A Network Flow Algorithm for Binary Image Reconstruction from Few Projections

Kees Joost Batenburg^{1,2}

¹ Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands

² CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Abstract. Tomography deals with the reconstruction of images from their projections. In this paper we focus on tomographic reconstruction of binary images (i.e., black-and-white) that do *not* have an intrinsic lattice structure from a small number of projections. We describe how the reconstruction problem from only two projections can be formulated as a network flow problem in a graph, which can be solved efficiently. When only two projections are used, the reconstruction problem is severely underdetermined and many solutions may exist. To find a reconstruction that resembles the original image, more projections must be used. We propose an iterative algorithm to solve the reconstruction problem from more than two projections. In every iteration a network flow problem is solved, corresponding to two of the available projections. Different pairs of projection angles are used for consecutive iterations. Our algorithm is capable of computing high quality reconstructions from very few projections. We evaluate its performance on simulated projection data and compare it to other reconstruction algorithms.

1 Introduction

Tomography deals with the reconstruction of images from a number of their projections [1, 2]. In many applications, such as the reconstruction of medical CT images, a large number of different pixel values may occur in the reconstruction. Typically, the number of projections that is required to obtain sufficiently accurate reconstructions is large in such cases (more than a hundred).

For certain applications, however, it is known in advance that only a few possible gray values may occur. Many objects scanned in industry for nondestructive testing or reverse engineering purposes are made of one homogeneous material, resulting in two possible gray values: the material and the surrounding air. Another example is medical digital subtraction angiography, where one obtains projections of the distribution of a contrast in the vascular system.

The field of *discrete tomography* deals with the reconstruction of images from a small number of projections, when the set of pixel values is known to have only a few discrete values [3]. By using this prior information about the set of possible values, it may be possible to dramatically reduce the amount of projection data that is required to obtain accurate reconstructions.

In [4] the author proposed an algorithm for reconstructing binary images that are defined on a lattice, using smoothness assumptions. This algorithm exploits the fact that the reconstruction problem for only two projections can be solved in polynomial time. The proposed reconstruction procedure is iterative: in each iteration a new reconstruction is computed using only two projections *and* the reconstruction from the previous iteration. The new reconstruction simultaneously resembles the image from the previous iteration and adheres to the two selected projections.

In this paper we describe a new iterative algorithm for reconstructing binary images that do *not* have an intrinsic lattice structure (i.e., subsets of the plane), which is based on ideas similar to those used in [4]. To solve the two-projection subproblems efficiently, a different pixel grid has to be used in each iteration, corresponding to the selected pair of projections. The reconstruction problem can then be solved as a special case of the *minimum cost network flow* problem in graphs, for which efficient polynomial time algorithms are available [5].

In [6] a more general version of our algorithm is described and more details are provided on the algorithmic steps. We refer the reader to that paper for further details.

We restrict ourselves to parallel beam tomography. Let $D = \{\theta_1, \dots, \theta_d\}$ be a set of disjoint real numbers in the interval $[0, \pi)$, the *projection angles*. Let n be a positive integer. For $i = 0, \dots, n$, put $t_i = -n/2 + i$. Let $\theta \in [0, \pi)$. For $t, t' \in \mathbb{R}$, $t < t'$, define the *strip* $S_\theta(t, t')$ as

$$S_\theta(t, t') = \left\{ (x, y) \in \mathbb{R}^2 : \begin{array}{l} x \cos \theta + y \sin \theta \geq t \quad \text{and} \\ x \cos \theta + y \sin \theta \leq t' \end{array} \right\}.$$

Fig. 1. (a) Left: Schematic view of the parallel beam geometry which contains the lines $L_1 : x \cos \theta + y \sin \theta = 0$ and $L_2 : x \cos \theta + y \sin \theta = t$. (b) Right: A two-projection image.

Figure 1a shows the geometric meaning of the last definition. Define the *imaging area* I as

$$I = \bigcap_{k=1}^d S_{\theta_k}(t_0, t_n).$$

We will now define the basic reconstruction problem of reconstructing a binary image (i.e., black-and-white) from few projections. We consider the unknown image as a mapping $f : I \rightarrow \{0, 1\}$.

