LA-UR-00-4761

Approved for public release;
distribution is unlimited.

Title: **Advanced Data Processing for Volumetric Computed Tomography Data**

Author(s): Matthew J. Sheats

Submitted to:

http://lib-www.lanl.gov/la-pubs/00393765.pdf

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

FORM 836 (10/96)

ADVANCED DATA PROCESSING FOR VOLUMETRIC COMPUTED TOMOGRAPHY DATA

by

Matthew J. Sheats

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

University of California Davis

December 1st, 2000

Approved by _____
Chairperson of Supervisory Committee

_____

_____

_____

Program Authorized
to Offer Degree_____

Date _____

ACKNOWLEDGMENTS

This document summarizes a project that has been four years in the making. During that time I have attended two schools, lived in three states, and all the while, been a Los Alamos National Laboratory employee. My personal success and the success of FlashCT are thanks to a multitude of people whom I have had the honor to work with. Their support shaped my future and I am forever grateful.

From the University of California Davis, I must first thank Dr. Ken Joy. His invitation to apply to UC Davis started me down the path to graduate school, and his teaching made me believe that maybe I could handle "this graphics stuff". Thanks also for office hours at the pub and the retreat to the cabin. I want to thank Dr. Bernd Hamann and Dr. Kwan-Liu Ma for ongoing visualization advice and guidance. I especially appreciate allowing me to shape my projects to fit my work at Los Alamos National Laboratory. A very special thank you to Kim Reinking for unraveling the knots that kept me from starting at Davis originally and for tying up the loose ends to help me graduate. Finally, a thank you to Glen Kramer, classmate and friend for providing constant sanity checks on long algorithm homeworks, hours of prelim study, and for watching my place during my long absences.

My experience at LANL would not have begun had it not been for Dr. David Stupin. As a third year computer science major, he brought me on to a key project and gave me room to develop it as best I could. Dave has not only been an excellent mentor, but a good friend, offering excellent advice and even putting his life on hold to help me move to California. I cannot thank you enough. An equally large thank you goes to Tony

Davis. Tony has been the key designer and promoter of the FlashCT system. He designed the Data Acquisition System, which in turn shaped the direction of the Data Processing System. His efforts always inspired me to do a little bit more. Even though we've had our rough spots, Tony has become an excellent friend. Like Dave, Tony dropped everything for five days to embark on a marathon trip to move my things back from California.

Of course, I need to thank the entire Non Destructive Testing and Evaluation team at LANL. I thank Dr. Martin Jones for continued support and for forging the waves so I could continue my research in Davis. He is always an excellent sounding board for new ideas and great source of encouragement. Thank you to Dr. Thomas Claytor for heading the FlashCT project and being a constant source of news, funding, and humor.
Dr. Deborah Summa guided me through my first stumbling attempts with IDL. Her input provided the first steps towards our current reconstruction algorithms when I didn't know where to begin. I thank Charles Hills for simulation support and for being the readily available user of FlashCT who wasn't one of the two authors. Finally I thank the remaining staff of ESA-MT for both administrative and technical support during the creation of the software.

Thank you to Hytec Inc. of Los Alamos who was our industrial partner during the development of FlashCT. Their input made the design of the system easier and their support during its production made the work worthwhile. It is gratifying to see a large project such as this finally placed in the hands of a user.

No acknowledgment section would be complete without thanking one's parents. Throughout my entire life they have encouraged me to do my best, and to use education to

get ahead in life. Even so, I think it shocked them just a little when I decided to go on to graduate school. I thank them for emotional support, airfare during all the major holidays, and the countless number of times they offered me money when they were convinced I was broke and starving.

This report (LAUR-004761) was prepared as an account of work sponsored by an agency of the United States Government. Neither The Regents of the University of California, the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by The Regents of the University of California, the United States Government, or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of The Regents of the University of California, the United States Government, or any agency thereof. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

TABLE OF CONTENTS

ACKNOWLEDGMENTS ii

ABSTRACT vii

LIST OF FIGURES viii

LIST OF ABREVIATIONS xi

CHAPTER

I. INTRODUCTION TO FULL-VOLUME INDUSTRIAL COMPUTED TOMOGRAPHY 1
- 1.1 Traditional CT Systems and Uses 1
- 1.2 Advances in CT Data Acquisition 5
- 1.3 Data Processing in the Past and Present 8
- 1.4 Overview: Acquisition to Reconstruction 9
  - 1.4.1 Image Collection 9
  - 1.4.2 Image Correction 10
  - 1.4.3 Sinogram Conversion 11
  - 1.4.4 Sinogram Correction 12
  - 1.4.5 Reconstruction 13
- 1.5 Reconstruction Algorithms in Detail 14
  - 1.5.1 Parallel Beam Reconstruction 14
  - 1.5.2 Fan Beam Reconstruction 17
  - 1.5.3 Cone Beam Reconstruction 18
- 1.6 FlashCT: A new idea in industrial computed tomography 21
  - 1.6.1 Data Acquisition System 21
  - 1.6.2 Data Processing System 22

II. DESIGN OF THE DATA PROCESSING SYSTEM 24
- 2.1 Hardware, Operating System, and Programming Language Decisions 24
- 2.2 The Overall Design of The Data Processing System 25
  - 2.2.1 The Design of the IDL Classes 26
  - 2.2.2 The Design of the Visual Basic Classes 27
  - 2.2.3 The Complete Package 28
- 2.3 A Tour of the Data Processing System 28
  - 2.3.1 The Main Menu 28
  - 2.3.2 The Image Viewer 29
  - 2.3.3 The Sinogram Converter 30
  - 2.3.4 The Centering Tool 31
  - 2.3.5 The Reconstruction Tool 32
  - 2.3.6 The Visualization Dialog 33
  - 2.3.7 File Management 34
- 2.4 Additional Design Issues for the Data Processing System 35

III. EXPLOITING THE MASSIVE PARALLELISM OF FULL VOLUME CT RECONSTRUCTION 36
- 3.1 The Batch-Queue System 36

3.2 Communication and Job Determination 39
3.3 The FlashCT – DPS Multi-Processor Test Bed 41
3.4 The FlashCT Cluster Manager 44
IV. DATA PROCESSING SYSTEM RESULTS 47
4.1 Reconstruction Algorithm Results 47
4.1.1 Parallel Beam Reconstructions 48
4.1.2 Fan Beam Reconstructions 49
4.1.3 Cone Beam Reconstructions 50
4.2 Data Sets of Interest 50
4.2.1 Rock Core 50
4.2.2 Coral 51
4.2.3 Mag Light 52
4.2.4 Microsoft Mouse 53
V. CONCLUSIONS 55
5.1 Algorithm Effectiveness 55
5.2 System Speed 55
5.3 System Cost 56
5.4 Overall Functionality 56
5.5 Future Improvements 56
REFERENCES 57

