User Guide to
The Computational Morphometry Toolkit[1]

Release 1.00

Torsten Rohlfing

November 25, 2009

Neuroscience Program, SRI International, Menlo Park, CA[2]

Abstract

This guide is intended as a very brief introduction of the main tools in the Computational Morphometry Toolkit (CMTK), which is available in source code and as pre-compiled binaries from http://www.nitrc.org/projects/cmtk/. The target audience of this document are CMTK users, who might use this document as a reference to the most common processing tasks, and prospective users, who may find this information useful to determine whether CMTK provides functionality that they can use. We focus in particular on a simplified workflow for deformation morphometry studies based on magnetic resonance images: DICOM conversion, artifact correction, affine and nonlinear image registration, reformatting, Jacobian determinant map generation, and statistical hypothesis testing.

---

[2] Continued development and maintenance of CMTK is funded by the NIBIB under Grant No. R01 EB008381.

Contents

1 Introduction

The Computational Morphometry Toolkit, or short CMTK, is a set of software tools that perform various types of processing and analysis on three-dimensional (3D) image data. CMTK is available both in source code (licensed under the GNU GPL3) and as pre-compiled binaries from `http://www.nitrc.org/projects/cmtk/` .

CMTK is primarily a collection of command line tools, which make the toolkit ideally suited for unattended batch processing of large amounts of data. In addition, CMTK's back-end libraries, which are shared by all command line tools, can be used as a relatively lightweight, yet powerful, platform for implementation of new image processing algorithms.

LaTeX source for this User Guide, including all figures, can be checked out from the CMTK Subversion repository via

```
svn co https://nitrc.org/svn/cmtk/trunk/doc/UserGuideCMTK/
```

1.1 Coordinate Conventions

For medical image data, CMTK uses an anatomy-based coordinate system, which we refer to as "RAS" coordinates. This means that the x direction of the coordinate space increases towards the anatomical "Right," the y direction increases towards the anatomical "Anterior," and the z direction increases towards the anatomical "Superior." The coordinate space origin, $(0,0,0)$, thus coincides with the "Left-Posterior-Inferior" corner of the image volume.

All images that are read into one of CMTK's tools are first reoriented to fit this coordinate system. This means that the storage order of image pixels in memory is such that the fastest-varying of the three pixel indexes corresponds to the "Left"–"Right" anatomical direction, the second fastest to the "Posterior"–"Anterior" direction, and the slowest varying to the "Inferior"–"Superior" direction. Consequently, the first pixel in memory is the one that is the Left-Posterior-Inferior-most pixel anatomically.

For image file formats that define subject orientation based on direction vectors within an anatomy-based coordinate space, which is the majority of modern formats, CMTK determines the nearest anatomical orientation of the image within ± 45 degrees around each rotation axis.

To confirm that images are read correctly, and to diagnose problems, CMTK comes with a very simple triplanar image viewer (see screen shot in Fig. 1), adequately named "`triplanar` ." The coordinates shown in this viewer for any image are exactly the coordinates that all CMTK tools use. Note that for the triplanar viewer to be available, CMTK must be built with support for the Qt toolkit[1] (version 4.3.0 or higher), and the "`BUILD_GUI` " build option must be enabled.

1.2 Registration Terminology

Since one of the primary strengths of CMTK is its selection of powerful and well-tested registration tools, we shall first clarify some important registration terminology. In pairwise registration, throughout this guide as well as in all tools and source code, we shall refer to one image as the reference and the other as the floating image. Others may refer to these as the fixed and the moving image, respectively. By definition, all coordinate transformations computed by CMTK are functions that map *from* the space of the reference

---

[1] `http://qt.nokia.com`

120 120 77

Pixel Index: [120,120,77] Value: 350.000000

Figure 1: Screen shot of CMTK's triplanar image viewer.

(fixed) image *to* the space of the floating (moving) image. As a result, when reformatting one image to match the other, it is the floating image by default that will be transformed to match the reference image.

Note that when we speak about transforming coordinates of features, such as landmarks or the nodes in a mesh, then the coordinates of the reference image will be transformed to match the floating image.

