

Direction-Dependency of a Binary Tomographic Reconstruction Algorithm

László Varga, Péter Balázs*, and Antal Nagy

Department of Image Processing and Computer Graphics
University of Szeged
Árpád tér 2, H-6720 Szeged, Hungary
{vargalg,pbalazs,nagya}@inf.u-szeged.hu

Abstract. We study how the quality of an image reconstructed by a binary tomographic algorithm depends on the direction of the observed object in the scanner, if only a few projections are available. To do so we conduct experiments on a set of software phantoms by reconstructing them from different projection sets using an algorithm based on D.C. programming (a method for minimizing the difference of convex functions), and compare the accuracy of the corresponding reconstructions by two suitable approaches. Based on the experiments, we discuss consequences on applications arising from the field of non-destructive testing, as well.

Keywords: discrete tomography, reconstruction, non-destructive testing, D.C. programming; GPU-accelerated computing.

1 Introduction

The goal of *tomography* is to reconstruct an image from its projections. In the general case this problem can be solved, e.g., by the filtered backprojection method that can reconstruct the image when a sufficiently great number of projections - usually a few hundreds - are available [9]. However, in certain applications of tomography it is not possible to make so many projections of the observed objects. *Discrete tomography* deals with the case when the objects to be reconstructed consist of just a few different materials, with known attenuation coefficients [7,8]. With this prior information, algorithms were developed capable of reconstructing the original image (or a similar one) from just a few - usually 2-10 - projections. Several works have been done to investigate how small perturbations in the projection data can affect the result of discrete tomographic reconstruction, and how the original and the reconstructed image can differ from each other in such cases (see, e.g., [1,4]). However, having so few projections, defined by very differing angles brings up additional questions. Does

* Corresponding author. This research was partially supported by the TÁMOP-4.2.2/08/1/2008-0008 program of the Hungarian National Development Agency and by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

the result of the reconstruction depend on how we choose these angles? How should we choose the angles of the projections to obtain the best result possible for a given number of projections?

Such studies are motivated by practical applications. For example, in non-destructive testing (NDT) of objects made of homogeneous materials, the acquisition of the projections may be very expensive, so it is important to keep the number of projections as small as possible. In NDT often a blueprint image is available, and the task is to determine how much the object of interest differs from the given blueprint. Usually, the object studied might be placed with some rotation into the scanner, which may affect the accuracy of the reconstruction, and make the comparison to the blueprint impossible. Even if the effect of rotation is somehow eliminated, projections of a given object from certain directions can be more informative than other ones, which makes sense to use the blueprint image to determine how to put the objects into the scanner to get better results without making additional projections.

The aim of this paper is to determine – at least empirically – how dependent a discrete tomographic reconstruction can be, on the angles chosen for the projections. We do this by performing experimental tests on a set of software phantoms, trying to reconstruct them from different projection sets. Such results have been briefly mentioned before in [10], but to our knowledge no detailed research in this topic has been done so far.

The paper is structured as follows. We start out by stating the reconstruction problem in Section 2. Then, in Section 3 we go into more details to specify the test framework. Section 4 describes the experiments we conducted, while we give our experimental results in Section 5. In Section 6 we discuss how our results are related to applications of non-destructive testing. Finally, Section 7 is for the conclusion.

2 The Reconstruction Problem

We study the reconstruction of binary images of size $n \times n$ from their projections. The projection data is measured by line integrals taken from a set of directions defined by different angles, using parallel beam geometry. In this case, the binary reconstruction problem can be represented as a system of equations

$$\mathbf{Ax} = \mathbf{b}, \quad \mathbf{A} = (a_{i,j})_{n^2 \times m} \in \mathbb{R}^{n^2 \times m}, \quad \mathbf{x} \in \{0, 1\}^{n^2}, \quad \mathbf{b} \in \mathbb{R}^m, \quad (1)$$

where m is the total number of projection rays used, $a_{i,j}$ gives the length of the line segment of the i -th ray through the j -th pixel, and b_i gives the projection of the image along the i -th projection ray as illustrated in Figure 1.

