River-Bar-Toolbox

River-Bar-Toolbox contains two main codes (***Bar_detection***, ***MASK***), five sub-fucntions (CreateBW, CreateBWCLAHE, DrawBar, FindBar, prop) and two directories for input images (Images) and output data (Results). The example input images (image 1, image 2 and image 3) are a product of the post-processing of images acquired by the Sena Gallica Speculator (SGS) videomonitoring station, located in the harbour of Senigallia (Marche, Italy). The images display the final stretch of the Misa river estuary, where an emerged bar is visible. The output for these example images are in the Results directory. Note that the output obtained by the user may vary slightly from those reported in the directory. In fact, some operations (creation of a mask; drawing of a Region Of Interest), which must be performed directly by the user, affect the result.

Bar_detection is a semi-automatic procedure to detect the emerged area of a sediment deposit from images either of a river or of the sea. Through a sequence of image processing operations, the code identifies the emerged part of the deposit and measures its area, centre of mass and perimeter. The code also stores the coordinates of the pixels included in the detected area, in order to allow the reproduction of the shape of the deposit after the analysis.

The success of the code in identifying the bar depends on the difference of colour between the bar and the water. In the input images contained in this repository, the bar is adjacent to the left riverbank, which is of the same colour of the bar. Thus, in order to prevent the code from identifying the pixels of the riverbank as belonging to the bar, a mask is necessary to cover the riverbank.

If the user has a similar problem with its own images, he can use the code ***MASK*** before running ***Bar_detection***.

The code ***MASK*** lets the user choose among the images the one he wants to use to draw the mask. The MATLAB function roipoly is then used to allow the user to draw a polygon on the chosen image. roipoly returns the mask as a binary image, setting pixels inside the Region Of Interest (ROI) to 1 and pixels outside the ROI to 0.

Figure 1 displays the steps of the code:

i) drawing of the polygon by clicking on its vertices (during this operation the pointer is a cross hairs);

ii) moving of the pointer over the initial vertex of the polygon (the pointer changes to a circle) and double-clicking to create the mask;

iii) displaying of the image with the mask superimposed.

The user can also move the polygon and adjust the vertices. For more information see MATLAB Documentation about roipoly.

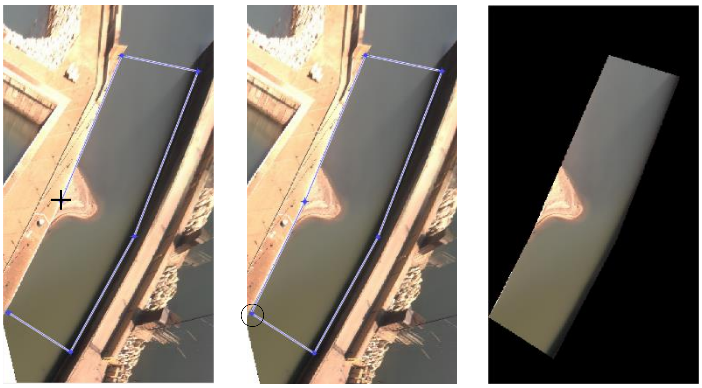


Figure 1. Steps i), ii) and iii) of the code MASK.

If the user does not want to create a mask, he can skip this passage and directly run ***Bar_detection***: the code will use a fictitious mask.

Bar_detection input parameters are:

- D = the name of the input directory that contains the images;
- start = the index of the first image to analyse in D;
- outDir = the name of the output directory.

The first passage of the code is to keep track in the output of the type of analysed image. In fact, the code shows the image and asks the user to classify it by choosing among four different *Types*:

- **Dark Image**: when the image is too dark (bad meteorological/lighting conditions, malfunctioning of the cameras, etc.) that not even the user's eye can identify the emerged deposit;
- **No bar**: when the image does not show any emerged area;
- **Visible Breaking**: when the image shows a wave breaking, which indicates the presence of a submerged deposit;
- **Visible Bar**: when the deposit is visible from the image and the user wants the code to detect it.

If the user selects *Visible Bar*, the code goes on with the detection of the emerged area, otherwise it stores the *Type* in the output file and goes on with the following image.

