K-Spapp

the K-Space app

Rudolf Springorum

Table of Contents

Introduction

K-Spapp is a simple tutorial to help understanding how K-Space works in MRI. Manipulate the K-Space of an MRI image by modifying parameters such as scan percentage, partial Fourier, K-Space shutters or add artificial noise. View the reconstructed image change in real-time.

Image Features:

- Change window width & level
- Zoom
- Import from Gallery
- Import from camera

K-Space features:

- Apply scan percentage
- Apply half scan factor
- Apply K-Space shutter
- Apply central K-Space shutter
- Apply Hanning filter
- Add noise
- Add motion
- Add spikes
- Add undersampling

K-Spapp is under active development by volunteers.

K-Spapp is free, has no-ads and does not collect any personal information.

Change request, future wishes

Functionality

- Done! Add spikes. Let the user pin point in K-space and generate a spike. Vary the intensity of the spikes. Keep track of spikes in an separate array such that they can be applied any time.
- Use sensors to change values:
  - Mic for adding noise (blow) or spikes (clap)
  - Shake for adding motion
- Add panning. Let the user move the image around when zoomed in.
- Add images. Allow the user to select other image

UX-Design

- Improve image information. Image information now is cluttered and very technical

Performance

- Use realForward(). ComplexForward() is twice as slow and consumes twice as much space. Requires redesign of K-space algorithms.
- Divide UI buttons in two groups: single values and [x,y] combinations. Replace the switch case() with simple if else statement. Saves lots of code lines
- Rewrite code using map(), norm() and constraint() functions

Version overview

1 – 1.0
- Initial version of K-Spapp
- Window, scan percentage, shutter

2 – 1.1
- Fixed: #1 DC offset when generating noise.

3 – 1.2
- Fixed: #2 Intermittent crash on launch.
- Fixed: #3 Bug on reset value while drag.
- Fixed: #4 Visualization of K-Space.
- Fixed: #5 Vertical drag now works also.

4 – 1.20
- Added: Zoom
- Added: Motion

5 – 1.20
- Fixed: #6 Correct implementation of half scan.

6 – 1.21
- Added: Select different images

7 – 1.22
- Added: Spikes

8 – 1.22
- Fixed: Full screen again

9 – 1.5
- New user interface
- Improved performance

10 – 1.51
- Load image from gallery
- Take image from camera

Bug list

#1 DC offset when generating noise.

Noise must have positive and negative values. The generated noise value has now equal distribution between -max < noise < max.

#2 Intermittent crash on launch.

A null exception was thrown at launch suggesting an attempt to draw an empty image. Images are loaded from disk. A short loop is added to check if image is loaded:

```
While (Img == null) delay();
```

#3 Bug on reset value while drag.

At the moment the button was pressed long enough to reset the value, almost immediately the value jumped to the one last used. This was caused by the mouse dragging routine which was changing the value even though the button was pressed. Fix: added a boolean to keep track is dragging happens on button or anywhere else.

#4 Visualization of K-Space.

The log function is too good, also the peripheral areas of K-space are very bright. The log function is replaced with a simple threshold (for the moment hard-coded) only showing a small range of K-space values.

#5 Vertical drag now works also.

Values can be changed by horizontal and vertical swipe, or combination.

```
MousePressed()
	downX = X - parameters[iParameter].down();
	downY = Y;

MouseDragged()
	parameters[iParameter].set(X-downX+Y-downY);
```

#6 Correct implementation of half scan.

Half scan uses conjugate symmetry which was not yet implemented. Symmetrical points in K-space have the following relation: Re[x,y] → Re[n-x,n-y] and Im[x,y] → -Im[n-x,n-y].

```
x2 = size – x
y2 = size – y
data[x][y*2]   =  data[x2][y2*2];   //Re
data[x][y*2+1] = -data[x2][y2*2+1]; //Im
```

