



# Discrete Reconstruction Techniques<sup>★</sup>

Attila Kuba<sup>1</sup>, László Ruskó<sup>2</sup>, Zoltán Kiss<sup>3</sup>, and Antal Nagy<sup>4</sup>

*Department of Image Processing and Computer Graphics  
University of Szeged  
Szeged, Hungary*

---

## Abstract

*Discrete REConstruction Techniques (DIRECT)* is a program system for generating test images, for computing their projections, for performing reconstructions, for comparing reconstructed images, and for visualizing the results. Our framework supplies a solid workflow to be performed during testing DT methods. A collection of DT reconstruction methods is planned to be implemented in this system to help further comparisons and software developments in DT.

*Keywords:* discrete tomography, program system.

---

## 1 Introduction

*Tomography* is an imaging procedure where the cross-sections of the 3D object being studied are determined from its projection images. Usually several projections of the object are acquired from different directions. Then the task is

---

<sup>★</sup> This work was supported by OKTA Grant T 048476 and NSF Grant DMS 0306215

<sup>1</sup> Email: [kuba@inf.u-szeged.hu](mailto:kuba@inf.u-szeged.hu)

<sup>2</sup> Email: [rusko@inf.u-szeged.hu](mailto:rusko@inf.u-szeged.hu)

<sup>3</sup> Email: [kiss@inf.u-szeged.hu](mailto:kiss@inf.u-szeged.hu)

<sup>4</sup> Email: [nagya@inf.u-szeged.hu](mailto:nagya@inf.u-szeged.hu)

to compute the cross-sections of the object via some mathematical procedure [4] called *reconstruction from projections*. This imaging technique is routinely used in *computerized tomography* (CT) for example, where the section images of the human body are computed from a huge number of measurements using transmitted X-rays. The general method for the reconstruction of 3D objects is that the 2D cross-sections of the objects are reconstructed from the projections measured in the plane of the selected section, effectively reducing the 3D reconstruction problem to a series of *2D reconstruction problems*. In the case of so-called *truly 3D reconstruction* the whole 3D object is reconstructed using rays in the whole 3D space.

DT is a special kind of tomography that can be applied if the object to be reconstructed consists of only a few known homogeneous materials (e.g. metal and wood). This information can be incorporated into the reconstruction process, giving one the opportunity of reconstructing simple objects from a much smaller number of projection values than is necessary for more complex objects. For this reason DT seems to be important in applications where the object consists of homogeneous regions and there is no opportunity or it is too costly to acquire lots of projections, like those in non-destructive testing, electron microscopy, and medicine. For a summary of the theory and applications of DT, see [5].

Several software packages exist for reconstructing of the discrete images (e.g., SNARK [14]). General problems come from the exchanging, transferring, and comparing of the result of the different DT methods. Our aim was to provide a software tool to solve these problems. DIScrete REConstruction Techniques (DIRECT) includes the phantom and projection generation, reconstruction, and comparison processes which are connected to DT methods.

The structure of the paper is as follows. In Section 2 we introduce our DIRECT framework. It contains two types of files: DIRECT data and instruction files. The structure of the DIRECT data files is described in Section 3. The DIRECT instruction files are presented in Section 4. An example showing the use of DIRECT is given in Section 5. Finally, we present conclusions connected with the system.

## 2 The DIRECT framework

DIRECT is a programming system designed to provide a uniform framework for implementing, comparing, and evaluating reconstruction algorithms used in discrete tomography. A number of reconstruction algorithms are incorporated already and the users can integrate their own methods into the system.

