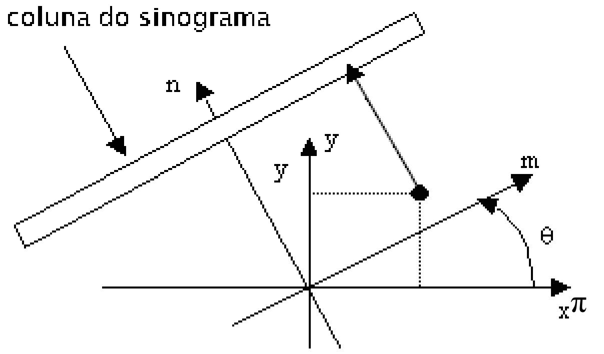
Radon transform and backprojection

1. Radon transform

The Radon (or projection) transform at angle θ corresponds to the line integrals of an image f(x,y) perpendicular to the direction θ. In practice, the angle is sampled uniformly between [-π/2 π/2]. The projection at the angle θ_k is stored in the sinogram p(k, m) column.



The coordinate (x,y) is obtained by rotating the coordinate grid (m,n) by an angle of θ around the center of the square image (c,c). The relation gives it:

$$\begin{pmatrix} x-c \\ y-c \end{pmatrix} = \begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix} \cdot \begin{pmatrix} m-c \\ n-c \end{pmatrix}$$

The transformRadon() method in the Radon.java file implements the Radon Transform of a square image. The method requires a 2D tween that is performed by the getInterpolatedPixel2D() method (linear interpolation).

Implementation Notes:

- To shorten compute time, the image is first copied into a 2D "array" that uses the method:
```
double[][] array = input.getArrayPixels();
double v = array[i][j]; // access to pixel (i,j)
```
- The default Java Math.floor() method is very slow, so we will use an available floor() method that runs faster.
- The image scan (index m, n) is restricted to a disk of diameter equal to the size of the image.

Understand the code and run on the test images (point.tif, lines.tif, circles.tif and phantom.tif) specifying different numbers of projections. By choosing the appropriate options in the dialog box, you can perform all three operations independently:
(1) Radon transform, (2) filtering, and (3) backprojection. Don't forget to select the correct input image.

2. Retro-projection transform.

The resulting image b(x,y) is obtained by backprojection of the sinogram p(k, m):

$$b(x,y) = \frac{\pi}{nbAngles}\sum_{k} p\big(k, m(x,y,\theta_k)\big)$$

with m expressed as a function of x, y, and θ_k.

Note that the backprojection is essentially the transposed flowchart of the transformRadon(). Code the inverse Radon() method that computes the inverse of the Radon transform of a sinogram in the Radon.java file.

The method requires a 1D (one-dimensional) tween that you must implement in the getInterpolatedPixel1D() method (linear tween).

To make a more efficient computational method, calculate a 2D "array" (double b[] []) and put this "array" in an ImageAccess at the end of the process.

```
double b[][] = new double[size][size];
  . . .
ImageAccess ReconstudedImage = new ImageAccess(b);
```

As imagens de teste são: point.tif, lines.tif, circles.tif e phantom.tif.

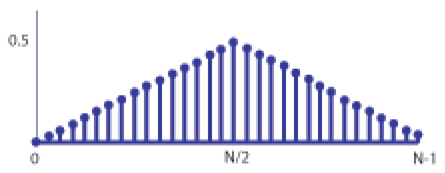
3. Filtered sinogram

We propose to implement three filters: (1) Ram-Lak and (2) Cosine in the Fourier domain, and (3) Laplacian in the spatial domain.

3.1 Ram-Lak filter

The Ram-Lak filter is defined by its frequency response H(w) = | w |. The applyRamLakFilter() method rotates by the angles (k) and filters each column of the sinogram through multiplication with the transfer function in the Fourier domain.

Encode the generateRamLakFilter() method, this returns an order that corresponds to the following diagram.



Implementation Note:
The fft.transform(real, imaginary), fft.inverse(real, imaginary) methods process the data locally (by reference); This means that the outputs are placed in the same arrays of the input.

3.2 Cosine filter

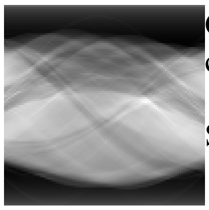
Encode the cosine filter in a similar way by using the applyCosineFilter() method. The frequency response of the cosine filter is:

$H(w) = | w |. \cos (\pi. | w|)$

3.3 Laplacian filter

Encode a digital version of the Laplacian filter in the applyLaplacianFilter() method. Here too, the Laplacian filter that is defined by the mask [1,-2,1], is applied separately to each column of the sinogram (mirror boundary conditions applied).

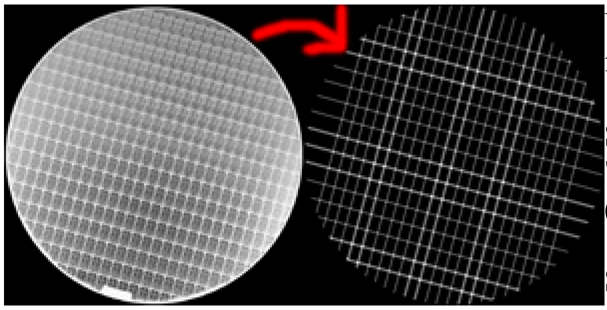
4. Reconstruction of a sinogram



Choose an appropriate filter to reconstruct this synogram given in the ctslicesino file.tif

Save the image (8-bit) and insert it into the report.doc.

5. Location of lines



Use the Radon Transform to locate the lines in a wafer image.tif.

Suggestion: The sinogram can be "thresholded" using the ImageJ command (Image->Adjust->Threshold).

Save the image (8-bit) and insert it into the report.doc.