Problem 1. Let $p_1 = (p_{11} \dots p_{1n})^T, \dots, p_d = (p_{d1} \dots p_{dn})^T \in \mathbb{R}^n$ be vectors of nonnegative real numbers (the measured strip projections for projection angles $\theta_1, \dots, \theta_d \in D$ respectively). Construct a function $f : I \rightarrow \{0, 1\}$ such that

$$\iint_{S_{\theta_k}(t_{i-1}, t_i)} f(x, y) \, dy \, dx = p_{ki} \quad \text{for } i = 1, \dots, n, \, k = 1, \dots, d.$$

We call an integral of the form $\iint_{S_{\theta_k}(t_{i-1}, t_i)} f(x, y) \, dy \, dx$ a *strip projection*.

In the next section we deal with the reconstruction problem from only two projections, which is severely underdetermined. In Section 3 we describe our iterative algorithm for reconstruction from more than two projections. Experimental results of the algorithm from Section 3 are presented in Section 4.

2 Reconstruction from Two Projections

This section deals with the reconstruction problem from only two projections, for angles θ_1 and θ_2 . To represent a mapping $f : I \rightarrow \{0, 1\}$ in a computer we have to resort to an approximate representation. An image f is represented as a 2D array of pixels. Every measured strip projection then gives rise to a linear equation on the pixel values of f . The resulting system of linear equations can be solved by methods from linear algebra, but in this way one cannot guarantee that a *binary* solution is found. We now introduce a particular grid, the *two-projection grid*, for which a binary solution of the reconstruction problem can be found efficiently. The rows and columns of this grid correspond to the strips for the two projections angles θ_1 and θ_2 . Define *grid cell* C_{ij} ($1 \leq i, j \leq n$) as

$$C_{ij} = S_{\theta_1}(t_{i-1}, t_i) \cap S_{\theta_2}(t_{j-1}, t_j).$$

A *two-projection image* is a mapping $\{1, \dots, n\} \times \{1, \dots, n\} \rightarrow \{0, 1\}$ which assigns a binary value to each grid cell of a two-projection grid. Figure 1b shows an example of a two-projection image.

It is often convenient to consider a two-projection image X as a *matrix* (x_{ij}) , where x_{ij} denotes $X((i, j))$ (the pixel value of cell C_{ij}). The strip projections of X for the projection angles θ_1 and θ_2 can be computed directly by summation of all entries in each row of X , or column respectively, and multiplying the result by the cell area a . For $k = 1, 2$, define $P_k(X) \in \mathbb{R}^n$ as the vector of strip projections

of X for angle $\bar{\theta}_k$. Define the *one-count* of X by $S(X) = \sum_{1 \leq i, j \leq n} x_{ij}$. We denote the area of a single grid cell by a . Note that all grid cells have the same shape and size.

We will now define a reconstruction problem for two-projection images. As the projection data may contain noise or other errors, we don't require that the image adheres perfectly to the measured projection data. In the next section, where we consider reconstruction from more than two projections, we require a generalization of the reconstruction problem which incorporates prior knowledge. We state this general reconstruction problem here.

Problem 2. Let $\bar{\theta}_1, \bar{\theta}_2 \in [0, \pi)$ be two disjoint projection angles. Let $p_1, p_2 \in \mathbb{R}^n$ be two vectors of nonnegative real numbers (the *measured projection data*). Let $W = (w_{ij}) \in \mathbb{R}^{n \times n}$, $\bar{T} \in \mathbb{N}_{>0}$ and $\alpha \in \mathbb{R}$. Construct a matrix $X \in \{0, 1\}^{n \times n}$ such that $S(X) = \bar{T}$ and

$$\alpha(|P_1(X) - p_1|_1 + |P_2(X) - p_2|_1) - \sum_{1 \leq i, j \leq n} aw_{ij}x_{ij}$$

is minimal.

In any instance of Problem 2, the one-count $S(X)$ is considered to be fixed at $\bar{T}$. A good value for $\bar{T}$ can be computed from the measured projection data by taking $\bar{T} = (|p_1|_1 + |p_2|_1)/2a$. Putting $\alpha = 1$ and $w_{ij} = 0$ for $1 \leq i, j \leq n$, yields the basic reconstruction problem (without the use of prior knowledge). The matrix W is called the *weight map*. The weight map is used extensively in the algorithm for reconstructing a binary image from more than two projections that we describe in the next section. It is used to express a preference for each pixel to obtain a value of either 0 or 1 in the reconstruction.

Problem 2 can be solved efficiently by formulating it as a minimum cost flow problem in a particular graph. Efficient algorithms are available for solving the resulting network flow problem. The basic idea of using network flow methods for the reconstruction of binary images from two projections was first described by Gale in 1957 [7], in the context of reconstructing binary *matrices* from their row and column sums.

