Costanza 1.0 Userguide

M. Green, P. Krupinski, P. Melke, P. Sahlin, H. Jönsson

December 10, 2008

1 Introduction

This document describes how to use COnfocal STack ANalyZer Application (Costanza), an ImageJ[1] plugin for segmentation and analyzing stacks of image data. Such data come very often from fluorescent microscopy and this is where Costanza should prove to be most useful. The main purpose of Costanza is to segment nuclei marked cells in three dimensions, making available quantitative measures such as positions, sizes, and average intensities of GFP markers for the segmented compartments. However since a user is given the control on the parameters used by Costanza filters and image processors, they may be tweaked and combined freely to give good results also for other image processing tasks. The main algorithm is independent of intensity thresholds, allowing for segmentation of data of varying intensity.

Costanza assumes a greyscale stack of images as input, and return the result both as cells/intensities marked in images and as an ImageJ data list.

2 Installation instructions

In order to use Costanza you must have ImageJ (`http://rsb.info.nih.gov/ij`) installed on your system. Download the file Costanza.zip from `http://www.thep.lu.se/costanza` . Unzip the Costanza archive in the plugins directory of your ImageJ installation. Next time you start ImageJ the plugin will show up as Costanza in the Plugins drop down menu.

3 Algorithms

The algorithms provided in Costanza can be divided into the main segmenting algorithm, preprocessing, and post-processing. An important feature

of Costanza is the fact that all the algorithms operate in three dimensions instead of performing tasks on the set of two dimensional images. This increases the complexity and times of calculations but contributes to much more reliable results of segmentation and analysis, and is an advantage of Costanza in comparison to other tools. The user has the possibility to use ImageJ x, y, and z scales for the stack or define custom ones. The results are thus displayed in real length units.

3.1 Steepest gradient ascent for segmentation

This is the main algorithm for Costanza. It starts at each voxel in the stack and tries to find local intensity maximum by "walking" uphill in intensity landscape among its neighbors. All paths leading to the same maximum are saved to the basin of attractor of this maximum, and represent a compartment. If more than one neighbor has an intensity value higher then the current voxel, a step is performed to the neighbor with highest

$$\frac{\Delta I}{\Delta \mathbf{x}} = \frac{I_{neigh} - I_{current}}{\mathbf{x}_{neigh} - \mathbf{x}_{current}} \tag{1}$$

where $\mathbf{x}_{current}(\mathbf{x}_{neigh})$ is the current (neighbor) position, and $\mathbf{I}_{current}(\mathbf{I}_{neigh})$ are the respective intensities. The neighborhood of the voxel can be chosen either as the six closest voxels, *i.e.* $(x \pm 1, y, z)$, $(x, y \pm 1, z)$, and $(x, y, z \pm 1)$ or as all the twenty six surrounding voxels $\{(\{x, x \pm 1\}, \{y, y \pm 1\}, \{z, z \pm 1\})\} - (x, y, z)$.

Optionally the treatment of intensity plateaus as one maximum can be turned on. This helps with analysis of data where large regions of identical intensities are present by choosing the maximum voxel for each of such regions to be the centroid of the plateau.

3.2 Preprocessing

3.2.1 Intensity inversion

As the gradient ascent algorithm assumes bright objects in a dark background as input, the stack intensity should be inverted if the objects of interest are dark in a bright surrounding. This algorithm replaces each voxel intensity I with its inverted value $I_{max} - I$, where I_{max} is the maximal value allowed in the greyscale images. This algorithm does not require any user provided parameters.

3.2.2 Background extraction by thresholding

It may be beneficial to remove some voxels before the application of the segmentation algorithm. This should be done both for removing uninteresting regions of the stack and for speeding up the processing. The background can be extracted via a threshold filter that assigns all voxels with intensity I less than a threshold value $I_{threshold}$ to the background. The background voxels will be subsequently excluded from consideration by all the other algorithms. The user has to provide a value for the parameter $I_{threshold}$.

3.2.3 Denoising filters

Segmentation by gradient ascent works the best if the intensity landscape is smooth and void from bogus maxima which can be present due to the noise. Elimination of the noise can be done by different filters in ImageJ, but Costanza provides two 3D filters for smoothing the data. As previously noted 3D filtering is preferred by the segmentation algorithm in Costanza. Thus even if a set of 2D filters was previously applied to the images in the stack it may be advantageous to apply Costanza's 3D mean filter before the analysis.

