Parallel algorithms for erosion and dilation of label images.

Release 0.00

Richard Beare[1] and Paul Jackway[2]

February 12, 2013

Richard.Beare@monash.edu
Department of Medicine
Monash University
Melbourne
Australia[1]
CSIRO Mathematics Informatics and Statistics
Dutton Park
Queensland
Australia[2]

Abstract

It is sometimes useful to be able to apply binary morphological operations, such as erosions and dilations, to labelled images in a fashion that preserves the labels. This article introduces a specialised class implementing parallel methods described in [1] that provide very fast dilations by circles and spheres of arbitary size. Comparisons with other implementations using currently available building blocks are also made.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3399]

Contents

1 Introduction

The link between Euclidean distance transforms and binary morphological operations is well known - erosions and dilations by circles and spheres can be performed by thresholding the distance transform. Efficient and readily parallelizable distance transform algorithms can, in turn, be based on erosions and dilations by parabolic structuring elements. The classes outlined in this article extend the contact point algorithm used for parabolic erosions and dilations to facilitate operations on label images. A more complete discussion is available in [1]. The classes introduced here are able to separate touching labels during erosion and split touching labels at the midpoint during dilation.

2 The classes

The *itk::LabelSetDilateImageFilter* and *itk::LabelSetErodeImageFilter* implement dilation and erosion of label images. They share a common parent class. The methods controlling class behaviour are:

- *UseImageSpacing*: Defines whether the radius refers to voxels or world dimensions. Default is false, meaning radius is in voxels.
- *Set/GetRadius*: Set the size of the dilation or erosion. There are versions to set the size in all directions to be the same, corresponding to a circular or spherical structuring element, and independently, corresponding to an ellipsoid structuring element (with axes parallel to image axes).

Examples of use are available with the package.

Please note that these are specialised classes which cannot use arbitrary structuring elements.

The code from this contribution is also available at `https://github.com/richardbeare/LabelErodeDilate`.

2.1 Notes for label erosion

The class provided for label erosion **will separate** touching labels. If this is not the desired behaviour, then implement label erosion via binary erosion and masking.

3 Alternative approaches to label dilation

As advised on the ITK mailing list, label dilation can be implemented via distance transforms and watershed transforms. This algorith is illustrated in SimpleITK python code below (courtesy of Bradely Lowekamp):

```
def MultilabelDilation(img, radius=1,kernel=sitk.BinaryDilateImageFilter.Ball):
    distImg = sitk.SignedMaurerDistanceMap(img != 0,
                                           insideIsPositive=False,
                                           squaredDistance=False,
                                           useImageSpacing=False)
    dilatImg = sitk.BinaryDilate(img!=0, radius, kernel)
    wsImg = sitk.MorphologicalWatershedFromMarkers(distImg, img)
    return dilatImg*wsImg
```

There are a couple of altnernatives to this algorithm implemented in C++ and provided in *multilabelDilation.h*. The first version is a variant of the code above, which avoids using the binary dilate operation and thresholds the distance transform instead. This version is called *multilabelDilation*. The second version uses the *DanielssonDistanceMapImageFilter* to produce both a distance map and a Voronoi tesselation. The distance map is thresholded to produce a dilation which is then used to mask the Voronoi tesselation. This version is called *multilabelDilationDanielsson*. The Danielsson filter is slower than the Maurer filter, but this approach avoids the watershed transform step as the information provided by the Voronoi map is removes the need for the watershed. Comparisons of performance are below.

3.1 Dilations via distance maps versus binary dilation

There are subtle differences between the effective structuring element produced when thresholding distance maps versus those provided by the *BinaryBallStructuringElement*. The latter is a Bresenham circle or sphere, which means that any voxel which is partly inside the specified radius is included in the structuring elements. Distance maps, on the other hand, typically compute distances to voxel centres. Thus any voxel whos centre is closer than the specified radius is included, resulting in a slightly smaller structuring element.

