Manual to Object detection, Registration and Segmentation

Author: Leon R. Watzema
leonwatzema@gmail.com
12 March 2021

Prerequisites

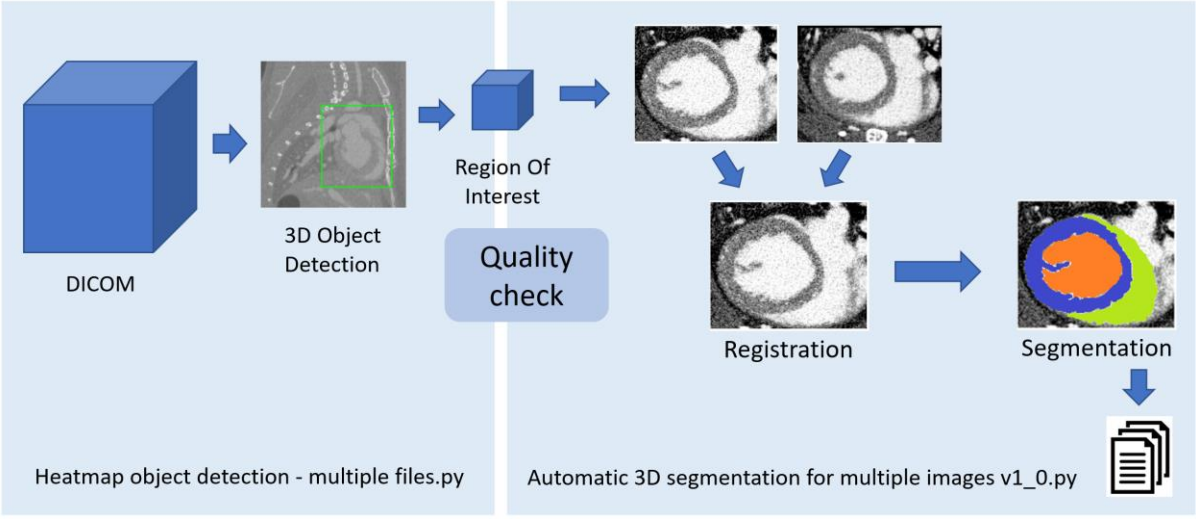
- A recent version of Python. For building and testing Python 3.8 was used
- SimpleITK can be installed by entering `pip install SimpleITK` from within Python

Workflow overview

The objective is to automatically measure certain features of a Region Of Interest (ROI) within a CT scan. To do this we must first crop the input image to the desired ROI. These cropped versions can be used as input for registration. The transformation performed for registration can be applied to a labelled image. The resulting transformed label image should match the cropped input as close as possible. Finally we can measure the segments within the transformed label image.

Three Python scripts were made to perform these tasks:

- Heatmap object detection - multiple files.py
- Heatmap object detection - single file.py (for testing)
- Automatic 3D segmentation for multiple images v1_0.py



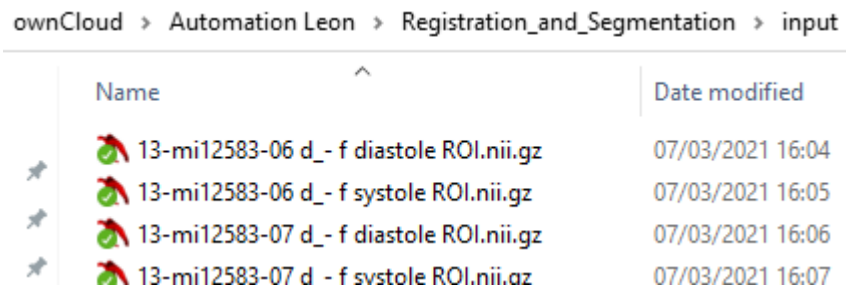
More details about the general workings and limitations of these programs can be found in HF-image 20210312.pdf

Object detection

The easiest way is to run 'Heatmap object detection - multiple files.py' from within a Jupyter notebook. Before executing we must specify the right folders:

```
19 dirInput = 'D:\\ownCloud\\uCT data DICOM\\Ercc1\\6 wks\\'
20 dirOutput = 'D:\\ownCloud\\Automation Leon\\Registration_and_Segmentation\\input\\'
21 dirRef = 'D:\\ownCloud\\Automation Leon\\Heatmap_object_detection\\reference_imgs\\'
22 dirMeta = 'D:\\ownCloud\\Automation Leon\\Registration_and_Segmentation\\metadata\\'
23 voxelsize = [0.16, 0.16, 0.16] # 0.4 to 0.16 --> resize 512 to 128
```

- dirInput – the folder that contains the DICOM files we want to crop. Note that we do not modify the input files in any way. The cropped versions are stored in dirOutput.
- dirOutput – stores the cropped ROI's DICOMs in NIfTI file format with the ' ROI.nii.gz' extension. These will be used as input for registration and segmentation.

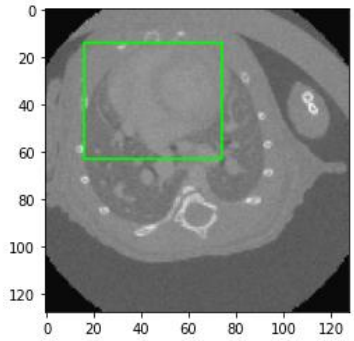


- dirRef – contains the reference images used for object detection. These require no attention
- dirMeta – stores metadata gathered from the DICOM header in a ROI_metadata.csv file. This data can be combined with the volume metrics generated by Registration and Segmentation later on.

