The PI-FAST Method for Approximate Helical Cone-Beam Reconstruction

Henrik Turbell and Per-Erik Danielsson
Image Processing Lab, Department of Electrical Engineering
Linköping University, SE-581 83, Sweden
turbell@isy.liu.se ped@isy.liu.se

Abstract— **The original PI-method is a non-exact method for 3D-reconstruction using cone-beam projections acquired from a helical source trajectory. In the new PI-FAST method, our aim is to reduce the artifacts while keeping the algorithmic simplicity. The detector is still bounded by the Tam-window, which makes the data capture complete and non-redundant. Also, the filtering step still consists of 1D ramp-filtering but comprises the following novelty. Using the principles of fast backprojection we are able to do the backprojection in two steps interleaved with by rampfiltering. This unusual order is shown to be advantageous by exploiting frequency-distance coherence in projection data. The first backprojection step lumps together projection data over shorter angular intervals along directions which correspond to various velocities in projection space. By rampfiltering these data instead of the original projection data we obtain less unwanted interaction in the z-direction of the volume and a substantial improvement in image quality.**

I. Two-step filtered backprojection using links

Filtered backprojection of 2D parallel data, $p(\theta, t)$, can be expressed as filtering with the ramp-filter $g(t)$

$$\tilde{p}(\theta, t) = p(\theta, t) * g(t) \tag{1}$$

followed by backprojection

$$f(x, y) = \sum_{i=0}^{N_\theta - 1} \tilde{p}(\theta_i, y\cos\theta_i - x\sin\theta_i) \tag{2}$$

over a projection angle interval of length $\theta_{N_\theta} - \theta_0 = \pi$.

It has been shown to be efficient to perform the backprojection summation (2) recursively [1], [2] in $\log_2 N_\theta$ steps. In [3] the process is descibed as iterative summing of filtered projection values along a sinusoid in the sinogram. We will utilize a simplified version of this fast backprojection technique that performs the calculations in two steps. First, small intermediate summations along *linear* segments approximating the sinusoid are calculated and stored. We call the linear segments *links*. In the second step the values of the links are combined in building over π a sinusoid to yield the final result. The reason for the computational gain is that the value of a link will be used for several pixel values in the second step. It should be noted, however, that this potential speedup is not our main reason for using the two-step backprojection. More interestingly we will show that we may postpone the ramp filtering till the first step of the backprojection has been performed.

Figure 1 shows an arbitrary link from (θ_{i_A}, t_A) to (θ_{i_B}, t_B) where $i_B = i_A + n$ and $t_B = t_A + d$. The link

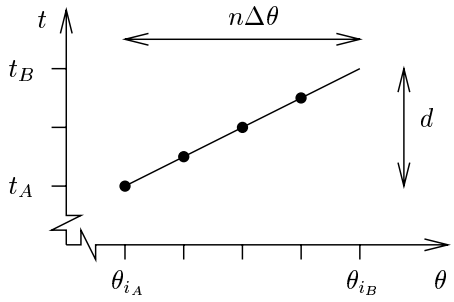


Fig. 1. The link value is a summation of projection values along the link.

value $\tilde{I}(\theta_{i_A}, t_A, d)$ is calculated as

$$\tilde{I}(\theta_{i_A}, t_A, d) = \sum_{i=i_A}^{i_A+n-1} \tilde{p}(\theta_i, t_A + d\frac{i - i_A}{n}) \tag{3}$$

The tilde on $\tilde{I}$ indicates that the link value is a summation of filtered projection data. Although the end points (θ_{i_A}, t_A) and (θ_{i_B}, t_B) preferably are choosen as sample points in the sinogram, the intermediate values along the link require an implicit 1D-interpolation each in (3). The link length n is constant in the algorithm. It has to be short enough for the line elements not to deviate significantly from the true curved sinusoid. A reasonable choice [3] is $n \approx \sqrt{N_\theta}$.

