Diagnostic Medical Image Processing
Prof. Dr.-Ing. Andreas Maier
Exercises (DMIP-E)
WS 2016/17

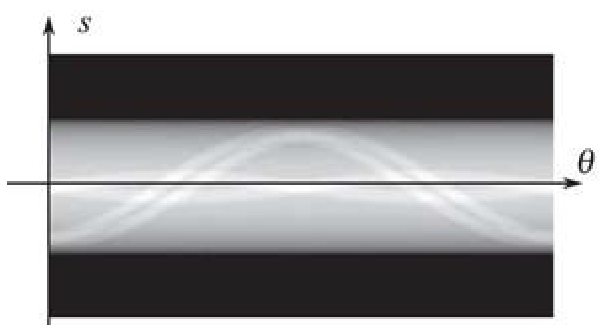
# Parallel Beam Reconstruction

Frank Schebesch, Tobias Würfl, Matthias Utzschneider,
Yixing Huang, Asmaa Khdeir, Houman Mirzaalian

Exercise Sheet 5

## 16 Parallel Beam Projection & Backprojection

(i) Look at the following figure:




What is this type of image called? Explain what information the image contains and how it can be obtained.

(ii) Consider the following $2 \times 2$ image:

$$\begin{bmatrix} w & x \\ y & z \end{bmatrix}.$$

Given four projections:

$$
\begin{aligned}
p_0 &: x + z = 9, w + y = 11, \\
p_{\frac{\pi}{4}} &: w + z = 7, \\
p_{\frac{\pi}{2}} &: w + x = 12, y + z = 8, \\
p_{\frac{3\pi}{4}} &: y + x = 13,
\end{aligned}
$$

build the system matrix and compute the $2 \times 2$ image values. Check your result.

(iii) Now we acquire four projections of the image:

$$
\begin{bmatrix} 1 & 3 & 2 \\ 6 & 1 & 2 \\ 0 & 5 & 3 \end{bmatrix}
$$

at angles $0$, $\frac{\pi}{4}$, $\frac{\pi}{2}$, $\frac{3\pi}{4}$. Obtain the sinogram. When computing the diagonal projections, assume, for simplicity, that the image intensities contribute to equal parts to the values of the detector pixels $1, 2, 3$ according to the following scheme (analoguously for the other diagonal angle):

| | | |
|---|---|---|
| 2 | 1 | 1 |
| 3 | 2 | 1 |
| 3 | 3 | 2 |

(iv) In the same fashion, backproject the data of the sinogram. Is this a perfect reconstruction, and why?

(v) Which angular range is necessary for a complete parallel beam backprojection: $[0, \pi]$ or $[0, \pi)$? Explain graphically.

**1+1+1+1+1**

(i) The image is called sinogram. Each line of the image contains the projection of the object from an angle on a trajectory around the object. The image can be obtained by Radon transform.

(ii)

$$
\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}
\begin{bmatrix} w \\ x \\ y \\ z \end{bmatrix}
=
\begin{bmatrix} 9 \\ 11 \\ 7 \\ 12 \\ 8 \\ 13 \end{bmatrix}
\quad \Rightarrow \quad
\begin{bmatrix} w & x \\ y & z \end{bmatrix}
=
\begin{bmatrix} 5 & 7 \\ 6 & 2 \end{bmatrix}
$$

(iii)

$$\begin{bmatrix} 7 & 9 & 7 \\ 7 & 5 & 11 \\ 6 & 9 & 8 \\ 10 & 3 & 10 \end{bmatrix}$$

(iv)

$$\begin{bmatrix} 28 & 32 & 23 \\ 37 & 26 & 33 \\ 29 & 38 & 30 \end{bmatrix}$$

It is not perfect, first of all these are not enough projections, second the filtering is missing.

(v) $[0, \pi)$, because the rays at 0 and $\pi$ are the same rays.