University of California Davis

Abstract

ADVANCED DATA PROCESSING FOR VOLUMETRIC COMPUTED TOMOGRAPHY DATA

by Matthew J. Sheats

Computed tomography (CT) has long been used by industry as a non-destructive evaluation technique for discovering flaws in parts before their use. Traditional linear array computed tomography systems acquired data at a rate that could be reconstructed simultaneously with data acquisition. With new area detectors, the data rate of acquisition has increased exponentially, and the reconstruction algorithm does not scale linearly with increased data. FlashCT – DPS (Flat Panel Amorphous Silicon High Resolution Computed Tomography, Data Processing System) is a software package developed by M. Sheats, and A. Davis at LANL to meet the data processing demands of new flat panel array computed tomography. FlashCT – DPS combines several features unique to industrial computed tomography systems. It addresses traditional usability problems by providing an intuitive graphical user interface and powerful analysis tools. It includes the three major CT reconstruction algorithms: parallel backprojection, fan beam resorting, and Feldkamp cone beam. It also provides visualization tools for examination of data after processing. Finally, it is able to perform distributed data reconstruction with a near linear speed increase as a function of the number of processors used. Additionally, utility programs have been developed to support project editing and computing cluster management for the FlashCT system. FlashCT – DPS runs on a standard PC platform and operates well on a variety of processor and memory configurations.

LIST OF FIGURES

Number *Page*

Figure 1: A typical radiography setup. 2

Figure 2: A radiograph of the sample in Figure 3. It is not possible to determine which pin is bent. 2

Figure 3: A radiography / CT sample object. Note the bent pin towards the middle. 2

Figure 4: A traditional linear detector array CT system. 3

Figure 5: A sample reconstruction from a linear detector array CT scan. 4

Figure 6: Conversion from 2D reconstructions to a 3D volume. 4

Figure 7: An area detector CT configuration using a CCD camera. 6

Figure 8: A flat panel detector for use in a CT system. 7

Figure 9: A sample CT configuration using a flat panel detector. 7

Figure 10: The five steps from data acquisition through data processing. 9

Figure 11: The data acquisition setup. 10

Figure 12: From uncorrected to corrected image. 11

Figure 13: Extracting sinograms from a block of corrected images. 12

Figure 14: Correcting sinograms by adjusting their center of rotation. 12

Figure 15: From corrected sinogram to reconstructed slice. 13

Figure 16: Combining all reconstructed slices into a "volume visualization". 13

Figure 17: Side view of a parallel CT geometry. 14

Figure 18: Performing filtered backprojection: from a sinogram row to a partially reconstructed slice. 15

Figure 19: A sinogram row is backprojected into a reconstruction slice. 16

Figure 20: Backprojection as it progresses over all angles of a sinogram. 16

Figure 21: Finding the needed sinograms for cone beam reconstruction using the extrema of a plane. 19

Figure 22: A ray is cast through a pixel to be reconstructed, with its value being interpolated from the sinogram subset. 20

Figure 23: Data acquisition hardware. Back: flat panel detector; middle: three-axis staging; foreground: microfocus x-ray source. 21

Figure 24: Data acquisition system software interface. 22

Figure 25: Active-X relationship between Visual Basic and IDL. 26
Figure 26: Main IDL classes. The arrows indicate a Contains-A relationship. 27
Figure 27: Relationship between Visual Basic classes and IDL classes. 27
Figure 28: Main menu of FlashCT - DPS. 29
Figure 29: Image viewer from FlashCT - DPS. A radiograph of a Microsoft Mouse is displayed. 30
Figure 30: FlashCT - DPS sinogram processor. 31
Figure 31: FlashCT - DPS centering tools displaying a sinogram from the Microsoft Mouse. 32
Figure 32: FlashCT - DPS reconstruction tool. 33
Figure 33: FlashCT - DPS Visualization dialog. The memory required for the Microsoft Mouse data set causes a math overflow. 34
Figure 34: FlashCT - DPS file manager. 35
Figure 35: FlashCT - DPS Batch-Queue system. 37
Figure 36: Selecting the "Include In Batch" checkbox includes the settings in the batch-queue process. 38
Figure 37: FlashCT - DPS status screen shown during data processing steps. 38
Figure 38: Queue where several projects can be issued for batch processing. 39
Figure 39: Drop directory is specified in the batch processing dialog and monitored on by the distributed reconstructor monitor. 40
Figure 40: Each drop file is intended for a specific machine with the same processor identification. 40
Figure 41: FlashCT Platinum Cluster. The first dedicated DPS reconstruction cluster. 42
Figure 42: Second DPS cluster, built from commercially available Cubix computing units. 43
Figure 43: FlashCT Cluster Manger sitting idle, monitoring the drop directory for a new job. 44
Figure 44: Cluster manager showing the progoress of a job. The machine chart illustrates the progress of each processor. 45
Figure 45: Cluster shutdown screen allows the user to see which machines are currently running and shutdown or reboot them as needed. 46

Figure 46: CAD representation of a test piece simulated using MCNPX for diagnostics on the reconstruction algorithms. 47
Figure 47: Simulated cone beam radiograph from the test piece. 48
Figure 48: Slice from the test piece (left) and a parallel beam reconstruction created from the simulated radiographs (right). 49
Figure 49: Slice from the test piece (left), a parallel beam reconstruction of a fan beam slice (center), and an accurate fan beam reconstruction (right). 49
Figure 50: Slice from the test piece (left), A fan beam reconstruction of a cone beam slice (center), and an accurate cone beam reconstruction (right). 50
Figure 51: Concrete core data set reconstructd using FlashCT - DPS's parallel reconstrution algorithm. Photograph (left), a single slice (cneter), a volume (right) are displayed.51
Figure 52: Coral data set reconstructed using FlashCT - DPS's fan beam reconstruction algorithm. A photograph (right), single reconstructed slice (top), and volume rendering (right) are displayed. 52
Figure 53: Mag light data set. A picture and radiograph (left), and a slice from the reconstruction with an isosurface rendering (right). 53
Figure 54: Microsoft mouse data set. Reconstructed by FlashCT – DPS's cone beam reconstruction algorithm. A picture (top left), a reconstructed slice (center), and a volume rendering (right) are displayed. 54

LIST OF ABBREVIATIONS

A-Si	Amorphous silicon.
CCD	Charged coupled device.
CT	Computed Tomography.
DAS	Data acquisition system.
DPS	Data processing system.
FlashCT	Flat Panel Amorphous Silicon High-resolution Computed Tomography.
IDL	Interactive Data Language.
GB	Gigabyte.
GUI	Graphical User Interface.
LANL	Los Alamos National Laboratory.
LANCSE	Los Alamos Neutron Science Center.
MB	Megabyte.
NDT	Non Destructive Testing.
RSI	Research Systems Inc.
VB	Visual Basic (Microsoft).