The code converts the image to grayscale and applies the mask on the image. Then, the user is asked to draw a ROI, i.e. a window inside the image where to execute the detection of the bar. The code performs the gradient of the image (Sobel) and converts it in binary form (black=background, white=bar). Then, it applies some morphological operations on the shape of the bar to fill any void and to smooth the contours. Figure 2 displays the input image, the image of the gradient, the binary image and the final image for the input images in the repository (image 1, image 2, image 3).

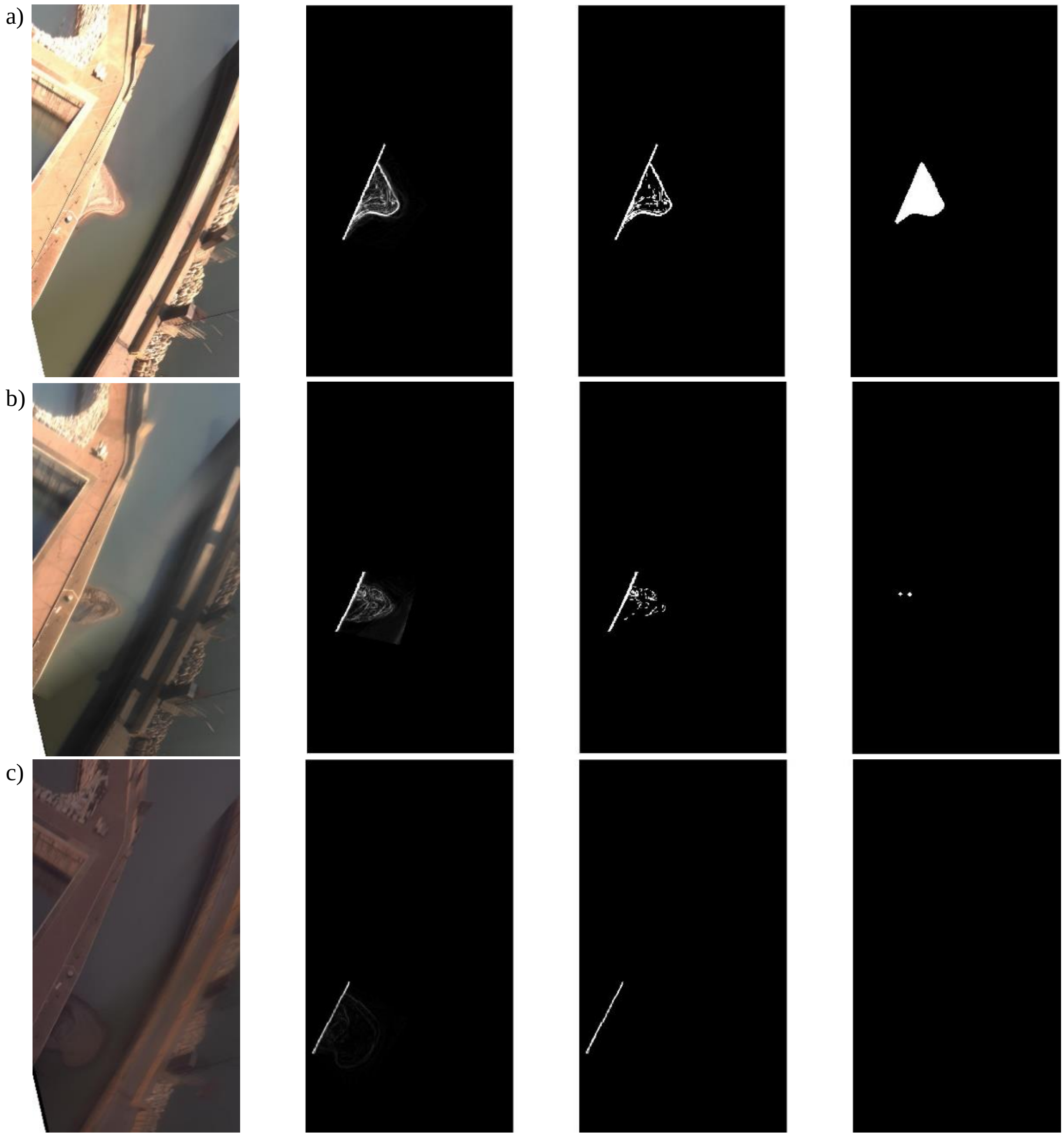


Figure 2. Input image (first column), gradient image (second column), binary image (third column) and final image (fourth column) for a) image 1, b) image 2 and c) image 3.