The endpoints of a link corresponds to a ray each in the image domain. The intersection (x, y) of the two rays (θ_{i_A}, t_A) and (θ_{i_B}, t_B) is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -\sin\theta_{i_A} & \cos\theta_{i_A} \\ -\sin\theta_{i_B} & \cos\theta_{i_B} \end{pmatrix}^{-1} \begin{pmatrix} t_A \\ t_B \end{pmatrix} \tag{4}$$

A pixel value is computed by summing the values of the links that build up the sinusoid of the pixel. See Figure 2. For each step l along the θ-axis, we use values of the four links surrounding the sinusoid segment to interpolate the contribution to the pixel value. For notational simplicity we denote the four link-values $\tilde{J}_l$, $\tilde{K}_l$, $\tilde{L}_l$, and $\tilde{M}_l$. The pixel value is obtained as

$$f(x, y) = \sum_{l=1}^{N_\theta/n} (w_l \left(w_{l+1}\tilde{J}_l + (1 - w_{l+1})\tilde{K}_l \right) + (1 - w_l) \left(w_{l+1}\tilde{L}_l + (1 - w_{l+1})\tilde{M}_l \right)) \tag{5}$$

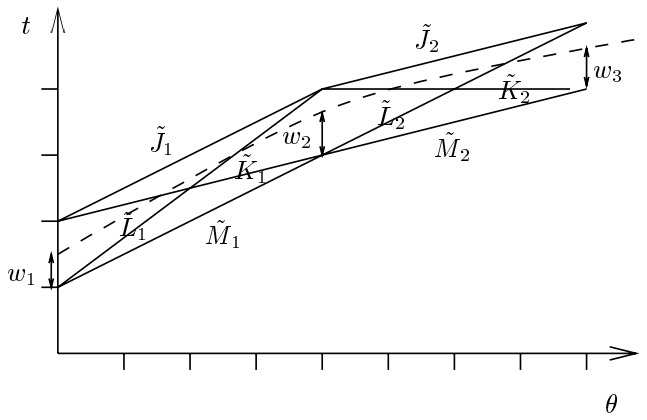


Fig. 2. A pixel value is the sum of interpolations between groups of four links.

where the interpolation weights w_l depend on the t-distance between the sinusoid and the link starting point.

By combining equations (1) and (3), the link value can be written as

$$\tilde{I}(\theta_{i_A}, t_A, d) = \sum_{i=i_A}^{i_A+n-1} \left(p(\theta_i, t_A + d\frac{i-i_A}{n}) * g(t) \right) = \left(\sum_{i=i_A}^{i_A+n-1} p(\theta_i, t_A + d\frac{i-i_A}{n}) \right) * g(t) = I(\theta_{i_A}, t_A, d) * g(t) \tag{6}$$

where $I(\theta_{i_A}, t_A, d)$ naturally denotes the link value calculated from unfiltered projection data. The change of order between summation and convolution is possible since the links are linear and equidistantly spaced along the t-axis. Equation (6) tells us that we may indeed switch the order between rampfiltering and the first backprojection step, the link computation. The right hand side convolution should be seen as a one-dimensional convolution along the t-axis where all links of constant θ_{i_A} and d participate in one filtering event.

II. The original PI-method

The original PI-method [4] is an approximate reconstruction method for the helical cone-beam geometry in Figure 3. The source moves along a helical trajectory of radius R and pitch P around the z-axis. We define the pitch as the distance, measured in an arbitrary length unit, between two consecutive turns of the helix. The effective area of the 2D detector is limited in height to a beam window between two consecutive turns of the source helix. See Figure 3(b). The physical construction and geometry of the detector may vary as long as all measurements are confined to this window, the PI-window. The cone-beam projections $p^C(\beta, \gamma, s)$ and the corresponding rays inside the PI-window are parameterized so that β is the projection angle, γ is the fan-angle, and $2s$ is the detector height coordinate on a detector wrapped onto the source trajectory cylinder. The rows of this detector are aligned with the curved helix. The top boundary of the PI-window is

