N-D Linear Time Exact Signed Euclidean Distance Transform

Nicholas J. Tustison, Marcelo Siqueira, and James C. Gee

February 17, 2006

Penn Image Computing and Science Laboratory
University of Pennsylvania

Abstract

Fast computation of distance transforms find direct application in various computer vision problems. Currently there exists two image filters in the ITK library which can be used to generate distance maps. Unfortunately, these filters produce only approximations to the Euclidean Distance Transform (EDT). We introduce into the ITK library a third EDT filter which was developed by Maurer *et al.* [3]. In contrast to other algorithms, this algorithm produces the exact signed squared EDT using integer arithmetic. The complexity, which is formally verified, is $O(n)$ with a small time constant where n is the number of image pixels.

1 Introduction

For the purposes of our implementation, we consider a binary image, I_B, to consist of a set of elements of a N-D grid. These elements are each assigned a value from the dyadic set $\{0,1\}$. We label these values as 'background' and 'foreground', respectively. The EDT of a binary image, I_{EDT} is also a N-D grid in which the value of the element, $e_{EDT} \in I_{EDT}$ is the Euclidean distance between the corresponding binary image element $e_B \in I_B$ and the nearest element with a 'foreground' label in I_B. For a review of the various algorithms for calculating the EDT and its relatives, we refer the reader to the **Introduction** section of [3].

The two existing classes within ITK for calculating approximations to the EDT of binary images are

- `SignedDanielssonDistanceMapImageFilter` and
- `ApproximateSignedDistanceMapImageFilter`.

The first class, based on the 2-D work of [2] and later extended to arbitrary dimensions and anisotropic voxel dimensions in [6] and [5], respectively, computes a close approximation to the EDT such that the maximum error < 1 pixel in 2-D, *i.e.* the identified neighboring pixel could be up to a pixel away from the true nearest neighbor. However, it is slower than the algorithm contained in the `ApproximateSignedDistanceMapImageFilter` class. This work, based on the developments discussed in [4], computes the Chamfer distance which is a weighted L_1 metric that approximates the EDT. In this paper, we provide implementation details of Maurer's algorithm although it will be noted that excellent pseudocode is provided in [3]. We also compare the Maurer and Danielsson filters in terms of their output difference as well as their time performance. Comparison with the `ApproximateSignedDistanceMapImageFilter`

class is not included since the outputs from a "straight-out-of-the-box" usage are not quite analogous. However, informal comparison demonstrated that its running times were comparable to the running times of the Maurer filter.

2 Implementation

Most of the parameters set by the user are identical to those found in the class SignedDanielssonDistanceMapImageFilter. These parameters are:

- `bool m_SquaredDistance`
- `bool m_UseImageSpacing`
- `bool m_InsideIsPositive`

In addition, we designate the value which identifies the set of background pixels (default is zero).

- `InputPixelType m_BackgroundValue`

The boolean variable `m_SquaredDistance` is set to 'true' if the squared EDT is desired. A value of 'false' returns the actual EDT. The advantage of the squared EDT is that only integer values are used in calculating the output. If the boolean variable `m_UseImageSpacing` is set to 'true', the output reflects the pixel spacing. This variable should be set to 'true' when handling the case of anisotropic pixels. Note that the modification to handle anisotropic pixel sizes is very minor (as explained in [3]). Finally, the value of the boolean variable `m_InsideIsPositive` determines whether the foreground, or 'inside', distance values are negative or positive.

Another point of interest regarding the Maurer filter concerns the possibility of parallelizing the computation. Although this is not how the current algorithm is implemented, distribution amongst p processors decreases the algorithmic complexity to $O(n/p)$.

3 Comparative Results

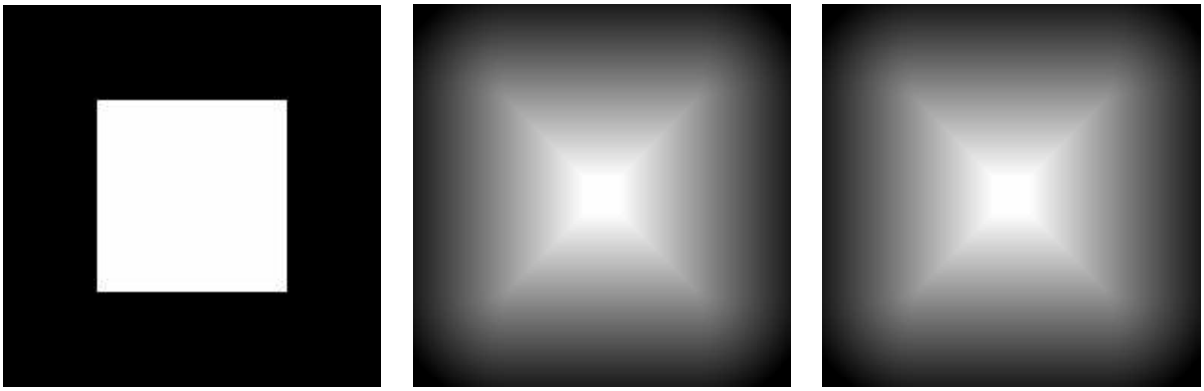
To demonstrate the difference in speed (in 2-D and 3-D) between the three signed distance filters, we used the following code snippet where `SignedFilterType` is one of the three filter types discussed previously. All tests were run using Red Hat Linux on a Pentium 4 (3 GHz), GCC version 3.2.2, and ITK 2.2.

```
SignedFilterType::Pointer filter = SignedFilterType::New();
filter->SetInput(reader->GetOutput());

itk::TimeProbe timer;
for (unsigned int i = 0; i < 10; i++)
{
  timer.Start();
  filter->Update();
  timer.Stop();
}
std::cout << "elapsed time: " << timer.GetMeanTime() << std::endl;
```

