3D RECONSTRUCTION WITH X-RAY IMAGES

[1] Hong-Yi Chen (陳弘毅), [2] Chiou-Shann Fuh (傅楸善), , Cheng-Shih Wong (翁丞世), Tzu-Chia Tung (董子嘉), You-Hsien Lee (李祐賢), Ting-Chen Tsan (昝亭甄)

Dept. of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
E-mail: r04922099@ntu.edu.tw; fuh@csie.ntu.edu.tw; mob5566@gmail.com;
d04944016@ntu.edu.tw; ab122858280@gmail.com; tusbasa1@gmail.com;

ABSTRACT

In our project, we attempt to reconstruct the 3D model of solder balls on the PCB (Printed Circuit Board) according to many projections of X-ray images of those solder balls. We mainly use two methods: SART (Simultaneous Algebraic Reconstruction Technique) and FBP (Filtered Back Projection). Consequentially, we finally obtain the 3D model of those solder balls, which can be used to inspect their inner structure. We also optimize our execution time and compare our result with that of Volume Graphics.

Keywords: *X-ray inspection, PCB, SART, FBP*

1. INTRODUCTION

1.1. Overview

We aim to implement the 3-dimensional reconstruction with X-ray images. The X-ray images are obtained by taking several photographs from different angles. With these X-ray images, we can reconstruct the images of PCB surface at different levels to inspect defect without breaking the PCB.
In this thesis, our goal is not only to produce high quality of the output images so that the void in solder balls are clear but also to make our program as fast as possible. We mainly focus on two parts: reconstruction and acceleration.

1.2. Reconstruction

Reconstruction plays a vital role in inspection because the better result of the reconstruction, the more effective we inspect the defects. The structure in our projection system is illustrated as follows:

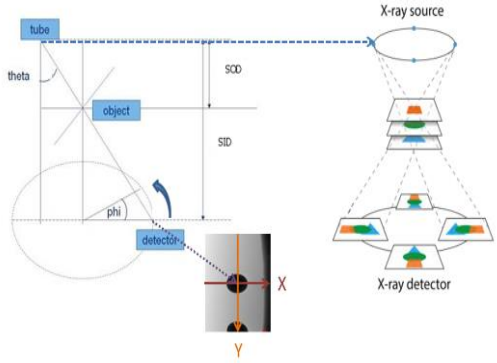

Figure 1 1: System setup. Theta is the angle between tube to detector and vertical line. SOD is Source to Object Distance. SID is Source to Image Distance, and phi is the angle of rotation.

By the relative motion between the source and detector, the system can obtain an image including tilt object after rotating phi degrees (=(360°)/(the number of projection images)). When the source and detector go through 360 degrees, we can use those images to implement our reconstruction.

Many algorithms were proposed for reconstruction such as Algebraic Reconstruction Technique (ART) [13]. ART is the simplest method for reconstruction based on Radon transform, however, the quality of output images and computation efficiency are not as satisfactory [15].

Thus, we find another way to reach our goal. We decide to implement two different common approaches: Simultaneous Algebraic Reconstruction Technique (SART) and Filtered Back Projection (FBP) [10]. For FBP, we also compare the results with different kinds of filters including ramp filter, Hann filter, Hamming filter, cosine filter, and Shepp-Logan filter.

1.3. Acceleration

After obtaining the correct reconstruction result, we will further pursue the shortest execution time in order to

come up to the expected throughput. We aim at adjusting the process of the reconstruction algorithm to make program executing more effective. Furthermore, Intel® Threading Building Blocks library is used to help the parallel loop execution to be accelerated.

2. BACKGROUND

2.1. Radon Transform

In mathematics, the Radon transform was introduced in 1917 by Johann Radon, who also provided a formula for the inverse transform [15]. The transform is the integral transform which takes a function *f* defined on the plane to a function $g(\phi, s)$ defined on the (two-dimensional) space of lines in the plane, whose value at a particular line is equal to the line integral of the function over that line. The Radon transform is widely applicable to tomography, the creation of an image from the projection data associated with cross-sectional scans of an object [15].