Although this gives an exact formulation of the reconstruction problem, solving the related equation system is usually not the best approach. As mentioned before, in discrete tomography just a handful of projections are available, therefore the corresponding system of equations is usually underdetermined, and due to errors in the measured projection data, it can be inconsistent as well. There

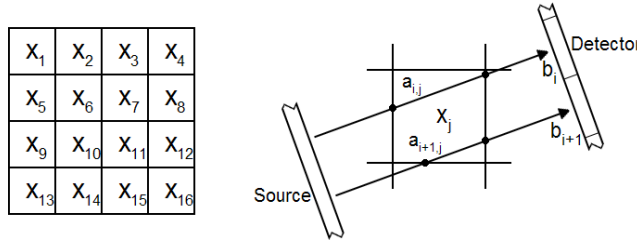


Fig. 1. Representation of the parallel beam geometry used

are two main approaches to overcome these problems. First, one can apply iterative algorithms for finding an approximate solution. These methods are the different versions of the so-called algebraic reconstruction technique [2,6,9]. The other main approach is based on defining a function

$$f(\mathbf{x}) = \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda \cdot g(\mathbf{x}) , \tag{2}$$

where \mathbf{A} , \mathbf{b} , and \mathbf{x} are the same as defined in (1) and $g(\mathbf{x})$ is a function representing prior information about the image to be reconstructed, with a given λ weight. After this reformulation, the reconstruction problem can be redefined as finding the minimum of the function $f(\mathbf{x})$ by some optimization strategies, like genetic algorithms, simulated annealing, or other numerical methods. Examples for this kind of algorithms can be found in [5,13,14].

For our experiments we were using the numerical method specified in [13] (in the following referred to as DC algorithm where the abbreviation DC stands for the difference of convex functions), which performs the reconstruction by minimizing the function

$$J_\mu(\mathbf{x}) := \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \frac{\gamma}{2} \sum_{j=1}^{n^2} \sum_{l \in N_4(j)} (\mathbf{x}_j - \mathbf{x}_l)^2 - \mu \frac{1}{2} \langle \mathbf{x}, \mathbf{e} - \mathbf{x} \rangle , \quad \mathbf{x} \in [0, 1]^{n^2} , \tag{3}$$

where γ is a given constant controlling the weight of the smoothness term on the image, $N_4(j)$ is the set of pixels 4-connected to the j -th pixel, and \mathbf{e} denotes the vector with all n^2 coordinates equal to 1. In the beginning of the optimization process $\mu = 0$, so the best continuous solution is found. In the sequel, μ is iteratively increased by μ_Δ , to force binary results. This algorithm is suitable to our task for several reasons:

- luck must not have an influence on the results, so a deterministic algorithm is needed,
- DC is proved to be an accurate algorithm that can work with a small number of projections, and
- its parallel implementation for a GPU makes the algorithm capable of performing a large number of tests required, in a relatively short time.

3 Technical Specification of the Reconstructions

The parameters of the DC reconstruction algorithm were mostly set as specified in [14], for example we used $\gamma = 0.25$. Though, instead of calculating μ_Δ from the first continuous reconstruction and the projection matrix \mathbf{A} , we explicitly set $\mu_\Delta = 0.1$, to avoid performing a large number of computation, and keep the running time of the algorithm as low as possible.

Reconstruction of the test objects were done from projection sets with different numbers of angles. Each time the angles were uniformly placed on the half circle as follows

$$S(\alpha, p) = \{90^\circ + \alpha + i \frac{180^\circ}{p} \mid i = 0, \dots, p - 1\}, \quad (4)$$

where p (the number of angles) and α (the starting offset) are given values. Figure 2 gives an example using 4 projection angles. For each image the number of projections p ranged from 2 to 16, and for each such projection sets the starting angle α ranged from 0° to $\left(\left\lceil \frac{180}{p} \right\rceil - 1\right)^\circ$ with a step of 1° as illustrated in Figure 3. That makes a total of $\sum_{i=2}^{16} \lceil \frac{180}{i} \rceil = 431$ different reconstruction tasks for each image. For each projection the rays were placed at equal distances,

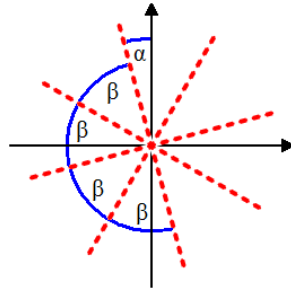


Fig. 2. The projection sets $S(\alpha, 4)$ used in the tests (α is predefined, $\beta = 45^\circ$)