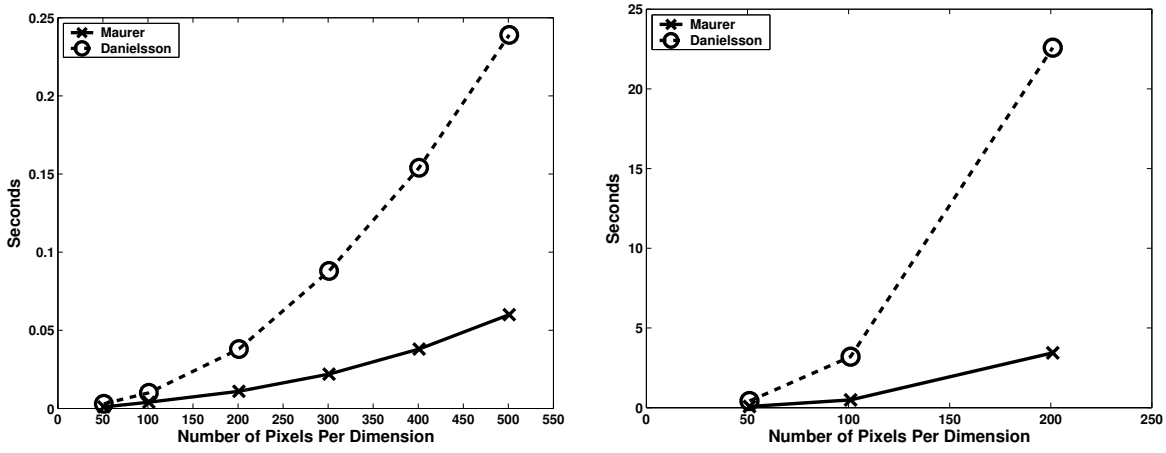
3.1 Simulated Images

Different resolutions of the image shown in Figure 1(a) were used as inputs to the three filters. The corresponding output images for the Maurer and Danielsson filters are shown in Figures 1(b) and 1(c), respectively. Note that the output images of the Maurer and Danielsson filters for these particular images are identical to within working precision even though the Danielsson filter is theoretically not exact (unlike the Maurer filter).



(a) (b) (c)

Figure 1: Results from applying the different filters to the (a) segmented lungs image. Distance map produced by the (b) Maurer filter and the (c) Danielsson filter.



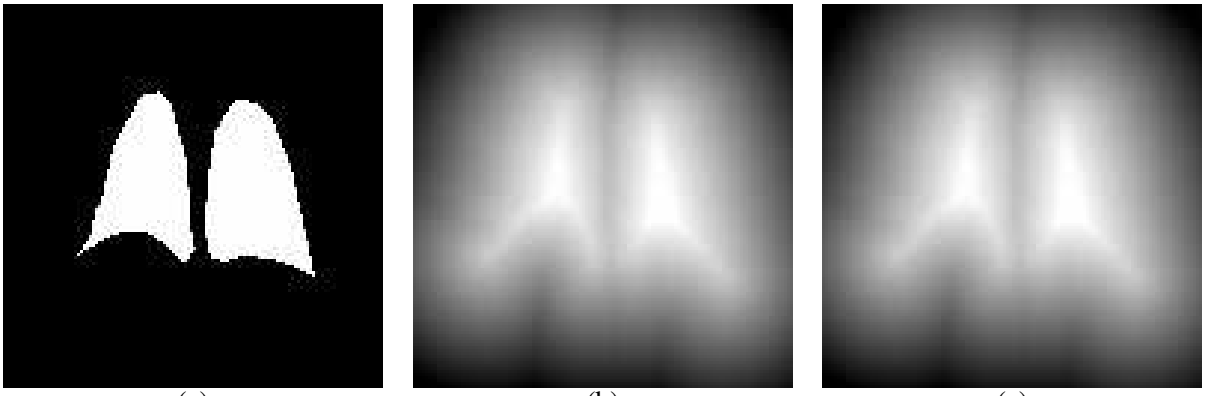
(a) (b)

Figure 2: Running times (in seconds) for the three filters. Times for the two-dimensional images are shown in (a) whereas the three-dimensional image times are shown in (b).