A function $g(\varphi, s)$ is the line integral of the image intensity $f(x, y)$, along a line l that is distance s from the origin and at angle φ off the *x*-axis [7].

$$g(\phi, s) = \int_l f(x, y) dl$$

All points on this line satisfy the equation:

$$xsin(\phi) - ycos(\phi) = s$$

Therefore, the forward projection function $g(\phi, s)$ can be rewritten as:

$$g(\phi, s) = \int \int f(x, y)\delta(xsin\phi - ycos\phi - s)dxdy$$

where the $\delta(n)$ is a delta function, if $n = 0$, $\delta(n) = 1$; else, $\delta(n) = 0$.

The collection of these $g(\phi, s)$ at all ϕ is called the Radon transform of Image.

If a function f represents an unknown density, then the Radon transform represents the projection data obtained as the output of a tomographic scan. Hence the inverse of the Radon transform can be used to reconstruct the original density from the projection data, and thus it forms the mathematical underpinning for tomographic reconstruction, also known as image reconstruction [15].

2.2. Fourier Slice Transform

In mathematics, applying the Fourier Slice Transform, central slice theorem, or projection-slice theorem [14] to two-dimensional image that the results of the following two computations are equal:

(i) Take a two-dimensional function, project it onto a (one-dimensional) line, and do a Fourier transform of that projection.

(ii) Take the same function above, but do a two-dimensional Fourier transform first, and then slice it through its origin, which is parallel to the projection line.

In operator terms, if F_1 and F_2 are the 1- and 2-dimensional Fourier transform operators mentioned above, P_1 is the projection operator (which projects a 2-D function onto a 1-D line) and S_1 is a slice operator (which extracts a 1-D central slice from a function), then:

$$F_1 P_1 = S_1 F_2$$

The projection-slice theorem is well proven for the case of two dimensions. Without loss of generality, we can take the projection line to be the *x*-axis and the law still applies if we use a shifted and rotated line. Using a shifted line (in *y*-axis) gives the same projection and therefore the same 1D Fourier transform results. The rotated function is the Fourier pair of the rotated Fourier transform, for which the theorem again holds.
If $f(x, y)$ is a two-dimensional function, then the projection of $f(x, y)$ onto the *x*-axis is $p(x)$ where

$$p(x) = \int_{-\infty}^{\infty} f(x, y) dy$$

The Fourier transform of $f(x, y)$ is

$$F(k_x, k_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-2\pi i(xk_x + yk_y)} dxdy$$

The slice is then $s(k_x)$

$$\begin{aligned} s(k_x) = F(k_x, 0) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-2\pi i x k_x} dxdy \\ &= \int_{-\infty}^{\infty} [f(x, y) dy] \int_{-\infty}^{\infty} e^{-2\pi i x k_x} dx \\ &= \int_{-\infty}^{\infty} p(x) e^{-2\pi i x k_x} dx \end{aligned}$$

According to the equation above, the slice is not associated with *y*-axis and the rotation is correspondent, so using a shifted line (in *y*-axis) gives the same Fourier transform results and the rotated function is the Fourier pair of the rotated Fourier transform.

2.3. Radon Transform with Fourier Slice Transform

The Fourier slice transform shows that the 1D Fourier transform of the projection function $g(\phi, s)$ mentioned in Radon transform is equal to the 2D Fourier transform

of the image evaluated on the line that the projection was taken on (the line that $g(\phi, 0)$ was calculated from) [7]. Thus, we know what the 2D Fourier transform of the image looks like (or at least what it looks like on certain lines and then interpolate), and we can simply take the 2D inverse Fourier transform and obtain the original image. We can show the Fourier slice theorem in the following way:
The 1D Fourier Transform of g is given by:

$$G(\phi, \omega) = \int e^{-j\omega s} g(\phi, s) ds$$

Substitute the expression for $g(\phi, s)$ into the Radon transform to get:

$$G(\phi, \omega) = \int \int \int f(x, y) e^{-j\omega s} \delta(x sin\phi - y cos\phi - s) dx dy ds$$

Use the sifting property of the Dirac delta function to simplify to get:

$$G(\phi, \omega) = \int \int f(x, y) e^{-j\omega(x sin\phi - y cos\phi - s)} dx dy$$

The definition of the 2D Fourier transform of f:

$$F(u, v) = \int \int f(x, y) e^{-j(ux+vy)} dx dy$$

It shows that Radon transform with Fourier slice transform is just $F(u, v)$ evaluated at $u = w * sin(\phi)$ and $v = -w * cos(\phi)$, which is the line corresponding to the projection $g(\phi, s)$. Therefore, we can apply some image operation in Fourier domain instead of spatial domain to get some acceleration and inverse to spatial domain to get our output.