1.3 Supported Image File Formats

CMTK supports a wide range of image file formats, both for import and export. When reading an image file into CMTK, its type is detected automatically. Note that in order to correctly identify the format of images with separate header and data files, it is necessary to provide CMTK with the path to the header file, not the data file.

Whether a particular file can be read into CMTK can easily be tested using CMTK's `describe` tool. For example, to test (and describe) the content of an Analyze 7.5[2] header/image pair, `example.hdr` and `example.img`, one would run the following command:

```
describe example.hdr
```

When writing files, CMTK determines the desired file format based on the suffix of the output path. The following suffixes are supported:

nii Single-file NIfTI-1 image[3].

img NIfTI image with detached header. Header file will be written with suffix `.hdr`

nrrd Single-file Nrrd[4].

---

[2] http://eeg.sourceforge.net/ANALYZE75.pdf

[3] http://nifti.nimh.nih.gov/nifti-1/

[4] http://teem.sourceforge.net/nrrd/

nhdr Nrrd with detached header. The data file will be written with `.raw` suffix.

hdr Analyze 7.5 detached header. The data file will be written with suffix `img`.

Note that both Analyze and NIfTI header/data file pairs use the suffixes `.hdr` and `.img`. For historic reasons, using `.hdr` as the output file suffix will always invoke Analyze export, whereas the `.img` suffix will invoke NIfTI export. Both formats need to be read using the `.hdr` file, however.

Note also that, by default, all data files are written with `gzip` compression. Because CMTK contains a bundled `zlib` library, this is true even when the `gzip` tool itself is not installed. This behavior can be disabled by defining the `CMTK_WRITE_UNCOMPRESSED` environment variable. On a Unix/Linux system using the `csh` shell, this would be achieved via

```
export CMTK_WRITE_UNCOMPRESSED=1
```

where only the definition of the variable is relevant, and its value is ignored. Thus, to re-enable compressed writing, rather than setting the variable to “0” for example, use

```
unset CMTK_WRITE_UNCOMPRESSED
```

or its appropriate equivalent inside your favorite shell.

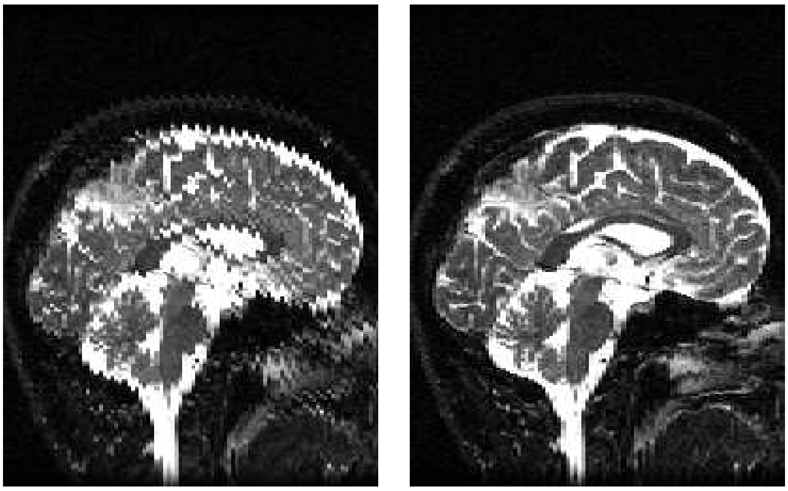

Figure 2: Example of interleaved image before (left) and after (right) correction of motion artifacts using the `film` tool. These are roughly mid-sagittal slices through a late-echo FSE image acquired in three interleaved passes.

2 Step-by-Step Morphometry

This section provides a step-by-step guide to the tools used in a typical morphometry study using the CMTK tools. It is not intended to provide a complete list of available tools. We are also not covering all available options of each tool. Note that a complete list of supported options can always be obtained by running a given tool with the `--help` command line option.

2.1 DICOM Image Stacker

When dealing with 3D medical image data in particular, the first step of processing is usually the conversion of a stack of single-slice image files in DICOM format to a single-file 3D image. To this end, CMTK provides a tool that can search through a file system tree, find all DICOM files in it, group the ones that form 3D image volumes, and write each of these volumes into a separate file in one of the supported formats.