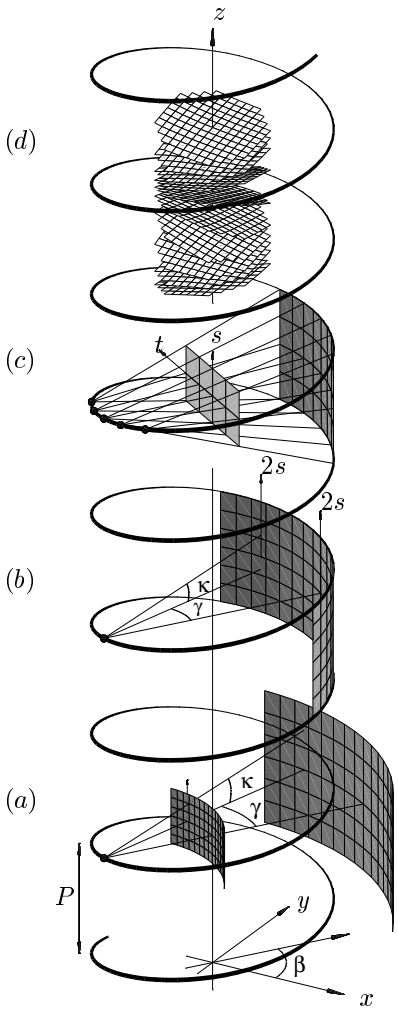


Fig. 3. Geometry for helical cone-beam acquisition. (a) Example of focus-centered detector. (b) A detector on the PI-window. (c) A pseudo-parallel beam together with the planar virtual detector. (d) Nutating PI-surfaces.

therefore at height $s = P/4$ and the bottom boundary at $s = -P/4$.

The first step of the PI-method is a re-sampling step from cone-beams to pseudo-parallel beams according to

$$p(\theta, t, s) = p^C(\beta, \gamma, s) = p^C(\theta - \arcsin\frac{t}{R}, \arcsin\frac{t}{R}, s) \tag{7}$$

Note that the detector height coordinate s is left unchanged. The re-sampling can thus be performed row-by-row and is then identical to the well-known 2D procedure known as parallel rebinning. We observe that the resulting beam shown in Figure 3(c) is divergent when seen from the side, but parallel when seen from along the rotation axis. Also shown is a virtual planar detector orthogonal to the projection direction placed on the rotation axis. The pseudo parallel beam intersects this planar detector on a perfectly Cartesian grid contained in a rectangle. The horizontal rows of this detector have constant value of s.

The PI-method then proceeds with a pre-weighting with the cosine of the cone-angle for each ray. This is followed by 1D ramp-filtering of data along the horisontal lines of the virtual planar detector. The voxel values are finally obtained by 3D backprojection of the filtered data.

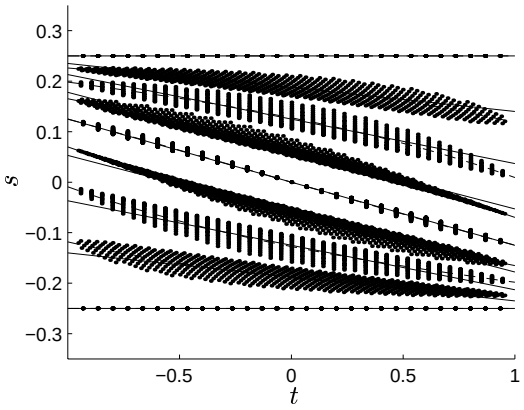


Fig. 4. Point on a PI-surface of a helix of pitch $P = 1$ as projected onto the planar virtual detector at different projection angles.