2.4. Filter

For Filtered Back Projection, filtering plays a significant role by changing the views in two ways. First, the top of the pulse is made flat, resulting in the final back projection creating a uniform signal level within the circle. Second, negative spikes have been introduced at the sides of the pulse. When back projected, these negative regions counteract the blur near the target object [10].

There are different methods to filter the reconstructed image. We compare with different kinds of filters including Ramp filter, Hann filter, Hamming filter, cosine filter, and Shepp-Logan filter.

- Ramp filter

  We expect that a high-pass filter would eliminate the artifact by amplifying high-frequency components in the back projection. Ramp filter is generally used in practice.

- Hann filter

  The statistical noise in the data manifests in Fourier space as high-frequency components. Thus the process of filtered back projection amplifies noise in the image. In order to reduce this effect, a range of modifications to the ramp filter can be used. The most commonly used of these is the Hann filter [1].

  $$w(n) = \frac{1}{2}\left[1 + cos\left(\frac{2\pi n}{N-1}\right)\right], \quad 0 \le n \le N-1$$

  where Nrepresents the width.

- Hamming filter

  The window with these particular coefficients was proposed by Richard W. Hamming [17]. The window is optimized to minimize the maximum (nearest) side lobe, giving it a height of about one-fifth that of the Hann window.

  $$w(n) = 0.54 - 0.46 * cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n \le N-1$$

  where Nrepresents the width.

- Cosine filter

  $$w(n) = \cos\left(\frac{\pi n}{N-1}\right), \quad 0 \le n \le N-1$$

  where Nrepresents the width.

- Shepp-Logan filter

  $$w(n) = \frac{\sin\left(\frac{\pi n}{N-1}\right)}{\frac{\pi n}{N-1}}, \quad 0 \le n \le N-1$$

  where Nrepresents the width.

3. METHODOLOGY

3.1. Overview

In this chapter, detailed methodology is completely discussed, including how to implement the two reconstruction methods: SART and FBP. In order to get higher throughput, we make our program executes in parallel by Threading Building Blocks (Intel® TBB) by Intel. For FBP, we additionally generate a filter and enhance the reconstructed images contrast. More details will be discussed in next sub-sections. The flow of our proposed algorithm is as follows:

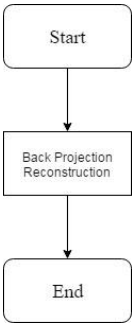

Figure 3-1: The flowchart of SART.

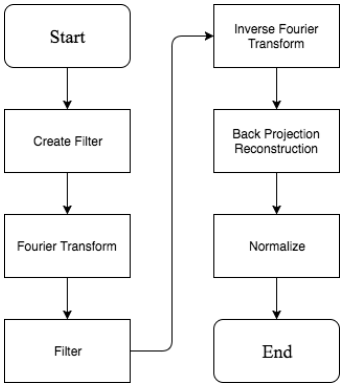

Figure 3-2: The flowchart of FBP.

3.2. Simultaneous Algebraic Reconstruction Technique

The Simultaneous Algebraic Reconstruction Technique [16], proposed by Anders Andersen and Avinash Kak in 1984, has had an improvement in Computed Tomography (CT). Although the projection data which the system possesses is limited, the SART still generates a good reconstruction in just one iteration, and it is superior to standard Algebraic Reconstruction Technique.

The ART computes an approximation of the solution of the linear systems of equations as follows [4],

$$f_{ij}^{q+1} = f_{ij}^{q} + \frac{p_j - \sum_{i=1}^{N} f_{ij}^{q}}{N}$$

where q is iteration; p_j is the measured data for a projection j; $\sum_{i=1}^{N} f_{ij}^{q}$ is the sum of the reconstructed elements along the ray; f_{ij} is an i-th element of the matrix A along the projection ray p_j; and N is the number of the reconstruction elements.

