TIMBIR Documentation

Release 0.0.2

Purdue University

April 15, 2016

Contents

The Time-Interlaced Model-Based Iterative Reconstruction (TIMBIR) is a method for 4D time-space reconstruction of data acquired using synchrotron X-ray computed tomography.

This guide is maintained on GitHub.

The Time-Interlaced Model-Based Iterative Reconstruction (TIMBIR) [*[A3]*, *[A2]*] is a method for 4D time-space reconstruction of data acquired using synchrotron X-ray computed tomography. TIMBIR is a synergistic combination of two innovations. The first innovation, interlaced view sampling, is a novel method of data acquisition which distributes the view angles more evenly in time. The second innovation is a 4D model-based iterative reconstruction algorithm (MBIR) which can produce time-resolved volumetric reconstruction of the sample from the interlaced views. In addition to modeling both the sensor noise statistics and the 4D object, the MBIR algorithm also reduces ring and streak artifacts by more accurately modeling the measurement non-idealities [*[A3]*, *[A2]*, *[A1]*].

CHAPTER 1

Install directions

This section covers the basics of how to download and install TIMBIR.

Contents:

1.1 Dependencies

- MPI compiler (Intel or Open MPI)
- OpenMP
- make utility

1.2 Dependencies Install

- Install MPI (Here I use openMPI as a example):

```
$ wget https://www.open-mpi.org/software/ompi/v1.10/downloads/openmpi-1.10.2.tar.gz
$ tar xvzf openmpi-1.10.2.tar.gz
$ cd <openmpi path>
$ ./configure --prefix=<your mpi install path>
$ make all install
```

- Install HDF5 library. First go to https://www.hdfgroup.org/HDF5/release/obtainsrc.html#conf to find the appropriate hdf5 library for your platform (here we use hdf5-1.8.16.tar as example:

```
$ tar xvf hdf5-1.8.16.tar
$ cd <hdf5 path>
$ ./configure --prefix=/clhome/KYUE/lib/hdf5 --enable-fortran --enable-cxx
$ make
$ make install
```

- Set your library path with HDF5 library and MPI library (here we use bash as example):

```
    $ vi env.sh (create a bash script)
    $ export HDF5_BASE=<hdf5 full path>
    $ export MPI_BASE=<MPI full path>
    $ export PATH = ${MPI_BASE}/bin:${HDF5_BASE}/bin:$PATH
    $ export LD_LIBRARY_PATH= = ${MPI_BASE}/lib:${HDF5_BASE}/lib64:$PATH
    $ source env.sh
```

1.3 Compiling TIMBIR

To compile the MBIR algorithm code:

```
$ git clone https://github.com/adityamnk/timbir.git timbir
$ cd timbir/src/MBIR_4D
```

This generates library files in timbir/src/lib. For more information, read the README in timbir/src/MBIR_4D.

1.4 Running the reconstruction algorithm

If the input data format is a standard binary, run the code in timbir/src/reconstruct/bin_data. For more information on data format and running the code, read the README in timbir/src/reconstruct/basic:

```
$ cd timbir/src/reconstruct/bin_data
$ make
```

This generates executables in the same folder.

If the input data is in HDF format used at APS, run the code in timbir/src/reconstruct/aps_data. For more information on data format and running the code, read the README in timbir/src/reconstruct/aps_data:

```
$ cd timbir/src/reconstruct/aps_data
$ make
```

This generates executables in the same folder.

If the input data is in standard HDF format, run the code in timbir/src/reconstruct/std_data. For more informantion on data format and running the code, read the README in timbir/src/reconstruct/std_data:

```
$ cd timbir/src/reconstruct/std_data
$ make #Generates executables in the same folder
```

CHAPTER 2

Data Collection

This section covers the basics of how to collect the data that will be analysed by TIMBIR including data collection scripts run at the APS.

Contents:

2.1 to be completed

CHAPTER 3

Examples

In this section, we list scripts that can be used to compile the code and run the reconstruction algorithm. Examples are provided for both 3D and 4D reconstruction on either a unix based system (Linux/Mac) or a super-computing cluster.