Mean filter uses a spherical filtering kernel and replaces each voxel intensity I by an average calculated from all voxels with a center positioned within a radius R_{max} measured in real length (*i.e.* taking the scale in each dimension into account). Since it can be more beneficial to run the filter multiple times with smaller R_{max}, rather than to increase R_{max}, the user should provide the two parameters, R_{max} and the number of times to run the filtering.

Costanza provides also 3D median filter as an optional noise reduction processor. In principle median filter acts in very similar manner to the mean filter with the difference that it assigns to the given pixel the value of intensity median of the pixels in the surrounding kernel rather than the intensity mean. Median filter tends to intensify differences between bright and dark regions and may be particularly useful in task like edge detection. The median filter also takes as input parameters radius of the filtering kernel R_{max} and the number of repetitions.

3.3 Postprocessing

After gradient ascent is applied the segmentation takes place, positions of the maxima for each basin of attraction (BOA) are identified and mean intensities for each BOA are computed. As the next step two post-processing options

are available.

3.3.1 BOA removal

Since the gradient ascent algorithm is insensitive to absolute intensity values, it may happen that it finds compartments in the noisy background. Thus we supply a post-processor for removing these false positives by using thresholds for minimal size of and minimal intensity in the extracted compartments. The user should provide a minimal size threshold S_{min} given in volume units, and minimal intensity I_{min}.

3.3.2 BOA merging

Local intensity fluctuations may cause that the bright regions corresponding to single cell in reality are interpreted as several compartments by the gradient ascent algorithm. The user has the possibility to merge such compartments, by setting a minimal distance between maxima R_{min}. Compartments with the distance between centers below R_{min} will be merged into a single compartment. This algorithm runs recursively.

3.4 Data collection

The algorithm results are provided in the form of marked cells/intensities in images as well as the actual quantitative numbers as a ImageJ data list. The options of which data is extracted can be found in Section 4.3.1.

4 User interface

4.1 Main

Costazna plugin interface consist of one window which displays a menu bar on the top, the tabbed panel which gives access to different program options and a button panel on the bottom which is used to start the analysis. When user presses the **Start analysis** button the current active stack and selected processors and parameter values will be used to perform the task.

4.2 Menu bar

The Costanza menu bar has two items: **File** and **Help**. **File** submenu can be used to save and load a configuration file where all the options and processors are stored. This helps a user to quickly store and retrieve previous settings

which they found useful for particular task. The last option is **Quit** which ends Costanza session. If user exits Costaza through the **Quit** the last used configuration will be saved in the file *last.ccfg* and automatically restored upon next start of the plugin. If this file is not available Costanza tries to open *default.ccfg* file and lastly uses hard coded set of parameters. By default the configuration files are stored in *plugins/Costanza/* directory in the main ImageJ directory.

4.3 Option tabs

Option tabs are used to set all features of Costanza. They are grouped into four different tabs (Fig. 1).