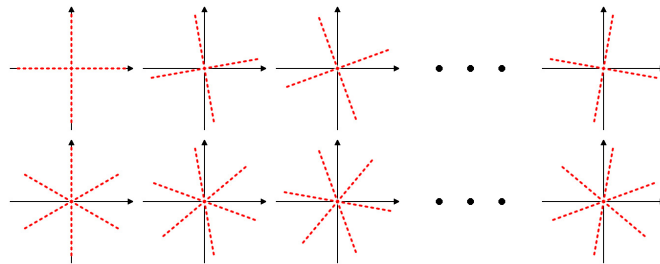


Fig. 3. The projection sets used for testing the DC algorithm with 2 (top row) and 3 (bottom row) projections. Dashed red lines represent the direction of the projections.

and covered the whole image. The distance between the detector elements was set to $1/2$ pixels to ensure that we have information of all the pixels in every projection.

4 Test Data and Experiments

We conducted several experiments using software phantoms from three different sources: three phantoms used for testing the algorithm in [14], two phantoms used in [2], and – in addition – 10 newly generated software phantoms, each containing five randomly positioned disks of random sizes. Where it was possible (in the case of the 10 new software phantoms and one taken from [14]) two altered versions of the phantoms were also generated: one with a ring, and one with a rectangular stripe around the original objects. All the 37 phantoms had the same size of 256 by 256 pixels. Some of the phantoms used in our tests can be seen in Figure 4.

The algorithm was implemented with GPU acceleration on the NVIDIA CUDA programming toolkit (detailed description of CUDA can be found in [11]). For the computation we used a 2.5 GHz Core 2 Quad CPU, and an NVIDIA GeForce 8800 GT GPU. The time required to perform the total 431 reconstruction tasks for each phantom was about 1-2 hours, depending on the phantom processed.

During the experiments we used two approaches to evaluate the results. The first one was to measure the accuracy of the reconstruction by counting the

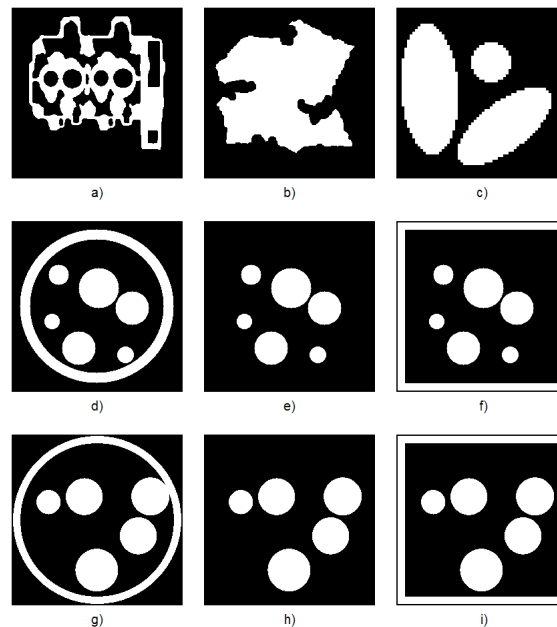


Fig. 4. Some of the software phantoms used for testing

number of pixels differing in the original phantoms and the result images for each given set $S(\alpha, p)$ of projections by

$$E(\mathbf{x}^*, S(\alpha, p)) := \|\mathbf{x}^* - \mathbf{x}_{S(\alpha, p)}\|_2^2, \quad (5)$$

where \mathbf{x}^* is the original software phantom and $\mathbf{x}_{S(\alpha, p)}$ denotes the result reconstructed from the set of directions $S(\alpha, p)$ defined in (4).

The other kind of evaluation was performed by computing the direction-dependency for each software phantom and for every number of projections with the formula

$$D_t(\mathbf{x}^*, p) := \frac{(E_{max}(\mathbf{x}^*, p) - E_{min}(\mathbf{x}^*, p))}{n^2} \left(\frac{\cos\left(\pi \frac{E_{min}(\mathbf{x}^*, p)}{n^2}\right) + 1}{2} \right)^q, \quad (6)$$

where

$$E_{min}(\mathbf{x}^*, p) := \min_{\alpha=0^\circ, \dots, (\lceil \frac{180}{p} \rceil - 1)^\circ} E(\mathbf{x}^*, S(\alpha, p)), \quad (7)$$

$$E_{max}(\mathbf{x}^*, p) := \max_{\alpha=0^\circ, \dots, (\lceil \frac{180}{p} \rceil - 1)^\circ} E(\mathbf{x}^*, S(\alpha, p)), \quad (8)$$

and q is determined as the root of the equation

$$\left(\frac{\cos(\pi t) + 1}{2} \right)^q = 1 - t, \quad (9)$$

with a given $t \in (0, 1)$ value.