For the sake of brevity, we make one simplifying assumption: we assume that all measured strip projections are integral multiples of the pixel area a . If the strip projections do not satisfy this assumption they can simply be rounded to the nearest multiple of a . If n is large, the effect of this rounding step is negligible. Details on how to compute an exact solution of Problem 2, without the assumption on the strip projections, are given in [6].

In the remainder of this section we assume that the reader is familiar with the basic concepts of *network flows*. The book [5] provides an excellent introduction to this subject.

With a pair of projection angles $(\bar{\theta}_1, \bar{\theta}_2)$ and their respective measured projections (p_1, p_2) , we associate a *directed* graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. We call G the *associated graph*, see Figure 2. The set V contains a node s (the *source*), a node t (the *sink*), one node for each

Fig. 2. Basic structure of the associated graph

strip of projection angle $\bar{\theta}_1$ and one node for each strip of projection angle $\bar{\theta}_2$. The node that corresponds to $S_{\bar{\theta}_k}(t_{i-1}, t_i)$ has label $n_{k,i}$. We call the nodes $n_{k,i}$ *line nodes*.

Each edge e of G has an associated *capacity* u_e and *cost* c_e . Every pair $(n_{1,i}, n_{2,j})$ of nodes is connected by a directed edge. We call these edges *pixel edges*. Each pixel edge $(n_{1,i}, n_{2,j})$ has capacity 1 and cost $-aw_{ij}$. For each pair $(s, n_{1,i})$ the graph G contains two parallel edges. The first edge has capacity p_{1i}/a and cost 0, the second edge has capacity $n - p_{1i}/a$ and cost $2\alpha a$. Similarly, G contains two parallel edges for every pair $(n_{2,j}, t)$ with capacities p_{2j}/a and $n - p_{2j}/a$ and costs 0 and $2\alpha a$ respectively.

An integral *flow* in G is a mapping $Y : E \rightarrow \mathbb{N}_{\geq 0}$ such that $Y(e) \leq u_e$ for all $e \in E$ and such that for all $v \in V \setminus \{s, t\}$:

$$\sum_{w: (w,v) \in E} Y((w,v)) = \sum_{w: (v,w) \in E} Y((v,w)).$$

Every integral flow Y corresponds to a two-projection image $X = x_{ij}$, defined by $x_{ij} = Y((n_{1,i}, n_{2,j}))$. Define the *total cost* of an integral flow Y by

$$C(Y) = \sum_{e \in E} c_e Y(e).$$

In [6] it is proved that any integral flow of minimal cost in G corresponds to a solution of Problem 2. Note that all edge capacities of G are integers. The integral minimum cost flow problem can be solved in polynomial time for graphs that

have integral arc capacities and costs. To obtain integral costs in our case, we multiply the edge costs by a sufficiently large number and round the results. Note that scaling all costs by the same factor does not change which flow has minimal cost.

3 More Than Two Projections

Unfortunately, there is no straightforward generalization of the network flow approach to the case of more than two projections. We propose an iterative algorithm, which uses the fact that the two-projection problem can be solved efficiently. The algorithm computes a reconstruction from more than two projections by solving a series of two-projection subproblems, each using two projection angles. The algorithm uses the concept of a weight map, as introduced in Problem 2. Our algorithm aims to find an *approximate* solution of the reconstruction problem. In each iteration a new pair of projection angles is selected. An instance of Problem 2 is then solved on the two-projection grid that corresponds to those two angles. The weight map is computed using the reconstruction from the previous iteration, in such a way that solutions are preferred which resemble the reconstruction from the previous iteration. Additionally, a preference for locally smooth regions is incorporated in the weight map. The reconstruction from the previous iteration was computed using a different pair of projections, which are thus incorporated into the new reconstruction. Repeating this argument, projections from earlier iterations are also incorporated.

```

Compute a start solution  $X^0$  on the standard square pixel grid;
 $A := (\sum_{k=1}^d |p_k|_1)/d$ ;
 $\tau := 0$ ;
while (stop criterion is not met) do
begin
     $\tau := \tau + 1$ ;
    Select a new pair of projection angles  $\bar{\theta}_1^\tau, \bar{\theta}_2^\tau \in D$ ;
    Compute the weight map  $W^\tau = (w_{ij}^\tau)$  for the two-projection grid
    corresponding to  $(\bar{\theta}_1^\tau, \bar{\theta}_2^\tau)$ , using the previous reconstruction  $X^{\tau-1}$ ;
    Compute  $\bar{T} := A/a_\tau$ ;
    Compute a solution  $X^\tau$  with  $S(X^\tau) = \bar{T}$  of Problem 2
    on the two-projection grid for angles  $(\bar{\theta}_1^\tau, \bar{\theta}_2^\tau)$ , using the
    weight map  $W^\tau$ ;
end
Return the final reconstruction  $X^*$ ;

