2D/3D Continuous Max-Flow (CMF) Software v1.0

Jing Yuan[1]

University of Western Ontario
London, Ontario, Canada N6A 5B7
cn.yuanjing@gmail.com

1 What does this software package solve?

This software package includes the programs which are designed to efficiently solve the 2D/3D image segmentation problem, based on the fast continuous max-flow method (CMF) proposed by J. Yuan, E. Bae, X-C Tai and Y. Boykov [5, 4].

Image segmentation problem can be mathematically formulated as:

$$\min_u \; \langle 1-u, C_s\rangle \, + \, \langle u, C_t\rangle + \int_\Omega \omega(x) \, |\nabla u| \; dx \tag{1}$$

where $u(x) \in \{0, 1\}$ gives the indicator function of the 2D/3D segmentation region/volume, and the total-variation function amounts to its weighted length.

By the work of Nikolova et al [3], the segmentation problem (1) can be globally solved by convex optimization with the relaxation of $u(x) \in [0, 1]$, such that

$$\min_u \; \langle 1-u, C_s\rangle \, + \, \langle u, C_t\rangle \, + \, \int_\Omega \omega(x) \, |\nabla u| \; dx \, , \quad \text{s.t. } u(x) \in [0,1] \, . \tag{2}$$

J. Yuan et al [5, 4] proposed the continuous max-flow approach to (2), for which the continuous max-flow model reads:

$$\max_{ps,pt,p} \quad \int_\Omega p_s \, dx \tag{3}$$

$$\text{s.t.} \quad (\text{div } p \, + \, p_t \, - \, p_s)(x) \, = \, 0 \tag{4}$$

$$p_s(x) \, \leq \, C_s(x) \, , \quad p_t(x) \, \leq \, C_t(x) \, , \quad |p(x)| \, \leq \, \omega(x) \tag{5}$$

Remark 1 *In mathematics, J. Yuan [5, 4] proved* (3) *is is dual to* (2) *and showed that* (3) *gives a variational flow-maximization explanation to* (2)*, i.e. the duality btw. continuous max-flow* (3) *and min-cut* (2)*. This is similar to the classical duality between max-flow and min-cut.*

Remark 2 *In numerics,* (3) *leads to the fast continuous max-flow algorithm [5, 4], which solves* (2) *efficiently and can be easily speeded-up by GPU. Essentially, its CPU version runs as fast as graph-cuts [1, 2] and its GPU version runs more than* 6 *times faster than graph-cuts in practice, especially in 3D!*

Remark 3 *The fast continuous max-flow algorithms are developed with three programming tools: matlab, C and GPU (CUDA based) respectively.*

They are ready for matlab usage and can also be easily embeded to other applications.

2 What does this software package include?

The software package includes the programs solving the 2D/3D image segmentation problem through the fast continuous max-flow algorithm [5, 4]. All the programs are developed to be ready for matlab usages, along with three implementations: matlab, C and CUDA C (GPU). *For the GPU programs, you need an Nvidia CUDA based GPU card.*

The included files are listed as follows:

- Matlab programs: CMF_Cut.m and CMF3D_Cut.m are the 2D and 3D matlab implementation of the fast continuous max-flow algorithm respectively.
- C programs: CMF_mex.c and CMF3D_mex.c are the respective 2D and 3D implementation of the fast continuous max-flow algorithm by C.
- GPU(CUDA) programs: CMF_GPU.cu, CMF_kernels.cu, CMF3D_GPU.cu and CMF3D_kernels.cu are the respective 2D and 3D implementation of the fast continuous max-flow algorithm by CUDA C.
  The files *_kernesl.cu give the GPU kernel implementation.
- Example files: test_CMF.m and text_CMF3D.m show how to use the C and GPU(CUDA) programs in matlab.
- LICENCE.TXT is the GNU Licence file.

3 How to use this software package?

To use the matlab programs inside: CMF_Cut.m and CMF3D_Cut.m, you can directly modify the data and parameters in the corresponding matlab files.

To use the C programs: CMF_mex.c and CMF3D_mex.c, you should first compile them by matlab mex.