The disparity in speed performance is clearly visible between the Danielsson and Maurer filters for two-dimensional images (Figure 2(a). This difference in speed is exacerbated when one considers three-dimensional images [Figure 2(a)] (which were 3-D analogs of the image shown in (Figure 2(a)). Higher resolution images were considered, *e.g.* $(301 pixels)^3$ but memory issues associated with the Danielsson filter prevented their acquisition.

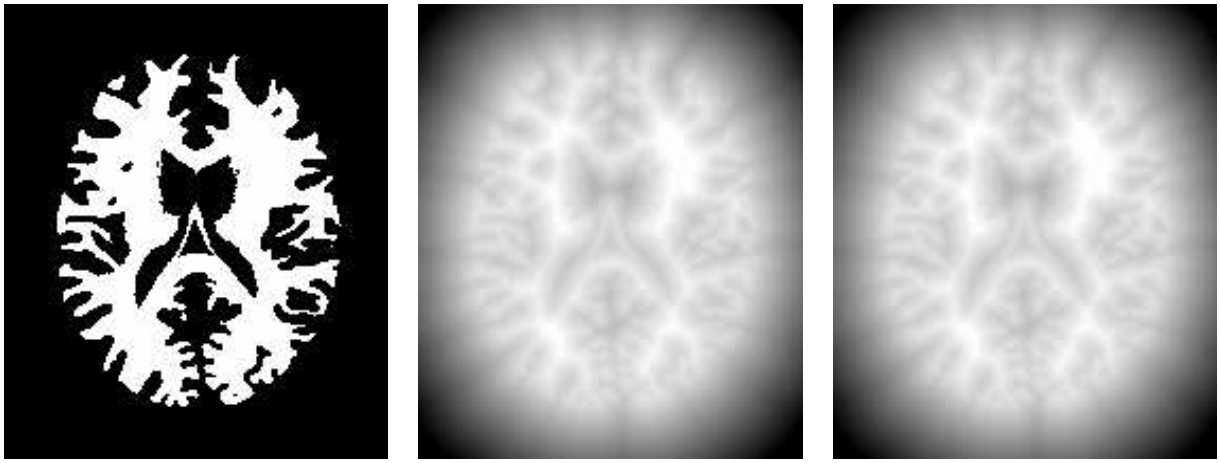
3.2 Segmented Lung and Brain Images

To further demonstrate the performance of our implementation of Maurer's algorithm, we show the results of finding the distance maps of 2-D 128×128 segmented lungs (Figure 3) and 3-D $181 \times 217 \times 181$ segmented white matter (2-D slice results are shown in Figure 4). The segmented lung images were obtained from current research in our lab. The segmented white matter was obtained from the brain web [1].



(a) (b) (c)

Figure 3: Results from applying the different filters to the (a) segmented lungs image. Distance maps produced by the (b) Maurer filter and (c) Danielsson filter.



(a) (b) (c)

Figure 4: Results from applying the different filters to a 2-D slice of the (a) segmented white matter image. Distance maps produced by the (b) Maurer filter and (c) Danielsson filter.

The maximum squared difference between the output of the Maurer and Danielsson filters for the binary lung images (Figure 3(b) and Figure 3(c)) is 0.029. The times to produce the output were 0.005 seconds and 0.015 seconds, respectively. Similarly, the maximum squared difference between the output of the Maurer and Danielsson filters for a 2-D binary brain image slice (Figure 4(b) and Figure 4(c)) is also 0.029 pixels^2. The times to produce the output were 0.005 seconds and 0.015 seconds, respectively. Running the Maurer filter on the entire 3-D image required an average time 3.204 seconds whereas the Danielsson filter took 21.36 seconds. The maximum squared difference was 0.343146.

References

[1] Chris A Cocosco, Alex P Zijdenbos, and Alan C Evans. A fully automatic and robust brain MRI tissue classification method. *Med Image Anal*, 7(4):513–27, Dec 2003. 3.2

[2] P. E. Danielsson. Euclidean distance mapping. *Computer Vision, Graphics, and Image Processing*, 14:227–248, 1980. 1

[3] C. R. Maurer Jr., Qi Rensheng, and V. Raghavan. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):265–270, 2003. (document), 1, 2

[4] K. Krissian and C. F. Westin. Fast and accurate redistancing for level set methods. In *EUROCAST NeuroImaging Workshop*, pages 48–51. Ninth International Conference on Computer Aided Systems Theory, February 2003. 1

[5] J. C. Mullikin. The vector distance transform in two and three dimensions. *CVGIP: Graphical Models and Image Processing*, 54:526–535, 1992. 1

[6] I. Ragnemalm. The euclidean distance transform in arbitrary dimensions. *Pattern Recognition Letters*, 14:883–888, 1993. 1