```

Fig. 3. Basic steps of the algorithm

Figure 3 shows the basic steps of the algorithm. First, a start solution is computed, using all projections simultaneously. The start solution should provide a good first approximation of the unknown image, while being easy to compute. The start solution can be computed on the standard $n \times n$ square pixel grid. For the experiments in Section 4 we used the SIRT (Simultaneous Iterative Reconstruction Technique, see Chapter 7 of [1]) to compute the start solution, which yields a gray value reconstruction.

Subsequently, the “total area A of the white region” (i.e., the region where the function value of the unknown image f is 1) is computed as $(\sum_{k=1}^d |p_k|_1)/d$.

Next, the algorithm enters the main loop. In each iteration τ of the main loop a new pair $(\bar{\theta}_1^\tau, \bar{\theta}_2^\tau)$ of projection angles is first selected, which determines the two-projection grid for iteration τ . We refer to the cell (i, j) of this grid as C_{ij}^τ . To choose the projection angles, the projections of the current image for each of the angles $\theta_1, \dots, \theta_d$ are computed first. The new pair of angles is chosen such that the angle between $\bar{\theta}_1^\tau$ and $\bar{\theta}_2^\tau$ is at least $\pi/3$ and the total deviation of the computed projections from the prescribed projections is largest, according to the sum-norm.

Next, the number of white pixels $\bar{T} = \text{round}(A/a_\tau)$ is computed, where a_τ denotes the area of a grid cell in the current two-projection grid. Note that the grid cell area is different for each two-projection grid.

Subsequently the weight map $W^\tau = (w_{ij}^\tau)$ is computed from the previous reconstruction. We denote the grid cells of the new two-projection grid by C_{ij}^τ . Define $m_{ij} \in \mathbb{R}^2$ as the center of mass of cell C_{ij}^τ . The pixel weight w_{ij}^τ of pixel (i, j) depends directly on the average gray value of $X_{\tau-1}$ in a small neighbourhood of m_{ij} . Most common pixel neighbourhood definitions that are used in image processing, such as the 4-neighbourhood and 8-neighbourhood, are very suitable for square pixel grids. However, as our algorithm deals with many different pixel grids for which the pixel sizes and shapes may vary, we have to use a more general neighbourhood concept. Let r be a positive real number, the *neighbourhood radius*. We used $r = 1.5$ for all experiments in Section 4. Let Γ_{ij} be the average gray value inside a circle of radius r , centered in m_{ij} , in the two-projection image $X_{\tau-1}$. This value can be computed by intersecting each of the overlapping pixels of $X_{\tau-1}$ with the circular neighbourhood of m_{ij} and weighting the pixel values by the intersection area. Note that we always have $\Gamma_{ij} \in [0, 1]$.

Define $g : [-1, 1] \rightarrow \mathbb{R}$ by

$$g(v) = \begin{cases} v & \text{if } |v| \neq 1 \\ 2v & \text{if } |v| = 1. \end{cases}$$

We call g the *local weight function*. Together with the neighbourhood radius r , the local weight function determines the preference for locally smooth regions. The pixel weight w_{ij}^τ is computed as

$$w_{ij}^\tau = g(2(\Gamma_{ij} - \frac{1}{2}))$$

The basic idea of the local weight function is that, as the neighbourhood of a pixel becomes more white, the preference to make this pixel white in the next iteration increases. The same holds for black neighbourhoods. There is an additional increase in the (absolute value of) the pixel weight if the neighbourhood is completely homogeneous. If a pixel neighbourhood consists of 50% black and 50% white pixels, no preference is expressed as the pixel weight is zero.

The weight map W^τ , the value $\bar{T}$ and the projections for angles $(\bar{\theta}_1^\tau, \bar{\theta}_2^\tau)$ define an instance of Problem 2. Solving this problem by the network flow approach yields the new reconstruction X^τ .

To determine when the algorithm should terminate, the strip projections of X_τ are computed for all projection angles $\theta_1, \dots, \theta_d$. Subsequently the norm of the error with respect to the measured projections is computed. The algorithm terminates if the error has not decreased for T iterations, where T is a constant. In all experiments of Section 4 we used $T = 30$.