In computation, the ART focuses on pixels along a ray in each projection. Conversely, the idea of SART is based on that a pixel must be through by not just one ray in one iteration. It makes a great difference between SART and ART that ART only updates all the pixels along one ray in one iteration, but SART updates a pixel by a value considering all the ray through it in one iteration [5]. Consequently, SART is much faster with solving equation than ART.

The SART computes an approximation of the solution of the linear systems of equations with a weighting matrix W [11]. The element w_{ij} in W denotes the contribution of f_{ij} to p_j. We assume that the width of each ray is equal to the image pixel size. Then the element w_{ij} can be calculated (or, at least, estimated) according to the projecting procedure as follows:

$$w_{ij} = \begin{cases} 1, & \text{the } j\text{-th ray passes the center of the } i\text{-th pixel} \\ 0, & \text{else} \end{cases}$$

Sum up the main points of the above, the SART computes an approximation of the solution of the linear systems of equations as follows [4]

$$f_{ij}^{q+1} = f_{ij}^{q} + \frac{\sum_{p_j \in P_\phi} \lambda \dfrac{p_j - \sum_{k=1}^{N} f_{ij}^{q} w_{ik}}{\sum_{k=1}^{N} w_{ik}}}{\sum_{p_j \in P_\phi} w_{ij}}$$

where q is iteration; p_ϕ is all projection values in one degree; p_j is the measured data for a projection j; λ is relaxation parameter; $\sum_{i=1}^{N} f_{ij}^{q}$ is the sum of the reconstructed elements along the ray; f_{ij} is an i-th element of the matrix A along the projection ray p_j; and N is the number of the reconstruction elements.

3.3. Filtered Back Projection

The Radon transform goes with Fourier slice theorem and Fourier transform, often called the Filtered Back Projection, which is another way to reconstruct image. Unlike ART or SART which are iterative methods, FBP is a direct method. It is a technique to correct the blurring encountered in simple back projection.

For FBP, each view is filtered before the back projection to counteract the blurring point spread function. That is, each of the one-dimensional view is convolved with a one-dimensional filter kernel to create a set of filtered views. These filtered views are then back projected to provide the reconstructed image which is much more correct than the result from back projection.

In mathematics, first recall the definition for the 2D inverse Fourier Transform [7]

$$f(x,y) = \frac{1}{(2\pi)^2} \int\int_{R^2} F(u,v) e^{j(ux+vy)} dxdy$$

Make a change of variable from rectangular to polar coordinates and replace $F(\phi, w)$ with $G(\phi, w)$, we get

$$f(x,y) = \frac{1}{4\pi} \int \int_{R^2} G(\phi,\omega) e^{j\omega(xsin\phi + ycos\phi)} |w| dxdy$$

where $|w|$ is the determinant of the Jacobian of the change of variable from rectangular to polar coordinates. Notice that we have to multiply our projections by $|w|$ in the Fourier domain. Thus we can see $|w|$ as a filter.

$$Q(\phi,\omega) = G(\phi,\omega)|w|$$

This product is called the filtered back projection at angle ϕ.

To sum up, the FBP algorithm consist of the following steps [8]:

(i) Compute a one-dimensional Fourier transform of each projection.
(ii) Multiply the Fourier transform of each projection by the weighting filter.
(iii) Compute the inverse one-dimensional Fourier transforms of the filtered projection.
(iv) Apply back projection to the processed projections in spatial domain to obtain the two-dimensional reconstructed object.

For the filter we use in this thesis, we generate a modified Hann filter

$$w(n) = \frac{1}{3}\left[1 + cos\left(\frac{2\pi n}{N-1}\right)\right], \qquad 0 \le n \le N-1$$

where Nrepresents the width.

The difference between Hann filter and our modified filter is the constant. We substitute $\frac{1}{3}$for $\frac{1}{2}$. Our filter may make the image much smoother than Hann filter.

3.4 Back Projection

3.4.1. Value

The characteristics of the X-ray system as mentioned in Figure 1-1, the relative motion between the source and detector generates projections with angle θ $(= 35°)$.