III. Approximations in some previous helical cone-beam algorithms

In order to track the inexactness of the PI-method, we will now study the set of object points entering the rectangular window of the planar detector at the same time. These points lie on a surface, which we will call a PI-surface. It can be shown that each object point belongs to one and only one PI-surface [5]. The complete set of PI-surfaces has a nutation around the rotation axis and fills up the complete volume to be reconstructed. See Figure 3(d).

We assume that the projection system is rotating upwards in which case the points of a PI-surface enter the rectangular detector window on a perfect line on the upper boundary simultaneously. See Figure 4. As the rotation of the projection system continues, the projection of the PI-surface moves downwards on the detector. Unfortunately, it starts immediately to deviate from the line shape and occupies instead an elongated area with a non-horizontal mid-line. However, after a rotation of exactly $180°$, all points on the PI-surface are again lined up horizontally, now along the lower boundary of the window, and exit the rectangular window simultaneously.

The algorithms found in [6], [7], [4] utilize the observation that the object points on nutating surfaces are concentrated along slanted lines in-between entrance and exit. The 1D ramp-filtering in these algorithms is performed along these slanted lines or curves. The filtered data is then backprojected in two dimensions onto the nutating surface or in three dimensions into the volume. The fact that some of the object points on the nutating surface are projected above such a slanted line and some below is not handled in the filtering step of these algorithms, but will be addressed by the new PI-FAST method.

Consider a neighbourhood of points on a PI-surface. They are projected onto a neighbourhood on the detector. Furthermore, they have similar velocities in the t-direction on the detector. Unfortunately, the projection values of this detector neighbourhood are contaminated by object points belonging to other PI-surfaces. However, on the detector, the projection of these contaminating points have different velocities in the t-direction. Only links of the same slope $\frac{d}{n}$ have the same projected t-velocity [8]. By not filtering links of different slopes together, PI-FAST aims to decrease unwanted interaction between neighbouring PI-surfaces during the filtering event.

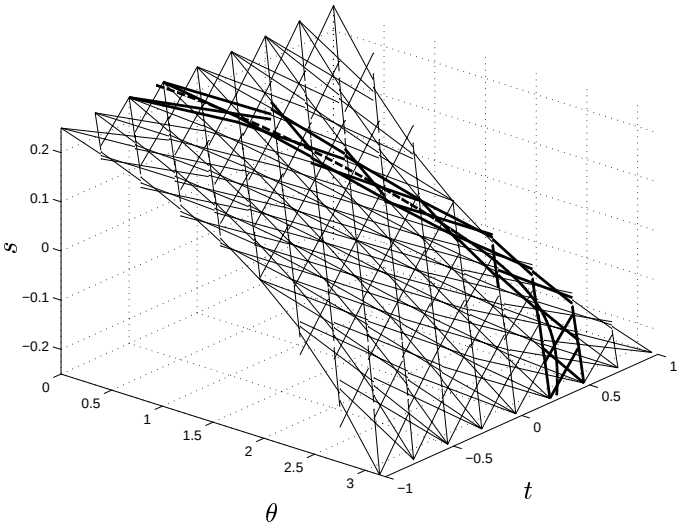


Fig. 5. The links needed for a PI-surface of a helix of pitch $P = 1$. Note that the number of links in both θ- and t-direction is small compared to a typical case to simplify the illustration. This makes the links sprawl out in the s-direction more than in a typical case.

IV. The PI-FAST reconstruction method

For each PI-surface let us construct a complete set of link values $I(\theta, t, d)$. The links are positioned in the 3D (θ, t, s) projection space. The (θ, t)-coordinates of the links are identical to the 2D case since we may regard the reconstruction as 2D when seen from along the rotation axis. The s-coordinates require some further analysis.

Given a PI-surface and the values of θ_{i_A}, t_A, θ_{i_B}, and t_B we use (4) to compute the (x, y)-coordinate of the point on the PI-surface corresponding to the link. Knowing (x, y), the z-coordinate of the point is uniquely given by the equation of the PI-surface. The link endpoint coordinates s_A and s_B can then be calculated by projecting the point (x, y, z) onto the virtual planar detector from projection angles θ_{i_A} and θ_{i_B} respectively. Analytical expressions of these geometry calculations are found in [9].