The function (6) simply gets the ratio of the missed pixels in the result and multiplies it with a correction function. The task of the correction function is to guarantee that the value of $D_t(\mathbf{x}^*, p)$ will not be too large, if the ratio of the missed pixels on the best result is much greater than t , therefore we can set an approximate threshold of accuracy we are interested in, with the parameter t . It is obvious, that the larger value means that the quality of the reconstruction is more dependent on the choice of the directions.

5 Experimental Results

We used the function defined in (6) to find the "software phantom-projection number" pairs which are the most sensitive to rotation, and to compare the best and worst reconstruction results. Of course, in a real application only accurate results would be acceptable, so we were interested in the cases when the results had only small errors, therefore we set the parameter $t = 0.001$. Figure 5 shows the values of our direction-dependency measurement $D_t(\mathbf{x}^*, p)$ for Figures 4d-f.

First, we examined the simple phantoms, where there was no ring or rectangular stripe around the objects. In this case, for most of the phantoms we found that the direction-dependency is the largest when about 3-5 projections are available for the reconstruction. More precisely, for most of the phantoms, there is a

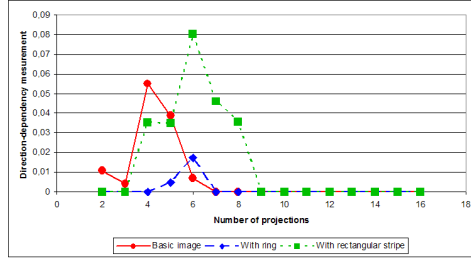


Fig. 5. Direction-dependency of the software phantoms of Figures 4d-f (the higher the values are the more dependent the phantom is to the choice of directions)

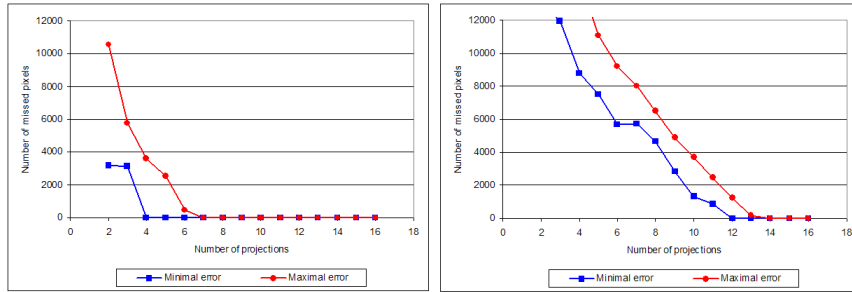


Fig. 6. Minimal and maximal number of missed pixels as it depends on the number of projections, for software phantoms of Figure 4e (left) and Figure 4a (right)

certain minimal number of projections where the algorithm can reconstruct the original objects almost perfectly, if there is a proper set of projections available, but we might need more projections when we take the wrong angles. For example, on the left of Figure 6 we can see that with the right angles we can get the original phantom from 4 projections, but if we have the worst possible angles we need 7 projections for the same results. Figures 8a-c give another example of a greatly direction-dependent software phantom, with the original phantom, and the best and worst reconstructions from 3 projections, respectively.

Although all the software phantoms showed this kind of sensitivity to the rotation of the projection sets, such great differences did not always occur. For example, on the right-hand side of Figure 6, we can see that there is no significant difference between the minimal and maximal error independently on the number of projections, so we can not get a much better result by finding the best angles. The results of this phantom reconstructed from 10 projections can be seen on Figures 8d-f, representing the original phantom, the best, and the worst results, respectively.

It is also useful to have a look at the exact errors for some of the phantoms, and projection numbers, according to the starting angles. Figure 7 shows the

number of missed pixels for the software phantoms shown in Figure 4g-i reconstructed from 3 and 4 projections. It immediately becomes visible that there is a big difference between the minimal and maximal values on each curve, which coincides with the previous statements, and proves that the choice of the projection angles can significantly influence the quality of the reconstruction. We can also observe that the curves depicted in Figure 7 are relatively smooth which suggests that it is not necessary to find the optimal angles for the projections to obtain a good reconstruction. Any angles close to the optimal ones can give acceptable results.