## 17 Filtered Backprojection

In this exercise we want to show the filtered backprojection algorithm in the Fourier domain. From the lecture, we know that simple backprojection of parallel beam data is not sufficient to obtain the original object. Filtering has to be applied which is usually implemented as a multiplication in Fourier domain. Here we derive the necessary steps:

(i) Start with the inverse Fourier transform of $f(x, y)$:

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v) e^{2\pi i(xu+yv)} \,\mathrm{d}u \,\mathrm{d}v$$

and transform the integration variables from cartesian (detector) coordinates $(u, v)$ to polar coordinates $(\omega, \theta)$. The transformation of $F(u, v)$ is denoted by $F_{\text{polar}}(\omega, \theta)$.

(ii) Since we do not know $F_{\text{polar}}$ directly, we want to connect this formula with something we can compute, for instance the 1-D Fourier transform of the sinogram $p(s, \theta)$:

$$P(\omega, \theta) = \int_{-\infty}^{\infty} p(s,\theta) e^{-2\pi i \omega s} \,\mathrm{d}s.$$

Show that $F_{\text{polar}}(\omega, \theta) = P(\omega, \theta)$ for all $\omega \geq 0$, $\theta \in [0, 2\pi)$.

(iii) Finally, use what you have shown so far to state the filtered backprojection algorithm in Fourier domain. What steps do you have to compute?

(iv) What is the name of the filter you have found? Why do we need it in the algorithm? Please describe the reason from the aspect of sampling.

**1.5+1.5+1+1**

(i) The coordinate transform

$$u = \omega \cos\theta, \quad v = \omega \sin\theta, \quad \omega \geq 0,$$

yields the Jacobian matrix:

$$J_{(u,v)}(\omega,\theta) = \begin{vmatrix} \frac{\partial u}{\partial \omega} & \frac{\partial u}{\partial \theta} \\ \frac{\partial v}{\partial \omega} & \frac{\partial v}{\partial \theta} \end{vmatrix} = \begin{vmatrix} \cos\theta & -\omega\sin\theta \\ \sin\theta & \omega\cos\theta \end{vmatrix} = |\omega| \,.$$

Transform the integral:

$$f(x,y) = \int_0^{2\pi}\int_0^{\infty} F_{\text{polar}}(\omega,\theta)\,|\omega|\,e^{2\pi i\omega(x\cos\theta + y\sin\theta)}\,\mathrm{d}\omega\,\mathrm{d}\theta$$

(ii) You find the proof in “Technical Box 1” here: `https://www5.cs.fau.de/fileadmin/persons/MaierAndreas/maier/Hornegger16-CRB.pdf`

(iii)

$$f(x,y) = \int_0^{2\pi}\int_0^{\infty} \left( \int_{-\infty}^{\infty} p(s,\theta) e^{-2\pi i\omega s}\,\mathrm{d}s \right) |\omega|\,e^{2\pi i\omega(x\cos\theta + y\sin\theta)}\,\mathrm{d}\omega\,\mathrm{d}\theta$$

So steps are: 1-D FT of rows/columns in sinogram, filtering (multiplication with ramp filter), backprojection

(iv) Ramp filter: According to the central slice theorem, each projection at angle $\theta$ samples a line passing through the origin of the Fourier space. Thus, we get higher sampling density at the low frequency components and lower sampling density at the high frequency components. The sampling density is about $1/\sqrt{w_x^2 + w_y^2} = 1/|w|$. Therefore, we need a factor $|w|$ to make the sampling density uniform.

## 18 Ram-Lak Filter – Programming Exercise

In this exercise we have a look at the filtered backprojection of a Shepp-Logan phantom. Your task is to implement the appropriate filter methods. Therefore, please fill in the missing parts in `exercise5.java`. You can set the type of filtering in the `main()`-method.

(i) First, try backprojecting without filtering. Then implement the Ram-Lak filter and apply it in Fourier domain.

(ii) Compare your result with another filter. For that purpose implement the Shepp-Logan filter as well.

**4.5+1.5**

(i) code implementation

(ii) code implementation

**Total: 16**