For example, the command

```
dcm2image --recurse --out-pattern image_%02d.nii /path/to/dicom
```

or short

```
dcm2image -r -O image_%02d.nii /path/to/dicom
```

would recursively search the file system under `/path/to/dicom` and write all resulting image volumes to consecutively numbered image files in NIfTI format, `image _01.nii` , `image _01.nii` , and so on.

2.2 Interleaved Image Motion Artifact Correction

When MR images are acquired as multiple interleaved sparse image stacks (“passes”), subject motion between the passes can lead to characteristic artifacts in the final, interleaved image stack (see Fig. 2 for an example). CMTK implements an algorithm for post-reconstruction correction of these artifacts [12] in the `film` tool (for “Fix InterLeaved Motion”).

The `film` tool operates in three stages: first, the interleaved image stack is separated into the original passes, and all passes are co-registered using rigid intensity-based registration to determine the inter-pass motion parameters. Second, volume injection is used to obtain a coarse reconstructed, motion-corrected image, which is then refined in the third stage using an iterative inverse interpolation algorithm (see Ref. [12] for details).

For proper operation, the `film` tool needs to be given the number of passes in the interleaved images, for example for a three-pass image:

```
film --passes 3 input.nii corrected.nii
```

In most cases, the through-plane acquisition direction can be guessed from the data.

2.3 MR Intensity Bias Field Correction

CMTK implements a model-free algorithm for intensity bias field correction based on minimization of image entropy [5]. The `mrbias` tool, which implements this algorithm, is typically called as follows:

```
mrbias --degree-mul 2 --mask foreground.nii spgr.nii spgr_corrected.nii
```

which computes a second-order polynomial multiplicative bias field. Computation is constrained via a (binary) mask that is read from the `foreground.nii` image. Alternatively, the tool can generate its own mask via the `--thresh-min` and `--thresh-max` command line parameters.

To generate foreground masks automatically, CMTK provides a very simple “levelset-type” segmentation tool:

```
levelset --binarize spgr.nii foreground.nii
```

In very broad terms, the tool implements an extreme simplification of the algorithm for segmentation without edges by Chan & Vese [3]. By default, the tool write an image that is the resulting levelset function, but using the `--binarize` switch turns the output into a thresholded, binary mask that is appropriate for use by the `mrbias` tool.

2.4 Affine Image Registration

The basic pairwise image registration tool in CMTK, `registration`, implements an algorithm similar to the multi-resolution algorithm by Studholme *et al.* [18]. More technical detail about our implementation in particular can be found in Ref. [10], albeit only in German.

In order to compute an affine registration between two images, the registration tool can be run as follows:

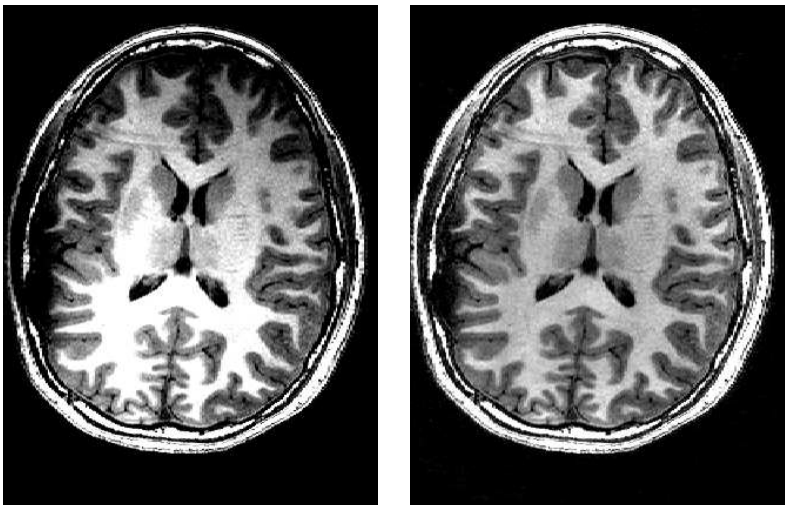

Figure 3: Example of MR intensity bias field correction using a second-order polynomial multiplicative bias field computed by the `mrbias` tool and applied to an SPGR image acquired at 3T.

```
registration --initxlate --dofs 6,9 --auto-multi-level 4 \
        -o affine.xform ref.nii flt.nii
```

This performs a registration of the floating image, `flt.nii` , to the reference image, `ref.nii` , where all optimization and image resampling parameters are automatically determined for a 4-level multi-resolution procedure.