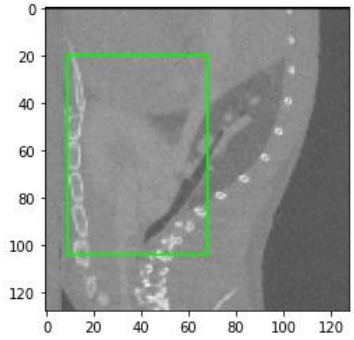
Single file object detection

The ‘Heatmap object detection - single file.py’ version is mainly used for testing with different parameters. This version will also display the detection results on screen:

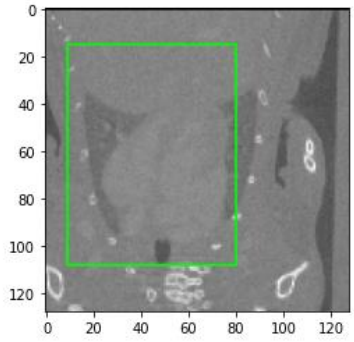
```
Start: 2021-03-08 20:40:57
00:00:02 - loading dicom done
00:00:16 - resampling done
00:00:41 - object detection done

Accuracy        4.6600691477457685
Center x        45
Center y        39
Center z        62
Axial
```




```
Sagittal
```




```
Coronal
```




```
00:00:45 - cropping done
00:01:01 - all done
```

Note: one DICOM file requires at least 60 seconds to process on a reasonably powerful desktop computer.

Registration and segmentation

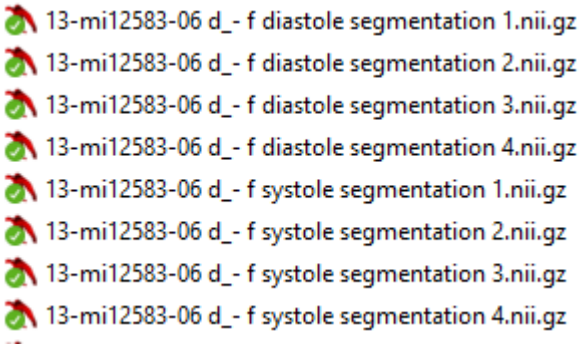
For this we use ‘Automatic 3D segmentation for multiple images v1_0.py’. Before executing we must specify the right folders:

```
dirInput = r"D:\\ownCloud\\Automation Leon\\Registration_and_Segmentation\\input\\"
dirMoving = r"D:\\ownCloud\\Automation Leon\\Registration_and_Segmentation\\moving\\"
dirOutput = r"D:\\ownCloud\\Automation Leon\\Registration_and_Segmentation\\output\\"
fileOutput = 'HFimage_segmentation.csv'
fileMetadata = 'ROI_metadata.csv'
marker = ' ROI.nii.gz'
```

- dirInput – the folder that contains the cropped ROI files we want to register and segment. Note that we do not modify the input files in any way.
- dirMoving – contains the moving and label images used for Registration and Segmentation. These require no attention unless one wants to use different moving images.
- dirOutput – stores the transformed and segmented input images in NIfTI file format. For each moving image a segmentation file will be generated. The same sort order as in the code is used:

  Diastole
  movingImage2_crop.nii.gz → … segmentation 1.nii.gz
  movingImage3_crop.nii.gz → … segmentation 2.nii.gz
  101800-diastole_crop.nii.gz → … segmentation 3.nii.gz
  110703-diastole_crop.nii.gz → … segmentation 4.nii.gz

  Systole
  101800-systole_crop.nii.gz → … segmentation 1.nii.gz
  110703-systole_crop.nii.gz → … segmentation 2.nii.gz
  MovingImage_sys.nii.gz → … segmentation 3.nii.gz
  movingImage_sys2.nii.gz → … segmentation 4.nii.gz



The segmentation files in dirOutput can be used for comparison with the inputfiles. This way we can determine the accuracy of the registration. For convenience a standard deviation indicator has been added to the ‘HFimage_segmentation.csv’. With the standard deviation we can quickly asses the quality of our volume metrics without having to scrutinize every segmentation file.

Progress is displayed during execution:

```
Start: 2021-03-10 14:48:26

Loading 1/4 moving images for Diastole done
Loading 2/4 moving images for Diastole done
Loading 3/4 moving images for Diastole done
Loading 4/4 moving images for Diastole done
Loading 1/4 moving images for Systole done
Loading 2/4 moving images for Systole done
Loading 3/4 moving images for Systole done
Loading 4/4 moving images for Systole done
ROI_metadata.csv found

00:00:21 - 192 images to process
        Registration 1/4 done
        Registration 2/4 done
        Registration 3/4 done
        Registration 4/4 done
00:05:52 - image 1/192 done
        Registration 1/4 done
```

During execution a file with DICOM headerdata (if available in ROI_metadata.csv) and volume metrics will be created or appended to. This file is called 'HFimage_segmentation.csv' and can be found in the dirOutput.

Note: running 'Automatic 3D segmentation for multiple images v1_0.py' on a reasonably powerful desktop computer will require at least 6 minutes for 1 inputfile.