4 Experimental Results

In this section we present reconstruction results of our algorithm for a set of four phantom images. For each of the phantoms we computed the measured projection data by simulation. In this way the noise level can be controlled, which is difficult for datasets obtained by real measurements.

We implemented the iterative network flow algorithm in C++, using the RelaxIV library [8] to solve the min cost flow problems. All experiments were performed on a 2.4GHz PentiumIV PC.

The four phantom images that we use to evaluate the reconstruction time and the reconstruction quality of our algorithm are shown in Figure 4.

Fig. 4. Four phantom images

First, we compare our reconstruction results from perfect projections to two alternative approaches. It is well known that continuous tomography algorithms such as Filtered Backprojection require a large number of projections. Algebraic reconstruction methods, such as ART and SIRT (see Chapter 7 of [1]) typically perform much better than backprojection algorithms if the number of projections is very small. Our algorithm uses the SIRT algorithm to compute a start

solution. As a first comparison, we compare the final reconstruction computed by our algorithm to the continuous SIRT reconstruction. In [9, 10], Weber et al. describe a linear programming approach to binary tomography which incorporates a smoothness prior. We implemented the *R-BIF* approach which is described in [10] using the ILOG CPLEX interior point solver [11]. The real-valued pixel values are rounded to binary values as a postprocessing step, which was also done in [9]. Besides the projection data, the linear program depends on a parameter α , which determines the preference for smoothness. For our set of four phantoms we found that $\alpha = 0.2$ works well. Figure 5 shows the reconstruction results for the four phantoms. Table 1 shows a quantitative comparison.

Fig. 5. Reconstruction results for the four phantoms from parallel beam projections, using SIRT (top), R-BIF from [10] (middle) and our network flow algorithm (NF, bottom). The figure captions show the number d of projections that was used.

The number d of projections is chosen for each phantom as the minimum number for which our algorithm computes an accurate reconstruction from the projection data. The projection angles are equally spaced between 0 and 180

Table 1. Quantitative comparison between the R-BIF reconstruction and the reconstruction computed by our network flow algorithm. The table shows the reconstruction time in minutes and the total number of pixel differences between the reconstruction and the original phantom.

phantom	size	#proj.	R-BIF		Network Flow	
			#errors	time(min)	#errors	time(min)
single object	300×300	5	220	9.2	58	2.0
50 ellipses	256×256	8	216	3.5	152	1.5
turbine blade	276×276	7	214	3.2	108	1.8
cylinder head	276×276	10	375	10	665	1.9

degrees. Each projection consists of n strip projections, that each have a width equal to the phantom image pixel width.

Our algorithm is capable of computing an accurate reconstruction of each of the four phantom images from a small number of projections. The reconstruction quality of the R-BIF algorithm appears to be similar, although there are some small differences between the reconstructions by both methods. Both algorithms clearly outperform the continuous SIRT algorithm.

We observed that the number of projections that is required to compute an accurate reconstruction is the same for the R-BIF algorithm as for our algorithm, for each of the four phantoms. As there are major differences between both algorithms, this suggests that this minimal number of projections is an intrinsic property of the images themselves. It may be very difficult or even impossible to do better by using a different algorithm.

The minimal number of projection that is required to reconstruct a given (unknown) image depends strongly on characteristics of the unknown image, as can be seen by the varying number of projections that is required for the four phantom images.

Our algorithm has several advantages compared to the R-BIF linear programming approach. First, the network flow algorithm is faster than general linear programming as used in [10], even though we used a high performance interior point solver for solving the linear program. It was demonstrated in [12] for the case of lattice images that the network flow approach for 2D reconstruction can be extended to a highly efficient algorithm for 3D reconstruction. Within each iteration a series of 2D reconstruction problems is solved, instead of one big 3D problem. The 2D reconstructions can be computed fast (because each subproblem is relatively small) and in parallel. A similar extension is possible for our new algorithm. Dealing with large volumes seems to be very difficult when using general linear programming, as the number of variables becomes huge for large 3D volumes.

Another advantage of our algorithm is that it can deal with noise effectively, as we will show in the next subsection. The linear programming approach from [9, 10] is not capable of handling noisy data. It can be extended to the case of noisy data, as described in [13]. However, the presented algorithm for dealing

with noisy data solves a *series* of linear programs, which results in far longer reconstruction times.

More recently, Schüle et al. developed a different reconstruction algorithm based on D.C. programming [14]. The results of this algorithm seem to be very promising and it does not have some of the drawbacks of the linear programming approach. We intend to perform a comparison between a wider set of reconstruction algorithms in a future publication.