CHAPTER I
INTRODUCTION TO FULL-VOLUME INDUSTRIAL COMPUTED TOMOGRAPHY

Non-destructive testing (NDT) is the field concerned with the examination of objects for defects without inflicting damage. Radiography is a popular and long-standing NDT technique, providing a two-dimensional projection of the internal structure of an object without harm. But, while radiography is capable of revealing many defects, it is incapable of providing detailed structural information. Computed tomography (CT) was developed as an extension to radiography. By using many radiographs, CT is able to reconstruct the internal structure of an object. Linear array CT systems have been employed for many years to examine single planar "slices" through the object of interest. Modern NDT standards demand the analysis of structural data in three-dimensions, creating a need for a 3D full-volume CT system.

1.1 Traditional CT Systems and Uses

Radiography has long been a standard defect detection technique. It typically employs a neutron or x-ray source and a piece of film as show in Figure 1. Radiation passes through the object, is attenuated, and exposes film creating a projection of the internal structure. The resulting projection frequently provides sufficient information to determine if certain defects exist, but does not provide detailed volumetric structural information. The sample radiograph in Figure 2 shows that it is possible to determine that the central pins are present, but does not show which pin is bent as in Figure 3. In order to determine if all pieces are in correct position or state, structure information is critical.

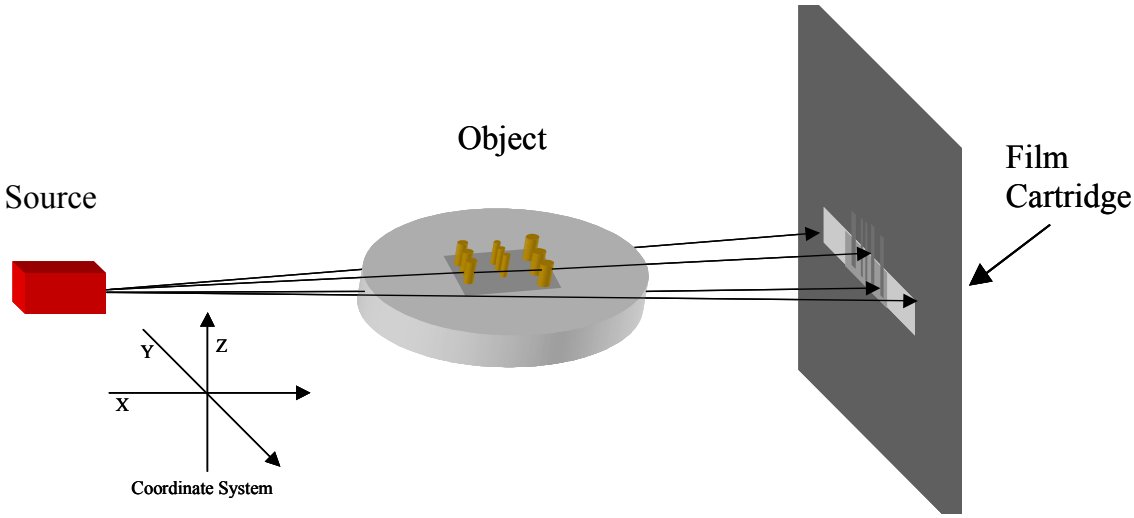

Figure 1: A typical radiography setup.

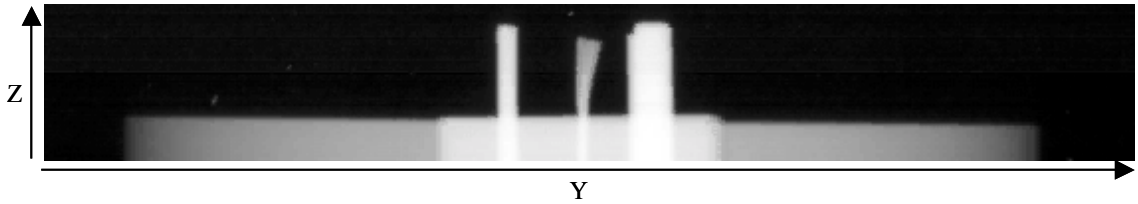

Figure 2: A radiograph of the sample in Figure 3. It is not possible to determine which pin is bent.

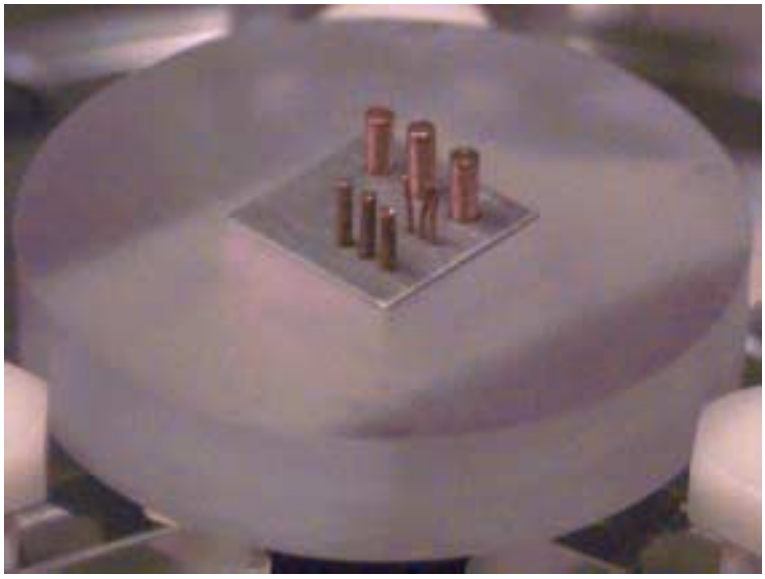

Figure 3: A radiography / CT sample object[1]. Note the bent pin towards the middle.

