BioImage Suite Web

Xenophon Papademetris and Dustin Scheinost
Dept. of Radiology & Biomedical Imaging
Yale School of Medicine

Adapted from a talk given at Yale, October 2018
What is It?

• BioImage Suite ported to run in a browser

• No SERVER – everything is running locally

• Desktop version (with essentially embedded browser) also available

• Ported over some of the functionality from the “legacy” version
Video: Cheryl Lacadie & Haley Garbus

https://youtu.be/zWS68aOWUBQ
Welcome to BioImage Suite Web.

- Select any tool from the Applications menu above, or simply click on the application picture, to open the application. Please visit our download site for commandline and desktop versions of the software.

- This software is not approved for clinical use. Use at your own risk.

- BioImage Suite Web is written in Javascript and has been mostly tested on Google Chrome. All applications are running on your local machine. No data is ever uploaded to any server.

- We gratefully acknowledge support from the NIH Brain Initiative under grant R24 MH114805 (Papademetris X. and Scheinost D. Pts, Dept. of Radiology and Biomedical Imaging, Yale School of Medicine.)

If you are looking for the old desktop based BioImage Suite software you may download it from the Legacy BioImage Suite Webpage.
BioImage Suite Versions

- 1995 – “Surface Editor” – cardiac segmentation on Silicon Graphics (MOTIF, Open Inventor)

- 2001 – Moved to multiplatform and adapted to brain use (with Todd) (VTK, Tcl)

- 2009 – 2017 : Many failed attempts to port to web

- 2017 – BioImage Suite Web (JavaScript/WebGL)
Why Web?

• Nothing to install – just navigate to www.bioimagesuite.org

• Works on almost anything (best on Chrome) (try it on your phone!)

• Software is always up-to-date

• But web based does not preclude desktop – we provide
  • Command line apps
  • Desktop Apps
  • Web Applications

• Same code, (almost) same functionality
What can it do?

• Provide six applications (plus two hidden ones!)

• Overlay images to create image visualizations

• Connectivity Visualizations

• Interactive Image Segmentation

• Run various image processing, segmentation and registration algorithms

• MNI2TAL Conversions
Connectivity Viewer

Inputs = Parcellation + Matrices
Overlay Viewer

Inputs = Anatomical + Functional Images
Image Editor

Interactive Segmentation
Image Editor II

VOI Analysis
Image Editor II

VOI Analysis
Single Viewer

Image Display + Processing
Dual Viewer

Image Display + Processing + Registration ...

A

B

C

D

E

F

G
Registration III
Welcome to BioImage Suite Web.

- Select any tool from the Applications menu above, or simply click on the application picture, to open the application. Please visit our download site for commandline and desktop versions of the software.
- This software is not approved for clinical use. Use at your own risk.
- BioImage Suite Web is written in Javascript and has been mostly tested on Google Chrome. All applications are running on your local machine. No data is ever uploaded to any server.
- We gratefully acknowledge support from the NIH Brain Initiative under grant R24 MH14805 (Papademetris X. and Scheinost D. Pi, Dept. of Radiology and Biomedical Imaging, Yale School of Medicine.)

For more information and to access the documentation, visit the BioImage Suite Web documentation page.
Module Testing II

Running Tests

Executing tests 0:87 (Max index=87). Only running tests with name=computeCorrelation

Test 3: computeCorrelation

- Command: computeCorrelation -i testdata/ButterWorthOutput.csv --zscore false
- Test details: --test_target testdata/newtests/goldcorrelation.matr --test_type matrix --test_comparison ssd --test_threshold 0.01
- Should pass: true

Test completed, now checking results.