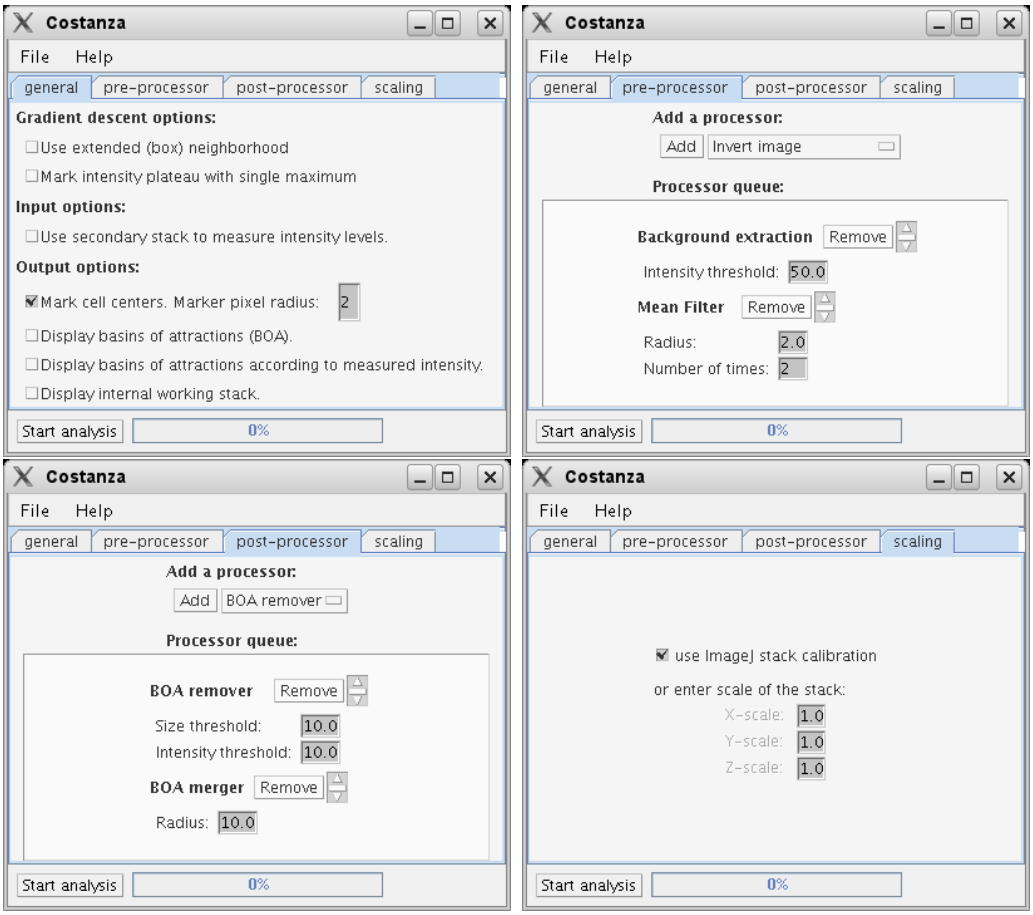


Figure 1: Graphical user interface for Costanza.

4.3.1 General options tab

This tab allows for setting options for gradient ascent algorithm as well as input and output options. The checkbox **Use extended (box) neighborhood** controls properties of the neighborhood of each pixel during segmentation by gradient ascent algorithm. When the box is checked 26 pixel (box like) neighborhood will be used. Otherwise 6 pixel neighborhood (cross like) will be used. The user has the option to treat intensity plateaus as a single BOA by checking **Mark intensity plateau with single maximum** checkbox. The active stack is automatically chosen for segmentation input. If an additional stack is to be used for measuring other intensities in previously found compartments, the box **Use secondary stack to measure intensity levels** has to be checked.

As default, the segmentation results are provided in a table displayed by ImageJ upon competition of analysis. Additionally other types of graphical output can be selected:

- Cell centers can be marked with red boxes of desired radius in pixels.
- Basin of Attractions (BOAs), attributed to the different cell centers can be colored with random colors.
- BOAs can be colored and displayed according to measured intensities. The color schema is from blue via green yellow to red.
- The working stack (after applying all the pre-processing options) can be displayed. This can be useful to see how the pre-processing has changed the original images.

4.3.2 Pre-processing options tab

Here different pre-processing steps can be chosen, added and removed by the user. Processors in the queue will be applied to the stack in top to bottom order. Order of the processors can be adjusted by up and down arrows to the right of the **Remove** button. The alternatives at the moment are:

- Invert - inverts the pixels in the stack. This is needed if the objects are dark in a brighter background, *e.g.* if bright membranes are marking cells.
- Background threshold - this will put all pixels below a threshold intensity to the background, and hence they will not be processed by other algorithms and the segmentation in particular.

- Mean Filter - this will reduce noise in the images by applying a 3D mean filter with provided radius. As it is often useful to apply this more than once, repeat option is also provided for the user.
- Median Filter - another alternative to reduce image noise is to apply 3D median filter. Radius and repeat options are available to set by the user as in previous case.

4.3.3 Post-processing options tab

After the gradient ascent algorithm is applied for segmentation, it might be useful to do some post-processing. As previously order of application of processors is from top to bottom. The available algorithms that can be chosen, added and removed are:

- BOA remover, where BOA regions attributed to compartments can be removed due to being too small or having too low intensity.
- BOA merger, where compartments with centers too close to each other can be merged into a single compartment.

