APR 01, 2024

OPEN ACCESS

CODA (part 6): register the nuclear coordinates and construct 3D cell matrix | HuBMAP | JHU-TMC

Kyu Sang Han[1], Pei-Hsun Wu[1], Joel Sunshine[2], Ashley Kiemen[2], Sashank Reddy[2], Denis Wirtz[1,2]

[1]Johns Hopkins University; [2]Johns Hopkins Medicine

Human BioMolecular Atlas Program (HuBMAP) Method Development Community

TMC - Johns Hopkins University

Kyu Sang Han
Johns Hopkins University

ABSTRACT

Using the registration transformations calculated in low-resolution (as described CODA (part 2): calculate registration on low-resolution tissue images), register the nuclear coordinates generated in CODA (part 5): nuclear coordinate generation

DOI:
dx.doi.org/10.17504/protocols.io.n2bvjnrnxgk5/v1

Protocol Citation: Kyu Sang Han, Pei-Hsun Wu, Joel Sunshine, Ashley Kiemen, Sashank Reddy, Denis Wirtz 2024. CODA (part 6): register the nuclear coordinates and construct 3D cell matrix | HuBMAP | JHU-TMC. **protocols.io** https://dx.doi.org/10.17504/protocols.io.n2bvjnrnxgk5/v1

MANUSCRIPT CITATION:
Kiemen, A.L., Braxton, A.M., Grahn, M.P. *et al.* CODA: quantitative 3D reconstruction of large tissues at cellular resolution. *Nat Methods* **19**, 1490–1499 (2022). https://doi.org/10.1038/s41592-022-01650-9

Protocol status: Working
We use this protocol and it's working

Created: Mar 28, 2024

Last Modified: Apr 01, 2024

PROTOCOL integer ID: 97515

Keywords: CODA, deeplearning, machinelearning, semanticsegmentation

Funders Acknowledgement:
Institute of Arthritis and Musculoskeletal and Skin Diseases
Grant ID: U54AR081774
National Cancer Institute
Grant ID: U54CA143868

register the nuclear coordinates

1 As input, this function needs the path to the low-resolution images that were registered in CODA-part2 (pth1x), the path containing the coordinates calculated in CODA-part4 (pthcoords), and the scale factor between these images. For cell detection performed in 10x and registration calculated in 1x,

scale=10;
register_cell_coordinates(pth1x,pthcoords,scale);

*if the yellow dots do not line up over the tissue, there is a problem in your code.

2 This function will create a subfolder inside pthcoords named 'cell_coordinates_registered.' Inside that subfolder will be a mat file corresponding to each tif image. Inside each mat file will be a variable named 'xy' (unregistered cell coordinates), 'xyg' (globally registered cell coordinates), and 'xye' (elastically registered

cell coordinates), all saved at the same low-resolution as the original registration images used in CODA-part2.

pthcoordsE=[pthcoords,'cell_coordinates_registered'];

construct 3D cell matrix

3 Next, create **volcell**, a matrix containing the nuclear labels in a volumetric matrix the same size and resolution of the tissue matrix constructed in CODA-part4. To call this function, you need to define the folder containing the registered, classified images created in CODA-part4, the folder containing the registered cell coordinates, the folder containing the tissue matrix created in CODA-part4, and the scale between the high-resolution, classified images and the high-resolution images used for cell detection (this is probably 1).

4 Define the folders containing the registered, classified images, the registered nuclear coordinates, and the tissue volume matrix:

pthclassifiedE=[pthclassified,'registeredE'];
pthcoordsE=[pthcoords,'cell_coordinates_registered'];
pthvolume=[pth,'lung_data'];

5 Next, define the scale between the high-resolution images used for tissue classification in CODA-part3 and the high-resolution images used for cell detection in CODA-part5. If the classification and cell detection were applied to the same resolution images (this is likely), the scale is 1.

scale=1;

6 Finally, nwhite is the whitespace class from the deep learning model. Here, let's again use nwhite=6 (the background class number from the sample lung model.

nwhite=6;

7 Now, call the make_volcell function:

build_cell_volume(pthclassifiedE,pthcoordsE,pthvolume,scale,nwhite)

This function will create a 3D matrix containing nuclear coordinates. The matrix will be at the same resolution as the tissue volume matrix, and automatically crop out the same background space that was manually selected for removal in CODA-part4.

8 To validate that the function creates a variable **volcell** that is the same resolution and crop as the tissue matrix **vol**, the function will display two images.

1. the function will display an overlay of the center image from vol and **volcell**.

2. the function will display an overlay of the z-projections of **vol** and **volcell**.

If either of these images do not appear overlayed (you see two separate, unaligned tissues), there was a problem with the construction of either **vol** or **volcell**.

9 Beyond the scope of this guide, consider **volshow** or **patch**

Notes on quantification

10 Primarily beyond the scope of this guide, as most spatial calculations must are custom to the biological question. Some basic considerations are in the following sections

Smooth data

11 To smooth your data. Your segmentation algorithm and 3D volumetric matrix will likely contain small false positive and false negative data. Try smoothing your volumetric data with the following.

vol = volumetric matrix of size [m n z]
containing 6 labels for 5 tissue structures plus background

ws = 6; % background class for volumetric matrix

to lightly smooth the tissue labels inside vol:
volS=vol; %

create a smoothing variable

```
for  b=1:max(vol(:))
    tmp=vol==b;
    tmp=imclose(tmp,strel('sphere',2));
% morphological closing to fill small holes
    tmp=imopen(tmp,strel('sphere',1));
% morphological opening to remove small noise
```

```
    tmp=bwareaopen(tmp,500);
% delete 3D objects that are fewer than 500 voxels;
    volS(volS==b)=ws;
    volS(tmp==1)=b;
end
```

Note: be VERY CAUTIOUS using **imopen** to smooth thin structures (glands and vasculature), as **imopen** may eliminate “true positive” thin walls. Consider skipping **imopen** when you smooth

Quantify volume

12 To quantify volumes. volume of tissue class 2 (use **vol** or your smoothed **volS**):

```
sxy=4; %
resolution of ‘vol’ matrix in xy in units of micron / pixel
sz=4; %
distance between serial histological sections
volume_type_2=sum(vol(:)==2)*sx*sx*sz;
```

% volume of type 2 in units of micron^3

```
volume_type_2=volume_type_2/(10^9);
```

% volume of type 2 in units of mm^3

Quantify cellularity

13 To quantify cellularity. number of cells in class 2 (use **vol** or your smoothed **volS**):

```
tmp=double(volcell).*double(vol==2);
cell_type_2=sum(tmp(:));
```

And more

14 For more complex quantifications:

Consider the functions **bwdist**, **regionprops3**, and custom functions you create.