MATITK: Matlab-ITK Interface for Medical Image Processing

V. Chu,, G. Hamarneh

School of Computing Science, Simon Fraser University, Burnaby, BC, V5A 1S6, Canada

Introduction and Motivation

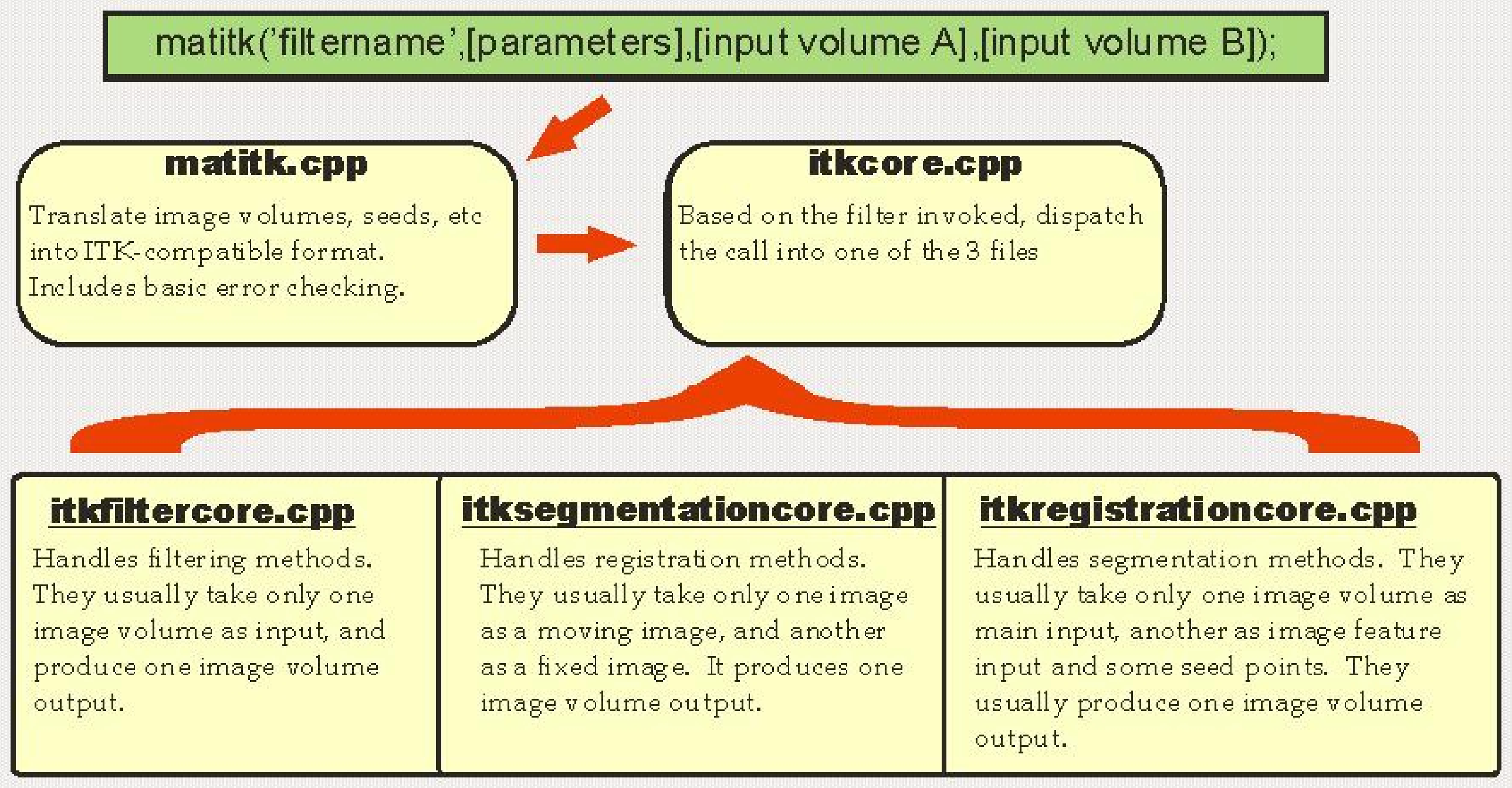
- MATLAB (MATrix LABoratory)
  - ➢environment developed by Mathworks
  - ➢often used by engineers and researchers
  - ➢facilitates matrix manipulations, numeric computations and graphics viewing.
  - ➢provides a high-level programming language that alleviates the users from low-level details such as memory management.
  - ➢available MATLAB toolboxes handle complicated peripheral tasks, such as GUI creation, graph plotting, statistical analysis, and 2D image acquisition.
  - ➢many researchers and engineers are comfortable with the one dimensional biomedical signal processing capabilities of MATLAB
  - ➢as the dimensionality of the data increases (to two and three dimensional images) both the unavailability of advanced algorithms and slow processing speed quickly become a bottleneck.
- ITK
  - ➢part of the Visible Human project
  - ➢free open-source toolkit written and used in C++ environment.
  - ➢contains various multidimensional filtering, segmentation and registration algorithms designed for medical image analysis.
- It is desirable to
  - ➢use the state-of-the-art, compiled, fast, 3D (and higher) medical image processing capabilities of ITK; and
  - ➢work in the fast-prototyping, high-level, environment of MATLAB that doesn't require intimate knowledge of C++, generic programming, and other advanced ITK programming concepts.
- Notably, because medical image data volumes are often huge, it is ***impractical*** to
  - ➢write the image volumes to disk in a suitable format; and read the volumes back with pre-compiled ITK algorithms.
- With the help of the wrapper that we introduce, biomedical computing researchers familiar with MATLAB can harness the power of ITK while
  - ➢avoiding learning C++ and dealing with low-level programming issues.
  - ➢avoiding manual marshalling (translation performed by the wrapper is done in memory, which is an order of magnitude faster)
- New additions to extend MATLAB medical image processing functionality can readily be used by the
  - –Statistical Parametric Mapping (SPM), and
  - –Extensible MATLAB Medical Analysis (EMMA) communities.