Algorithms & Data formats

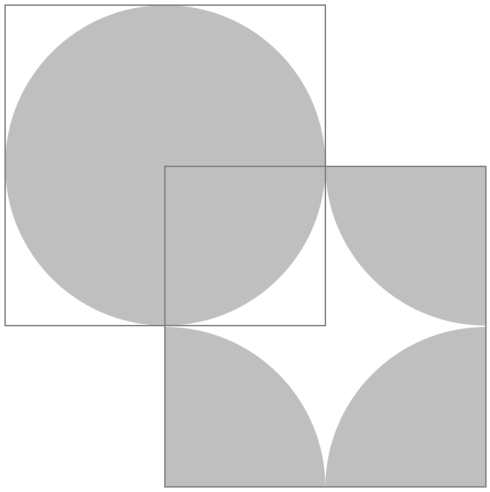
FFT: complexForward()

public void complexForward(double[][] a)

Computes 2D forward DFT of complex data leaving the result in a. The data is stored in 2D array. Complex data is represented by 2 double values in sequence: the real and imaginary part, i.e. the input array must be of size rows by 2*columns. The physical layout of the input data has to be as follows:

```
a[k1][2*k2]   = Re[k1][k2],
a[k1][2*k2+1] = Im[k1][k2], 0<=k1<rows, 0<=k2<columns,
```

Data format



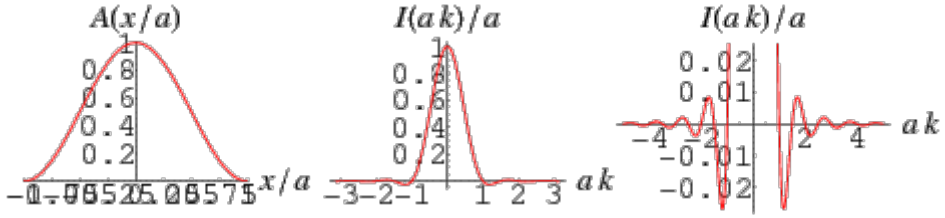
Upper left: K-space as it should be shown in the UI

Lower right: K-space as it is used by JTransforms

Hanning filter

An apodization function, also called the Hann function, frequently used to reduce leakage in discrete Fourier transforms. The illustrations above show the Hanning function, its instrument function, and a blowup of the instrument function sidelobes. It is named after the Austrian meteorologist Julius von Hann (Blackman and Tukey 1959, pp. 98-99). The Hanning function is given by:

$$A(x)=\cos^2\left(\frac{\pi x}{2a}\right) \qquad A(x)=\frac{1}{2}\left[1+\cos\left(\frac{\pi x}{a}\right)\right]$$



W: http://mathworld.wolfram.com/HanningFunction.html

Software used

Processing 3.2.1 (Linux-64)

Processing is a flexible software sketchbook and a language for learning how to code within the context of the visual arts. Since 2001, Processing has promoted software literacy within the visual arts and visual literacy within technology.

W: https://www.processing.org

JtransForms 3.1

JTransforms is the first, open source, multi-threaded FFT library written in pure Java. Currently, four types of transforms are available. The code is derived from General Purpose FFT Package written by Takuya Ooura and from Java FFTPack written by Baoshe Zhang.

W: https://github.com/wendykierp/JTransforms

Android Mode 0250 (3.0-RC4)

Processing for Android allows you to create Android apps using Processing. You can run your Processing sketches on Android devices with little or no changes in the code. You can also export your sketch as a signed package to upload to the Google Play Store.

W: http://android.processing.org/

Ke:tai v12

Let's you create mobile apps using Processing. Ke:tai is good at everything a mobile Android device does, and the desktop doesn't. It's a extensive library that gives you straight-forward access to sensors, cameras, and networking hardware.

W: http://ketai.org/

Android Studio 2.2.1

Android Studio provides the fastest tools for building apps on every type of Android device. World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.

W: https://developer.android.com/studio/index.html

Linux Mint 18 “Sarah” Cinnamon edition

Linux Mint 18 is a long term support release which will be supported until 2021. It comes with updated software and brings refinements and many new features to make your desktop even more comfortable to use.