Comparing matrix using ssd and threshold=0.01 Module computeCorrelation test passed.
deviation (ssd) from expected: 2.537576312709809e-7 < 0.01

Details


"command" : "computeCorrelation -i testdata/ButterWorthOutput.csv --zscore false",
"test" : "--test_target testdata/newtests/goldcorrelation.matr --test_type matrix"
Connectivity Display Testing
Testing Protocol for Bioimage Suite: MNI2TAL

**Background:** Imaging of the brain is a relatively new technique in medicine, neuroscience, and psychology. The Montreal Neurological Institute (MNI) template is used to identify reference spaces in imaging studies to identify structures and regions within the brain. The Talairach (TAL) Atlas is a mapping technique used to reduce the intersubject anatomic variability in mapping studies and allows for comparisons between coordinate-based data. Each point in the brain has both a set of MNI and Talairach coordinates.

**About BioImage Suite:** BioImage Suite is a collection of image analysis programs available as a software and online that can be used to view brain regions. One of the applications in BioImage Suite, MNI2TAL, is used to provide the MNI and Talairach coordinates of a given point in the brain. Given any MNI coordinate, the program outputs the Talairach equivalent coordinate, and for any Talairach coordinate, the program provides the corresponding MNI.

Testing MNI2TAL on a Mac:

1. Open the following hyperlink [http://bioidicesuiteweb.github.io/unstableapp/](http://bioidicesuiteweb.github.io/unstableapp/),

   ![Image 1](http://bioidicesuiteweb.github.io/unstableapp/

   ![Image 2](http://bioidicesuiteweb.github.io/unstableapp/)

   a. Find and click on “3. The MNI 2 Talairach Converter program”
Documentation

BiolImage Suite Web User Documentation

Table Of Contents

First we cover some core information in this document:

- Starting BiolImage Suite Web – how to get and run the software.
- Some Key Information – information about configuring your browser and default image orientations.

The following documents describe the different applications in BiolImage Suite Web.

- Displaying Images – we discuss here how the core viewer in BiolImage Suite web works (including colormapping etc.). This is shared by the almost all the applications.
- The Overlay Viewer Application – an application optimized for displaying functional overlays, including a brief description of the Mosaic Viewer Tool included here.
- The Image Editor Application – an application that can be used for interactive segmentation and VOI analysis of images.
- The Connectivity Visualization Tool – an application to visualize connectome matrices.
- The MNI2TAL Tool – an application to map MNI to Talairach coordinates and vice-versa.
- The Dual Viewer Tool – an advanced dual viewer application. This is intended to be primarily used for image registration tasks.

These documents describe how to perform specific tasks in one or more of the applications in the software.

- Image Processing Tasks – here we describe how to perform basic image processing tasks (e.g. image thresholding, cropping, resampling) using the tools in BiolImage Suite Web.
- Image Segmentation – this describes image segmentation functionality in our software.
- Image Registration – the image registration tools are described in the document describing the
BioImage Suite Web User Documentation

Image Defacing

This is a common procedure required to fully anonymize an MR brain image. This procedure masks out the face in an MRI making it impossible to “recognize” the subject (if the image is volume rendered or the face surface is extracted). Our implementation derives from the OpenfMRI Project. We use some data from their source code as well.

Note: If you need help with the basics of Bioimage Suite Web's viewers, please see the intro to viewers.

Method 1 — Simple Version

https://youtu.be/a8Cbs4SJABs
Key Technological (Software) Innovations

• Multi-context: same code commandline (via Node.js), desktop (via Electron) and Web
  • Each context has unique strengths
  • Tune to technical sophistication of the user and her needs
• Formal module architecture for automatic GUI and command line implementations of algorithmic modules.
• WebAssembly implementation of computationally expensive code (C++ compiled to WASM) e.g. Non Linear Registration ~90% native performance
• Custom Web Elements to modularize the applications
• No server, all computation is done in the client (your browser)
• A lot of this work builds on our previous experience with the Yale BioImage Suite (www.bioimagesuite.org) software package ➔ This is really the web version of this.
Software Architecture

Web-based (or Desktop) Applications modularized as Custom Web Elements (Browser or Electron)

JavaScript Computational Layer (Modules)

JavaScript Wrapper Layer

Data Serialization/deserialization Layer in C++

Low-Level Processing in C++ (compiled either to WebAssembly or native shared library)