4 Performance

The specialised version was developed due to ease of parallel implementation and performance results show that it is indeed much faster than versions that can be built using existing tools. It also scales moderately well with increased execution threads. The execution times and speedups for a 12 core Intel(R) Xeon(R) CPU X5650 @ 2.67GHz on a $182 \times 218 \times 182$ brain atlas are shown in Figures 1. None of the methods have a significant dependence on dilation size, as the structuring element is not explicit. Speedup at 8 cores for the Maurer method is 1.86 versus 4.71 for the parabolic method. The lack of scalability of the Maurer method is likely to be largely caused by the watershed step, which is not parallel. The Danielsson approach is much slower, despite avoiding the need for an explicit watershed transform. There is some redundancy in the Maurer approach, as a signed distance transform is computed by not used. In the single threaded the specialised version is 7 times faster than the Mauer approach and 180 times faster than the Danielsson-based method.

5 Sample results

Label images occur in many situations. This is an example of a brain atlas in which different labels represent different anatomical regions. Examples of 2D processing (operations applied to a single slice of the atlas) are shown in Figure 2. Examples of a 3D processing are shown in Figures 3 to 5.

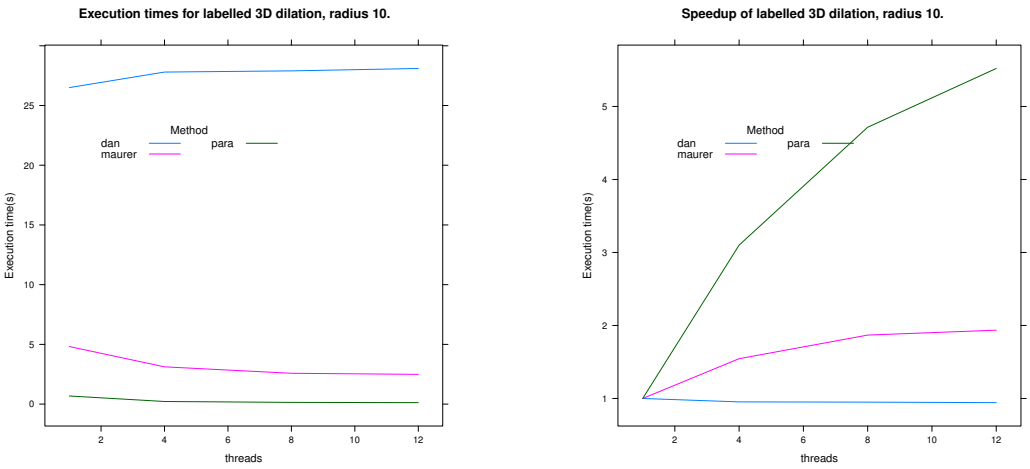

Figure 1: Execution times and speedups for all methods.

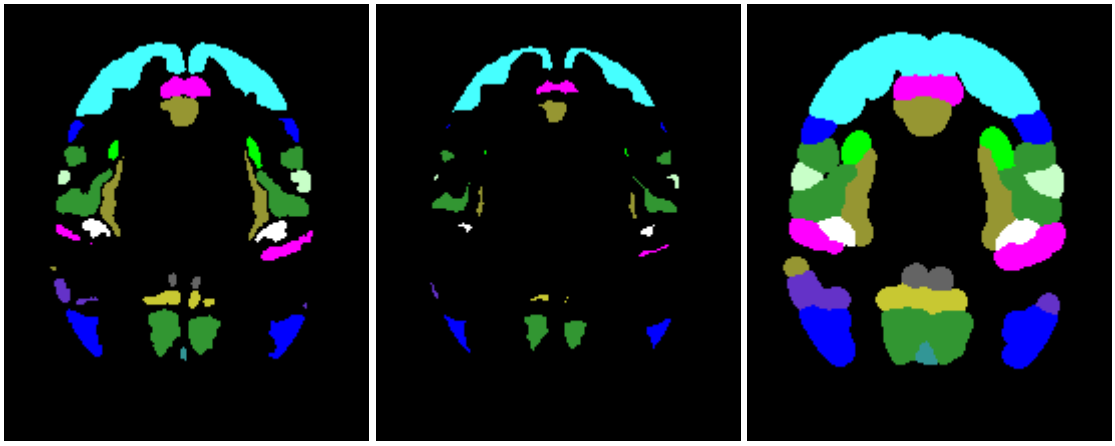

Figure 2: Original, eroded and dilated atlases - a single slice with a 2d circular structuring element.

