Getting Started With CellOrganizer v2.0

cellorganizer.org

About this document

This document covers the minimum materials required for learning model distributions of cellular organization and synthesizing images from those distributions. For more in-depth examples, please refer to the **demos** and the **templates** directories contained within the CellOrganizer distribution.

Step 0

- Download the latest or desired copy of CellOrganizer from
  `http://cellorganizer.org/Downloads`
- To start using CellOrganizer, start a Matlab session and change directory to the location of CellOrganizer and run setup.m

```
cd( '/path/to/cellorganizer' );
setup;
```

Model Training

Required Materials

A copy of CellOrganizer
A set of fluorescently tagged images, each containing a single cell.

Main Function

```
success = img2slml( dimensionality, img_dna_wildcard,
img_cell_wildcard, img_protein_wildcard, param)
```

Recommended Setup

1) Images

Cell images must satisfy the following requirements:
1) Either a single cell per image, or a binary-valued mask must be provided.
2) Images must be wildcard accessible (ex. 'img*-channel1.tif' could be the wildcard for all nuclear images, 'img*-channel2.tif' could be the wildcard for all cell images
3) 3D images must be contained within a single multi-stack TIF per channel (ex. image1-channel1.tif, image1-channel2.tif, image2-channel1.tif, image2-channel2.tif, …)

Example:

```
img_dna_wildcard =  '/your_image_directory/img*-channel1.tif';
img_cell_wildcard = '/your_image_directory/img*-channel2.tif';
img_prot_wildcard = '/your_image_directory/img*-channel3.tif';
```

3) Parameters

The parameter structure defines the settings of how different components of the model are learned.

param.model.trainflag - Specifies which type of model is being trained. One of the following strings:

'nuclear'	- trains a nuclear model
'framework'	- trains a nuclear and cell shape model
'all'	- trains a nuclear, cell shape, and protein model

Please visit http://cellorganizer.org/Downloads/v2.0/parameters.pdf for a detailed description of the param structure.

param.model.resolution - A 1x3 vector specifying voxel dimensions in um

Training a model of cellular organization

The following code is the bare minimum to train a 3D model:

```
dimensionality = '3D';

img_dna_wildcard =  '/your_image_directory/img*-channel1.tif';
img_cell_wildcard = '/your_image_directory/img*-channel2.tif';
img_prot_wildcard = '/your_image_directory/img*-channel3.tif';

param.model.trainflag = 'all';
param.model.filename = 'myModel.mat'
param.model.resolution =  [0.049, 0.049, 0.200];

success = img2slml( dimensionality, img_dna_wildcard,
img_cell_wildcard, img_protein_wildcard, param)
```

Advanced Demos

If you are interested in learning about training models you can start by investigating the demos. All the demos included with the source code use images from the Murphy Lab dataset collection.

demo2D01 - Trains a 2D generative model of protein location using one of the four patterns.
demo3D11 - Trains a generative model of the cell framework using the four patterns in the HeLa dataset.
demo3D12 - Trains a generative model of the cell framework and protein pattern using one of the four patterns in the HeLa dataset.
demo3D20 - Trains a generative model of the cell framework using diffeomorphic modeling with a subset of the HeLa dataset.

Synthesis

Once a CellOrganizer model has been trained CellOrganizer can be used to synthesize novel instances of the patterns learned from bioimaging data. CellOrganizer supports the synthesis of nuclei, cells, and protein patterns. A user may also choose to synthesize any number of protein patterns within a single cell and nuclear "framework". Currently CellOrganzier 2.0 supports the training and synthesis for vesicular and network type protein patterns.

Required Materials

A copy of CellOrganizer
At least one CellOrganizer model - see **Model Training** above

1) Model

CellOrganizer 2.0 synthesis supports models saved in .mat format.

Supported outputs:

1) TIFF images - Multi-stack TIFF images. In 2D each slice corresponds to a channel. In 3D each image corresponds to a channel.

2) Indexed TIFF image - Similar to 1 above, but here channels are given indexed values starting with one resulting in a single 2D image for 2D synthesis or a single 3D image for 3D synthesis.

3) Obj mesh files - Mesh representations for synthesized geometries. These can be imported to CellBlender(http://www.mcell.psc.edu/) using the blender import function. It is intended for complex shapes such as cell and nuclear bodies. Vesicles should utilize the SBML-Spatial Constructed Solid Geometry(CSG) primitives to avoid artifacts from meshing.

4) SBML-Spatial - Creates an XML file for SBML-Spatial v0.82a(http://sbml.org/Documents/Specifications/SBML_Level_3/Packages/spatial).

Synthesis related functions:

`slml2img`

This is the main function for synthesizing instances from CellOrganizer models.
Example call: `success = slml2img(models,param);`
Inputs: models - Cell array of paths to the CellOrganizer models you wish to synthesize from.
param - Param structure defining synthesis options. See Please visit http://cellorganizer.org/Downloads/v2.0/parameters.pdf for a detailed description of the param structure.
Outputs: success - Boolean flag indicating the success or failure of CellOrganizer to synthesize an image from the given inputs.

`syn2surfaceplot`

This function creates a surface plot for a set of TIFF images passed to it.