JavaScript Command Line Tools (Node.js)

Matlab Wrapper Layer

Matlab Scripts

Python Scripts

Python Wrapper Layer

JavaScript I/O Layer (Cloud, Network, Database, File)
C++/Web Assembly

- ~18,000 lines of C++ code (per wc) (much of this adapted from BioImage Suite but stripped of dependencies to VTK/ITK/..)
- Only external dependency is Eigen numerical library
- Compile into bytecode using Emscripten
- Serialization/Transfer layer
  - JS \(\leftrightarrow\) C++ data transfer is restricted to simple “C”-style objects, essentially numbers and arrays
C++ JS Integration

JS Code (or Python Code) → Auto-Generated JS Interface Code → C++ Interface Code → C++ Computational Code

/** Compute butterworthFilter Output
 * @param input the input matrix to filter (time = rows)
 * @param jsonstring the parameters { "type": "low", "cutoff": 0.15, 'sampleRate': 1.5 }; 
 * @param debug if > 0 print debug messages 
 * @returns a pointer to the filtered matrix (rows=frame, cols=rois) 
 */

// BIS: { 'butterworthFilterWASM', 'Matrix', [ 'Matrix', 'ParamObj', 'debug' ] } 
BISEXPORT unsigned char* butterworthFilterWASM(unsigned char* input, const char* jsonstring, int debug);
JS Code

• 45,000 lines of JS Code

• Abstraction of File I/O

• Computational Modules
  • Automatic user interface generation
  • Automatic commandline generation

• User Interface packaged as Custom Web Elements
**Formal Module Architecture**
(Inputs, Outputs, Parameters specified using JSON)

**JS**

```javascript
class SmoothImageModule extends BaseModule {

  constructor() {
    super();
    this.name = 'smoothImage';
  }

  execute(vals) {
    console.log('oooo executing: smoothImage with vals', vals);
    return new Promise((resolve, reject) => {
      let input = this.inputs['input'];
      let s = parseFloat(vals.sigma);
      biswrap.initialize().then(() => {
        this.outputs['output'] =
          biswrap.gaussianSmoothImageWASM(input, {
          "sigmas": [s, s, s],
          "inmm": super.parseBoolean(vals.inmm),
          "radiusfactor": parseFloat(vals.radiusfactor),
        }, super.parseBoolean(vals.debug));
        resolve();
      }).catch((e) => {
        reject(e);
      });
    });
  }
}
```

**Python**

```python
class smoothImage(bis_basemodule.baseModule):

def __init__(self):
  super().__init__();
  self.name='smoothImage';

def execute(self,vals):
  print('oooo executing: smoothImage with vals', vals);
  input = self.inputs['input'];
  s = (vals['sigma']);
  try:
    self.outputs['output'] =
      libbis.gaussianSmoothImageWASM(input, {
        "sigmas": [s, s, s],
        "inmm": self.parseBoolean(vals['inmm']),
        "radiusfactor": vals['radiusfactor'],
      }, debug=self.parseBoolean(vals['debug']))
  except:
    return False
  return True
```
Data Provenance

• Store data provenance information as JSON-extensions to files

• Who generated this file, when, where, with what and how?

• For images embed the JSON info as extensions to NIFTI Headers

• For matrices and transformations we are formalizing new JSON based file formats