First, we build a three-dimensional matrix which contains the whole PCB we aim to reconstruct where I_ϕ is a projection image with rotated angle ϕ $(= \frac{360° \times i}{\text{the number of projection images}}, 0 < i <$ the number of projection images), and we apply every I to reconstruct the image of slice S with different heights. Each pixel on slice may be passed by many X-ray. As a result, the intensity of each pixel is the accumulation of values of every X-ray passing through.

$$f_{ij} = \sum p_k$$

where f_{ij} is an intensity of i-th pixel on slice S with height j, and p_k is k-th projection passing through pixel f_{ij}.

3.4.2. Coordinate

Now we have the value to assign to target pixel. The next step is to define which pixel on the slice with different height corresponding to the pixel in projection images. Based on an idea that the objects on focal plane stay consistent in all projections, we apply focal plane as a reference to derive the corresponding coordinate. Illustrated as follows:

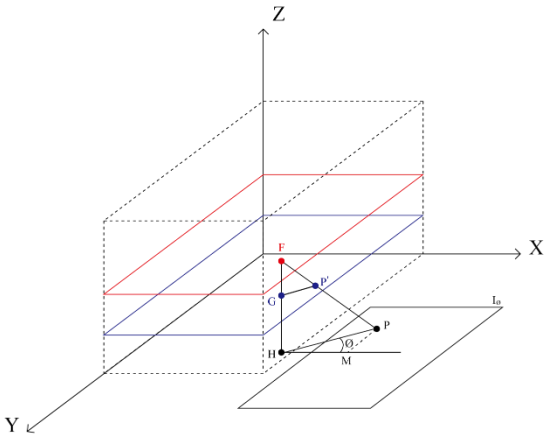

Figure 3-3: Illustrate the architecture to find the pixel on $\boldsymbol{z = z_{p'}}$ passed by a certain X-ray.

According to Figure 3-3, the point F is on the focal plane $z = z_F$, and P is its corresponding point in the projection image. The projection ray $\overline{FP}$ intersects with plane $z = z_{P'}$ at P' where $\overline{FH}$ is a perpendicular line with plane X-O-Y, which intersects with plane $z = z_{p'}$ at G; ϕ is rotated degree; and θ is the angle between tube to detector and vertical line.

Assume the coordinate of F to be

$$F(x_F,\ y_F,\ z_F)$$

We can get the coordinates of G, H

$$G(x_F,\ y_F,\ z_{p'})$$
$$H(\ x_F,\ y_F, 0)$$

The points P and F are on the same line. In triangle HMP,

$$\Delta\ x = z_F * tan\theta * cos\theta$$
$$\Delta\ y = -z_F * tan\theta * sin\theta$$

Thus, we can get the coordinate of P

$$P(x_F + z_F * tan\theta * cos\theta,\ y_F - z_F * tan\theta * sin\theta, 0)$$

Triangle FHP is similar to triangle FGP'. Thus

$$\frac{\overline{GP'}}{\overline{HP}} = \frac{\overline{FG}}{\overline{FH}}$$

$$\Rightarrow \begin{cases} x_{P'} = x_F + tan\theta * cos\theta * (z_F - z_{p'}) \\ y_{P'} = y_F - tan\theta * sin\theta * (z_F - z_{p'}) \end{cases}$$

Finally, we can get the target point coordinate

$$p'(x_F + tan\theta * cos\theta * (z_F - z_{p'}), y_F - tan\theta sin\theta * (z_F - z_{p'}), z_{p'})$$

3.5. Contrast Enhancement

The image shown to users should be 8-bit image, but our reconstructed image is 16-bit in low-contrast. Rather than operating on 8-bit image, we decide to normalize the image with narrow range when converting 16-bit image to 8-bit image. We use half of minimum intensity and $\frac{1}{128}$ of maximum intensity in 16-bit image to normalize to 8-bit image. As a result, it can enhance the contrast and preserve some information which may be lost if we use normal converting.

3.6. Acceleration

Although we can correctly obtain the reconstructed images, comparing with the throughput of Volume Graphics, ours is not good enough. Based on the program including many for-loops, we decide to make them parallel. We choose Threading Building Blocks by Intel.

TBB is a library that supports scalable parallel programming using standard ISO (International Standards Organization) C++ code [3]. TBB does not require special languages or compilers. TBB is designed to promote scalable data parallel programming. Additionally, TBB fully supports nested parallelism, so users can build larger parallel components from smaller parallel components. To use the library, users specify tasks, not threads, and let the library map tasks onto threads in an efficient manner. With TBB, we not only change the original for-loop to the parallel for-loop, but need to consider modifying the architecture of the program.