3.1 3D Reconstruction of Shepp-Logan Phantom

- **Unix/Linux/Mac OS::** Run the script timbir/demo/recon_3d/shepp-logan-3D/run_unix.sh
- **Super-computing cluster (Rice cluster at Purdue)::** Run the script timbir/demo/recon_3d/shepp-logan-3D/run_cluster.sh

3.2 4D Reconstruction of Cahn-Hilliard Phantom

- **Unix/Linux/Mac OS::** Run the script timbir/demo/recon_4d/cahn-hilliard-4D/run_unix.sh
- **Super-computing cluster (Rice cluster at Purdue)::** Run the script timbir/demo/recon_4d/cahn-hilliard-4D/run_cluster.sh

CHAPTER 4

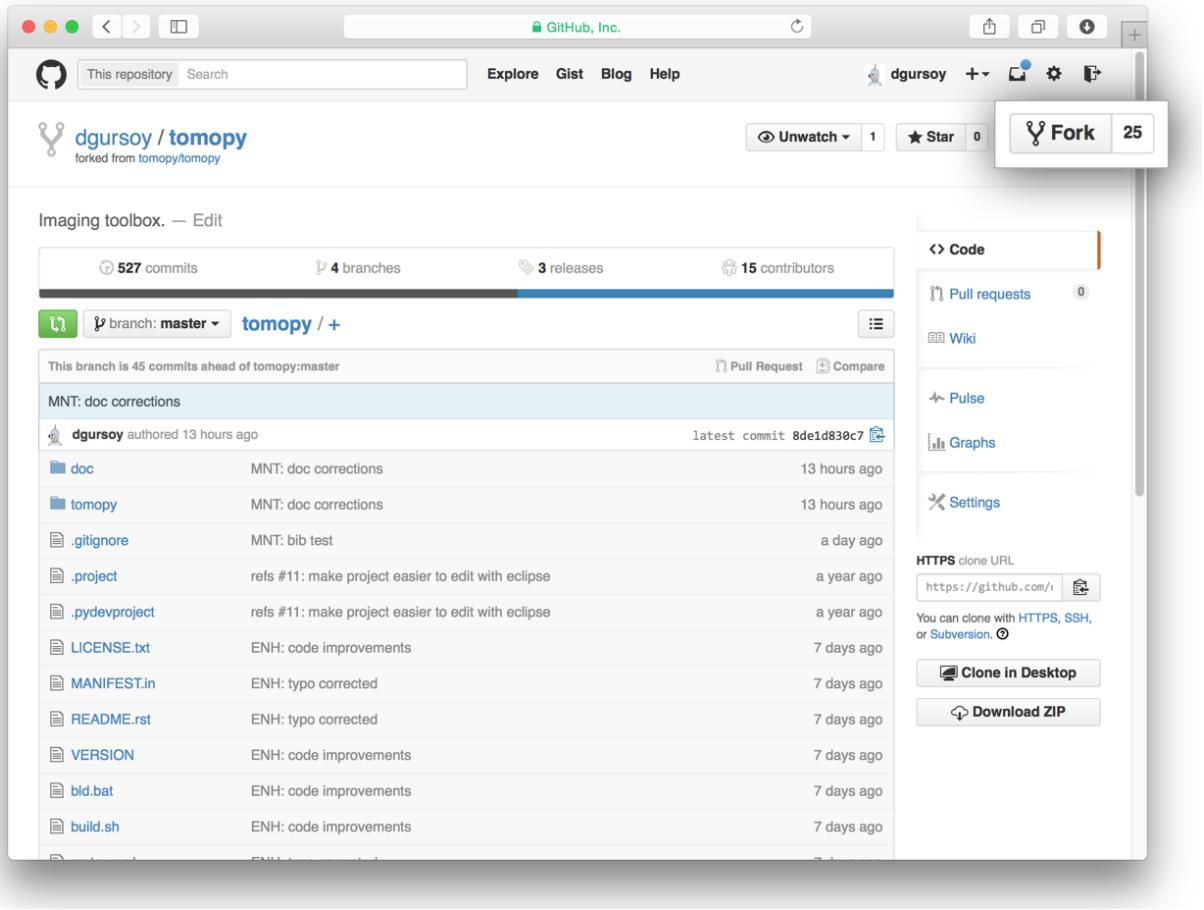
Development

This section explains the basics for developers who wish to contribute to the TIMBIR project.