At each resolution level, the registration first optimizes 6 degrees of freedom (DOF), i.e., translation and rotation of a 3D rigid transformation. Afterwards, 9 DOFs are optimized, i.e., three anisotropic scale factors in addition to the translational and rotational parameters. Supported DOF numbers are: 0 (no registration, for testing), 3 (translation only), 6 (rigid: translation, rotation), 7 (similarity: translation, rotation, global scale), 9 (translation, rotation, anisotropic scale), and 12 (full affine: translation, rotation, scale, and shears).

By default, registration uses the normalized mutual information [19] image similarity measure. Other available similarity measures are: standard mutual information [6, 21] (`--mi`), mean squared difference (`--msd`), normalized cross-correlation (`--ncc`), and correlation ratio [9] (`--cr`)

In the above example, the registration transformation is initialized (via `--initxlate`) by translating the floating image's center to that of the reference image. For more complex initializations, the `make_initial_affine` tool can be used, which supports centers of mass, principal axes [1], and image orientation vectors (e.g., as provided by the original DICOM data).

For example, in order to first initialize a transformation using principal axes and then use the result as the initial transformation for intensity-based refinement, one would use the following sequence of commands:

```
make_initial_xform --principal-axes ref.nii flt.nii initial.xform
registration --initial initial.xform --dofs 6,9 --auto-multi-level 4 \
        -o affine.xform ref.nii flt.nii
```

