Read and Write Support for MevisLab Dicom/Tiff Format

Release 1.00

Rashindra Manniesing[1]

January 18, 2009

[1]Biomedical Imaging Group Rotterdam (BIGR)
Erasmus MC - University Medical Center Rotterdam,
the Netherlands
r.manniesing@erasmusmc.nl

Abstract

MevisLab [2] is a development environment for medical image processing and visualization, which supports the reading and writing of combined dicom/tiff images. In this document we provide the source code (ImageIO factory) and testing data for the Insight Toolkit (ITK) framework [4].

Contents

1 Introduction

MevisLab [2] makes use of a combined header-image file format to support the reading and writing of images. The header file consists of one dicom file, usually copied from e.g. a raw dicom file and in case of three dimensional data also storing the interslice spacing, and of one TIFF file possibly compressed, for the image data. In this document we describe the available classes, required libraries and suggestions on how to include the new fileformat for your applications.

2 Available Classes

The following classes are provided:

- `itkMevisDicomTiffImageIOFactory.h/cxx`

  Basic image IO factory class to create an instance of the object.

- `itkMevisDicomTiffImageIO.h/cxx`

  The actual implementation of the dicom/TIFF file format. These classes have been developed using ITK 3.10.0, gdcm 2.0.10 and TIFF 3.8.2. It currently supports 2D and 3D scalar images, and uchar, char, ushort, short, uint, int and float pixel types. Furthermore, it always assumes tiled TIFF images when reading or writing, and always uses LZW compression when writing.

3 Required Libraries

The following libraries must be installed: GDCM, version 2.0 [1] and the TIFF library [3] (version 3.8.2 is recommended). GDCM can be build with cmake, TIFF uses autoconfigure. When (re-)compiling the ITK tree, set the corresponding flags to make use of the system installed versions.

4 Adding MevisIO to your Application

One possibility is to add the new file format to the ITK tree, by copying these files to the directory Insight/Code/IO/ and modifying the files CMakeLists.txt and `itk::ImageIOFactory.cxx`. It is recommended to register the new factory before the GDCM and TIFF factories. Clearly, this requires a recompilation of libITKIO.

Another possibility is to add the new file format to a specific application (see for example itkMevisTest) by including the following header files `itk::ObjectFactoryBase.h`, `itk::MevisDicomTiffImageIOFactory.h` and `itk::MevisDicomTiffImageIO` into your application, and explicitly register the new factory to base before reading and/or writing the images:

```
typedef itk::MevisDicomTiffImageIOFactory IOF;
IOF::Pointer iof = IOF::New();
itk::ObjectFactoryBase::RegisterFactory(iof);
```

It may be convenient to have the new factory compiled separately in a shared library for usage in several applications:

```
ADD_LIBRARY(MevisIO
        itkMevisDicomTiffImageIO.cxx
        itkMevisDicomTiffImageIOFactory.cxx)
TARGET_LINK_LIBRARIES(app MevisIO)
```

A final note concerns the use of RESCALE_INTERCEPT and RESCALE_SLOPE, stored in the DICOM header file for linear rescaling of the pixel intensities. These values are *not* taken into account when reading/writing images. How to test for non-default values is shown in the following example of source code:

```
typedef itk::ImageIOBase IOB;
IOB * iob = reader->GetImageIO();

typedef itk::MevisDicomTiffImageIO IO;
if (dynamic_cast<IO*>(iob))
{
    IO * io = dynamic_cast<IO*>(iob);
    const double intercept = io->GetRescaleIntercept();
    const double slope = io->GetRescaleSlope();

    if (intercept != itk::NumericTraits<double>::Zero  ||
            slope != itk::NumericTraits<double>::One)
    {
        std::cout << "applying inter/slope\t" << intercept << " " << slope << std::endl;
    }
}
```

5 Extensions

The following can be considered to extend the functionality of these classes: the support of streaming, RGB pixel types, 4D imaging data, non-tiled TIFF images and the possibility to let the user decide which compression scheme to use.

References

[1] GDCM, release 2.0. http://apps.sourceforge.net/mediawiki/gdcm/index.php?title=GDCM_Release_2.
3

[2] MevisLab, software for medical image processing and visualization. http://www.mevislab.de. (document), 1

[3] libTIFF - TIFF Libary and Utilities. http://www.libtiff.org. 3

[4] L. Ibáñez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide*. Kitware, Inc., second edition, 2005. (document)