Detailed internal volumetric structural information is provided by CT. The traditional method employs a source and a linear array of detector elements that can measure the radiation transmitted through the object (Figure 4). Rather than a full two-dimensional projection of the object, the detector records a one-dimensional projection as if a single row were captured from the radiograph. The object is rotated over 360 degrees with one-dimensional projections being captured at regular intervals. Using the projections, a reconstruction of the object's structure can be created. The reconstruction (Figure 5), like the radiograph, provides an image representing the object, but instead of a ZY-planar projection of the entire object, the reconstruction depicts the XY-plane structural elements of the object on the two-dimensional slice examined. The object can be examined fully in three-dimensions by translating the object vertically and repeating the process to collect further XY-planes. The result is a stack of two-dimensional reconstructed slices that can be combined into a three-dimensional volume. Figure 6 shows a completed volume where the stack of XY-planes makes it possible to resolve the bent pin shown in Figure 3.

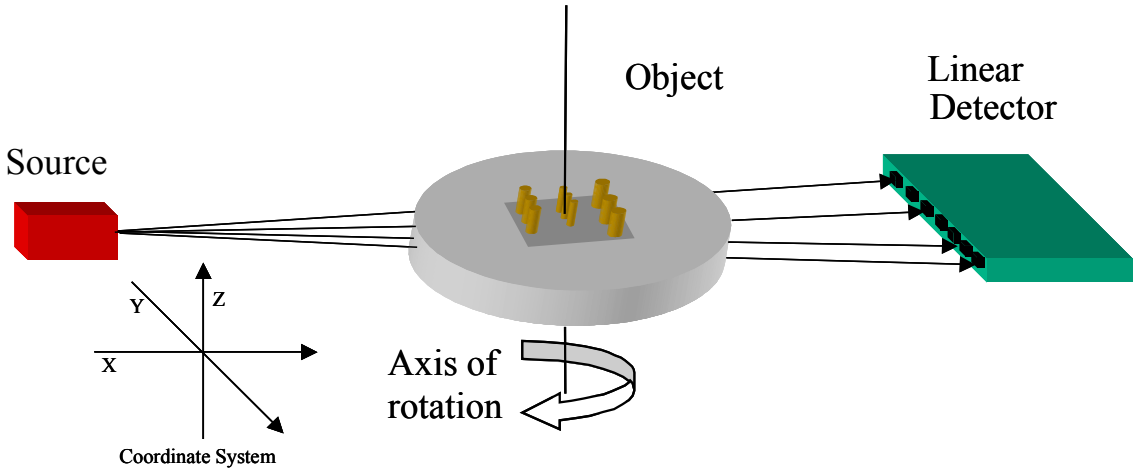

Figure 4: A traditional linear detector array CT system.

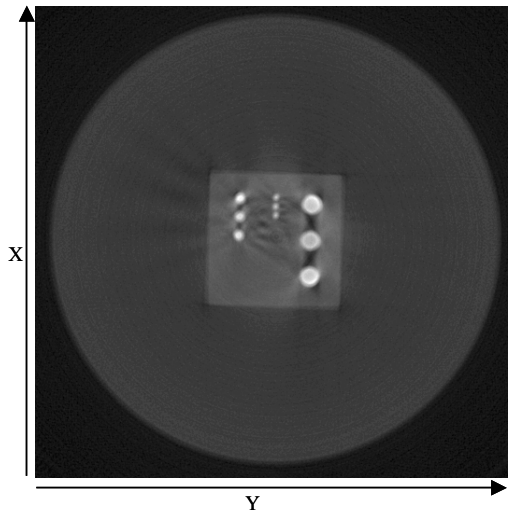

Figure 5: A sample reconstruction from a linear detector array CT scan.

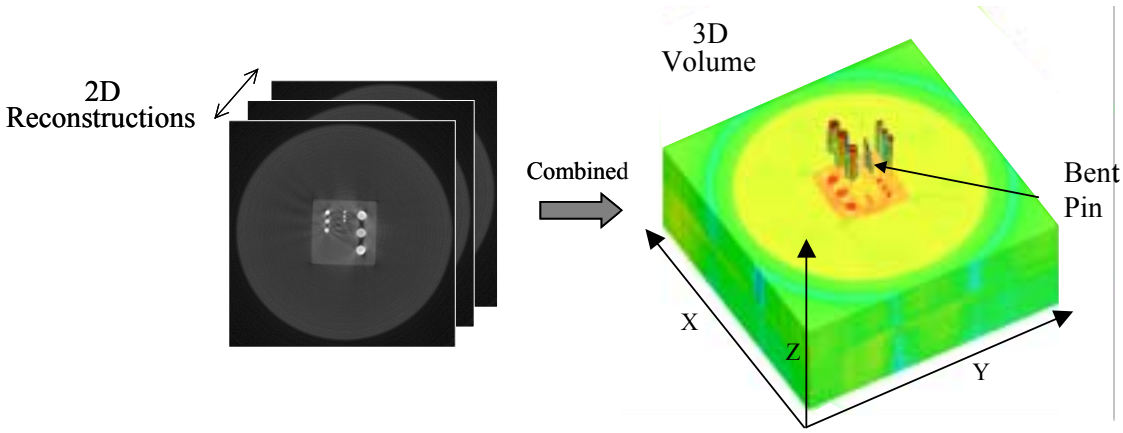

Figure 6: Conversion from 2D reconstructions to a 3D volume.

CT employing linear arrays is a solved problem. Given sufficient time, the full structural composition of an object can be examined in three dimensions. Unfortunately, for a large object, "sufficient time" could mean weeks, days, or years. Linear array systems are slow, requiring many rotations and translations of the object to complete a data set. Until recently, performing CT quickly remained a difficult issue to address.

1.2 Advances in CT Data Acquisition

An obvious solution to the limited data acquisition speed of linear arrays is to collect data in two-dimensions. Essentially, stacking many linear arrays on top of each other will remove the requirement to continuously translate the part. Sufficient array elements would allow data for an entire part to be collected in the same amount of time as a single slice took previously. Current linear arrays are not easily capable of such a configuration, but new technology is allowing the technique to be performed.

Two-dimensional data acquisition is done using area detectors, equipment capable of acquiring CT data in two-dimensions. The first available equipment of this type was the charged coupled device (CCD) camera. Modern CCD cameras have resolutions of several million pixels (1024 x 1024) or better: the equivalent to 1024 stacked linear array detectors. CCD cameras detect visible light, so the x-ray or neutron radiation must be converted. A sheet of material called a scintilator is placed behind the object in similar fashion to a piece of film in radiography. The scintilator illuminates in the presence of radiation converting the attenuated radiation into visible light proportional to the incident radiation. Radiation passing through the entire object creates a projection image on the scintilator similar to that provided by traditional film radiography. The CCD camera acquires the visible light image and the data is stored to a computer. Unfortunately, CCD camera electronics are vulnerable to radiation and cannot sit in its direct path. As a result, CCD cameras require additional optics and shielding that tend to make them difficult to use. Figure 7 illustrates the required equipment including a turning mirror and radiation shielding. CCD cameras require long periods of time to capture each image and are susceptible to electrical noise.