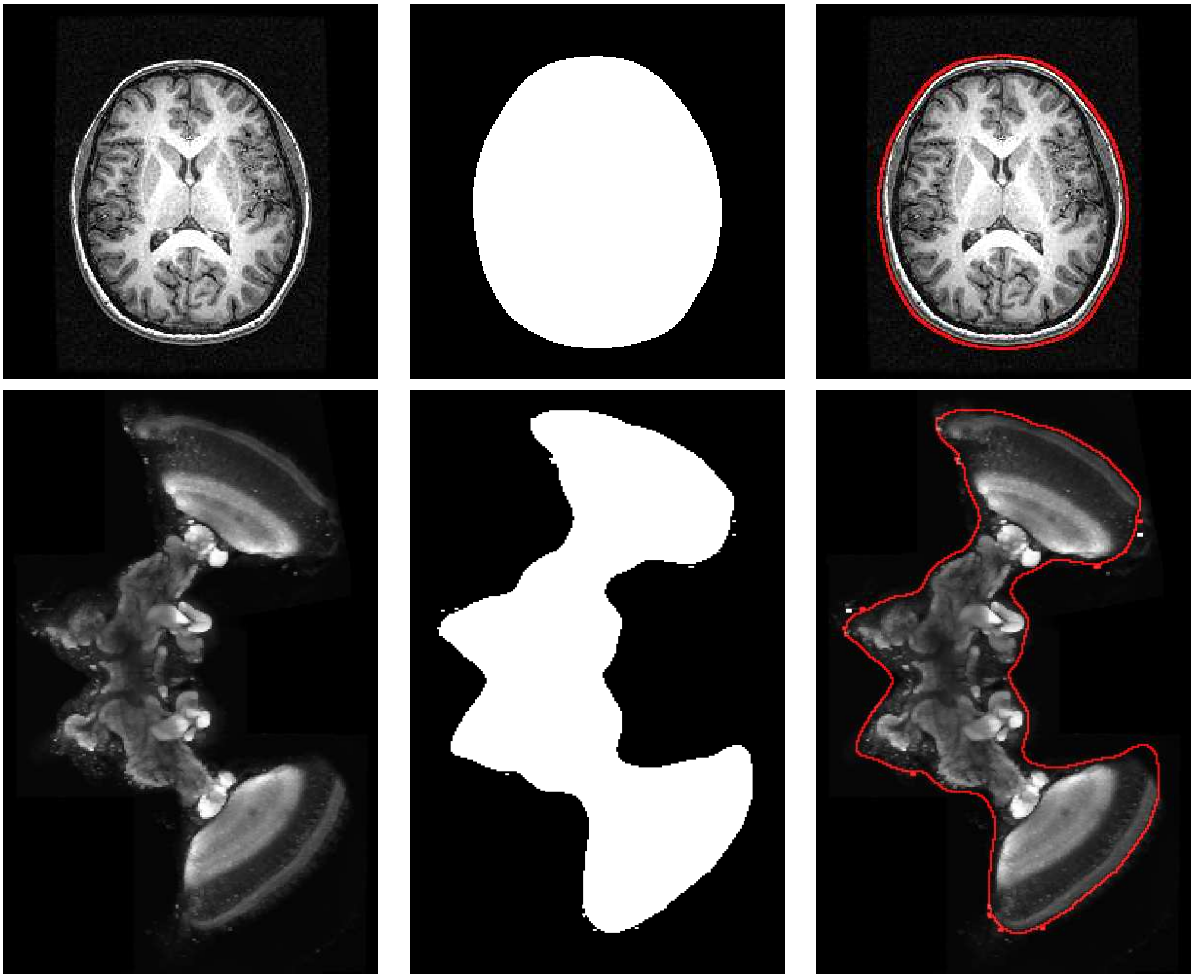

Figure 4: Examples of foreground/background segmentation using the `levelset` tool. *Top row:* SPGR image acquired at 3T. *Bottom row:* fluorescent confocal laser scanning microscopy image of a locust brain [4]. Both examples were computed by running the `levelset` tool with default settings and no image-specific parameters. [Locust images courtesy of U. Homberg, Universität Marburg (Germany).]

2.5 Nonrigid Image Registration

Pairwise nonrigid image registration in CMTK implements an algorithm introduced by Rueckert *et al.* [15], which uses as its transformation model multi-resolution free-form deformations based on cubic spline interpolation between sparse, uniformly distributed control points. Our particular implementation, which uses SMP parallelism to take advantage of multi-CPU systems, was described in Ref. [11].

A very simple nonrigid registration using a 40 mm control point grid, registering floating image `flt.nii` to reference image `ref.nii` based on an affine transformation `affine.xform` can be run as follows:

```
warp -o ffd40.xform --grid-spacing 40 --initial affine.xform ref.nii flt.nii
```

Typically, however, one would want to run a more sophisticated multi-level deformation, say with three refinements (each reducing the grid spacing by 1/2 for a final spacing of 5 mm), and constrain the deformation using grid bending energy:

```
warp -o ffd5.xform --grid-spacing 40 --refine 3 --energy-weight 1e-1 \
        --initial affine.xform ref.nii flt.nii
```

To prevent folding of the deformation grid, it is possible to instead constrain the Jacobian determinant of the deformation to be nonzero, which is achieved by changing the above command as follows:

```
warp -o ffd5.xform --grid-spacing 40 --refine 3 --jacobian-weight 1e-5 \
        --initial affine.xform ref.nii flt.nii
```

2.6 Reformating Registered Images

To reformat the registered floating image following the examples in the previous section, CMTK

```
reformatx  -o reformat.nii --floating flt.nii ref.nii ffd5.xform
```

The somewhat unintuitive order of arguments on the command line is due to the versatility of the `reformatx` tool, which allows for the concatenation of arbitrary transformations (and their inverses), such as

```
reformatx  -o reformat.nii --floating img3.nii \
        img1.nii img1_to_2.xform --inverse img3_to_2.xform
```

By default, `reformatx` uses trilinear interpolation, but it also supports cubic (`--cubic`) and cosine-windowed sinc (`--sinc-cosine`) interpolation for intensity images, partial volume interpolation [6] (`--pvi`) for label images, and nearest neighbor (`--nn`) interpolation for all types of images.