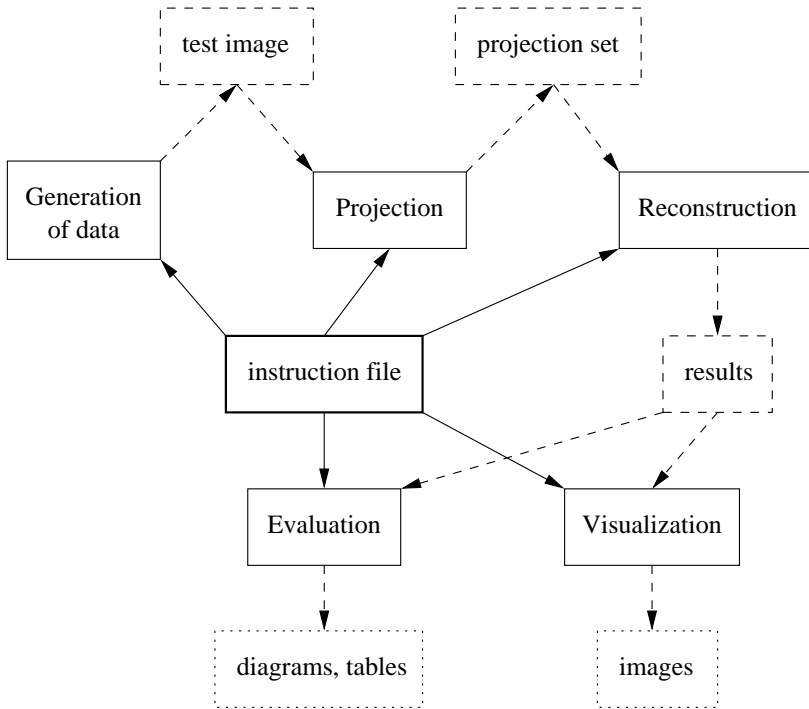


Fig. 1. The DIRECT framework: solid boxes are functions, dashed boxes are data files, dotted boxes are other data files. Solid lines represent control flow, while dashed lines represent data flow.

Our aim is not only to give a set of reconstruction algorithms but also to provide a tool for the statistical evaluation of the reconstruction algorithms. Beyond the reconstruction methods there are several programs for data (phantom or projection) generation, evaluation, and visualization. All these functions use the same format for parameter specification and data representation. The outline of DIRECT can be seen in Fig. 1.

The programs of DIRECT implement the following functions: phantom generation, projection, reconstruction, evaluation, and visualization. Each program can be used separately, but there is a framework with that the user can make complex experiments. Such experiments make it possible to run different reconstruction methods on the same projection data or to apply the same method with different parameters. The user can create *instruction files* containing control commands for the system, which executes the specified programs with the given parameters. The input data and the results are

located in so-called *data files*.

The next two sections describe the structure of the DIRECT data and instruction files.

### 3 DIRECT data files

Three types of data objects can be represented in DIRECT data files: *discrete images*, *projection sets*, and *additional information*. The first two types are used to represent image or projection data, while the latter is used to determine how data should be implemented or to describe the relation between different data objects.

#### 3.1 Data objects

We introduce the concept of  $d$ -dimensional ( $d \in \mathbb{Z}_+$ ) *discrete image*  $f$  as a function. Let  $I^d = \{x \in \mathbb{R}^d \mid 0 \leq x_i < N_i \text{ for all } 1 \leq i \leq d\}$  define the image domain, where the vector  $N \in \mathbb{Z}_+^d$  denotes the size of the domain. Given a finite set  $D$  of non-negative real numbers as the range of  $f$ , the discrete image is a function  $f : I^d \rightarrow D$  such that  $f$  is constant in each  $d$ -dimensional unit cube of the lattice  $Z^d$ .

Discrete images are represented in DIRECT by the vector  $\mathbf{x} \in \mathbb{R}^{N_1 \cdot N_2 \cdot \dots \cdot N_d}$  where each  $x_n$  represent the value of  $f$  in a  $d$ -dimensional cube of the lattice  $Z^d$ . Figure 2 shows a  $3 \times 2$  discrete image that can be represented by the vector  $\mathbf{x} = (1, 2, 0, 0, 2, 1)$ .

Data files can contain two types of discrete images. *Phantom images* are used for providing simulated data for reconstruction, while *result images* represent intermediate or final results of a reconstruction method. Only one phantom can be saved in a data file, but the number of result images is not limited.

In DT special measurements are available from the discrete images. These measurements, called *projections*, are usually integrals over subspaces of the continuous domain of  $f$ . An example for such a measurement (see Fig. 2.) is the line integral along straight lines of the  $d$ -dimensional Euclidean space.

In general, let  $\Theta = \{\theta \in \mathbb{R}^{d-1} \mid 0 \leq \theta_i < 2\pi\}$  denote a set of directions and let  $M_\theta$  denote the number of measurements belonging to direction  $\theta$ . The set  $p_\theta = \{m_{\theta,j} \mid 1 \leq j \leq M_\theta\}$  of measurements is referred as projection belonging to direction  $\theta$ . The *projection set* consists of projections belonging to different directions. Data files can contain one projection set that can consist of any number of projections.