Overview

- MATLAB has the functionality to access dynamically linked library compiled in another language such as C and Fortran.
- Such dynamically linked library, also referred to as MEX file (for MATLAB EXecutables), can be run from the MATLAB environment like MATLAB M-functions or built-in functions
- The work presented is a MEX that serves as a wrapper, or an interface between MATLAB computation environment and ITK, hereafter referred to as MATITK.
- Because image data is represented differently, the wrapper provides the necessary translations in an efficient manner.

Methods (Implementation of MATITK)

- We used the following in our development environment:
  - ➢MATLAB 7.0 on MS Windows
  - ➢ ITK toolkit v1.80
  - ➢A MATLAB compatible compiler. MS Visual Studio 2003 C++ compiler was used.
- MATITK itself is written in C++.
- When MATITK command is issued in MATLAB the code compiled from matitk.cpp is executed.
  - ➢matitk.cpp is responsible for error handling and translation of image data passed from MATLAB into ITK-compatible format.
  - ➢This includes dealing with indexing differences between MATLAB and C++ arrays (ordering of dimensions and zero- vs. one-based array indexing).
- The image passed from MATLAB will be stored in an ITK Image container.
- itkcore.cpp calls the code in one of the three files, depending on whether the command invoked is a medical image filtering, segmentation, or registration command.
- Helper classes seedcontainer.cpp and parametercontainer.cpp contain the user-supplied seeds and parameters respectively.
  - ➢ITK methods can access the necessary seed points and parameters in the three ITK core files.



matitk('filtername',[parameters],[input volume A],[input volume B]);

matitk.cpp
Translate image volumes, seeds, etc into ITK-compatible format. Includes basic error checking.

itkcore.cpp
Based on the filter invoked, dispatch the call into one of the 3 files

itkfiltercore.cpp
Handles filtering methods. They usually take only one image volume as input, and produce one image volume output.

itksegmentationcore.cpp
Handles registration methods. They usually take only one image as a moving image, and another as a fixed image. It produces one image volume output.

itkregistrationcore.cpp
Handles segmentation methods. They usually take only one image volume as main input, another as image feature input and some seed points. They usually produce one image volume output.

Figure 1: Architectural Design of MATITK Framework

- With the developed framework, additional ITK methods can be added and be accessible from MATLAB environment.
- For example, new filtering method should be added to itkfiltercore.cpp and assume the following format:

```
#include "itkDiscreteGaussianImageFilter.h"
   ...
 0 void filterGaussian(){
 1   const char* PARAM[]={"gaussianVariance","maxKernelWidth"};
 2   const char* SUGGESTVALUE[]={"",""};
 3   const int nParam = sizeof(PARAM)/sizeof(*PARAM);
 4   ParameterContainer paramIterator(PARAM,SUGGESTVALUE,nParam);
 5   double gaussianVariance=paramIterator.getCurrentParam(0);
 6   unsigned int maxKernelWidth=(unsigned
                       int)paramIterator.getCurrentParam(1);
 7   typedef
     itk::DiscreteGaussianImageFilter<InternalImageType,InternalImageType>
             FilterType;
 8   FilterType::Pointer filter = FilterType::New();
 9   filter->SetInput(importFilter[IMPORTFILTERA]->GetOutput());
10   filter->SetVariance( gaussianVariance );
11   filter->SetMaximumKernelWidth( maxKernelWidth );
12   filter->Update();
13   pixelContainer=filter->GetOutput()->GetPixelContainer();
}
```

- With specific syntaxes in line 1 to line 5, ITK filtering method can access the parameters passed from the MATLAB environment. From line 9, the ITK method can access the data volume from MATLAB and with line 13, it returns the results back to MATLAB.