2.7 Jacobian Determinant Maps

Jacobian determinant maps, which are a staple ingredient of deformation-based morphometry studies [2], can also be computed using the `reformatx` tool. In the simplest case, we may want to compute the Jacobian determinant map for a transformation `time1_to_time2.xform` between two images, say images `time1.nii` and `time1.nii` acquired from the same subject at two time points. The command to compute the appropriate Jacobian determinant map, `jacobian.nii`, is then

```
reformatx  -o jacobian.nii time1.nii --jacobian time1_to_time2.xform
```

More interestingly, say we want to compare these Jacobian maps from multiple subjects, all in the space of as common atlas coordinate system. Then, instead of computing each map first in each subject's coordinate system and then reformatting these maps into atlas space, we can directly compute the maps in atlas space by concatenation of transformations:

```
reformatx  -o jacobian.nii atlas.nii \
        atlas_to_time1.xform --jacobian time1_to_time2.xform
```

Here, every sample coordinate in atlas space is first mapped to subject time 1 space via `atlas_to_time1.xform`. For the resulting location, the Jacobian determinant of the longitudinal transformation, `time1_to_time2.xform`, is then computed.

Because the nonrigid transformations computed by the `warp` tool are generated via continuously differentiable B-spline basis functions, we can compute the Jacobian analytically at any location in the domain of the transformation, which means that the direct computation of Jacobians into atlas space does indeed avoid one interpolation of the Jacobian determinant map.

Note that the `reformatx` tool allow an arbitrary number of transformations to be listed both before and after the `--jacobian` switch, and any transformation can additionally be inverted by prefixing it with `--inverse` (affine transformations are inverted explicitly, whereas nonrigid transformations are inverted numerically).

2.8 Statistical Testing

For group comparisons of, for example, Jacobian determinant maps between different subject groups, the `ttest` tool computes different types of t-tests (all two-tailed) and statistics. In the simplest case, two populations A and B of maps can be tested against one another as follows:

```
ttest -o pvalues.nii --tstats-file tstats.nii \
        jacobianA1.nii jacobianA2.nii -- jacobianB1.nii jacobianB2.nii
```

This computes a pixel-wise two-tailed *unpaired* t-test between the two lists of images separated with "--" on the command line. The resulting *p*-values image is then written to `pvalues.nii`, and the t-statistics are also written to `tstats.nii`.

To compute a two-tailed *paired* t-test, make sure that there are an equal number of images before and after the "--" separator and that corresponding images in both groups are in the same order, then add the `--paired` option and run

```
ttest -o pvalues.nii --tstats-file tstats.nii --paired \
        jacobianA1.nii jacobianA2.nii -- jacobianB1.nii jacobianB2.nii
```

Invoking `ttest` with only a single group of images (without "--" anywhere in the image list), will compute a single-sample t-test, that is, a test for significant differences from zero:

```
ttest -o pvalues.nii --tstats-file tstats.nii jacobianA1.nii jacobianA2.nii
```

2.9 Atlas-based Segmentation

Atlas-based segmentation uses correspondence between a previously segmented image (the atlas) and a new, unsegmented image to create a segmentation of the latter [8]. This relatively simple idea can easily be implemented using CMTK's `registration`, `warp`, and `reformatx` tools. For convenience, however, CMTK also provides an integrated atlas-based segmentation tool, which can be run as follows:

```
asegment input_image.nii atlas_image.nii atlas_labels.nii output_labels.nii
```

Here, it is assumed that `input_image.nii` is a new, unsegmented image, for example an MR scan of a new subject, `atlas_image.nii` is the intensity image of the atlas, and `atlas_labels.nii` is the label image of the atlas, i.e., the segmentation corresponding to the atlas intensity image. The tool will then register the atlas to the new image, reformat that atlas label map onto it, and write the result to the file `output_labels.nii`.

The standard atlas of CMTK is the SRI24 atlas [13, 14], which comprises several different channels of MR image information, as well as scalar diffusion measures, tissue probability maps, and segmentation maps. If CMTK is configured and built with SRI24 support (by setting the `CMTK_ROOT_PATH_SRI24` CMake variable), then a simplified segmentation tool, `asegment_sri24` , which uses the SRI24 atlas is also built.