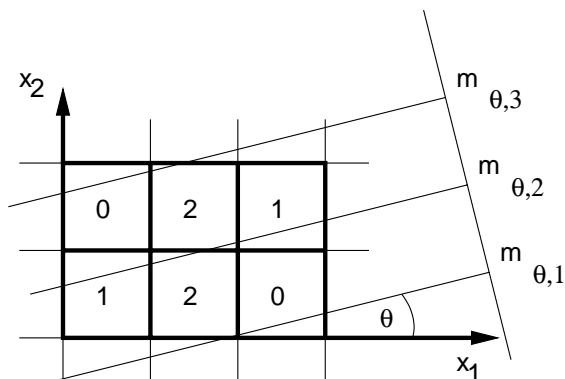


Fig. 2. 2-dimensional discrete image (left) where  $N = (3, 2)$ , and projection (set of line integrals) belonging to direction  $\theta$ .

*Additional information* can be any alphanumeric data which describe the attributes of the different data objects. In case of discrete images these information can be, e.g., the size of the image, in case of projection sets they describe how the measurements were taken (see Fig. 4).

### 3.2 Data format

DIRECT uses eXtensible Markup Language (XML) [11] to represent instructions and data. XML is a simple text format designed for electronic publishing and exchange of a wide variety of data on the Web and elsewhere. There are many advantages of using this format. First of all, XML is text format so its source is human readable and editable by any simple text editor. On the other hand, XML documents are portable between different operating systems.

The structure of the XML file can be specified using XML Schema Definition (XSD) which makes it easy to process these files by any application and permits of checking the internal consistency of the data stored in XML file, if it was created by the user. Using the eXtensible Stylesheet Language (XSL) the layout of the XML documents can be specified, so it does not need any program to be visualized but Internet browser.

XML allows the representation of the data objects to be internal or external. In the first case data is present in the XML file, so that images and projections are in human readable form. This is very useful, when a new method is developed, since it is very easy to verify the data stored in the file. The other way for data representation is to save the data in binary file and have a reference to that. This way permits of using any common file format

and saves disk space.

Beyond the data and instruction files, the result of various applications of DIRECT (e.g. evaluation, visualization) are available in standard format (HTML, VRML, PostScript) so that they can be imported in several applications with no regard to the software environment.

## 4 DIRECT instruction files

DIRECT consists of several programs which provide the following 5 functions: data generation, projection, reconstruction, evaluation, visualization. These functions are implemented by separated programs. In order to standardize how the user specifies the data and the parameters for any program of the system, we introduced some control commands. These commands can be placed in an *instruction file* that is executed by the DIRECT framework. The control commands are represented by XML structures called *nodes*. Each node can have attributes and other sub-nodes which are optional in some cases.

During the execution of an instruction file several data files can be created by different applications. The principle of file handling is that data files can only be appended and can not be modified (this helps to maintain the consistency between data objects). Instructions can belong to each data object, but these optional attributes do only describe what has already happened to the data (how it was achieved) and don't specify what to do with the data. The following sub-sections describe the five main groups of DIRECT according to the five function mentioned earlier.

### 4.1 Generation of test images

A subset of DIRECT applications was developed for creating images which can be used for testing reconstruction methods. Test images usually consist of geometric shapes which can be specified by some parameters. For example, the 3D object to be reconstructed can be a cylinder containing spheres. This 3D image in DIRECT can be described by geometric parameters (radii, heights, and coordinates) and the discrete values characterizing the materials of the objects.

### 4.2 Calculation of projections

DIRECT involves several applications for calculating projections of discrete images. We can use different models of projection measurements: line integrals, strip integrals, fan-beam, and cone beam models. In all these cases,

the projections can be defined using a linear equation system of the form:  $A\mathbf{x} = b$ , where  $b$  represents the projection set,  $\mathbf{x}$  is the representation of a discrete image, and the matrix  $A$  describes the projection geometry. In order to simulate real data noise can be added to the projections.

### 4.3 Reconstruction

Presently, we can reconstruct discrete objects from projections using one of the following methods:

- reconstruction of hv-convex discrete sets [1],
- reconstruction of hv-convex discrete sets with absorption [6],
- reconstruction of binary circles [7],
- reconstruction of images with more discrete levels [7].