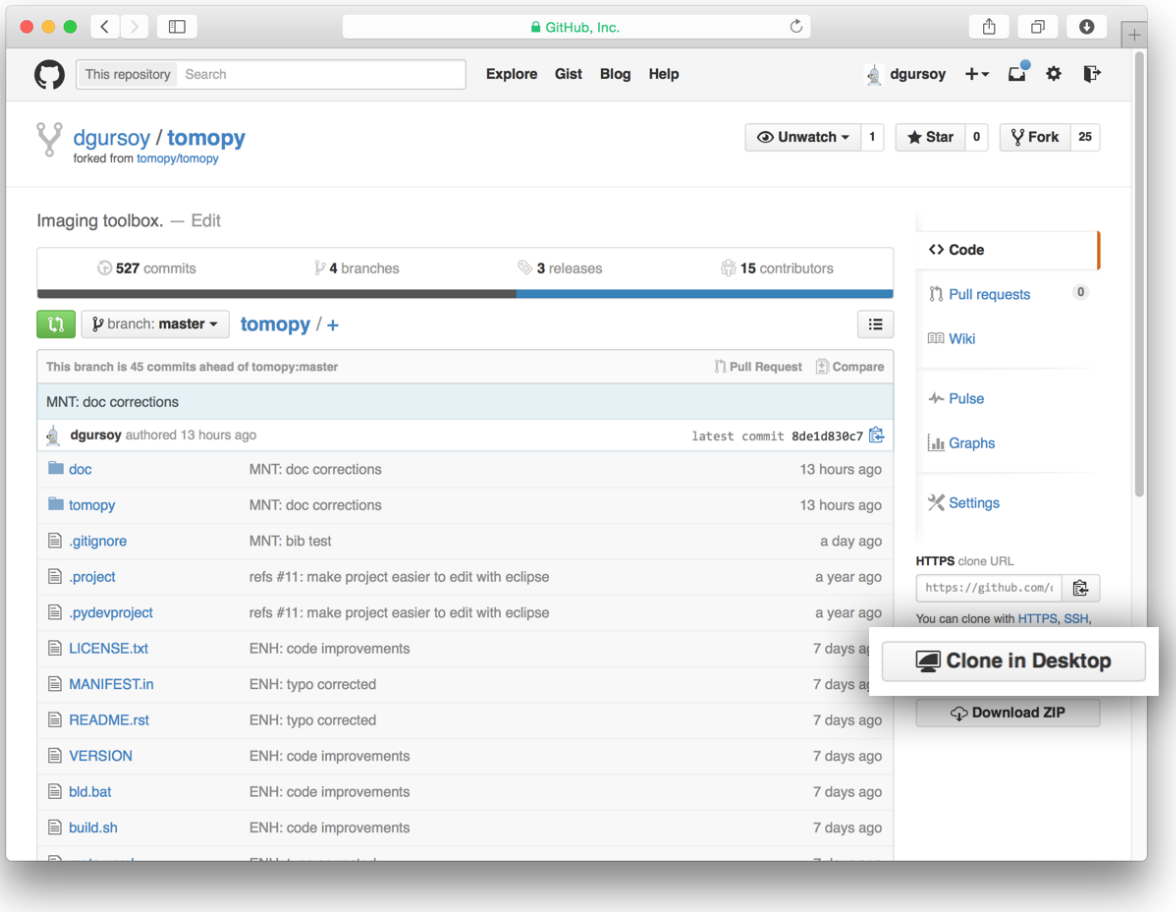
Contents:

4.1 Cloning the repository

The project is maintained on GitHub, which is a version control and a collaboration platform for software developers. To start first register on GitHub and fork the TIMBIR repository by clicking the **Fork** button in the header of the TIMBIR repository:



This successfully creates a copy of the project in your personal GitHub space. The next thing you want to do is to clone it to your local machine. You can do this by clicking the **Clone in Desktop** button in the bottom of the right hand side bar:



This will launch the GitHub desktop application (available for both Mac and Win) and ask you where you want to save it. Select a location in your computer and feel comfortable with making modifications in the code.

4.2 Coding conventions

We try to keep a consistent and readable code. So, please keep in mind the following style and syntax guidance before you start coding.