To use the GPU programs: CMF_GPU.cu and CMF3D_GPU.cu, you should first compile them by nvcc. The detailed command and configurations can be found at: http://developer.nvidia.com/matlab-cuda . **For Linux users, you may need the permission to access the GPU card!**

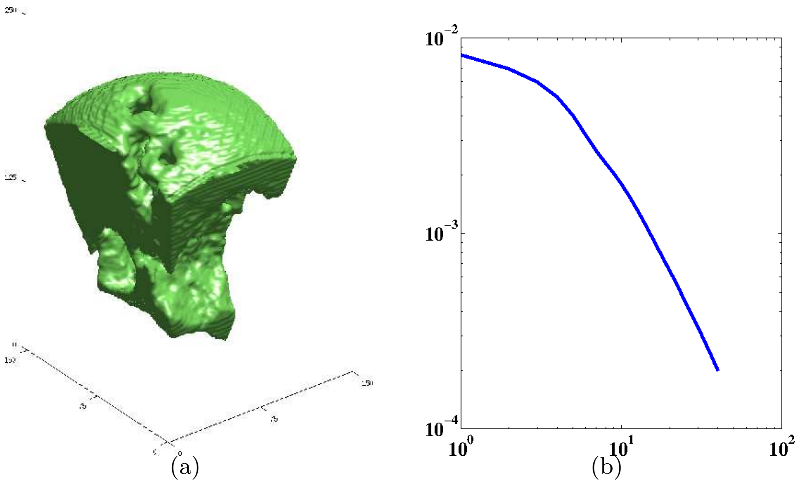
Once you finish compiling C and GPU programs, they can be used by any matlab program in a similar way.

For example, you can just follow the steps in matlab to use the 3D GPU program CMF3D_GPU (also see the two example files included):

```
>> define the volumes: w, Cs and Ct.
>> define the parameter vector: para
>> [u, erriter, i, timet] = CMF3D_GPU(single(w), single(Cs),
single(Ct), single(para));
```

The output u gives the segmentation volume. The following command is used to show its 3D segmentation surface in matlab (see Fig. 1a):

```
>> isosurface(u,0.5), daspect([1 1 1]);
```



(a) (b)

Fig. 1. (a) An example of 3D volume segmentation ($144 \times 112 \times 208$). (b) An example of the convergence plot.

The output *erriter* gives the error estimation at each iteration and it is used to show the convergence in matlab (see Fig. 1b):

```
>> loglog(erriter,'DisplayName','erriter');figure(gcf)
```

The output *i* and *timet* show at which step the algorithm converges and the total computing time respectively.

4 Licence

Please email Jing Yuan (cn.yuanjing@gmail.com) for any questions, suggestions and bug reports. If you use this software, you have to reference the papers [5, 4].

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

```
Copyright 2011 Jing Yuan (cn.yuanjing@gmail.com)
  https://sites.google.com/site/wwwjingyuan/
```

5 Contributors

The programs are not partcularly optimized. You are very welcome to provide your suggestions for the improvement of programs. I will incorporate you in the following contributors' list for your kind contributions:

Jing Yuan	Univ. of Western Ontario, Canada	cn.yuanjing@gmail.com
Egil Bae	Bergen Univ., Norway	Egil.Bae@math.uib.no
Xue-Cheng Tai	Bergen Univ., Norway	tai@cma.uio.no
Yuri Boykov	Univ. of Western Ontario, Canada	yuri@csd.uwo.ca

References

1. Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on PAMI*, 23:1222 – 1239, 2001.
2. Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts. *IEEE Transactions on PAMI*, 26:65–81, 2004.
3. Mila Nikolova, Selim Esedoglu, and Tony F. Chan. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J. App. Math.*, 66(5):1632–1648, 2006.
4. J. Yuan, E. Bae, X.-C. Tai, and Y. Boycov. A study on continuous max-flow and min-cut approaches. Technical report CAM-10-61, UCLA, 2010.
5. J. Yuan, E. Bae, and X.C. Tai. A study on continuous max-flow and min-cut approaches. In *CVPR*, USA, San Francisco, 2010.