The comparison of execution time with two code sections is shown as follows:

Execution time (s) / FBP	with For-loop	with Parallel For-loop
Make Filter	0.12348	0.069422
Fourier Transform	2.55927	0.232338
Back Projection	15.844818	1.186021
Histogram Specification	5.349486	0.556932
Total	24.000535	2.114135

Table 3-1: The comparison of execution times (approximately 10 times acceleration)

Experimental Environment1
1. CPU: Intel® Core i5 2.7GHz processor
2. Memory: 8GB
3. OS: OS X El Capitan
4. Programming Language: C/C++, OpenCV 2.4.9

4. EXPERIMENTAL RESULT

In our experiment, there are two samples: Sample1 contains 128 projection images with resolution 1496×1496 pixels reconstructing object with height 600 pixels, and Sample2 contains 16 projection images with resolution 1124×1000 pixels reconstructing object with height 200 pixels. We use Sample1 to compare with SART and FBP, and use Sample2 to compare with FBP and the results by Volume Graphics.

4.1. Comparison between SART and FBP

Method	SART	FBP
Execution time (s)	291	210

Table 4-2: Our execution time of SART and FBP.

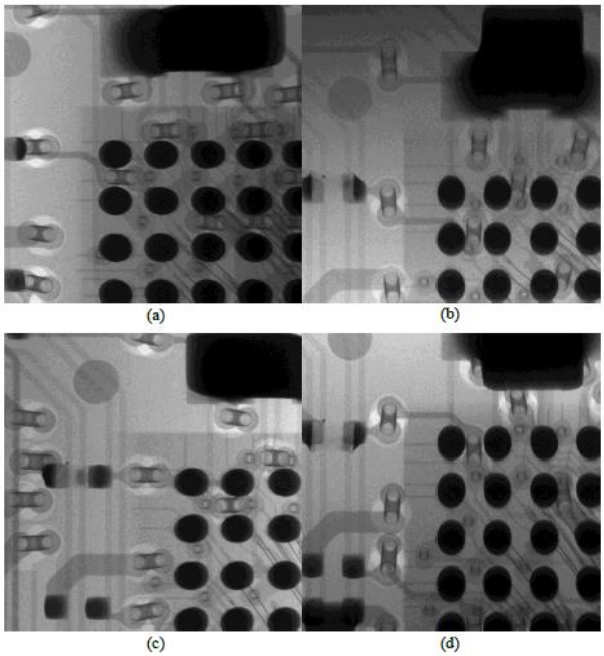

Figure 4-3: The projection images in Sample1 (totally 128 images). (a) The projection image with tilt angle **0**°. (b) The projection image with tilt angle **90**°. (c) The projection image with tilt angle **180**°. (a) The projection image with tilt angle **270**°.

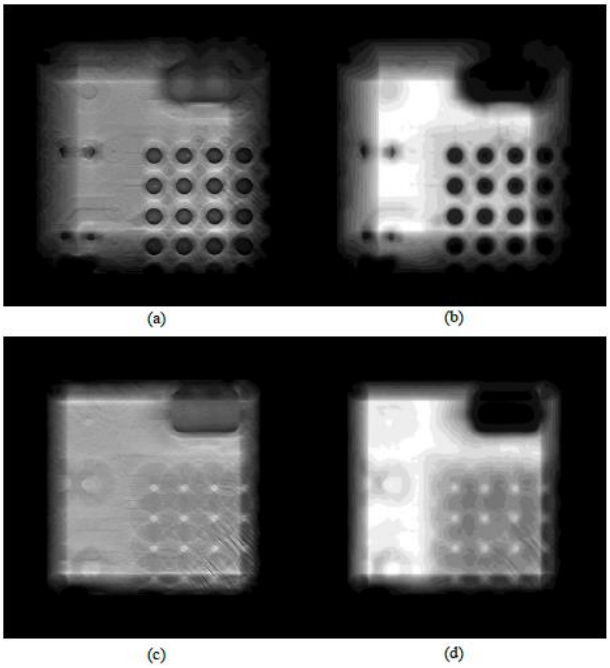

Figure 4-4: Our reconstructed images in Sample1 with SART and FBP. (a) The reconstructed image with height **100** pixels by FBP. (b) The reconstructed image with height **100** pixels by SART. (c) The reconstructed image with height **240** pixels by FBP. (d) The reconstructed image with height **240** pixels by SART.

