Construction and viewing 3D volumetric image from 2D Biomedical images

Sagaf Youssouf

Université Paris-Est Créteil (UPEC)
International Master of Biometrics and Intelligent Vision
https://www.international-master-biometrics-intelligent-vision.org/

Abstract. In image processing, we may have to deal 2D images or 3D images. 2D images are the images that we are used to seeing captured by regular cameras. In scientific literature, they are called natural images. On the other hand, we have 3D images that are obtained with very specific technologies such as CT scanners and mammography devices. These types of images allow health professionals to have access to very high-resolution angles and details for the analysis of a human body part. To analyze 3D medical imaging, modern technology depends strongly on 3D visualization techniques. In this exercise, we managed to build in python a 3D volume by stacking slices one on top of the other. We obtained volumetric data which can be displayed using an extension technique of the imshow function of matplotlib.

Keywords: 3D Volumetric · Biomedical images

1 Introduction

Most radiology experts agree to say that 3D technology is strongly transforming medical imaging. This evolution of 3D imaging is due to the computing power of current machines and the software that goes with it. Because of this, 3D volumetric data visualization is becoming more valuable in medical image analysis. To display 2D images, it's common to use a method such as imshow() function from matplotlib. But to view 3D volumes, one may use tools that are outside Python like ITK-SNAP software application. But this will maximize the need to switch contexts between data analysis and data exploration. Here we build a simple program to visualize 3D volumetric data and then to visualize it using the"scrolling" capabilities of the matplotlib library. In this report, you will find in the first section the illustration of the technique performed to construct the 3D volumetric from 2D slices of the chest captured by Computed Tomography scans. In the second section, the trick applied to visualize the 3D volumetric is reported.

2 Methods and materials

We have 21 slices in a directory named `SCD2001_files`. The goal is to read these images and stack them to form a 3D volume. For this, our program is made up of three steps :

- Loading slices;
- Construction of the 3D volume;
- Visualization of the 3D volume;

To implement these three steps, we used the following python libraries : `pydicom`, `numpy`, `matplotlib` and `OS`

Loading of the slices : to read the slices, we first use the `OS` module in python to list all the slice. This module is an interface which provide a simple way of using operating dependent functionality. The next step is to read every dicom file using `read_file()` function from `pydicom`. The result is sorted and stored in a python list. Below is the snippet code to do that. An example of slice is given on the figure 1.

```
path = "/home/yosagaf/devs/medical-biometrics/SCD2001_files"
CT_images = os.listdir(path)
slices = [dicom.read_file(path+"/"+s, force=True) for s in CT_images]
slices = sorted(slices, key=lambda x:x.ImagePositionPatient[2])
```

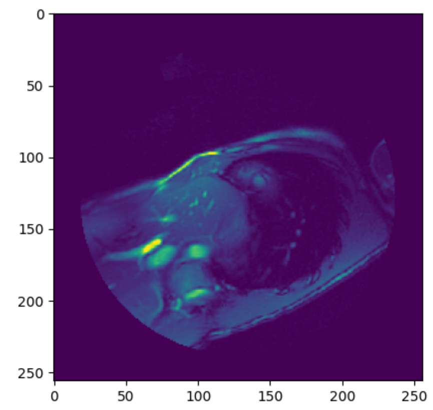

Fig. 1: An example of slice of the chest displayed using matplotlib

Construction of 3D volumetric image : to build the 3D volume, we could use the `volread()` function of `imageio` library. This function allows the reading of a volume from a directory. It returns an array of NumPy, which comes with a dictionary of metadata at its attribute. But to highlight our skills in python programming, we choose to code this step from scratch. We created a shape of the 3D array using the size of one slice and the number of slices as the third dimension. The obtained result is a tuple of `(21,256,226)`. The produced shape is the same as the one that could be given by the `volread()` function. To fill the 3D volume, we loop over the pixel array of each slice which forms the 2D array. The 3D volume is then filled with those 2D arrays to build the 3D volumetric image. This step is illustrated with the following code :

```
for i, s in enumerate(slices):
    array2d = s.pixel_array
    volume3d[i:,:,] = array2d
```

Viewing 3D Volumetric The most common way to display an image data in python is to use the `imwhow()` function of matplolib library. To display a 3D volumetric data, we extend this function using scrolling capabilities of matplotlib viewer. Here we define actions to perform change on the plot including the change of plot data using specific key. Here *l* and *n* keys are used to bind the previous and next slice respectively on the keyboard. The imshow function returns an `ImageAxes` object containing the matplotlib Axes object where drawing takes place in its `.images` attributes. All we have to do are described as follows :

- plotting a random index and store it for late use as an additional runtime;
- provide a function `nSlice()` and lSlice() to change the index for the given slices of the 3D volumetric;
- use canvas `draw` method to redraw the figure with the new data;

We simply add event handlers to Matplotlib and the last piles them on top of each other. Used keys must be removed from `matplotlib`'s default key maps, otherwise, when we will hold *n* or *l* buttons, a logarithmic scale will be observed because those keys are the shortcut to change the x-axis. The most important functions doing the job of 3D viewing is given below :

```
def view3D(volume):
    removeKeymapConflicts({'n', 'l'})
    fig, ax = plt.subplots()
    ax.volume = volume
    ax.index = volume.shape[0] // 2
    ax.imshow(volume[ax.index])
    fig.canvas.mpl_connect('key_press_event', processKey)
```

3 Results

After the construction of the 3D data volume, displaying the volume is done using a trick around the `imshow()` function of matplotlib. When we hold the n or l key, we can see a kind of interaction due to the slices changes in the third dimension of the 3D volumetric data.

4 Conclusion

Through this exercise, the objective was to build a 3D volume of data from 2D scans. The second step consisted in displaying this volume of data using a specific technique around the `imshow` function. The result is a kind of animation. The technique used can also be used to build complex applications around `matplotlib`'s visualization capabilities. This lab enabled us to highlight our skills in computer vision with medical imaging.

References

1. 3D Volume Plots in Python, `https://plotly.com/python/3d-volume-plots/`
2. Viewing 3D Volumetric Data With Matplotlib using Matplotlib's event handler API, `https://www.datacamp.com/community/tutorials/matplotlib-3d-volumetric-data`
3. Viewing Images with pydicom, `https://pydicom.github.io/pydicom/stable/old/viewing_images.html`
4. Getting 3D image from 2D image, `https://stackoverflow.com/questions/4436507/getting-3D-image-from-2d-image`
5. Several 2D images to 3D, `https://forum.image.sc/t/several-2d-images-to-3d/34467`