the borders of the vessel (**x_BORDERS.tif**) could be used together with the system image (**x_SYSTEM.tif**) in the recognition of the liquid surfaces and phase boundaries within the vessel. The **_BORDER.tif** output file is the the second parameter

(Iborder) in the function *Liquid_Surface_Line_Recognition(Is,Iborder,outname….)* used for the recognition of liquid surfaces in images.
The output images which end by '**_TEMPLATE.tif**' contain a binary the template of the vessel which could be used to recognise the boundary of the vessel in other images (see script for recognition of , using templates).

2. Instructions for use:

1) Open script *Directory_Extract_from_background.m*.
2) Insert the directory name in which the image of the object are stored, into *dirname* in line 4 of the script.
3) Run script.
4) After the script finished running the output should appear in the same directory of the original images (*dirname)*. Note that running time is usually few minutes per image but could be longer(>10 minutes per image).

3. Examples

See "EXAMPLE IMAGES" directory for example input images and can be used as test. The "EXAMPLE IMAGES" directory located in the source code directory.

4. Input:

Jpg color images located in library given in *dirname (*specify in line 4 of *Directory_Extract_from_background* script*)*

5. Restriction on input image

The images must be in color jpg format.

The objecy must be symmetric relative to the Y axis of the image.

The background should be as uniform as possible .

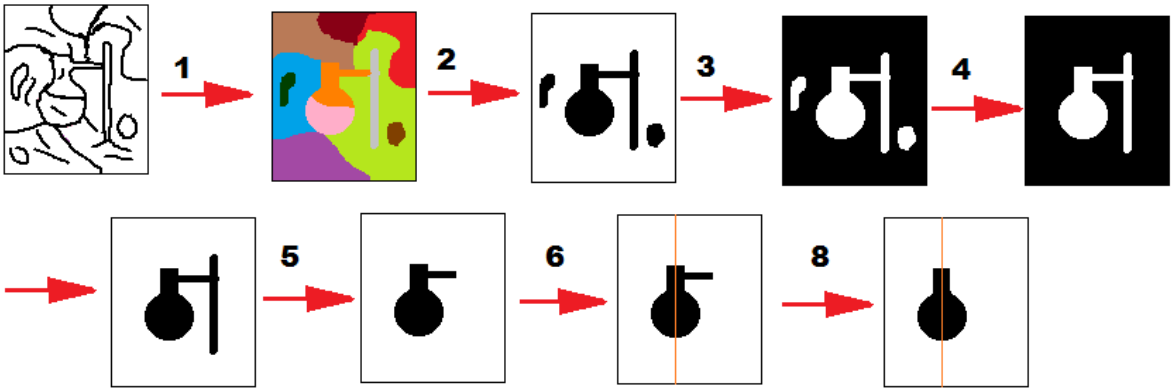
The object must not touch any of the outer edges of the image.

The vessel **could** be held by a Stand (chemistry lab apparatus) the Stand will be ignored in the recognition process.

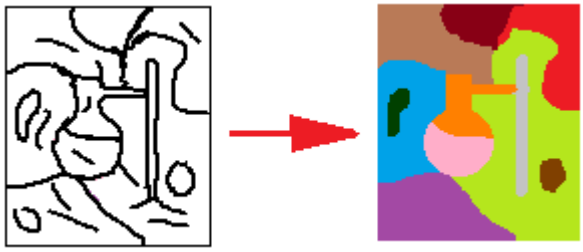
6. OUTPUT

See figure 1, section 1.

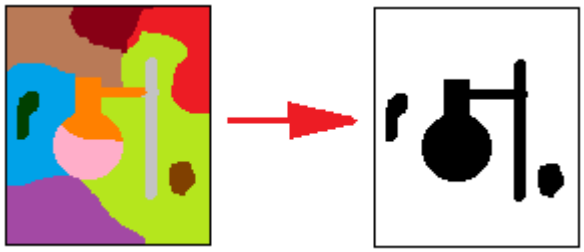
7. Algorithm:



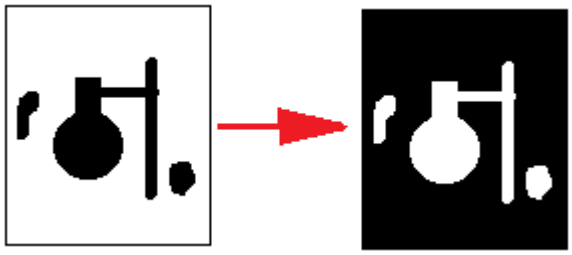
1) Segment the image by looking for edges that form closed contours. The edges are found by combination of canny and sobel edges detector.