These operations can provide a suitable image for the final identification of the bar (Figure 2a, fourth column; Figure 2b, fourth column) or a totally black image, impossible to use for this purpose, as for image 3 (Figure 2c, fourth column). In the first case, the code tries detecting the bar and displays the result, i.e. the grayscale image with the coloured bar overlaid, so that the user can check it. Figure 3 displays the result for image 1 and image 2. The code then asks the user if he wants to accept the result. For the example images, the result is accepted for image 1 and discarded for image 2.

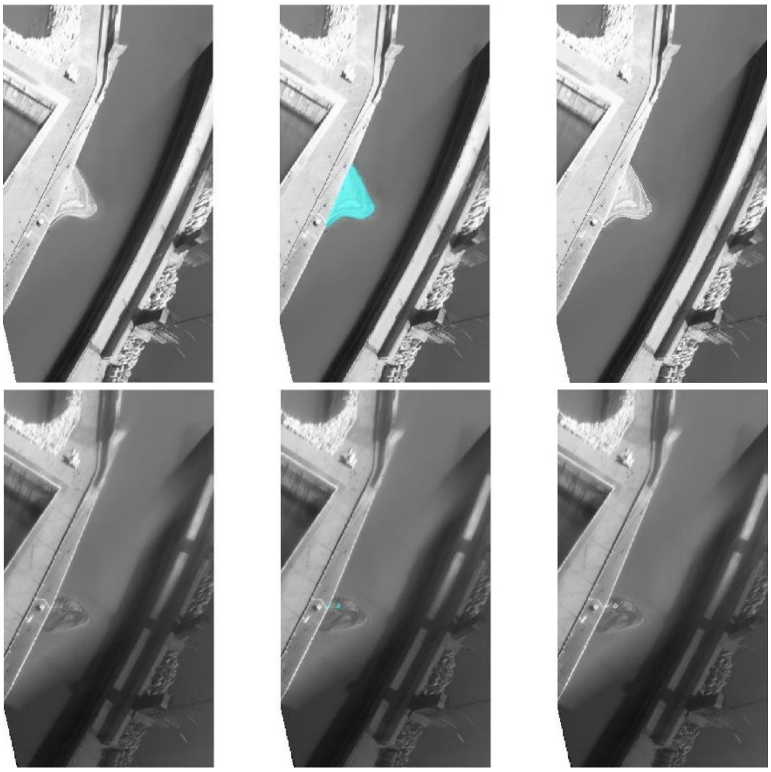
a)
b)


Figure 3. Result of the bar detection: grayscale image (first column), grayscale image with the bar in cyan (second column) and grayscale image with the perimeter of the bar in white (third column) for a) image 1 and b) image 2.

In the second case, the code moves back to the grayscale image and performs its contrast enhancement (CLAHE). In fact, in most cases, the failure in the making of a suitable binary image, depends on the fact that the image contrast is not high enough to allow the gradient operation to highlight the contour of the bar. After the contrast improvement, the code goes on with the other operations described above, i.e. image gradient, binarization, detection of the bar when the binary image is suitable, representation and inspection of the result. The CLAHE algorithm is also used whenever the first attempt to identify the bar fails and the user decides to discard the first result, as for image 2 (Figure 3b). So, the detection of the bar can be executed with two different sequences of image processing operations, and the best result can be accepted. Figure 4 displays the gradient of the image, the binary image and the final image after the application of CLAHE for image 2 and image 3.

If both methods fail to recognize the bar, as for image 3, the user can discard both results and draw the bar shape himself. Figure 5 displays the manual drawing of the bar and the result.