Example call:

```
syn2surfaceplot(tiffDirectory,colors,viewangles,alphaval);
```

Inputs: tiffDirectory - String containing the path to the TIFF images that will be displayed
colors - Cell array containing a list of valid colors for each channel
viewangles - Vector defining the viewing angle
alphaval - Double 0-1 value defining the transparency of the objects

`im2projection`

This function creates a projection along each dimension of a 3D image.

Example call: `img2D = im2projection(img);`

Inputs: img - 3D image array

Outputs: img2D - 2D image containing the three projected dimensions

`img2tif`

This function saves an image as a TIFF file.

Example call: `img2tif(img,savePath);`

Inputs: img - 2D or 3D image array
savePath - String specifying where to save the image (e.g. ‘./myimg.tif’)

`syn2blender`

This function saves an image as a OBJ file.

Example call: `syn2blender(imgDir,saveDir,param);`

Inputs: imgDir - String specifying a 3D image path (e.g. ‘./mycells/’)
savePath - String specifying where to save the image (e.g. ‘./mymeshes/’)
param - Struct specifying mesh options (see demo3D13 for details)

`syn2projection`

This function saves a 2D projection of a 3D image

Example call: `syn2blender(imgDir,saveDir,param);`

Inputs: imgDir - String specifying a 3D image path (e.g. ‘./mycells/’)
savePath - String specifying where to save the image (e.g. ‘./mymeshes/’)
param - Struct specifying options (see demo3D14 for details)

Advanced Demos

If you are interested in learning about training models you can start by investigating the demos. All the demos included with the source code use images from the Murphy Lab dataset collection.

2D

demo2D00 - This function demonstrates the synthesis of a 2D nucleus, cell shape and each 2D protein pattern included in the CellOrganizer 2.0 distribution(endosome,lysosome,mitochondrion,nucleolus). Outputs will be a single multistack TIFF image where each stack in the file corresponds to a “channel” (e.g. cell shape, nucleus, endosome, lysosome,...).Calls*: img2slml*

3D

demo3D00 - This function demonstrates the synthesis of a nucleus and cell containing lysosomes. Outputs will be multistack TIFF images with each image corresponding to a "channel". Calls*: img2slml*

demo3D01- This function demonstrates the synthesis of a nucleus and cell containing all the vesicular patterns in the CellOrganizerv2.0 release. Outputs will be multistack TIFF images with each image corresponding to a "channel". Calls*: img2slml*

demo3D02 - This function demonstrates using syn2surfaceplot to display the images generated using demo3D00. This demo therefore should be run after demo3D00. Outputs will be a displayed surface plot. Calls*: syn2surfaceplot*

demo3D03 -This function is identical to demo3D01 with the exception of param.sampling.method = 'sampled' and param.sampling.density = 75 instead of param.sampling.method = 'disc'. Calls*: img2slml*

demo3D04 - This function synthesizes one 3D image from microtubule model. Outputs will be multistack TIFF images with each image corresponding to a "channel". Calls*: img2slml*

demo3D05 - This function demonstrates the synthesis of a nucleus and cell containing all the patterns in the CellOrganizerv2.0 release and sampling set to 'sampled'. Outputs will be multistack TIFF images with each image corresponding to a "channel". Calls*: img2slml*

demo3D06 - This function demonstrates the synthesis of one 3D instance from all object models and a microtubule model and convolution with a point-spread function. Calls*: img2slml*

demo3D07 - This function demonstrates the synthesis of one 3D instance from all object models and a microtubule model with sampling set to 'sampled', a density of 25 and convolution with a point-spread function. Outputs will be multistack TIFF images with each image corresponding to a "channel". Calls*: img2slml*

demo3D08 - This function demonstrates the synthesis of a nucleus and cell containing all the patterns in the CellOrganizerv2.0 using the 'indexedimage' output. Outputs will be a multistack TIFF image with each numerical value corresponding to a "channel". Calls*: img2slml*

demo3D09 - This function is identical to demo3D00 but additionally creates a projection of the 3D output image for cell shape and saves it as a TIFF image. Calls*: slml2img,im2projection,img2tif*

demo3D10 - This function demonstrates the creation of OBJ mesh files for a cell, nucleus and lysosomes. Calls*: slml2img*

demo3D13 - This function demonstrates the creation of OBJ mesh files for images already synthesized using demo3D03. Calls*: syn2blender*

demo3D14 - This creates projections using the outputs from demo3D01. As such demo3D01 should be run first. Outputs will be a TIFF image. Calls*: syn2projection*

demo3D15 - Synthesizes 1 image using a transferrin model for the protein and a diffeomorphic model for the nuclear and cell shape. Calls: *slml2img*

demo3D16 - This function demonstrates using raw image segmentations to provide a framework for a protein pattern model. Calls: *slml2img*

demo3D17 - This method shows how to input an image to CellOrganizer to show the user they can use their own binary images from raw experimental data they can use to synthesize protein patterns.

demo3DDynamic - This function demonstrates the synthesis of a shape space walk using the

diffeomorphic model to generate frameworks. Calls: *sbml2img*
demo3DPrimitives - This function demonstrates the creation of an SBML-Spatial file containing specifications for geometric primitives(CSG objects) for the endosomal pattern and parametric objects for the cell and nuclear shape. Calls: *sbml2img*
demo3DMultiresSynth - Synthesizes images at three different resolutions.
demo3DObjectAvoidance - Synthesizes an image with non-intersecting endosomal objects. Calls: *sbml2img*