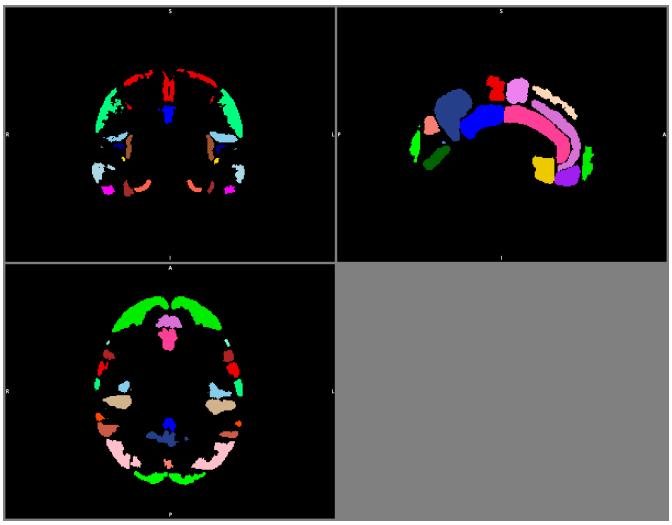

Figure 3: Axial, sagittal and coronal slices through a labelled brain atlas.

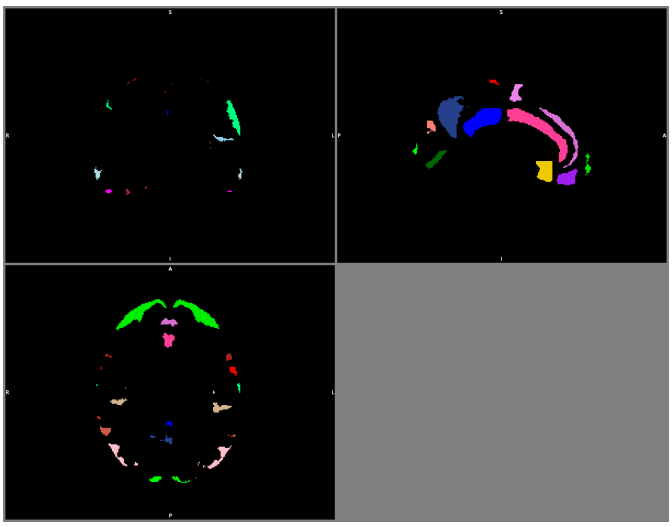

Figure 4: Atlas in Figure 3 after applying 3d labelled erosion.

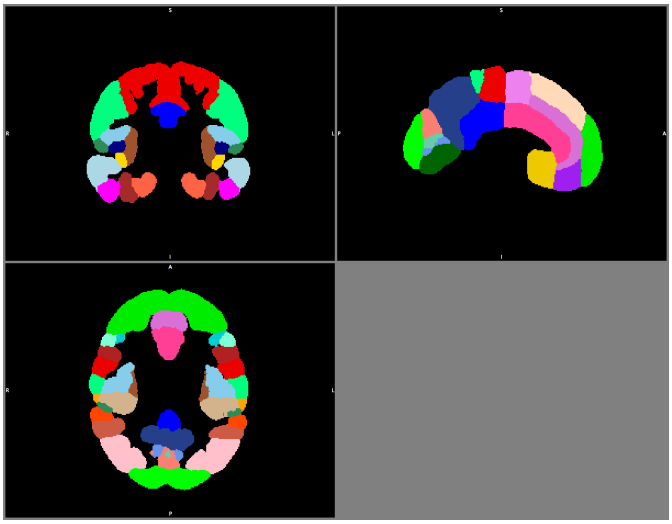

Figure 5: Atlas in Figure 3 after applying 3d labelled dilation.

6 Conclusion

This article provides two classes for erosions and dilations of label images using algorithms developed in a previously published work. These methods use parallel, scan-line base algorithms that offer very fast operations.

References

[1] R. Beare and P. Jackway. Parallel algorithms via scaled paraboloid structuring functions for spatially-variant and label-set dilations and erosions. In *International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 180–185. IEEE, 2011. (document), 1

[2] L. Ibanez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-10-6, http://www.itk.org/ItkSoftwareGuide.pdf, first edition, 2003.