However, despite their problems, they provided the first step into three-dimensional CT data acquisition and continue to be used in many systems.

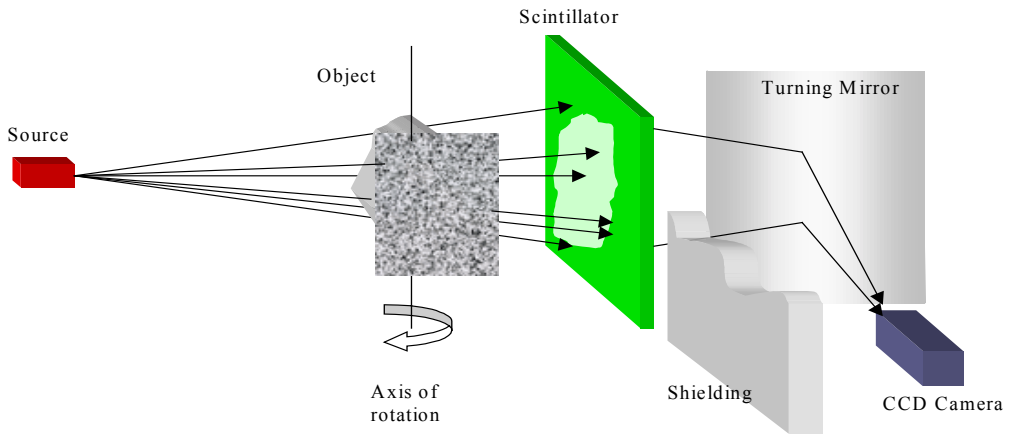

Figure 7: An area detector CT configuration using a CCD camera[2].

Recently, advances in flat panel display technology have lead to a new type of area detector. Amorphous silicon (A-Si) detectors (Figure 8) are technologically similar to standard desktop LCD displays, but instead of projecting light, they detect it. Flat panels also rely on a scintilator to convert radiation into light but do not suffer from many of the problems associated with CCD cameras. The amorphous silicon photodiode array is inherently rad-hard, and can be placed directly in the path of the radiation source (Figure 9). Due to their efficient detection of the visible light from the scintilator, image acquisition time is much faster, and the data is less prone to electrical noise. While flat panel resolutions lag behind the high end CCD cameras, their ease of use, speed, and quality of data make them an excellent option for volumetric CT.

Figure 8: A flat panel detector for use in a CT system[3].

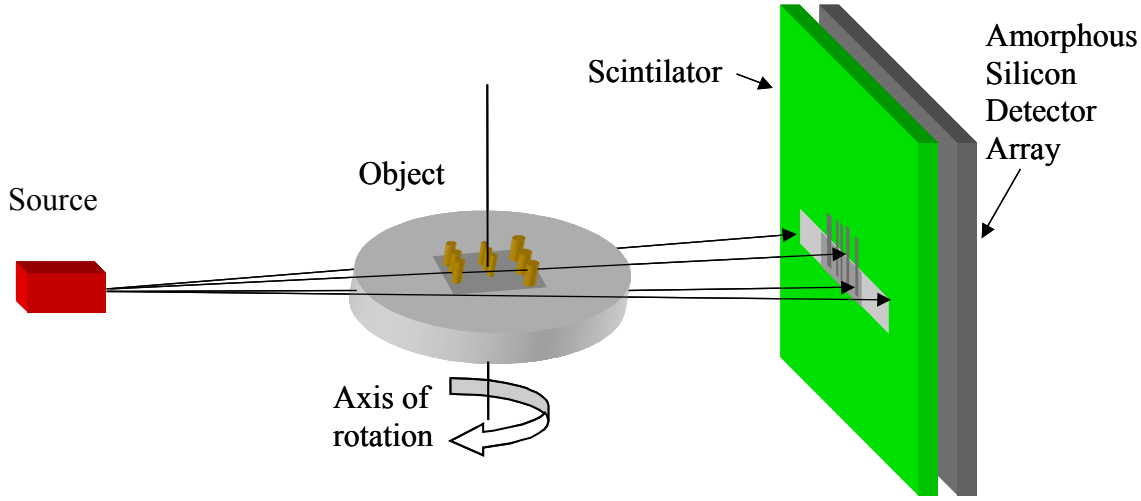

Figure 9: A sample CT configuration using a flat panel detector.

The problem that has arisen with area detectors is how to handle the vast quantities of data generated. Millions of pixels per image with 12 to 16 bits of data each can be

collected in just a few minutes, however the reconstruction of the data remains a computationally complex problem. Advances in data acquisition require new advances in data processing.

1.3 Data Processing in the Past and Present

Linear array systems acquire data at a rate that allows hardware to reconstruct simultaneously with acquisition. Even though two-dimensional reconstruction is an $O(n^3)$ algorithm where n is the size of the data set, acquisition was slow enough and the problem was sized such that processors could perform necessary reconstruction steps before the next piece of data was acquired. Linear array systems are cohesive and bound tightly to acquisition time instead of reconstruction time.

Area detectors create a new problem. Using an area detector, many orders of magnitude more data is available in the same time as it took for a linear array to provide data for a few slices. The processing hardware has kept pace with the new rate of acquisition, however it has not kept pace with the new rate of reconstruction. Reconstruction time quickly becomes the limiting factor in performing three-dimensional CT scans.

Vastly longer reconstruction time demands the separation of acquisition from reconstruction. Acquisition requires the use of expensive equipment and facilities plus constant monitoring by operators. Once the data is available, however, reconstruction can be done unattended by any computer, even if it has an exceedingly long run time. If a boundary is formed between acquisition and reconstruction, the expensive part can be done quickly leaving the processing to be done on inexpensive hardware unattended. The result

is fast and less expensive data acquisition with reconstruction performed away from costly resources.

1.4 Overview: Acquisition to Reconstruction

The CT process consists of five steps: image collection, image correction, sinogram conversion, sinogram correction, and reconstruction. As shown in Figure 10, the process is broken into the two sections of acquisition and processing. Image collection and correction are performed during data acquisition while the remaining three steps are performed during data processing.