This tool, by default, registers a given image to the SPGR channel of the SRI24 atlas. It then creates and writes a segmentation map based on the "tzo116plus" label map, which derived from the "automatic anatomic labelling" (AAL) parcellation map by Tzourio-Mazoyer *et al.* [20]. This is achieved simply by running

```
asegment_sri24 input_image.nii output_segmentation.nii
```

Different atlas channel can be used for registration, selected using the `--registration-channel` command line option: "`spgr` " for T1-weighted SPGR, which is the default,"`early-fse` " for early-echo (proton density-weighted) FSE , "`late-fse` " for late-echo (T2-weighted) FSE, and "`fa`" for DTI-derived fractional anisotropy.

Likewise, different label maps are available for the output, selected by the `--label-map` command line option: "`tzo116plus` " for the extended Tzourio-Mazoyer map, "`lpba40` " for a segmentation based on the 40-subject LONI Probabilistic Brain Atlas [17], and "`tissue` " for a maximum-likelihood three-tissue (CSF, WM, GM) segmentation.

Acknowledgments

Much of the effort required to get CMTK ready for release as open source software was performed by Mike Hasak at SRI. Calvin R. Maurer, Jr., wrote the original implementation of his linear-time algorithm for the Euclidean distance transform [7], which `cmtk::UniformDistanceMap` is based on, and kindly agreed to distribution of this derived code under the GPL. Likewise, Daniel Russakoff kindly agreed to GPL licensing of code he wrote for entropy computation based on covariance matrices, as he used it in his work on Regional Mutual Information [16]. Greg Jefferis provided numerous bug reports and fixes, including much of the details required to get CMTK compiled and working on the MacOS platform.

References

[1] N. M. Alpert, J. F. Bradshaw, D. Kennedy, and J. A. Correia. "The principal axes transformation – a method for image registration." *Journal of Nuclear Medicine*, **31**(10):1717–1722, 1990. 2.4

[2] J. Ashburner, C. Hutton, R. Frackowiak, I. Johnsrude, C. Price, and K. Friston. "Identifying global anatomical differences: Deformation-based morphometry." *Human Brain Mapping*, **6**(5–6):348–357, 1998. `http://dx.doi.org/10.1002/(SICI)1097-0193(1998)6:5/6<348::AID-HBM4>3.0.CO;2-P`. 2.7

[3] T. F. Chan and L. A. Vese. "Active contours without edges." *IEEE Transactions on Image Processing*, **10**(2):266–277, 2001. `http://dx.doi.org/S1057-7149(01)00819-3`. 2.3

[4] A. E. Kurylas, T. Rohlfing, S. Krofczik, A. Jenett, and U. Homberg. "Standardized atlas of the brain of the desert locust, schistocerca gregaria." *Cell and Tissue Research*, **333**(1):125–145, 2008. `http://dx.doi.org/10.1007/s00441-008-0620-x`. PMID 18504618. 4

[5] B. Likar, M. A. Viergever, and F. Pernus. "Retrospective correction of MR intensity inhomogeneity by information minimization." *IEEE Transactions on Medical Imaging*, **20**(12):1398–1410, 2001. `http://dx.doi.org/10.1109/42.974934`. 2.3

[6] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. "Multimodality image registration by maximisation of mutual information." *IEEE Transactions on Medical Imaging*, **16**(2):187–198, 1997. 2.4, 2.6

[7] C. R. Maurer, Jr., R. Qi, 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. 2.9

[8] M. I. Miller, G. E. Christensen, Y. Amit, and U. Grenander. "Mathematical textbook of deformable neuroanatomies." *Proceedings of the National Academy of Sciences of the U.S.A.*, **90**(24):11944–11948, 1993. 2.9

[9] A. Roche, G. Malandain, X. Pennec, and N. Ayache. "The correlation ratio as a new similarity measure for multimodal image registration." In W. M. Wells, III., A. C. F. Colchester, and S. Delp (eds.), "Medical Image Computing and Computer-Assisted Intervention - MICCAI'98, First International Conference, Cambridge, MA, USA, October 11-13, 1998, Proceedings," vol. 1496 of *Lecture Notes in Computer Science*, pp. 1115–1124. Springer-Verlag, 1998. 2.4