4.2. Comparison FBP with Volume Graphics

Experimental Environment2	Our Environment	Environment of Test Research Incorporation
CPU	Intel® Xeon E5-2620 2.00GHz processor	Intel® E5-2687 2.53GHz processor
Memory	65GB	96GB
OS	Linux	Windows
Programming Language	C/C++, OpenCV 2.4.9	C/C++

	Our Result	Result from Volume Graphics
Execution time (s)	2.2	2.5

Table 4-3: Execution times of our result and Volume Graphics result.

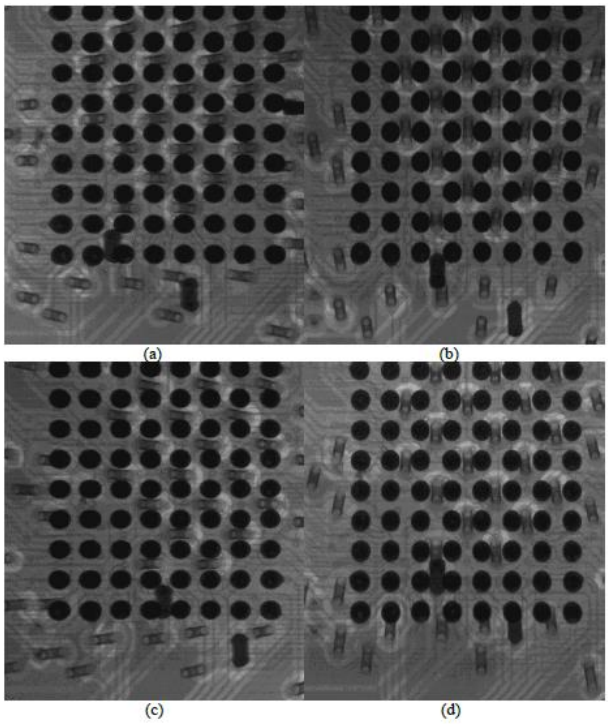

Figure 4-3: The projection images in Sample2 (totally 16 images). (a) The projection image with tilt angle **0**°. (b) The projection image with tilt angle **90**°. (c) The projection image with tilt angle **180**°. (a) The projection image with tilt angle **270**°.

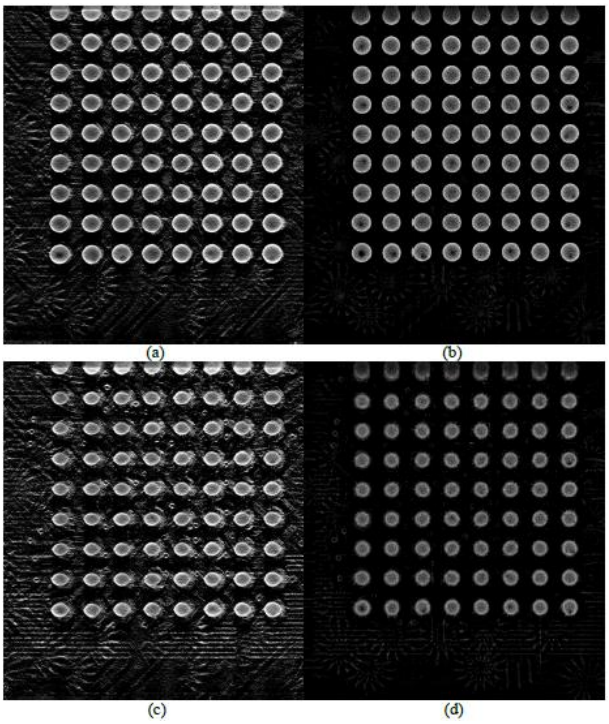

Figure 4-4: The reconstructed images in Sample2. (a) The FBP reconstructed image with height 105 pixels by our method. (b) The FBP reconstructed image with height 105 pixels by Volume Graphics. (c) The FBP reconstructed image with height 112 pixels by our method. (d) The FBP reconstructed image with height 112 pixels by Volume Graphics.