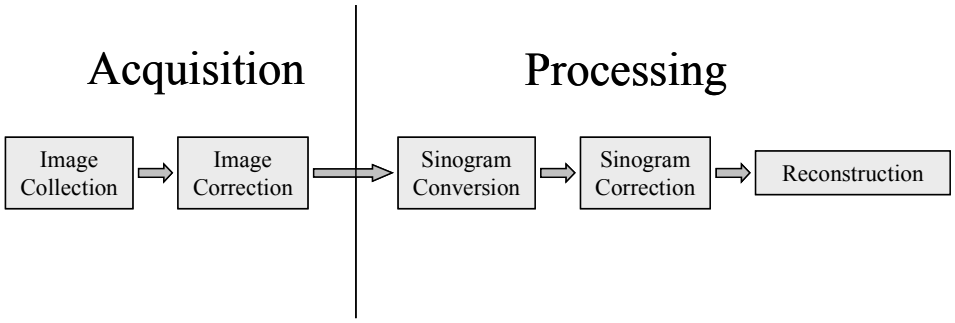

Figure 10: The five steps from data acquisition through data processing.

1.4.1 Image Collection

Image collection is the first step to a CT reconstruction. Figure 11 depicts the image collection system. An object is placed on the rotary stage and the radiation source is activated. The computer collects each image by rotating the stage incrementally and acquiring data from the detector. Each image is corrected before the process proceeds.

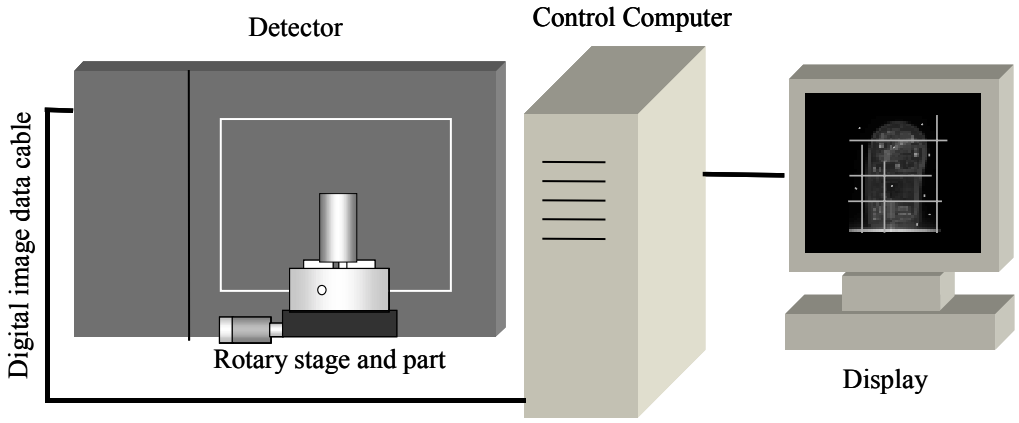

Figure 11: The data acquisition setup[2].

1.4.2 Image Correction

Due to their manufacturing process, most flat panels contain inherent defects. Bad columns, rows, and pixels may exist on the silicon of even a new detector[3]. Defects cause serious problems for CT reconstruction algorithms, so they must be eliminated. Image correction uses maps of bad pixels in combination with interpolation and smoothing routines to eliminate "bad data".

Additionally, the individual photodiodes of the detector have unique gains and offsets, which can cause the images to be uneven. A deterministic technique involving detector calibration is employed to remove these effects. The difference between a corrected and uncorrected image is dramatic, as displayed in Figure 12. The final converted images are stored to disk, ready for use by the data processing side of the CT system.

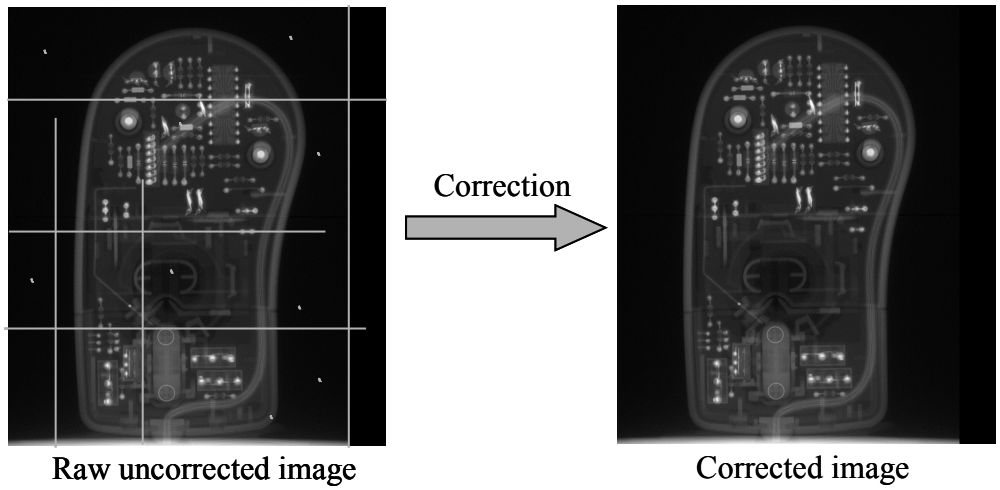

Figure 12: From uncorrected to corrected image.

1.4.3 Sinogram Conversion

The next step of the process requires a conversion from corrected images to a different image type called a sinogram. A sinogram can be described as the image created by a row on the detector as the part rotates. The top row of the sinogram is a row from the first image taken and the bottom row of the sinogram is the same row from the last image taken (Figure 13). The resulting image is made up of multiple sine waves of different amplitude. The waves represent features in the object that are being rotated over 360 degrees. Sinograms are acquired by extracting slices out of a volume of images. Given the proper geometry, a single sinogram is all that is necessary to perform a reconstruction on that row of data. It is sinograms that are ultimately used by the reconstruction algorithm.