First of all the code should be well documented, easy to understand, and integrate well into the rest of the project. For example, when you are writing a new function always describe the purpose and the parameters:

```
def my_awesome_func(a, b):
    """
    Adds two numbers.

    Parameters
    ----------
    a : scalar (float)
        First number to add

    b : scalar (float)
        Second number to add

    Returns
```

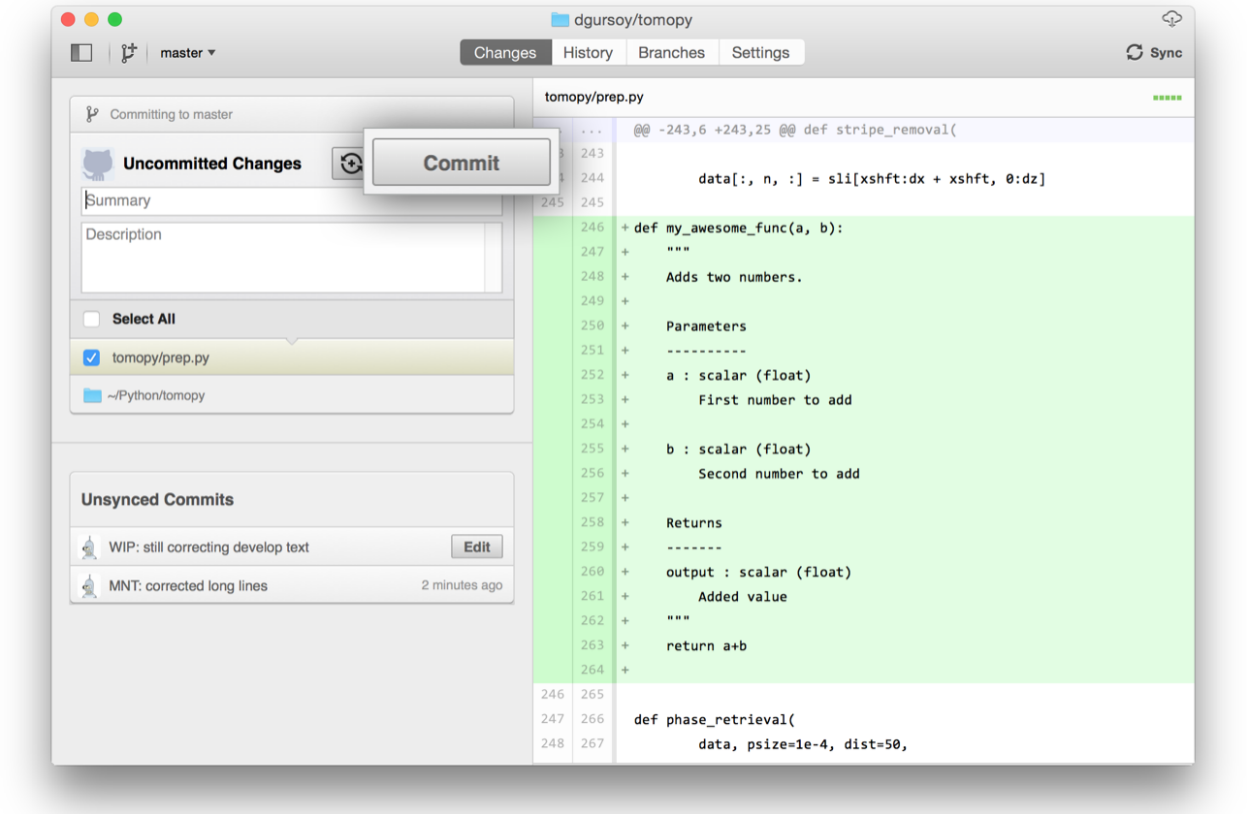
```
    -------
    output : scalar (float)
        Added value
    """
    return a+b
```

4.3 Package versioning

We follow the X.Y.Z (Major.Minor.Patch) semantic for package versioning. The version should be updated before each pull request accordingly. The patch number is incremented for minor changes and bug fixes which do not change the software's API. The minor version is incremented for releases which add new, but backward-compatible, API features, and the major version is incremented for API changes which are not backward-compatible. For example, software which relies on version 2.1.5 of an API is compatible with version 2.2.3, but not necessarily with 3.2.4.