### 4.4 Evaluation

In order to measure the differences between the original ( $\mathbf{x}$ ) and the reconstructed ( $\mathbf{x}'$ ) image, several measures of errors are implemented, such as the *fraction of misplaced voxels* (see [9])

$$MV = \frac{\sum |\mathbf{x}_j - \mathbf{x}'_j|}{2 \cdot \sum \mathbf{x}_j},$$

the *mean square error* (see [2,3]), the *shape error*

$$SE = \frac{2 \cdot \sum |\mathbf{x}_j - \mathbf{x}'_j|}{\sum \mathbf{x}_j + \sum \mathbf{x}'_j},$$

the *volume error*

$$VE = \frac{2 |\sum \mathbf{x}_j - \sum \mathbf{x}'_j|}{\sum \mathbf{x}_j + \sum \mathbf{x}'_j},$$

and the so-called *shape and volume conformity* (see [8,10]).

### 4.5 Visualization

Different kinds of data objects (images, projections) can be visualized in DIRECT. Due to the XML format the data files can be displayed using any web browser. On the other hand there are some applications which convert the different data objects into standard formats (image files, VRML objects, ...), so the user does not need any special program for visualization.

```

<direct_inst project="example">

  <generate>
    <comment>Example for test data generation</comment>
    <method name="polgen">
      <parameter name="ncols" value="12"/>
      <parameter name="nrows" value="8"/>
      <parameter name="conn" value="8"/>
    </method>
  </generate>

  <reconst>
    <comment>Example for reconstruction</comment>
    <method name="polrec_hybrid">
      <parameter name="conn" value="8"/>
    </method>
  </reconst>

  <display>
    <method name="convert_vrml">
      <parameter name="pixel" value="square"/>
      <parameter name="lattice" value="on"/>
      <parameter name="proj" value="graph"/>
      <parameter name="bgcolor" value="1.0"/>
      <parameter name="objcolor" value="0.5"/>
    </method>
  </reconst>

</direct_inst>

```

Fig. 3. Instruction file describing generation of test data, reconstruction, and visualization of the results.

## 5 Example for using DIRECT

In this Section we demonstrate the way of making experiments involving several applications of DIRECT. Figure 3 shows an instruction file that specifies the creation of a projection set, the reconstruction of the generated data, and the visualization of the results. The instruction files have a name (see attribute `project`) that specifies the name of the data file (`example.xml`) belonging to the given experiment. All functions executed during this experiment use this data file, unless other file is specified by the user.

The command `<generate>` is used to describe the parameters of a test data generation. As it can be seen from node `<method>`, the applied program called `polgen` has 3 parameters (the numbers of rows and columns of the image and the type of connectedness). This program generates an hv-convex binary image with the given parameters and saves its row and column sums into the data file.

At this point the resulted data file `example.xml` (see Fig. 4) consist of only

```

<direct_data dim="2">
  <projections ddist="1.000" dwidth="1.000" type="line">
    <comment>Example for test data generation</comment>
    <method name="polgen">
      <parameter name="ncols" value="12"/>
      <parameter name="nrows" value="8"/>
      <parameter name="conn" value="8"/>
    </method>
    <projection angle="0.000" datatype="int" ncols="8">
      <proj_ascii>3 4 7 10 9 6 5 1</proj_ascii>
    </projection>
    <projection angle="90.000" datatype="int" ncols="12">
      <proj_ascii>1 2 2 3 4 5 6 5 7 5 3 2 </proj_ascii>
    </projection>
  </projections>

  <reconstruction>
    <comment>Example for reconstruction</comment>
    <method name="polrec_hybrid">
      <parameter name="conn" value="8"/>
    </method>
    <image type="result">
      <comment>1. solution</comment>
      <image_bin datatype="byte" ncols="12" nrows="8">
        example.bin
      </image_bin>
    </image>
  </reconstruction>
</direct_data>

```

Fig. 4. Data file representing generated projection set and reconstruction results.

a node `<projections>`. This structure represents the generated projection set that in our case consists of 2 projections (belonging to 0 and 90 degrees). The attributes of the node `<projections>` show that the projections are line integrals where the detector distance (`ddist`) is equal to 1 (this is measured in the pixel size that is assumed to be 1 in DIRECT). The value of the attribute (`ddist`) is important only if the integrals are calculated along strips (`type="strip"`). The projection geometry is parallel (projection sets having divergent geometry have an additive node `<divergent>` within the structure `<projections>`). Since the projection data is small, it can be saved in human readable form into the data file using the node `<image_ascii>`. The node `<method>`, which is optional for all data objects, shows how the test data was created.