5. CONCLUSION AND FUTURE WORK

In our work, we implement Simultaneous Algebraic Reconstruction Technique (SART) and Filtered Back Projection (FBP) with modified filter to reconstruct three-dimensional object with two-dimensional projection images. We also apply Threading Building Blocks (Intel® TBB) by Intel to accelerate our program and achieve approximately 10 times acceleration.

Although our execution time is slightly faster than Volume Graphics, it can be further accelerated by computing with Graphics Processing Unit (GPU), such as using Computed Unified Device Architecture (CUDA). However, our FBP reconstructed images have similar quality to Volume Graphics. For the void in solder ball, the low-contrast problem still exists in our result. This is the issue to resolve in the future.

ACKNOWLEDGEMENT

This research was supported by the Ministry of Science and Technology of Taiwan, R.O.C., under Grants MOST 103-2221-E-002-188 and 104-2221-E-002-133-MY2, and by Test Research, Lumens Digital Optics, Delta Electronic, Egistec, D8AI, and Lite-on.

REFERENCES

[1]. R. Badawi, “Introduction to PET Physics,” http://depts.washington.edu/nucmed/IRL/pet_intro/toc.html, 2017.
[2]. L. Han, “Tools for 2-D Tomographic Reconstruction,” http://www.mathworks.com/matlabcentral/fileexchange/43008-tools-for-2-d-tomographic-reconstruction?s_tid=srchtitle, 2017.
[3]. Intel, “Introducing the Intel® Threading Building Blocks (Intel® TBB),” https://software.intel.com/en-us/node/506042#introducing_main, 2017.
[4]. H. K. Lin, “Algebraic Reconstruction Technique,” Technical Report, Department of Computer Science and Information Engineering, National Taiwan University, 2016.
[5]. Y. Y. Lin, “The Study of Using Simultaneous Algebraic Reconstruction Technique on BGA Inspection,” http://140.113.39.130/cgi-bin/gs32/hugsweb.cgi?o=dnthucdr&s=id=%22GH009533591%22.&searchmode=basic, 2017.
[6] D. R. Nadeau, “C/C++ Tip: How to Loop through Multi-Dimensional Arrays Quickly,” http://nadeausoftware.com/articles/2012/06/c_c_tip_how_loop_through_multi_dimensional_arrays_quickly#BenchmarkresultsndashcompiledwithoptimizationsDisabled, 2017.
[7]. C. Pan, “Image Projections and the Radon Transform,” https://www.clear.rice.edu/elec431/projects96/DSP/bpanalysis.html, 2017.
[8]. P. A. Penczek, “Fundamentals of three-dimensional reconstruction from projections,” https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3165033/, 2017.
[9]. N. Rezvani, “Iterative Reconstruction Algorithms for Polyenergetic X-Ray Computerized Tomography,” ftp://www.cs.toronto.edu/na/reports/Nargol.Rezvani.PhD.Thesis.pdf, 2017.
[10]. S.W. Smith, “The Scientist & Engineer's Guide to Digital Signal Processing,” http://www.dspguide.com/ch25/5.htm, 2017.
[11]. X. Wan, F. Zhang, and Z. Liu, “Modified Simultaneous Algebraic Reconstruction Technique and Its Parallelization in Cryo-electron Tomography,” http://ieeexplore.ieee.org/document/5395300/, 2017.
[12]. R. Wang, “Image Enhancement by Contrast Transform,” http://fourier.eng.hmc.edu/e161/lectures/contrast_transform/contrast_transform.html, 2017.

[13]. Wikipedia, "Algebraic Reconstruction Technique," https://en.wikipedia.org/wiki/Algebraic_Reconstruction_Technique, 2017.
[14]. Wikipedia, "Projection-Slice Theorem," https://en.wikipedia.org/wiki/Projection-slice_theorem, 2017.
[15]. Wikipedia, "Radon Transform," https://en.wikipedia.org/wiki/Radon_transform, 2017.
[16]. Wikipedia, "Simultaneous Algebraic Reconstruction Technique," https://en.wikipedia.org/wiki/Simultaneous_Algebraic_Reconstruction_Technique, 2017.
[17]. Wikipedia, "Window Function," https://en.wikipedia.org/wiki/Window_function#Hamming_window, 2017.