2) All blobs that touch the outer boundaries of the image are merged and considered as background.

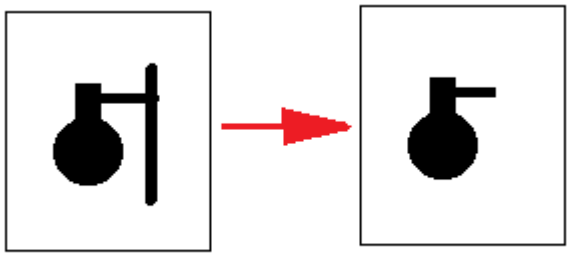


3) The negative of the background (found in step 2) is taken. This negative is a binary image in which every pixel that is not background have value of 1 and every background pixel is zero.

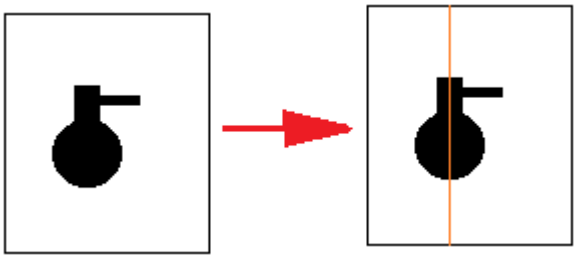


4) This binary image is again segmented, and the largest blob is taken as the vessel.

5) If the blob have two parallel region in the horizontal axis the thinnest of the two is deleted (other word if some line of the blob have more than two edges remove all but the edges that form thickest region. Done to remove stands (chemistry lab) from images)

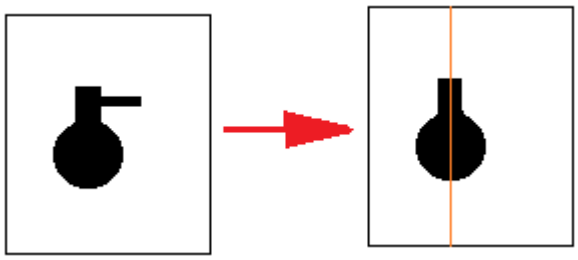


6) The symmetry axis and symmetry level of the blob is found by scanning every line of the blob and finding its center x value for each line. The most abundant center value (x) for lines in the blob is taken as the symmetry axis.



7) The fraction of lines that have center in the symmetry axis is taken as the symmetry level of the blob that will later be used to calculate its score.

8) For each line in the blob that don't have center in the symmetry axis change either the left or right boundary position of the line such that the new center of the line will be on the symmetry axis.



9) The contour of the resulting blob could be used as output for the edges of the boundaries in the image.

10) The symmetry level of the blob is used to score how good is the match of the blob to the object in the image.

11) Scan steps 9-10 on the image in various of sizes and with various of threshold for the segmentation step. Find the blob with the highest symmetry score.

12) Use the blob with the highest symmetry score as output.

8. Main functions and scripts

Directory_Extract_from_background: Script that perform the recognition process on every jpg file within given directory (the directory page is given in *dirname* in line 4 of the script). The output file will be created in *dirname after the script finished* (few minutes per file). See sections 1-6.

Exctract_object_from_background(*filename, segmentation_mode ,Symmetry_Mode)*

Description: Found the boundaries of symmetric object in an image with uniform background.

Input:

filename: The name +location of the image files.

Segementation_mode(optional parameter): The method that will be used to segment the image. Option *Segementation_mode ='BORDER_CANNY'* is a default mode and it will segment the image by using canny and sobel edges that form closed contours. Alternative segmentation is by 'THRESHOLD' which segment image using intensity threshold with OTSU limits, this give bad result for transparent vessels.

Symmetry_Mode (optional parameter)*:* The way in which the symmetry of the object will be adjusted. The values can be between 0 to 2. Value of 2 is default value of 1 give slightly different symmetry adjusting mode with almost same result. Value of 0 mean no symmetry adjustment (for a symmetric). The symmetry mode is explained in the function *symmetrized*.

Output files: All output are given as both parameters and files in the directory of the input file see section 1 and figure 1 for details on output files.

Output parameters:

Iresize: The input image (*filename*) in greyscale and resized to specific size that gave the best boundaries (The corresponding output file is *filename* _SYSEM.tif output files [Figure 1, Section 1])

Imarked: Same as Iresize only with the boundaries of object marked on the image (The corresponding to output file is *filename*_MARKED.tif [Figure 1, Section 1])

Iborders: Binary image of size of Iresize with all pixels correspond to the object boundaries marked white 1 and the rest marked black 0 (The corresponding file is *filename*_BORDER.tif [Figure 1, Section 1]).