Results and Conclusions

- The following ITK methods are currently supported:

Opcode	Corresponding human-readable filter name
FGA	filterGaussian
FCA	filterCurvatureAnsio
FCF	filterCurvatureFlow
FMMCF	filterMinMaxCurvatureFlow
FGM	filterGradientMagnitude
FGMS	filterGradientMagnitudeWithSmoothing
FSN	filterSigmoidNonlinearMapping
FBD	filterDilate
FBE	filterErode
FDM	filterDanielssonDistanceMapImageFilter
FDMV	filterDanielssonDistanceMapImageFilterGetVoronoiMap
FBL	filterBilateral
FBB	BinomialBlurImageFilter
FBT	BinaryThresholdImageFilter

Opcode	Corresponding human-readable filter name
FBB	BinomialBlurImageFilter
FD	DerivativeImageFilter
FDG	DiscreteGaussianImageFilter
FF	FlipImageFilter
FGAD	GradientAnisotropicDiffusionImageFilter
FGMRG	GradientMagnitudeRecursiveGaussianImageFilter
FLS	LaplacianRecursiveGaussianImageFilter
FMEANF	MeanImageFilter
FMEDIANF	MedianImageFilter
SCC	segmentationConfidenceConnected
SIC	segmentationIsolatedConnected
SNC	segmentationNeighbourhoodConnected
SCT	segmentationConnectedThreshold
SFM	segmentationFastMarch
SOT	segmentationOtsuThreshold

- MATITK commands can then be invoked in MATLAB environment by simply typing matitk; which writes the following to MATLABs window:

```
matitk(operationName,[parameters],[inputArray1],[inputArra
y2],[seed(s)Array],[Image(s)Spacing])
```

- The first argument to matitk, operationName, specifies the opcode of the ITK method to be invoked.
- The second argument to matitk, parameters, specifies the required parameters of the ITK method to be invoked (specified by operationName). The parameters that are required for a particular method can be found out by typing matitk(operationName);
- For example to perform anisotropic diffusion filtering on a 3D image, the user types matitk('FCA') and the following will be written to MATLAB's window that lists the required parameters:

```
FCA is being executed... You must supply parameters for
this function in an array, with the elements in this
order:
numberOfIterations,
timeStep (which usually has value equal to 0.0625),
conductance (which usually has value equal to 3.0)
3 parameters must be supplied.  You supplied 0.
```

- The third and fourth arguments to matitk, inputArray1 and inputArray2, specify the input image volumes. They must be three dimensional and contain double data type elements.
- To demonstrate the use of MATITK, the following commands are executed:

```
>> load mri; D=squeeze(D);
>> b=matitk(`FCA',[5 0.0625 3],double(D));
FCA is being executed... FCA has completed.
>> c=matitk(`SCC',[1.4 10 255],double(b),[],[102 82 25]);
SCC is being executed... SCC has completed.
```

- The purpose here, of course, is not to optimally analyze medical images but rather to demonstrate the use of MATITK. Execution takes a few second on a 3GHz PC in both cases.

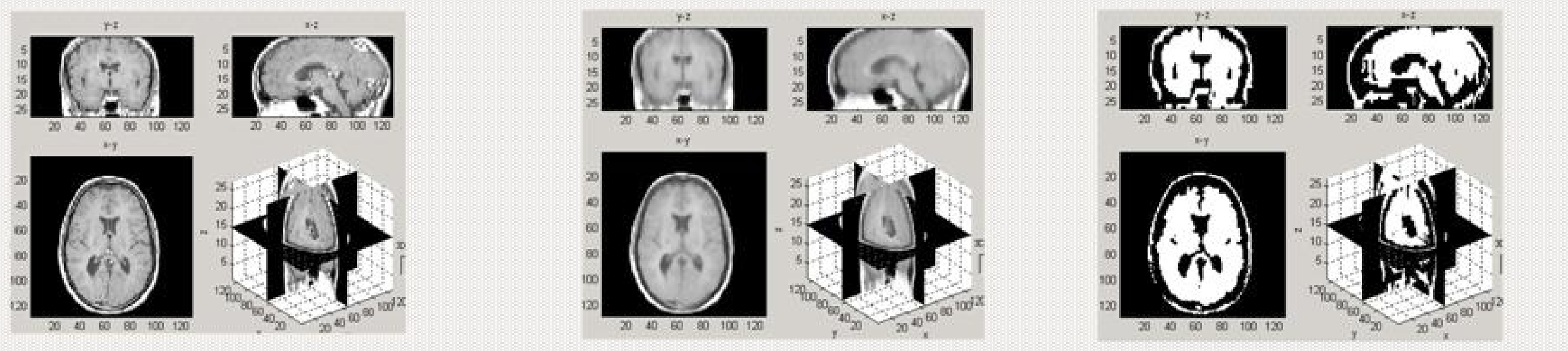


Figure 2: Example MATITK results. Left to right, top row: Original 3D image, anisotropic smoothing, connected component segmentation.

- We presented MATITK, an easy to install, use, and extend, MATLAB-ITK interface. MATITK enables researchers and scientists to easily and efficiently access advanced medical image processing and analysis methods of ITK from MATLAB.

Contact: Vincent Chu (vwchu@sfu.ca) or Ghassan Hamarneh (hamarneh@cs.sfu.ca)