• Same applies to higher level outputs (e.g. motion parameters)
```json
{
    "bisformat": "BisDataObjectCollection",
    "filename": "testdata/test_motion_correction__test_motion_correction__mot.json",
    "comments": [
        
    ],
    "ModuleOutput": {
        "command": "/usr/bin/node /home/xenios/javascript/bisclipjs/bisclip.js motionCorrection -t testdata/test_motion_correction_correction_nii.gz --doreslice true --o /home/xenios/winhome/Desktop/motionparams.json --resliced /home/xenios/winhome/Desktop/motioncorrect.nii.gz",
        "output": "output",
        "parameters": {
            "dorese": true,
            "default": true,
            "intscale": 1,
            "nups": 1024,
            "extrasmoothing": 0,
            "metric": "CC",
            "optimization": "Hillclimb",
            "steps": 0.25,
            "levels": 3,
            "iterations": 32,
            "resolution": 1.01,
            "debug": false,
            "steps": 4,
            "nups": 0,
            "systeminfo": {
                "os": "linux",
                "arch": "x64",
                "hostname": "2230pc",
                "user": "xenios",
                "date": "2018-04-02T0:13:52.377Z",
                "nodeversion": "v8.9.4",
                "biswebversion": "04/02/2018"
            },
            "numitems": 5,
            "itemlist": [
                {
                    "type": "transform",
                    "data": "\"BisLinearTransformation\"",
                    "filename": "\"identity.mat\"",
                    "comments": [],
                    "matrix": [[0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]],
                    "parameters": [0, 0, 0, 0, 0],
                    "metadata": {
                        "Frame": 0
                    }
                },
                {
                    "type": "transform",
                    "data": "\"BisLinearTransformation\"",
                    "filename": "\"identity.mat\"",
                    "comments": [],
                    "matrix": [[0.904924571542196,
                        0.42557207894516, 0.6154688262939453],
                        [0.42557207894516, 0.904924571542196,
                        0.141296, -39.15327453613281], [0.0, 0.0, 0.0, 1]],
                    "parameters": [2.65124988559082, -1.7674999237060547, 0.0, 0.0, -25.18687438964838],
                    "metadata": {
                        "Frame": 1
                    }
                },
                {
                    "type": "transform",
                    "data": "\"BisLinearTransformation\"",
                    "filename": "\"identity.mat\"",
                    "comments": [],
                    "matrix": [[0.904924571542196,
                        0.42557207894516, 0.6154688262939453],
                        [0.42557207894516, 0.904924571542196,
                        0.141296, -39.15327453613281], [0.0, 0.0, 0.0, 1]],
                    "parameters": [-1.7674999237060547, 2.65124988559082, 0.0, 0.0, -25.18687438964838],
                    "metadata": {
                        "Frame": 2
                    }
                },
                {
                    "type": "transform",
                    "data": "\"BisLinearTransformation\"",
                    "filename": "\"identity.mat\"",
                    "comments": [],
                    "matrix": [[0.9081797003746033,
                        -0.4185804724693293, 0.945373553515625],
                        [0.1, 0.0, 0.0, 0.0], [0.4185804724693293,
                        -0.9081797003746033, 0.945373553515625, 0.1],
                        [0.0, 0.0, 0.0, 1]],
                    "parameters": [0.837499618530273, -1.7674999237060547, 0.0, -24.74499893184766, 0],
                    "metadata": {
                        "Frame": 3
                    }
                }
            ]
        }
    }
}
```
Data Format Example

```json
{
    "bisformat": "BisDataObjectCollection",
    "filename": "testdata/test_motion_correction__test_motion_correction__mot.json",
    "comments": [
        {
            "ModuleOutput": {
                "command": "/usr/bin/node /home/xenios/javascript/bislib/js/bin/bisweb.js motionCorrection -t testdata/test_motion_correction -n.nii.gz -r testdata/test_motion_correction.nii.gz --doreslice true --output /nhome/Desktop/motioncorrect.nii.gz",
                "output": "output",
                "parameters": {
                    "doreslice": true,
                    "norm": true,
                    "intscale": 1,
                    "numbins": 1024,
                    "extrasmoothing": 0,
                    "metric": "CC",
                    "optimization": "HillClimb",
                    "stepsize": 0.25,
                    "levels": 3,
                    "iterations": 32,
                    "resolution": 1.01,
                    "debug": false,
                    "steps": 4,
                    "refno": 0
                }
            },
            "systeminfo": {
                "os": "linux",
                "arch": "x64",
                "hostname": "z230pc",
                "user": "xenios",
                "date": "2018-04-02T20:13:52.377Z",
                "nodeversion": "v8.9.4",
                "biswebversion": "04/02/2018"
            }
        }
    ]
}
```
Data Format Example

```json
{
    "bisformat": "BisDataObjectCollection",
    "filename": "testdata/test_motion_correction__test_motion_correction__mot.json",
    "comments": [
        {
            "ModuleOutput": {
                "command": "/usr/bin/node /home/xenios/javascript/bslib/js/bin/bisweb.js motionCorrection -t testdata/test_motion_correction__mot.json -o testdata/test_motion_correction__mot.__mot.json --doreSlice true --outputPath /home/xenios/winhome/Desktop/motionparams.json --resliced /home/xenios/winhome/Desktop/motioncorrect.nii.gz",
                "output": "output",
                "parameters": {
                    "doreSlice": true,
                    "norm": true,
                    "intscale": 1,
                    "numbins": 1024,
                    "extrasmoothing": 0,
                    "metric": "CC",
                    "optimization": "HillClimb",
                    "stepsize": 0.25,
                    "levels": 3,
                    "iterations": 32,
                    "resolution": 1.01,
                    "debug": false,
                    "steps": 4,
                    "refno": 0
                }
            },
            "systeminfo": {
                "os": "linux",
                "arch": "x64",
                "hostname": "z230pc",
                "user": "xenios",
                "date": "2018-04-02T20:13:52.377Z",
                "nodeversion": "v8.9.4",
                "biswebversion": "04/02/2018"
            }
        }
    ]
}
```
Some Limitations