4.4 Commiting changes

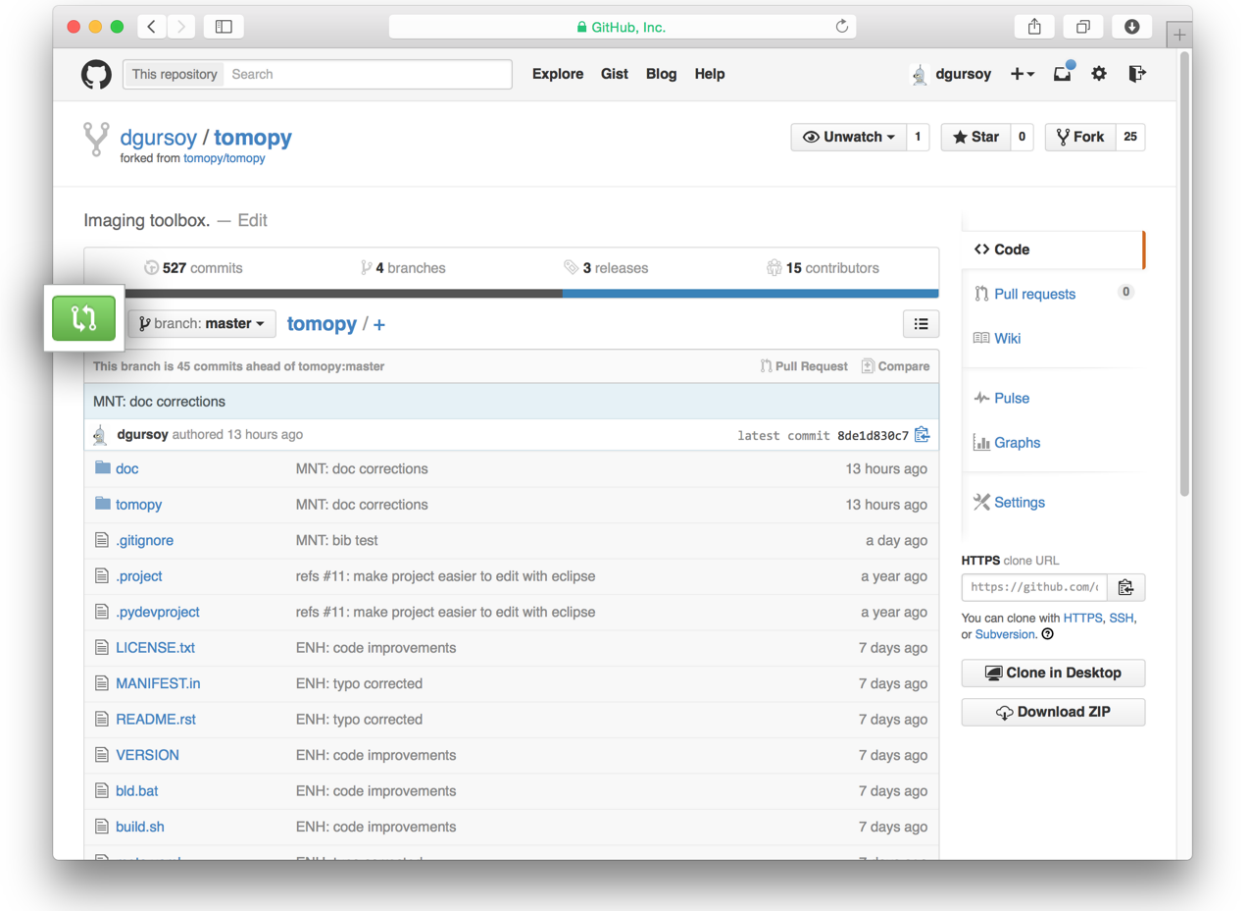
After making some changes in the code, you may want to take a *snapshot* of the edits you made. That's when you make a *commit*. To do this, launch the GitHub desktop application and it should provide you all the changes in your code since your last commit. Write a brief *Summary* and *Description* about the changes you made and click the **Commit** button:



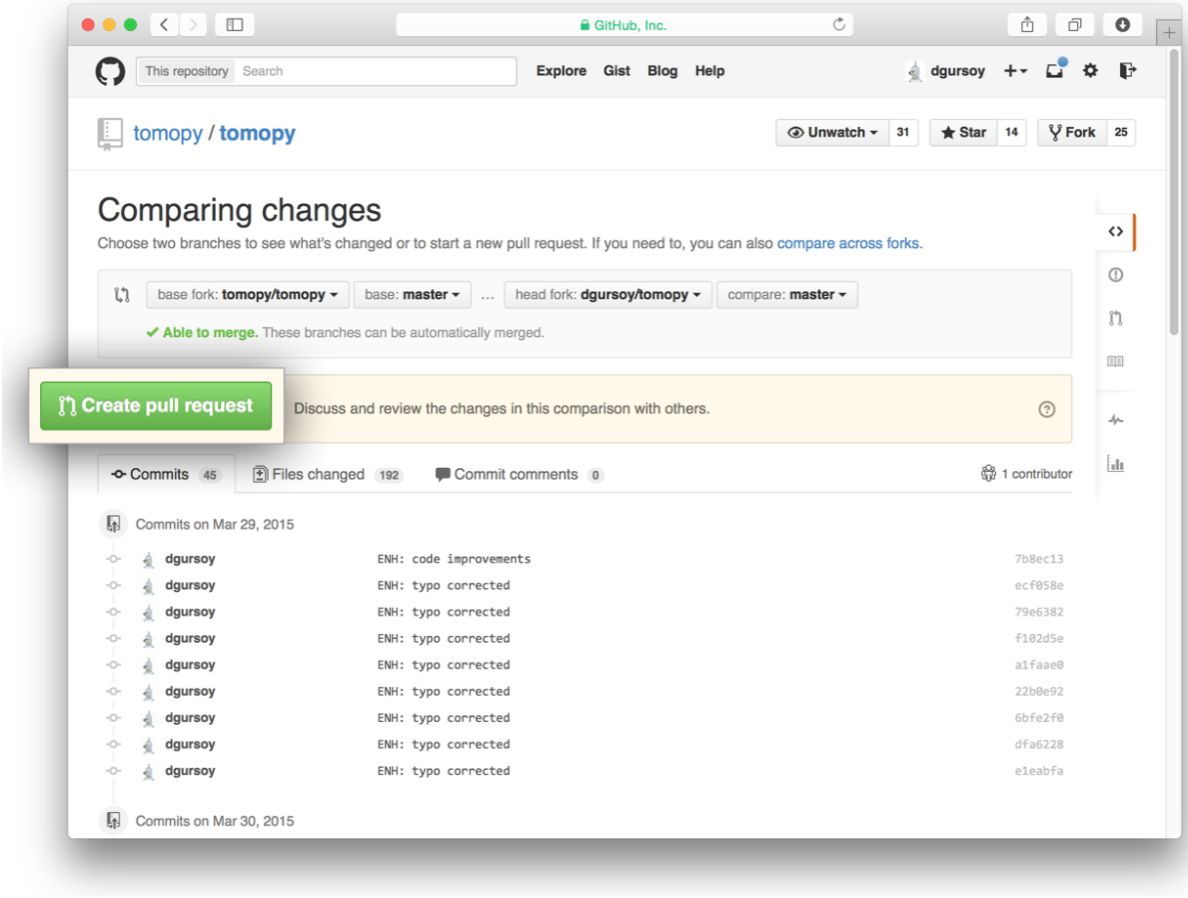
You can continue to make changes, add modules, write your own functions, and take more *Commit snapshots* of your code writing process.

4.5 Contributing back

Once you feel that the functionality you added would benefit the community, then you should consider contributing back to the TIMBIR project. For this, go to your online GitHub repository of the project and click on the *green* button to compare, review and create a pull request.



After clicking on this button, you are presented with a review page where you can get a high-level overview of what exactly has changed between your forked branch and the original TIMBIR repository. When you're ready to submit your pull request, click **Create pull request**:



Clicking on **Create pull request** sends you to a discussion page, where you can enter a title and optional description. It's important to provide as much useful information and a rationale for why you're making this Pull Request in the first place.

When you're ready typing out your heartfelt argument, click on **Send pull request**. You're done!

CHAPTER 5

Frequently asked questions

Here's a list of questions.

Questions

5.1 How can I report bugs?

The easiest way to report bugs or get help is to open an issue on GitHub. Simply go to the project GitHub page, click on Issues in the right menu tab and submit your report or question.

CHAPTER 6

Credits

We kindly request that you cite the following articles if you use TIMBIR:

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Bibliography

[A1] K.A. Mohan, S.V. Venkatakrishnan, L.F. Drummy, J. Simmons, D.Y. Parkinson, and C.A. Bouman. Model-based iterative reconstruction for synchrotron x-ray tomography. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, 6909–6913. May 2014. doi:10.1109/ICASSP.2014.6854939.

[A2] K.A. Mohan, S.V. Venkatakrishnan, J.W. Gibbs, E.B. Gulsoy, Xianghui Xiao, M. De Graef, P.W. Voorhees, and C.A. Bouman. 4d model-based iterative reconstruction from interlaced views. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, 783–787. April 2015. doi:10.1109/ICASSP.2015.7178076.

[A3] Kadri Mohan, Singanallur Venkatakrishnan, John Gibbs, E Gulsoy, Xianghui Xiao, Marc De Graef, Peter Voorhees, and Charles Bouman. Timbir: a method for time-space reconstruction from interlaced views. *IEEE Transactions on Computational Imaging*, 2015.