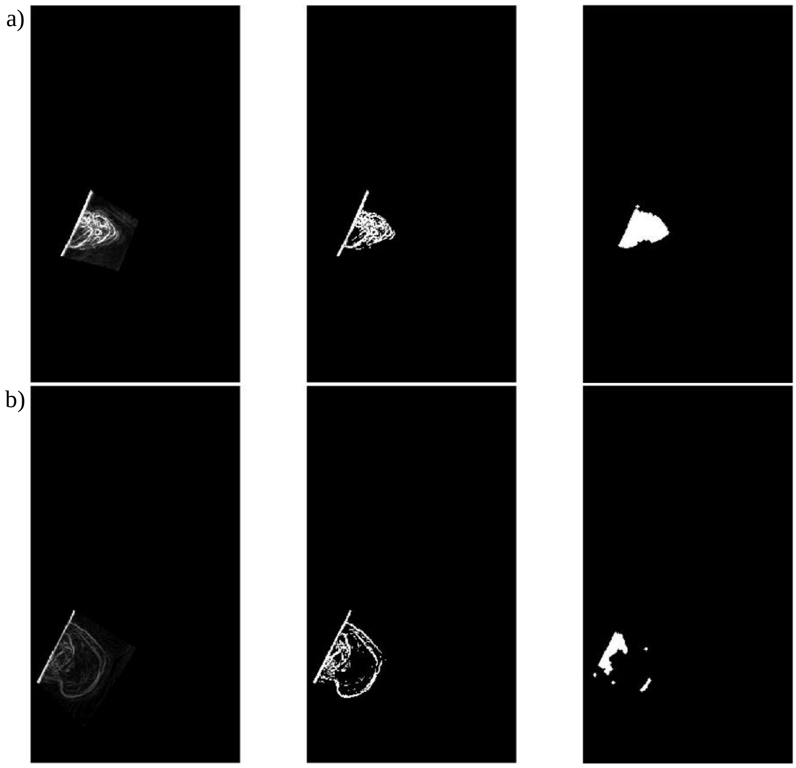


Figure 4. Gradient image (first column), binary image (second column) and final image (third column) after CLAHE for a) image 2 and c) image 3.

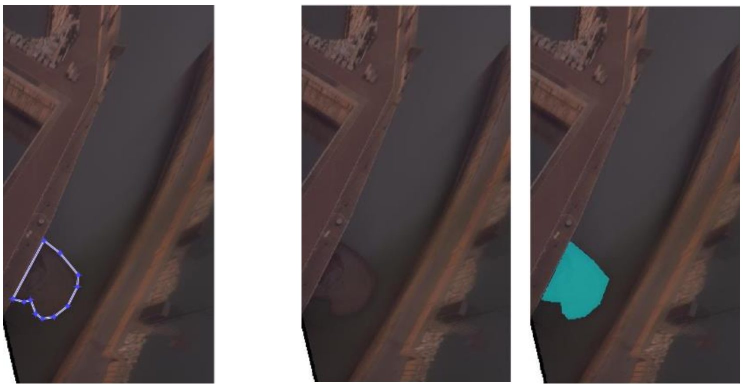


Figure 5. The first column displays the drawing of the contour of the bar; the second and third columns represent the result shown to the user by the code.

Even in the case of manual drawing, the user can either discard the result or save the data.

Once the result of the bar detection is accepted, the code computes and stores the *Area* of the bar, by counting the number of pixels in the detected region, and the coordinates of its *Centroid*. Moreover, the code stores the *Perimeter*, i.e. the coordinates of pixels included in the perimeter, and the *Pixel List*, i.e. the coordinates of pixels included in the area.

The output of the code is, for each analysed image, a file containing:

- ***Name***: the name of the image;
- ***Type***: the classification of the image;
- ***Area***: number of pixels included in the detected area;
- ***Centroid***: coordinates of the centre of mass of the detected area;
- ***Perimeter***: coordinates of pixels included in the perimeter of the detected area;
- ***Pixel List***: coordinates of pixels included in the detected area.

The code provides a semi-automatic and rapid procedure to identify emerged deposits from images and compute their geometrical characteristics. Using a suitable coordinate transformation, it is possible to refer the coordinates of the detected deposit to a world reference system. Thanks to the checking of the results by the user, this procedure could be used to analyse the migration of an emerged deposit with a quite good accuracy.

More information about the code can be found in: “Long-term evolution of an inner bar at the mouth of a microtidal river”, A. Baldoni, E. Perugini, L. Soldini, J. Calantoni & M. Brocchini. (article in progress)