CurveLab Toolbox, Version 2.0

Emmanuel Candès, Laurent Demanet, Lexing Ying
Applied and Computational Mathematics
California Institute of Technology, Pasadena CA 91125

1 Introduction

CurveLab is a collection of Matlab and C++ programs for the *Fast Discrete Curvelet Transform* in two and three dimensions.

For the 2d curvelet transform, the software package includes two distinct implementations: the wrapping-based transform and the transform using unequally-spaced fast Fourier transform (USFFT). Both variants are based on the Curvelet transform as described in 'New Tight Frames of Curvelets and Optimal Representations of Objects with Piecewise C^2 Singularities', *Comm. Pure Appl. Math.* **57** (2004) 219-266. The implementation is also discussed in detail in the Technical Report "Fast Discrete Curvelet Transforms" available at `http://www.curvelet.org`. We advise users to become familiar with these references.

The two implementations differ by the choice of spatial grid used to translate curvelets at each scale and angle.

- The USFFT version uses a decimated rectangular grid tilted along the main direction of each curvelet. There is one such grid per scale and angle, and this implementation is therefore very close to the definition given in the above reference. For the digital transform, tilting the grids induces a resampling of the Fourier transform on semi-regular grids, hence the use of a (perhaps novel) USFFT routine. For the inversion, a conjugate-gradient solver rapidly converges to the solution.

- The wrapping version uses, instead, a decimated rectangular grid aligned with the image axes. For a given scale, there are essentially two such grids (decimated mostly horizontally or mostly vertically). The resulting sampling is not as faithful to the original transform, but the basis functions are curvelets as much as in the USFFT-based implementation. Since no interpolation is necessary in the frequency plane, the transform is a numerical isometry and can be inverted by its adjoint.

For the 3d curvelet transform, the software in this package is an extension of the wrapping version in 2d. Due to the large size of the 3d data and the increasing redundancy of the curvelet transform, three different implementation are included:

- The in-core implementation which stores both the input data and the curvelet coefficients in the memory (suitable for small size data set).

- The out-core implementation which stores the input data in the memory and most of the curvelet coefficients on the disc (suitable for medium size data set).

- The MPI-based parallel implementation which distributes both input data and the curvelet coefficients on multiple nodes. The parallel implementation can successfully handle input data of size $1k \times 1k \times 1k$.

The C++ part of the package includes all the 2d and 3d implementations listed above. The Matlab part of this package includes only the 2d implementations of the USFFT and wrapping transforms.

2 Installation

Copy the CurveLab package into the directory of your choice, and expand it with

```
gunzip CurveLab-2.0.tar.gz
tar xvf CurveLab-2.0.tar
```

Matlab part. The Matlab implementation for the wrapping-based transform can be found in the directory `fdct_wrapping_matlab`, while the Matlab implementation for the USFFT-based transform is in the directory `fdct_usfft_matlab`. In both directories, you can find several demos which explain how to use the software. Before running the demos in the directory `fdct_usfft_matlab`, run `fdct_usfft_path.m` to set the correct searching path.

C++ part. The C++ implementation is tested on Unix-type platforms, including Linux, SunOS and MacOS. The installation requires g++ (version 3.2 or 3.3), and Matlab's mex compiler (version 13 or 14) if you want to compile the mex files for the C++ implementation (make sure the mex compiler use the same version of g++).

To install the C++ implementations (except the MPI-based parallel 3d transform)

- Download and install FFTW version 2.1.5 (`http://www.fftw.org`).
- In makefile.opt,
  set `FFTW_DIR` to be the path in which FFTW 2.1.5 is installed, and
  set `MEX` to be the correct mex compiler (optional).
- While in the main toolbox directory, type
  `make lib` to build the libraries for both the wrapping-based and the USFFT-based implementations
  `make test` to test the C++ installation (optional)
  `make matlab` to generate the mex files of the C++ implementation, which can be called in Matlab (optional)

For the 2d implementations, the source files of the "C++ wrapping version" are in the directory `fdct_wrapping_cpp/src`, while the mex files generated by `make matlab` and the demo files are in the directory `fdct_wrapping_cpp/mex`. The source files of the "C++ USFFT version" are in the directory `fdct_usfft_cpp/src`, while the mex files and the demo files are in the directory `fdct_usfft_cpp/mex`.

For the 3d implementations, the source files for the in-core version are in the directory `fdct3d/src`, while the mex files (generated by `make matlab` and the Matlab demo files are

in the directory `fdct3d/mex`. The source files of the out-core version are in the directory `fdct3d_outcore`.

To install the MPI-based parallel 3d implementation, please read `fdct3d_mpi/src/README` for details.

3 List of Matlab Files

The 2d Matlab implementations of the wrapping-based and USFFT-based transforms are in the directories `fdct_wrapping_matlab` and `fdct_usfft_matlab` respectively. The Matlab interface of the C++ implementations is located in the directories `fdct_wrapping_cpp/mex` and `fdct_usfft_cpp/mex`. Below is a (non-exhaustive) list of the `.m` files which can be found in those directories. `xxx` stands for either `wrapping` or `usfft`.

Basic transform functions.

- `fdct_xxx.m` – forward curvelet transform.
- `ifdct_xxx.m` – inverse curvelet transform.
- `afdct_xxx.m` – adjoint curvelet transform (only for USFFT-based transform since for the wrapping transform, the adjoint is the same as the inverse transform).
- `fdct_xxx_param.m` – this function returns the location of each curvelet in phase-space.

Useful Tools.

- `fdct_xxx_dispcoef.m` – a function which returns a matrix containing all the curvelet coefficients.
- `fdct_xxx_pos2idx.m` – for a fixed scale and a fixed direction, this function returns the index of the curvelet closest to a certain point in the image.

Demo Files.

- `fdct_xxx_demo_basic.m` – displays a curvelet in the spatial and frequency domain.
- `fdct_xxx_demo_recon.m` – partial reconstruction using the curvelet transform.
- `fdct_xxx_demo_disp.m` – displays all the curvelet coefficients of an image.
- `fdct_xxx_demo_denoise.m` – image denoising via curvelet shrinkage.

There are two extra Matlab demos in `fdct_wrapping_matlab` directory:

- `fdct_wrapping_demo_denoise_enhanced.m` – further techniques for image denoising.
- `fdct_wrapping_demo_wave.m` – wave propagation using curvelets.

An extra file `fdct_usfft_path.m` is included in `fdct_usfft_matlab` directory. This script needs to be called to append several subdirectories into the searching path.

The Matlab code for the in-core 3d transform are located in `fdct3d/mex`. Below is a list of the `.m` files.

Basic transform functions.

- `fdct3d_forward.m` – forward 3d curvelet transform.
- `fdct3d_inverse.m` – inverse 3d curvelet transform.
- `fdct3d_param.m` – a function which returns the location of each curvelet in phase-space.

Demo Files

- `fdct3d_demo_basic.m` – displays a curvelet in the spatial and frequency domain.

Extra information for each of these functions can be obtained by typing (at the Matlab prompt) `help` followed by the name of the function.

4 Changes

- Fixed bug in the 2d C++ implementation for input data with odd size.
- In 2d C++ implementation, changed the input data structure from class `CpxOffMat` to class `CpxNumMat`. The test files are changed accordingly.

5 Copyright and Contact

 For further information, please contact `curvelab@curvelet.org`.