Comparing the curves of Figure 7 belonging to the different versions of the phantoms we can see that the original objects give the smallest errors. If we add a ring around the original objects then the curve looks similar but with greater error values. The explanation of this could be that the ring brings more instability into the equation-system of the reconstruction problem, but this symptom is still a subject to our further studies. We can also realize that the ring makes the relative difference between the best and worst results smaller compared to the total number of misreconstructed pixels, therefore the phantom becomes less dependent on the choice of the angles of the projections. Figures 8g-i give an example for this case, with the original software phantom, and the best and worst reconstructions from 6 projections. The situation is quite different when we add a rectangular stripe to the objects. An example is given in Figures 8j-l, with the original phantom, the best and, the worst results from 4 projections. In this case we can see that the error functions shown in Figure 7 can take extremely large values related to their global minima. The rectangle stripe added to the objects can be entirely reconstructed if two projection angles are aligned to its sides, and in this case the stripe does not effect the reconstruction, and the result is the same (or at least very close to that) as if there were no rectangular stripe. Furthermore, the farther the projection angles are from the proper alignment, the less accurate the reconstruction is (similarly as in the case of adding a ring to the objects). This means that adding such objects to the phantom can greatly increase the direction dependency of the reconstruction. We used equiangular

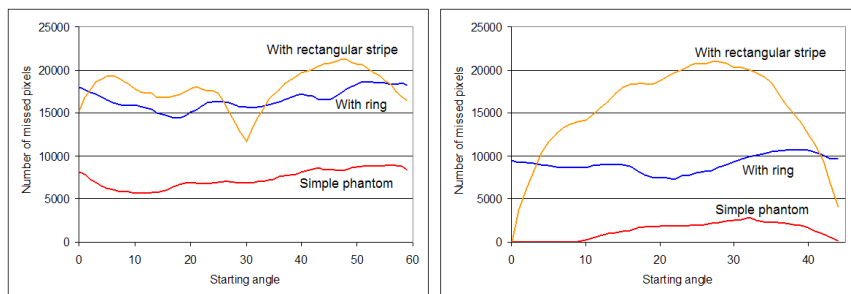


Fig. 7. Example for the number of missed pixels, for the software phantoms of Figures 4g-i, reconstructed from 3 (left) and 4 (right) projections

projection geometries, so we have two minima on the curves in the case when we had an odd number of projections, because only one projection can be aligned to the side of the rectangular stripe at once. In case of the ring we can not observe such minimal values because the ring is invariant to the rotation.

As a summation Table 1 shows the minimal number of projections required for a reconstruction with a ratio of missed pixels less than $t = 0.001$, for the best and worst projection sets of the phantoms. Again we can see that it is important to find the right angles, if we want to reduce the number of projections required for an acceptable reconstruction.

Finally, we had the assumption that the accuracy of the reconstruction is the best if the projections are taken from the direction defined by the first or second principal components of the objects. Although reconstructions using the projections from the directions including the principal components usually give good results, our tests revealed that - at least empirically - there is no straightforward connection between the best projection directions and the principal components.

Table 1. Minimal number of projections required for a reconstruction with a ratio of missed pixels less than $t = 0.001$, for the best and worst projection sets of the images, each column representing the results of a software phantom (s.p. - simple phantom; w.r. - phantom with ring; r.s. - phantom with rectangular stripe)

	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.	15.
s.p. - best	4	5	12	5	4	4	4	4	4	4	4	4	4	4	4
s.p. - worst	5	6	14	6	7	5	5	5	5	5	5	5	5	6	5
w.r. - best	-	-	-	-	6	6	7	6	6	6	7	6	7	7	7
w.r. - worst	-	-	-	-	7	7	7	7	7	6	7	7	7	7	7
r.s. - best	-	-	-	-	6	6	6	6	4	6	6	6	4	6	6
r.s. - worst	-	-	-	-	9	9	9	9	9	9	9	9	9	9	9

6 Direction-Dependency in Non-Destructive Testing

In industry, there is often a need to get information about the interior of objects (industrial parts) in a non-destructive way, i.e. without damaging the object itself. This process is called non-destructive testing. In these applications the information about the object is usually collected by transmission tomography using X-rays or neutron rays to form the projections of the object. Since the acquisition of such projections can be very expensive and time-consuming, it is important to keep the number of projections as small as possible. If the object is made of homogeneous material then an approach to achieve this is to apply binary tomography for the reconstruction [3].