4.3.4 Scaling options tab

By default Costanza will use stack scaling obtained from ImageJ. A user can change this by setting the approporiate values for the scale in x, y and z directions.

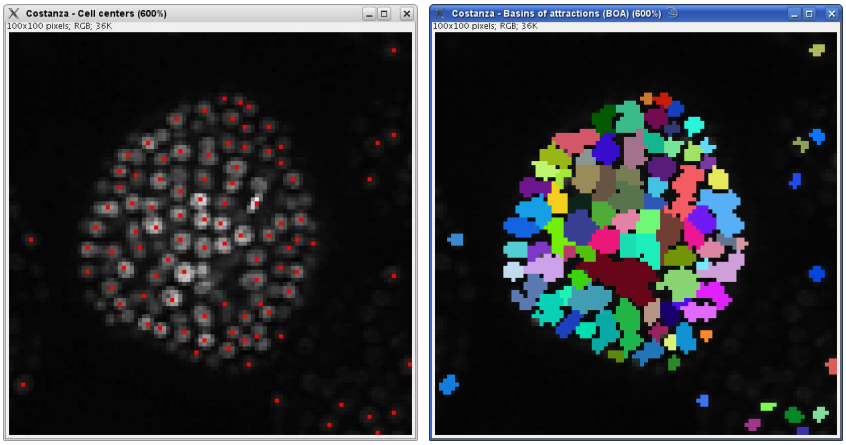
5 Examples

The performance of the Costanza application is dependent on the parameter values given to the different algorithms used for image processing. These values will be most likely different for different tasks. Getting satisfactory results in each case may require some experimenting and tweaking different parameters. Here we provide some examples of applications. The used parameter values are also presented to guide users into setting reasonable parameter values.

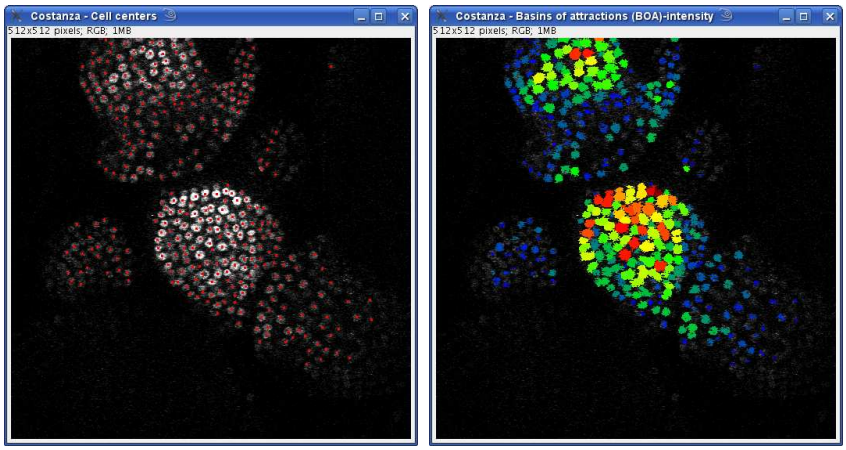
5.1 SAM nuclei data in 3D

This data set represents nuclei marked cells in the *Arabidopsis* shoot apical meristem. The data set consists of a stack of 20 images. For simplicity and readability reasons we present results of analysis of single 2D slice from the

stack but the whole 3D stack can be processed equaly well. The segmentation is run using two different resolutions, and the result is shown in Fig. 2. Parameters used for these two runs are given in Table 1, and should represent reasonable values also for 3D stacks.



(a) 100x100 pixels image



(b) 512x512 pixels image

Figure 2: Result shown for images of nuclei data. We thank M. Heisler for the original images.

5.2 SAM membrane data in 2D

This data set represents membrane marked cells in the *Arabidopsis* shoot apical meristem. The data set consists of single 2D images. The segmentation is run using two different resolutions, and the result is shown in Fig. 3.

Parameter	Small	Large
Bg threshold	20	30
MeanFilter R	1.0	2.0
MeanFilter num	2	2
Remove Int Th.	6	10
Remove Size Th.	6	10
Merger R	3	7
xy-scale	1	1
z-scale	1	1

Table 1: Nuclei extraction in 2D and 3D.