Itemplate: Binary image of the object border with object boundaries marked white (1). Can be use as template for finding the object in other images (The corresponding output file is *filename* _ TEMPLATE.tif [Figure 1, Section 1]). The template size match the vessel size/shape in Iresize.

Y_img_size: The final size of the image Y axis (for images Iresize and Iborders).

Algortihm:

Scan various of threshold and sizes of for segmenting image in filename. The recognition itself done by *Find_Vessel_Contour*. This function scan *Find_Vessel_Contour* with various of different image size for the image in filename and with various of threshold for the edges in the segmentation step. It pick the best result based on the its symmetry score (symmetry of the contour found).

Find_Vessel_Contour(figurefilename, Symmetry_Mode, Npix, threshold, SegmentationMode)

Description: Found the boundaries of symmetric object in an image with uniform background.

Input:

Figurefilename: The name +page of the image file.

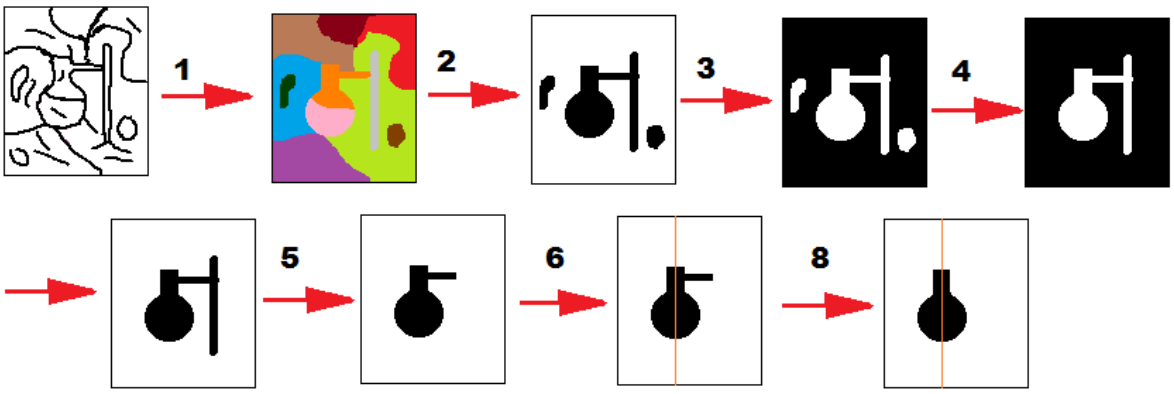
Segementation_mode(optional parameter): The method that will be used to segment the image. Option *Segementation_mode* ='BORDER_CANNY' is a default mode and it will segment the image by using canny and sobel edges that form closed contours. Alternative segmentation is by *Segementation_mode* = 'THRESHOLD' which segment image using intensity threshold with OTSU limits this give bad result for transparent vessels.

Symmetry_Mode (optional parameter): The way in which the symmetry of the object will be adjusted. The values can be between 0 to 2. Value of 2 is default. Value of 1 give slightly different symmetry adjusting mode with almost same result as 2. Value of 0 mean no symmetry adjustment. The symmetry mode is explained in the function *symmetrized*.

Npix(Optional parameter): Max number of pixels in the image examined (*Figurefilename*) . If the image have more pixels then *Npix* then it will be resized to the size of *Npix* pixels while maintaining proportion. Note that large image (above16000) can take really long time and should be avoided.
Threshold (optional parameter): The threshold that will be used in the canny/sobel edge detector in the segmentation step.

Output
*vessel_cont:*Binary image of the object border with object boundaries marked white (1). Can be use as template for finding the object in other images (similar to X_TEMPLATE.tif Figure 1).
symmetry_score: The symmetry level of the traced blob (fraction of lines in the original blob that have center in the blob symmetry axis).
brxy: List of x,y coordniates of boundary of the object in the image (after the image were resize).
imborder: The resized image with the boundaries of object marked on it (see IMarked [figure 1 section 1]).
YaxisSize: The final size of the image Y axis in which the recognition where performed

Algortihm:
Basically steps 1-10 in the algorithm section in section 7:



1) Segment the image by looking for edges that form closed contours. The edges are found by combination of canny and sobel edges.

2) All blobs that touch the outer boundaries of the image are merged and considered as background.

3) The negative of the background (found in step 1) is taken. This negative is a binary image in which every pixel that is not background have value of 1 and every background pixel is zero.

4) This binary image is again segmented, and the largest blob is taken as the vessel.

5) If the blob have two parallel region in the horizontal axis the thinnest of the two is deleted (other word if some line of the blob have more than two edges remove all but the edges that form thickest region).