A frequent task in NDT is to determine how similar the object studied to the given blueprint image is. The way to do it is the following. One places the object into the scanner, forms its projections from a few directions, and applies some (binary) reconstruction method to obtain an image from the object. Finally, the

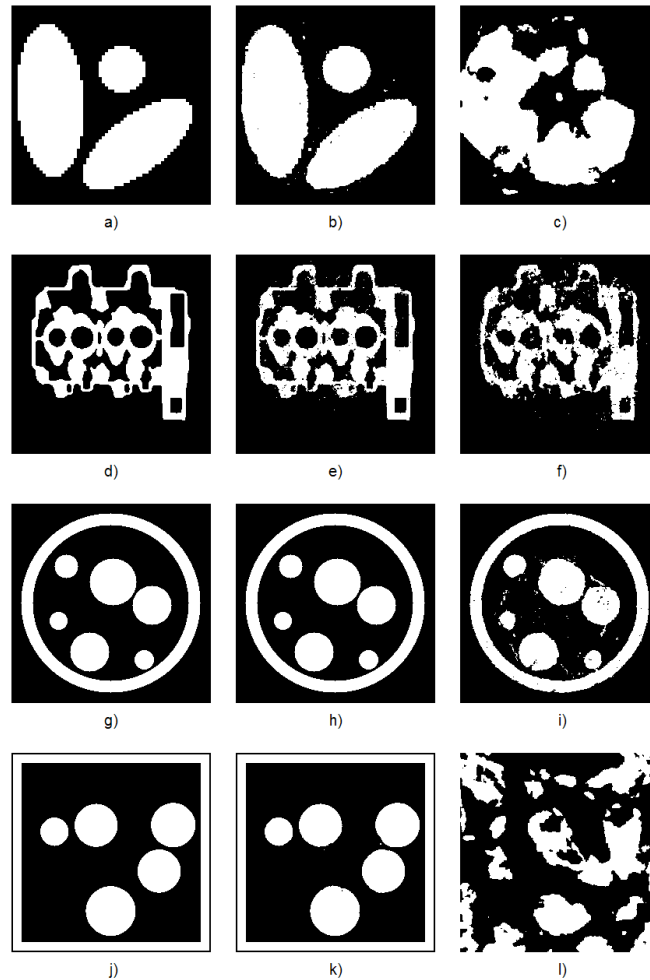


Fig. 8. Examples for the best and worst reconstructions of software phantoms, with given projection numbers. The original phantoms (first column), and the best and worst results (second and third column, respectively) for the same number of projections but different starting angles (b, c: 3 projections with starting angles 5° and 36° ; e, f: 10 projections with 0° and 5° ; h, i: 6 projections with 6° and 25° ; k, l: 4 projections with 0° and 27°).

difference of the blueprint and the reconstructed images is measured according to an arbitrary similarity metric. Since the blueprint image is available in advance, we can simulate its projections in arbitrary directions, and perform all the tests of Section 5 in order to characterize the blueprint image from the viewpoint of direction-dependency. This information turns out to be especially useful in several scenarios of NDT.

If there is a reference mark on both the benchmark and the studied objects then it is possible to place this latter one with a rotation of arbitrary known degree into the scanner. From the dependency function of the blueprint image, similar to that of Figure 7, we know when the best reconstruction quality can be achieved – we simply have to seek the global minimum of the function. This determines how (i.e., in which direction) to place the test object into the scanner to have the most accurate reconstruction from the available number of projections. Since the dependency function is smooth, it is sufficient to place the object with only approximately the same rotation as the dependency function suggests.

On the other hand, if there is no mark on the studied object, then it might be placed with an unknown rotation into the scanner. Again, from the dependency function of the blueprint image we can predict how sensitive our test will be to this rotation. In addition, from the graph of the blueprint image similar to that of Figure 6 we can deduce how many projections are needed to keep the maximal error acceptably low, i.e., to be sure that the effect of rotation will be eliminated. If it is impossible to acquire so many projections, then from the minimal error we can estimate the best reconstruction possible from the given number of projections. This knowledge is also useful, since it tells us whether DC algorithm is appropriate for the given industrial test. If the error of the best reconstruction is still high, then we might classify perfect objects as damaged ones and vice versa.