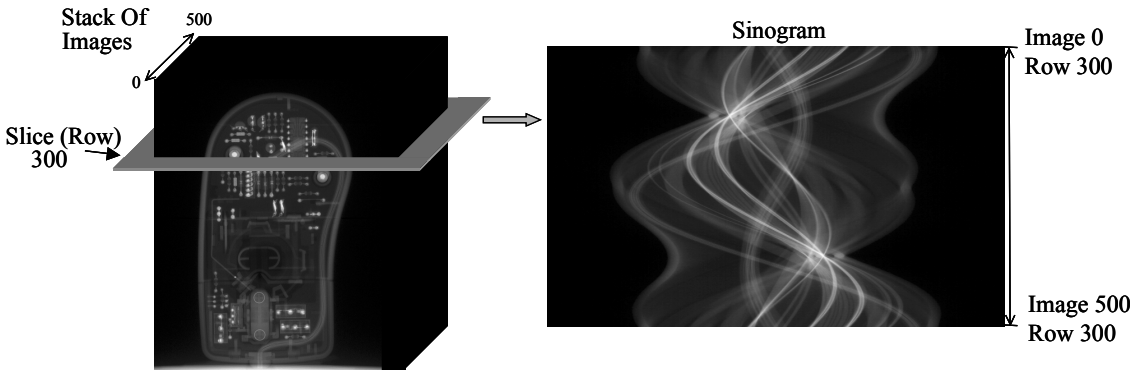

Figure 13: Extracting sinograms from a block of corrected images.

1.4.4 Sinogram Correction

Unfortunately, even though the images have been corrected for detector variations, there may remain an additional problem. If the center of rotation of the CT scan is not coincident with the center of the image, the resulting reconstruction will be deformed. The rotational stage could have been either off center or skewed with respect to the detector plane. Sinogram correction removes centering errors by detecting the skew in the sinogram and shifting it appropriately (Figure 14). Once corrected, the sinogram is ready to be processed by the appropriate reconstruction algorithm.

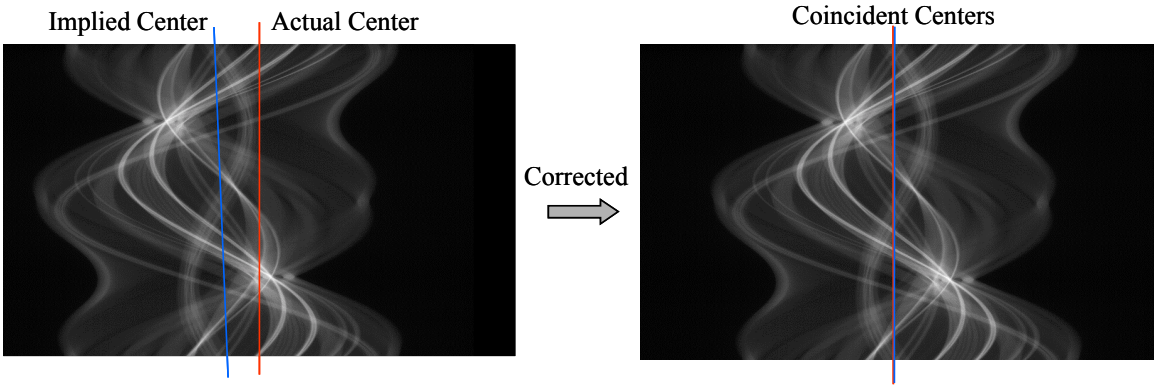

Figure 14: Correcting sinograms by adjusting their center of rotation.

1.4.5 Reconstruction

The final step in the process is reconstruction. At a high level, CT reconstruction consists of passing a corrected sinogram to an appropriate CT reconstruction algorithm. The algorithm applies a required reconstruction filter to the sinogram and then converts it into a complete two-dimensional CT slice (Figure 15). Once all slices have been reconstructed, they can be recombined into a single volume for visualization (Figure 16). CT reconstruction is an algorithmically complex operation, and is fully addressed in section 1.5.

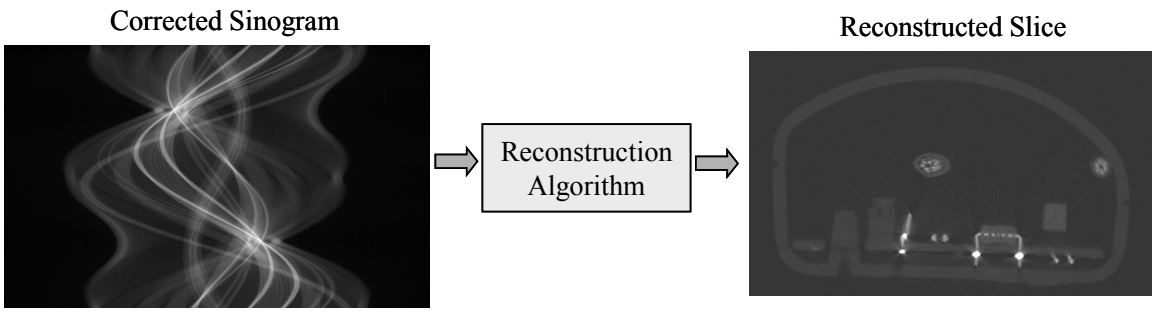

Figure 15: From corrected sinogram to reconstructed slice.

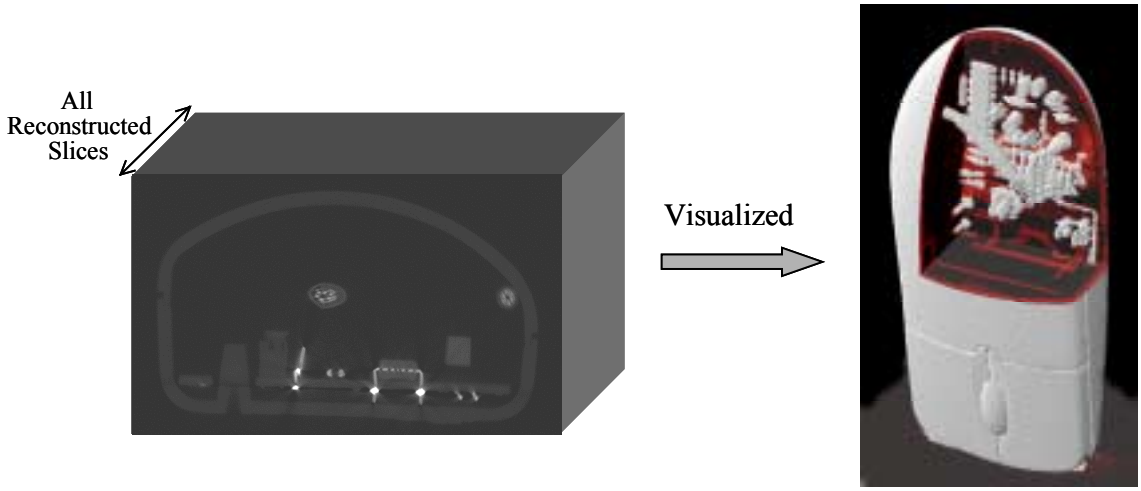

Figure 16: Combining all reconstructed slices into a “volume visualization”.