6) The symmetry axis and symmetry level of the blob is found by scanning every line of the blob and finding its center x value for each line. The most abundant center value (x) for lines in the blob is taken as the symmetry axis.

7) The fraction of lines that have center in the symmetry axis is taken as the symmetry level of the blob that will later be used to calculate its score.

8) For each line in the blob that don't have center in the symmetry axis change either the left or right boundary position of the blob such that the new center of the line will be on the symmetry axis.

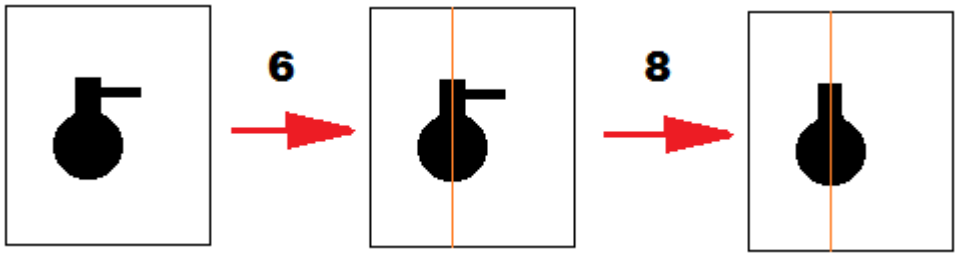
9) The resulting contour of the resulting blob could be used as output for the edges of the vessel in the image.

10) The symmetry level of the blob is used to score how good is the match of the blob to the vessel in the image.

.

symmetrized(*BW2,Symmetry_Mode*)

Description: Get binary image with one blob. Find its most likely symmetry axis for the blob. Adjust the blob boundaries so it will be completely symmetric around this symmetry axis. Found the image symmetry level (fraction of lines that were symmetrized around this line.



Input:

BW2: Binarry image containing one blob to be symmetrized.

Symmetry_Mode (optional parameter): The method that will be used to symmetrized the image. *Symmetry_Mode* =1 if a given line in the blob does not have center in the symmetry axis: Replace the either the left or right edge of this line depending on which edge is farther away from the edge of the closest line that center around the symmetry axis.

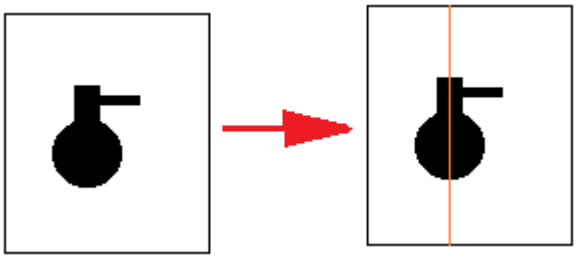
Symmetry_Mode =2 (default). 1 1 if a given line in the blob does not have center in the symmetry axis: Replace the either the left or right edge of this line depending on which edge is farther away from the average of the two closest edges (above and below) of the closest lines that have center in the symmetry axis.

OUTPUT

BW3: Binary image of the blob in *BW3* after it been symmetrized (all lines are center in the symmetry axis)
symmetry_score: Basically symmetry level of the original blob in BW2. The fraction of lines in BW2 blob that have center in the symmetry axis

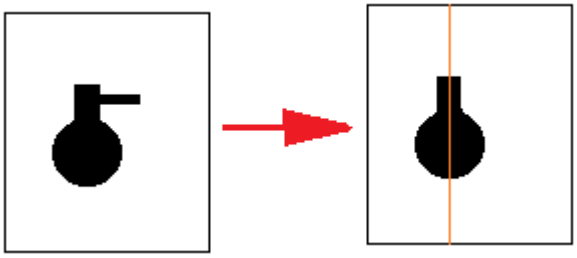
Algortihm:

Step 6-10 in the algorithm in section 7
6) The symmetry axis and symmetry level of the blob is found by scanning every line of the blob and finding its center x value for each line. The most abundant center value (x) for lines in the blob is taken as the symmetry axis.



7) The fraction of lines that have center in the symmetry axis is taken as the symmetry level of the blob that will later be used to calculate its score.

8) For each line in the blob that don't have center in the symmetry axis change either the left or right boundary position of the line such that the new center of the line will be on the symmetry axis.



9) The resulting contour of the resulting blob could be used as output for the edges of the vessel in the image.

10) The symmetry level of the blob is used to score how good is the match of the blob to the vessel in the image.

Remove_Parallel_Region(*BW2*)

Description: Receive binary image with one blob (input: *BW2*): If the blob have more then one parallel regions in the horizontal axis, all but the thickest of these regions is deleted (other word if some line of the blob have more than two edges remove all but the edges that form thickest region.). The blob with deleted region is return as binary image (*output:* BW2)