According to the instruction file the next function is reconstruction. In this phase the program called `polrec_hybrid` is executed. This method can reconstruct hv-convex binary images from two projections. It has one parameter,

namely, the type of connectedness of the object to be reconstructed. The result of the reconstruction is saved in the data file using a node `<reconstruction>`. This structure represents one image that is a final (not an intermediate) result according to the attribute `type` of the node `<image>`. The image data represented by the node `<image_bin>` is saved in the file `example.bin` that consists of  $12 \times 8$  bytes.

The last function of the instruction file specifies the visualization of the results. The information found in the data file can be displayed using a web browser (see Fig. 5), but the data object should be converted into a standard file format. The program `convert_vrml` converts binary images with two projections into VRML format. According to the parameters the pixels are displayed as squares, the lattice is visible, projections are drawn as bar-graphs, the background color is white, and the points of the object are gray. The result of this function can be seen in Fig. 6.

Examples for visualized discrete images can be seen in Figs. 7 - 11. The examples demonstrate various ways for displaying images with projections, comparing different images, and visualizing 3-dimensional discrete images using different styles. Figure 7 shows the comparison of two discrete objects. As it can be seen, pixels of the discrete objects can be displayed by disks or squares. The light and dark gray pixels represent the different objects while the intersection is medium gray. Projections can be visualized using numbers and bar-graphs as it is shown in Figs. 7 and 8. In most of the cases the discrete images are saved in a standard image format. Figures 9 and 10 show images with two and four discrete levels. Both examples consist of a phantom, an initial, a result, and a difference image. For visualizing 3-dimensional discrete images we can use VRML [12]. Figure 11 demonstrates a discrete volume having solid and transparent surfaces.

## 6 Conclusion

We intend to make all functions of DIRECT available for public via a web interface. There are function available soon for everybody (e.g., four reconstruction methods can be tested presently). The size of the images are limited to  $100 \times 100$  for such users. For special users we can offer more services after having e-mail contact.

### Projections:

**Comment:** Example for test data generation  
**Detector width:** 1.000  
**Detector distance:** 1.000  
**Type of projections:** line  
**Name of the generating method:** polgen  
**Parameters:**

Name	Value
ncols	12
nrows	8
conn	8

**Projection list:**

Angle	Size	Datatype	Location
0.000	8	int	Internal
90.000	12	int	Internal

---

### Reconstruction:

**Comment:** Example for reconstruction  
**Name of the generating method:** polrec\_hybrid  
**Parameters:**

Name	Value
conn	8

**Image list:**

Type	Number of iterations	Rows	Columns	Datatype	Location	Format	Comment
result	N/A	8	12	byte	example.bin	Binary	1. solution

Fig. 5. Example data file as it appears in the web browser.

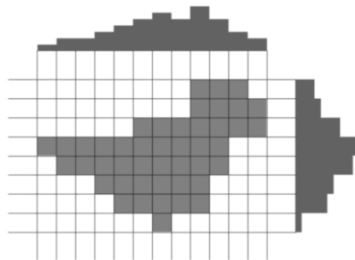


Fig. 6. Visualization of the result image and the projections using VRML.

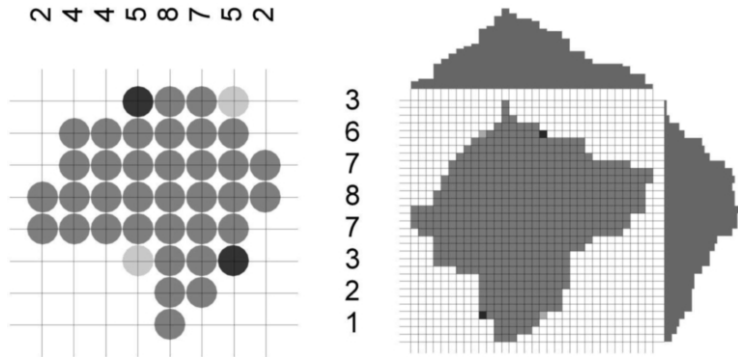


Fig. 7. Visualization of the difference between two discrete image.