The complete set of links for a PI-surface will approximately follow the projection tracks of the points in the PI-surface from their movements in the projection space. See Figure 5. At the entrance projection angle $\theta_{i_{\text{in}}}$ of the PI-surface, the links will all start at $s = P/4$ and at the final projection angle $\theta_{i_{\text{in}}} + \pi$ the links will all end at $s = -P/4$. In-between these two angles, the links will spread out in the s-direction.

When the end-points of the links have been calculated it is possible to calculate each link value along the line seg-

ment between the link end point as

$$I(\theta_{i_A}, t_A, d) = \sum_{i=i_A}^{i_A+n-1} p(\theta_i, t_A + d\frac{i-i_A}{n}, s_A + (s_B - s_A)\frac{i-i_A}{n}) \quad (8)$$

The implicit 1D interpolation in the t-direction for the 2D case in (3) has here become an implicit 2D interpolation on each (t, s)-plane. If we insert the re-sampling step in (7) into (8), we may construct the link values directly from the cone-beam data $p^C(\beta, \gamma, s)$ without the intermediate re-sampling step in (7). There is furthermore no need to initially re-sample the original cone-beam data to rows of constant s since the mapping between the actual physical detector sampling points and the (β, γ, s)-system also can be inserted into (8). The sampling points of the terms of the summation in (8) will then not coincide with the projection data sampling points even in the projection angle direction. An implicit 3D-interpolation is thus necessary for each term in (8). To avoid aliasing, it may be necessary to have a smaller step size in the summation along the θ-direction than the sampling distance step used in (8).

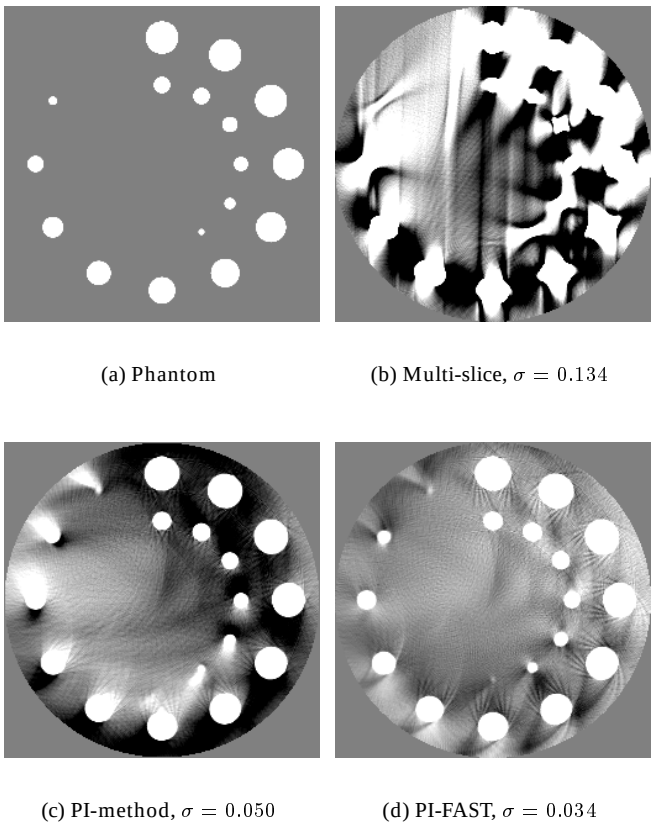
Once the link values of a PI-surface have been computed they may be ramp-filtered as described in the right hand side of (6). The filtered links are then combined into pixel values according to (5). This backprojection step is identical to the 2D case, but will nevertheless perform a 3D backprojection. It does not have to consider the link positions in the s-direction, since this information is already taken care of in the first backprojection step in (8). The pixels on the PI-surfaces are finally interpolated onto a suitable final 3D sampling grid, such as a cubic grid, for presentation and analysis. Note that this interpolation only is performed along the z-direction, since the pixels on the PI-surfaces already are positioned on a Cartesian (x, y)-grid.