[10] T. Rohlfing. *Multimodale Datenfusion für die bildgesteuerte Neurochirurgie und Strahlentherapie*. Ph.D. thesis, Technische Universität Berlin, 2000. 2.4

[11] T. Rohlfing and C. R. Maurer, Jr. "Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees." *IEEE Transactions on Information Technology in Biomedicine*, **7**(1):16–25, 2003. PMID 12670015. 2.5

[12] T. Rohlfing, M. H. Rademacher, and A. Pfefferbaum. "Volume reconstruction using inverse interpolation: application to interleaved image motion correction." In D. Metaxas, L. Axel, G. Fichtinger, and G. Székely (eds.), "Medical Image Computing and Computer-Assisted Intervention — MICCAI 2008. 11th International Conference, New York, NY, USA, September 6-10, 2008, Proceedings, Part I," vol. 5241 of *Lecture Notes in Computer Science*, pp. 798–806. Springer-Verlag, Berlin/Heidelberg, 2008. `http://dx.doi.org/10.1007/978-3-540-85988-8_95`. PMID 18979819, PMC 2646840. 2.2

[13] T. Rohlfing, N. M. Zahr, E. V. Sullivan, and A. Pfefferbaum. "The SRI24 multi-channel brain atlas: Construction and applications." In J. M. Reinhardt and J. P. W. Pluim (eds.), "Medical Imaging 2008: Image Processing," vol. 6914 of *Proceedings of SPIE*, p. 691409. Bellingham, WA, 2008. `http://dx.doi.org/10.1117/12.770441`. PMID 19183706, PMC 2633114. 2.9

[14] T. Rohlfing, N. M. Zahr, E. V. Sullivan, and A. Pfefferbaum. "The SRI24 multichannel atlas of normal adult human brain structure." *Human Brain Mapping*, 2010. `http://dx.doi.org/10.1002/hbm.20906`. In press. 2.9

[15] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. "Nonrigid registration using free-form deformations: Application to breast MR images." *IEEE Transactions on Medical Imaging*, **18**(8):712–721, 1999. 2.5

[16] D. B. Russakoff, C. Tomasi, T. Rohlfing, and C. R. Maurer, Jr. "Image similarity using mutual information of regions." In "Computer Vision - ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part III," vol. 3023 of *Lecture Notes in Computer Science*, pp. 596–607. Springer-Verlag, Berlin/Heidelberg, 2004. 2.9

[17] D. W. Shattuck, M. Mirza, V. Adisetiyo, C. Hojatkashani, G. Salamon, K. L. Narr, R. A. Poldrack, R. M. Bilder, and A. W. Toga. "Construction of a 3D probabilistic atlas of human cortical structures." *NeuroImage*, **39**(3):1064–1080, 2008. `http://dx.doi.org/10.1016/j.neuroimage.2007.09.031`. 2.9

[18] C. Studholme, D. L. G. Hill, and D. J. Hawkes. "Automated three-dimensional registration of magnetic resonance and positron emission tomography brain images by multiresolution optimization of voxel similarity measures." *Medical Physics*, **24**(1):25–35, 1997. 2.4

[19] C. Studholme, D. L. G. Hill, and D. J. Hawkes. "An overlap invariant entropy measure of 3D medical image alignment." *Pattern Recognition*, **32**(1):71–86, 1999. `http://dx.doi.org/10.1016/S0031-3203(98)00091-0`. 2.4

[20] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. "Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain." *NeuroImage*, **15**(1):273–289, 2002. `http://dx.doi.org/10.1006/nimg.2001.0978`. 2.9

[21] W. M. Wells, P. A. Viola, H. Atsumi, S. Nakajima, and R. Kikinis. "Multi-modal volume registration by maximization of mutual information." *Medical Image Analysis*, **1**(1):35–51, 1996. `http://dx.doi.org/10.1016/S1361-8415(01)80004-9`. 2.4