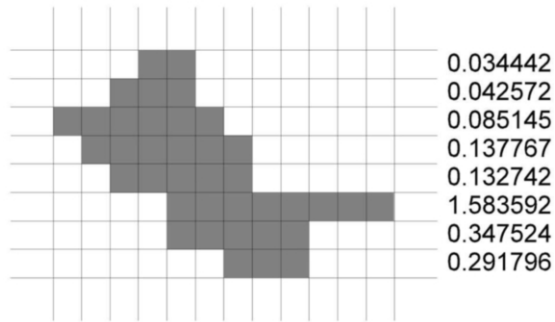


Fig. 8. Visualization of a discrete image from absorbed projections.



Fig. 9. Reconstruction of disks located inside a ring: phantom (left), initial, result, and difference (right).

## References

- [1] Balogh, E., A. Kuba, Cs. Dévényi, A. Del Lungo, *Comparison of algorithms for reconstructing hv-convex discrete sets*, Linear Algebra and Its Applications **339** (2001), 23–35.

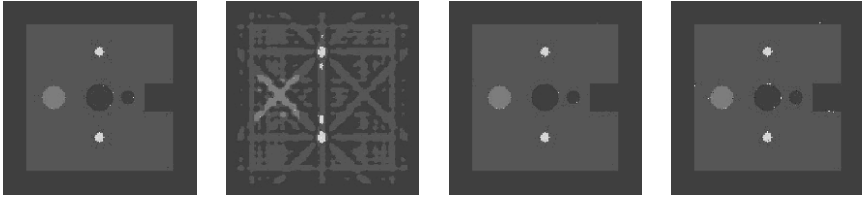


Fig. 10. Reconstruction of discrete images: phantom (left), initial, result, and difference (right).

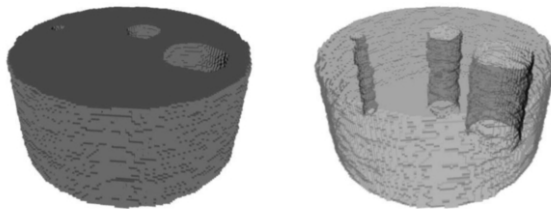


Fig. 11. Visualization of a 3-dimensional discrete image.

- [2] Chan, M. T., G. T. Herman, and E. Levitan, *Bayesian image reconstruction using image-modeling Gibbs priors*, *Int. J. Imaging Systems Techn.* **9** (1998), 85–98.
- [3] Chan, M. T., G. T. Herman, and E. Levitan, “Probabilistic modeling of discrete images,” in G.T. Herman, A. Kuba (Eds.), *Discrete Tomography. Foundations, Algorithms, and Applications*, Birkhäuser: Boston, MA, (1999), 213–235.
- [4] G.T. Herman, *Image Reconstruction from Projections*, Academic Press, (1980).
- [5] Herman, G.T., A. Kuba, A. (Eds.), “Discrete Tomography: Foundations, Algorithms and Applications” Birkhäuser, Boston, (1999).
- [6] Kuba, A., A. Nagy, *Reconstruction of hv-convex binary matrices from their absorbed projections*, *Electronic Notes in Theoretical Computer Science* **46** (2001), 383–393.
- [7] Kuba, A., L. Ruskó, L. Rodek, Z. Kiss, *Preliminary Studies of Discrete Tomography in Neutron Imaging*, *IEEE Trans. on Nuclear Science* **52** (2005), 380–385.
- [8] Prause, G. P. M. and D. G. W. Onnasch, *Binary reconstruction of the heart chambers from biplane angiographic image sequence*, *IEEE Trans. Medical Imaging* **15** (1996), 532–559.

- [9] Robert, N., F. Peyrin, and M. J. Yaffe, *Binary vascular reconstruction from a limited number of cone beam projections*, *Medical Physics* **21(12)** (1994), 1839–1851.
- [10] Slump, C. H., and J. J. Gerbrands, *A network flow approach to reconstruction of the left ventricle from two projections*, *Comp. Graphics Image Proc.* **18** (1982), 18–36.
- [11] eXtensible Markup Language, [www.w3.org/XML/](http://www.w3.org/XML/).
- [12] Virtual Reality Modeling Language, <http://www.web3d.org/x3d/vrml/>.
- [13] DIRECT homepage, <http://www.inf.u-szeged.hu/direct/>.
- [14] <http://www.cs.gc.cuny.edu/~gherman/snark2001.html>,  
Homepage of SNARK93 software system, “A Programming System for 2D Image Reconstruction from Projections”.