4.1 Noisy Projection Data

So far all experiments were carried out with perfect projection data. We now focus on the reconstruction of images from noisy projection data, which is practically more realistic. We also assume that the noise is independent for each projected strip. To be more precise, we assume that the noise is additive and that it follows a Gaussian distribution with $\mu = 0$ and σ constant over all projected strips. The standard deviation σ is expressed as $\sigma = vy$, where y denotes the average measured strip projection over all projected strips. For example, taking $v = 0.05$ results in independently distributed additive noise for each strip projection, following the $\mathcal{N}(0, 0.05y)$ distribution.

Figure 6 shows reconstruction results for the cylinder head phantom, using 10 parallel beam projections and varying noise levels. The reconstruction quality decreases gradually as the noise level is increased. Up to $v = 0.02$, the noise hardly has any visible influence on the reconstruction quality.

Fig. 6. Reconstruction results from 10 parallel beam projections for increasing noise levels

5 Conclusions

We have described a novel algorithm for the reconstruction of binary images from a small number of their projections. Our algorithm is iterative. In each iteration a reconstruction problem is solved that depends on two of the projections and the reconstruction from the previous iteration. The two-projection reconstruction problem is equivalent to the problem of finding a flow of minimal cost in the associated graph. This equivalence allows us to use network flow algorithms for solving the two-projection subproblems.

The reconstruction results show that the reconstruction quality of our algorithm is far better than for the SIRT algorithm from continuous tomography. A comparison with the R-BIF linear programming approach from [10], which uses a smoothness prior, shows that both algorithms yield comparable reconstruction quality. Our algorithm runs faster than the linear programming approach and can be easily extended to 3D reconstruction or reconstruction from noisy projections. These extensions are difficult to accomplish efficiently for the linear programming approach.

Our results for noisy projection data shows that the algorithm is capable of dealing with significant noise levels. The reconstruction quality decreases gradually as the noise level is increased.

In future research we intend to perform extensive comparisons between different discrete tomography algorithms. Such a comparison is often difficult, as each algorithm makes different assumptions on the class of images, the detector setting, etc. Generalization of our algorithm to 3D reconstruction is straightforward, following the same approach as in [12].

References

1. Kak, A.C., Slaney, M.: Principles of Computerized Tomographic Imaging. SIAM (2001)
2. Natterer, F.: The Mathematics of Computerized Tomography. SIAM (2001)
3. Herman, G.T., Kuba, A. (eds.): Discrete Tomography: Foundations, Algorithms and Applications. Birkhäuser Boston (1999)
4. Batenburg, K.J.: Reconstructing Binary Images from Discrete X-Rays. CWI Report PNA-E0418 (2004) (submitted for publication)
5. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Prentice-Hall (1993)
6. Batenburg, K.J.: A Network Flow Algorithm for Binary Tomography Without an Intrinsic Lattice. preprint: www.math.leidenuniv.nl/~kbatenbu (2006)
7. Gale, D.: A Theorem on Flows in Networks. Pacific J. Math. **7** (1957) 1073–1082
8. Bertsekas, D.P., Tseng, P.: RELAX-IV: A Faster Version of the RELAX Code for Solving Minimum Cost Flow Problems. LIDS Technical Report LIDS-P-2276. MIT (1994)
9. Weber, S., Schnörr, C., Hornegger, J.: A Linear Programming Relaxation for Binary Tomography with Smoothness Priors. Electron. Notes Discrete Math. **12** (2003)
10. Weber, S., Schüle, T., Schnörr, C., Hornegger, J.: A Linear Programming Approach to Limited Angle 3D Reconstruction from DSA Projections. Special Issue of Methods of Information in Medicine **4** (2004) Schattauer Verlag. 320–326
11. ILOG CPLEX: <http://www.ilog.com/products/cplex/>
12. Batenburg, K.J.: A New Algorithm for 3D Binary Tomography. Electron. Notes Discrete Math. **20** (2005) 247–261
13. Weber, S., Schüle, T., Hornegger, J., Schnörr, C.: Binary Tomography by Iterating Linear Programs from Noisy Projections. Proceedings of IWCIA 2004. Lecture Notes in Computer Science, Vol. 3322 (2004) 38–51
14. Schüle, T., Schnörr, C., Weber, S., Hornegger, J.: Discrete Tomography by Convex-Concave Regularization and D.C. Programming. Discrete Appl. Math. **151** (2005) 229–243