7 Conclusion and Further Work

The aim of this paper was to study how the accuracy of the DC binary tomography algorithm depends on the direction of the projections available for the reconstruction. We introduced two approaches to evaluate the direction-dependency of the DC algorithm, and conducted experiments on software phantoms. We found that certain objects behave considerably sensitively from the viewpoint of direction-dependency. On the other hand, there are objects for which the result of reconstruction is less dependent on the direction of the projections, but even in those cases choosing the proper directions can reduce the number of projections needed for an accurate reconstruction. Our investigations can be essentially useful in the non-destructive testing of industrial objects made of homogeneous materials.

The presented results can be extended in many different ways. In our future work we intend to perform similar studies on other reconstruction algorithms, and – in the same time – investigate the effect of noise added to the projections. We also want to examine whether the observations of this paper still hold if the projections are non-equiaxially acquired or if the investigated objects consist of more than one materials.

Finally, we are planning to extend the investigation on adaptive projection acquisition (like in [12]), i.e., to determine the the best angles half way through the data acquisition, by using information gathered from projections already made, when a blueprint image is not available.

Acknowledgments

The authors would like to thank Joost Batenburg and Christoph Schnörr for providing test images to the studies.

References

1. Alpers, A.: *Instability and Stability in Discrete Tomography*. Ph.D. Thesis, Technische Universität München. Shaker Verlag, Aachen (2003)
2. Batenburg, K.J., Sijbers, J.: DART: a fast heuristic algebraic reconstruction algorithm for discrete tomography. In: *IEEE Conference on Image Processing IV*, pp. 133–136 (2007)
3. Baumann, J., Kiss, Z., Krimmel, S., Kuba, A., Nagy, A., Rodek, L., Schillinger, B., Stephan, J.: *Discrete Tomography Methods for Nondestructive Testing*. In: [8], ch. 14, pp. 303–331 (2007)
4. van Dalen, B.E.: *Stability results for two directions in discrete tomography*. arXiv:0804.0316 [math.CO] (2008)
5. Di Gesu, V., Lo Bosco, G., Millonzi, F., Valenti, C.: A memetic algorithm for binary image reconstruction. In: Brimkov, V.E., Barneva, R.P., Hauptman, H.A. (eds.) *IWCIA 2008*. LNCS, vol. 4958, pp. 384–395. Springer, Heidelberg (2008)
6. Herman, G.T.: *Fundamentals of Computerized Tomography, Image Reconstruction from Projections*, 2nd edn. Springer, London (2009)
7. Herman, G.T., Kuba, A. (eds.): *Discrete Tomography: Foundations, Algorithms and Applications*. Birkhäuser, Boston (1999)
8. Herman, G.T., Kuba, A. (eds.): *Advances in Discrete Tomography and Its Applications*. Birkhäuser, Boston (2007)
9. Kak, A.C., Slaney, M.: *Principles of Computerized Tomographic Imaging*. IEEE Press, New York (1999)
10. Nagy, A., Kuba, A.: Reconstruction of binary matrices from fan-beam projections. *Acta Cybernetica* 17(2), 359–385 (2005)
11. NVIDIA CUDA Programming Guide, Version 2.0, http://developer.download.nvidia.com/compute/cuda/2_0/docs/NVIDIA_CUDA_Programming_Guide_2.0.pdf
12. Placidi, G., Alecci, M., Sotgiu, A.: Theory of adaptive acquisition method for image reconstruction from projections and application to EPR imaging. *Journal of Magnetic Resonance, Series B*, 50–57 (1995)
13. Schüle, T., Schnörr, C., Weber, S., Hornegger, J.: Discrete tomography by convex-concave regularization and D.C. programming. *Discrete Applied Mathematics* 151, 229–243 (2005)
14. Weber, S., Nagy, A., Schüle, T., Schnörr, C., Kuba, A.: A benchmark evaluation of large-scale optimization approaches to binary tomography. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) *DGCI 2006*. LNCS, vol. 4245, pp. 146–156. Springer, Heidelberg (2006)