We summarize the computation steps in the PI-FAST algorithm as

1: Pre-weight the projection data with the cosine of the cone-angle of each ray.
2: **for all** PI-surfaces **do**
3: Compute the link values according to (8) using the appropriate mappings to the acquisition geometry of the original data.
4: Ramp-filter the link values as in (6).
5: Compute the pixel values of the PI-surface using (5).
6: **end for**
7: Resample the pixels of the PI-surfaces in the z-direction onto a Cartesian grid.

V. Experimental results

Figure 6 shows the reconstruction results of a phantom consisting of homogeneous spheres. For comparison the result of a so called multi-slice method with 2D backprojection [10] found in present 4-row scanners is included. Noise-free projection data was generated on a 64-row detector with a fan-angle of $\pm 30°$ and cone-angle of $\pm 7.13°$. Further experiments and details are found in [9].



(a) Phantom (b) Multi-slice, $\sigma = 0.134$

(c) PI-method, $\sigma = 0.050$ (d) PI-FAST, $\sigma = 0.034$

Fig. 6. Reconstruction results of the sphere clock phantom [9] together with the root mean square error. Radius $R = 2.0$ length units, pitch $P = 1.0$ length units, $N_\theta = 256$ projections per half turn, fan-angle $\pm 30°$, cone-angle $\pm 7.13°$ on a 64 row detector with 255 elements per row. Slice reconstructed on $256 \times 256 \times 1$ voxels of side $\frac{1}{128}$ length unit. Greyscale interval $[-0.05, 0.05]$.

References

[1] Stefan Nilsson, *Application of Fast Backprojection Techniques for Some Inverse Problems of Integral Geometry*, Ph.D. thesis, Department of Mathematics, Linköping University, 1997, No. 499.

[2] Martin L. Brady, "A Fast Discrete Approximation Algorithm for the Radon Transform," *SIAM Journal of Computing*, vol. 27, no. 1, pp. 107–119, 1998.

[3] Per-Erik Danielsson and Malin Ingerhed, "Backprojection in $O(N^2 \log N)$ time," in *IEEE Medical Imaging, Nov 13–15, 1997, Alberquerque, New Mexico, USA*, 1997.

[4] Henrik Turbell and Per-Erik Danielsson, "Helical Cone-Beam Tomography," *International Journal of Imaging Systems and Technology*, vol. 11, no. 1, pp. 91–100, 2000.

[5] M. Defrise, F. Noo, and H. Kudo, "A solution to the long object problem in helical CB tomography," *Physics in Medicine and Biology*, vol. 45, pp. 623–643, 2000.

[6] G. L. Larson, C. C. Ruth, and C. R. Crawford, "Nutating slice CT image reconstruction," Patent Application WO 98/44847, 1998.

[7] M. Kachelriess, S. Schaller, and W. Kalender, "Advanced Single-Slice Rebinning in Cone-Beam Spiral CT," *Medical Physics*, vol. 27, no. 4, pp. 754–772, 2000.

[8] P. Edholm, R. M. Lewitt, and B. Lindholm, "Novel Properties of the Fourier Decomposition of the Sinogram," in *Int. Workshop on Physics and Engineeing of Computerized Multidimensional Imaging and Processing, Proc. of the SPIE*, 1986, vol. 671, pp. 8–18.

[9] Henrik Turbell, *Cone-Beam Reconstruction Using Filtered Backprojection*, Ph.D. thesis, Linköping University, 2001, ISBN 91-7219-919-9.

[10] K Taguchi and H Aradate, "Algorithm for image reconstruction in multi-slice helical CT," *Medical Physics*, vol. 25, no. 4, pp. 550–561, 1998, ISSN 0094-2405.