W: https://www.linuxmint.com/

Inkscape 0.91

All icons designed in Inkscape,

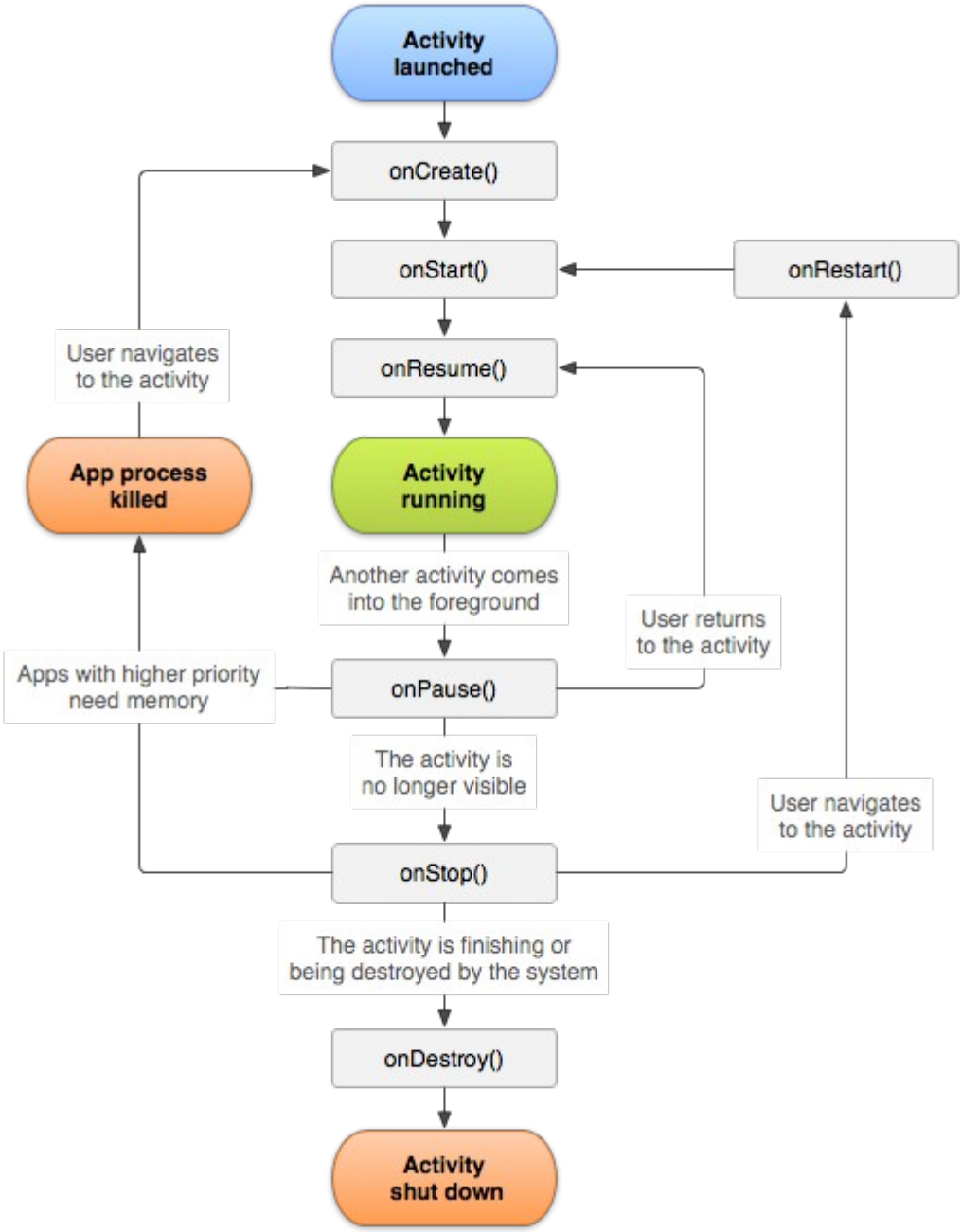
W: www.inkscape.org

Gimp 2.8.16

All MR images acquired with Philips Achieva and processed in Gimp.

W: www.gimp.org

Appendix A: Lifecycle android app



W: https://developer.android.com/reference/android/app/Activity.html