Parameter	WUS	PIN1
Bg threshold	1	1
MeanFilter R	5.0	10.0
MeanFilter num	2	2
Remove Int Th.	10	10
Remove Size Th.	10	10
Merger R	5	10
xy-scale	1	1
z-scale	-	-

Table 2: Cell extraction in 2D membrane data.

Parameters used for these two segmentations are given in Table 2.

References

[1] M. Abramoff, P. Magelhaes, and S. Ram. Image processing with imagej. *Biophotonics Int*, 11:36–42, 2004.

[2] B. S. E. M. H Jönsson, MG Heisler and E. Mjolsness. An auxin-driven polarized transport model for phyllotaxis. *PNAS*, 103:1633–1638, 2006.

[3] G. R. V. A. V. G. B. S. E. M. H Jönsson, M Heisler and E. Meyerowitz. Modeling the organization of the wuschel expression domain in the shoot apical meristem. *Bioinformatics*, 21:i232–i240, 2005.

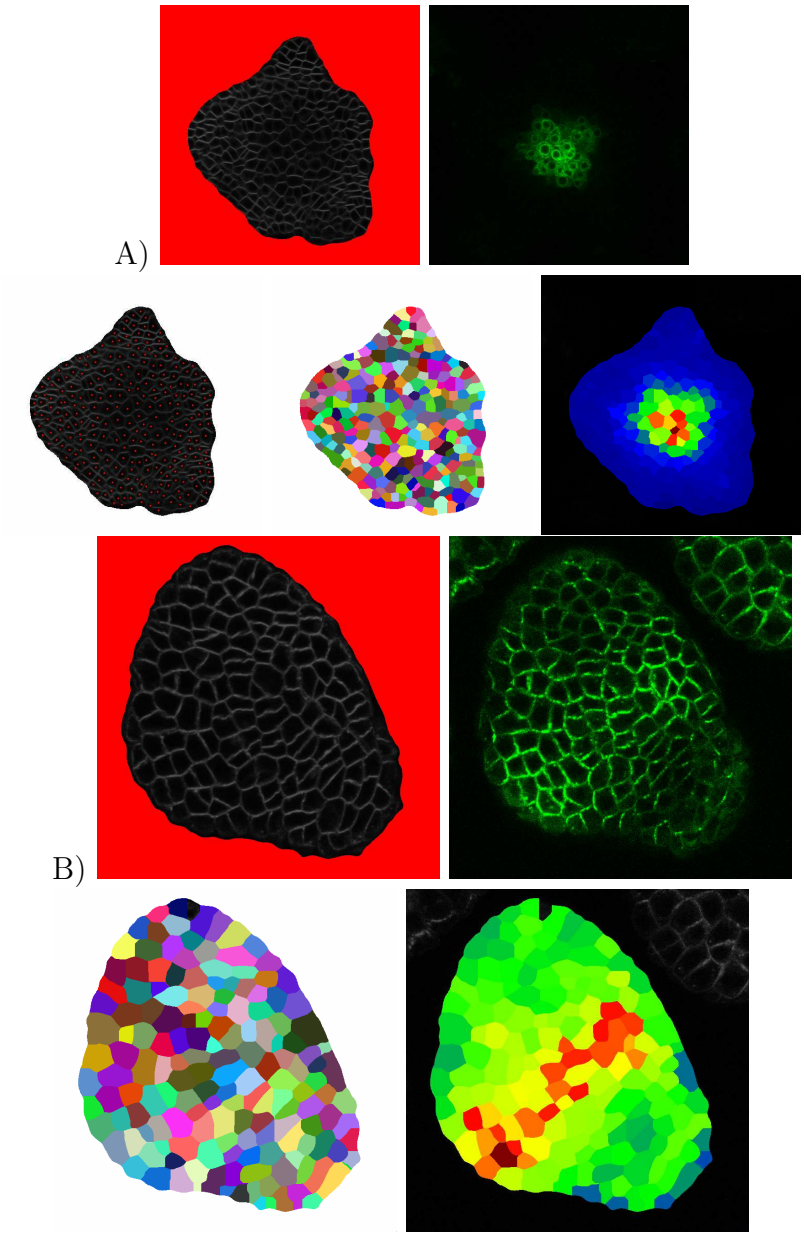


Figure 3: Result shown for images of membrane data, where segmentation and GFP quantification is displayed. The images are similar to the ones presented in [3] (A), and [2]. (B).