• There are some cases where the web (alone) is not enough

• Reading multiple files from disk at the same time is not easy to do in a browser
  • Each file|open and file|save operation must be approved by the user (security)

• Solution I – Cloud Storage (Amazon S3 and Local Server)
  • We provide a simple script to allow browser to access your filesystem directly

• Solution II – Desktop Version
  • Electron-based Desktop Apps
Desktop Apps

- Must be downloaded + installed
- Essentially package a version of Chrome (Electron) and our code
- Can access filesystem directly
- Looks more or less same as web-based version
Command Line Apps

- Some times it is useful to “batch” process data

- We provide command line applications of the same modules as in the web-based version (same exact code)

- Download and Install
- Also need node.js (commandline JS interpreter)
Some Cool Figures

- Simultaneous fMRI Ca(2+) Imaging Project (Crair, Constable PIs)
- Images using data from Eve Lake and Xinxin GE
- Custom code to merge 2D Projected Images and 3D MRI
Mapping 2D Optical to fMRI (via Angio MRI)
Mapping Parcellations from Optical to MRI
Lots more

• Google Drive and Dropbox integration (read only)
• Progressive Web Application (offline use via caching)
• All code on github
• Lots of programmer documentation on JS/WebAssembly etc. on Github (under docs).
• Python and Matlab (early) integration of computational code.
• Paravision Import functionality for small animal MRI
• Port of old diff-SPECT code for Epilepsy
• Landmark placement and editing
• Interactive Registration
• Copy & Paste Viewers across applications (via browser cache)
• Arbitrary Image Orientation and Registration (plus force “on load” if desired)
Current State

• Software is available (web, desktop, source code etc.)

• Closing in on v1.0 release (0.99!) – probably next week
  • Focus has been on getting low level functionality in place

• Next
  • Data organization (DICOM, BIDS)
  • Processing pipeline for resting state fMRI and optical Ca 2+ images
  • Better cloud integration
  • Machine learning via tensorflow.js
  • Form a basis for all future development work
Acknowledgments

• Zach Saltzman
• John Onofrey
• Cheryl Lacadie
• Haley Garbus
• Dana Casetti
• Daniel Coman
• Basav Ganganna
• Pranav Chandrasekaran
• Users, testers, ...

• Eigen
• CMake/CTest/CPack
• Emscripten
• Node.js
• Gulp
• Webpack
• Bootstrap
• JQuery
• dat.gui
• Three.js
• Electron
• Electron-Packager

We gratefully acknowledge support from the NIH Brain Initiative under grant R24 MH114805