1.5 Reconstruction Algorithms in Detail

There are three basic types of reconstruction algorithms tied to the three typical source geometries used. Depending on the source and detector configuration, a CT scan can take on the following geometries: parallel beam, fan beam, or cone beam. The geometry describes relative shapes of the source and the detector. In order to generate a correct reconstruction, the geometry must be accurately recreated and applied in the reconstruction process. The theory of CT reconstruction is beyond the scope of this paper, instead, a higher-level algorithmic approach to solving the reconstruction problem is addressed.

1.5.1 Parallel Beam Reconstruction

Parallel beam is the simplest geometry, where radiation passing through an object travels parallel to the XY-plane. All data needed to create a single slice reconstruction is available in a single sinogram. A parallel beam is created either by a two-dimensional grid of point sources, or by a single point source an infinite distance from the detector. In reality a true parallel beam is impractical to create, but can be approximated by placing the source far away with respect to the size of the part and the resolution of the detector, as in Figure 17.

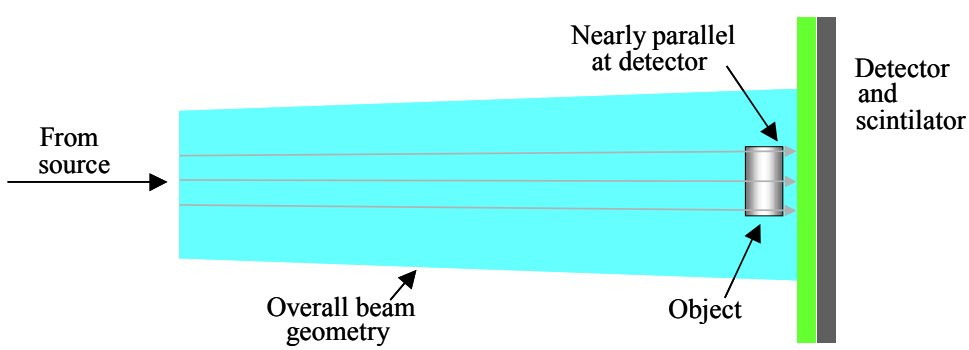

Figure 17: Side view of a parallel CT geometry.

Parallel beam reconstructions are performed using filtered backprojection[4]. Filtered backprojection derives from inverting the Radon Transform through the Fourier slice theorem. As radiation passes through the object, it is attenuated on its path to the detector. Filtered backprojection uses the stored attenuation values to reproduce the structure of the object by projecting the attenuation data back through the slice to be reconstructed. Figure 18 illustrates that a single row from a sinogram is extracted and summed across the output array where the reconstruction is being created. The summing is done similarly to a ray casting method used in volume visualization. A ray is projected from a point on the source to the matching point on the detector. The point on the detector is mapped to a data value in the filtered sinogram. As the ray progresses from the source to the detector, it encounters the reconstruction array rotated to the same angle the row of the sinogram represents. Each pixel in the reconstruction array the ray encounters is shaded with the value from the filtered sinogram, but interpolated to match its contribution to the pixel in the reconstruction. Figure 19 shows that a pixel's value is highly determined by how far the ray must cross through the pixel's space.

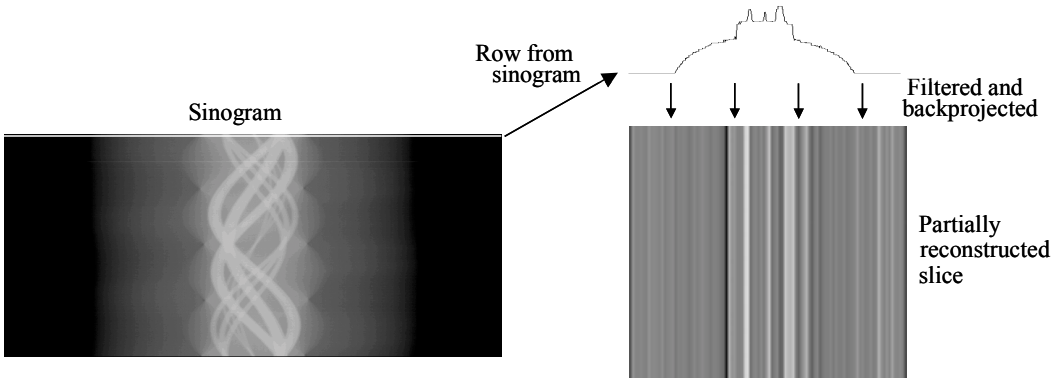

Figure 18: Performing filtered backprojection: from a sinogram row to a partially reconstructed slice.

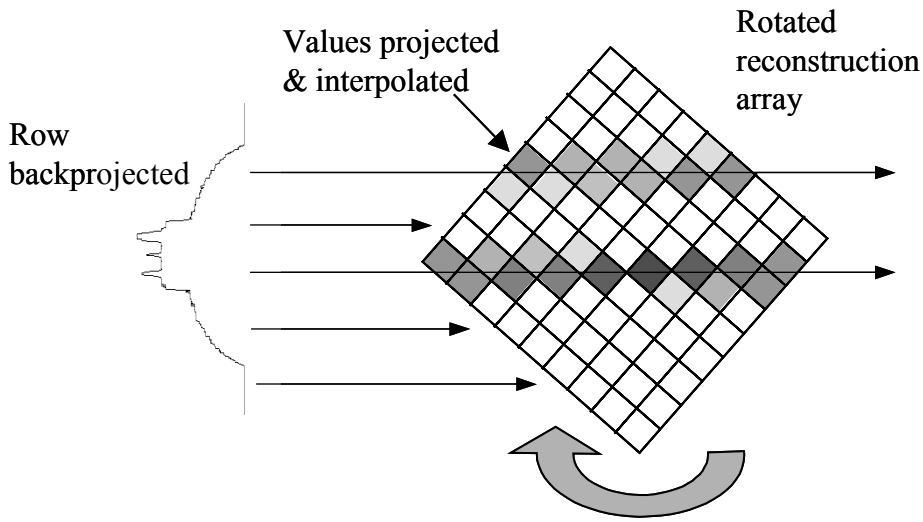

Figure 19: A sinogram row is backprojected into a reconstruction slice.

The process is repeated for all rows of the sinogram. Since each row in the sinogram represents a rotation of the original object, the orientation of the backprojection is rotated accordingly. Once all rows have been backprojected into the reconstruction array, a fully reconstructed slice is formed (Figure 20). This process is repeated for all sinograms to form a full three-dimensional CT reconstruction.

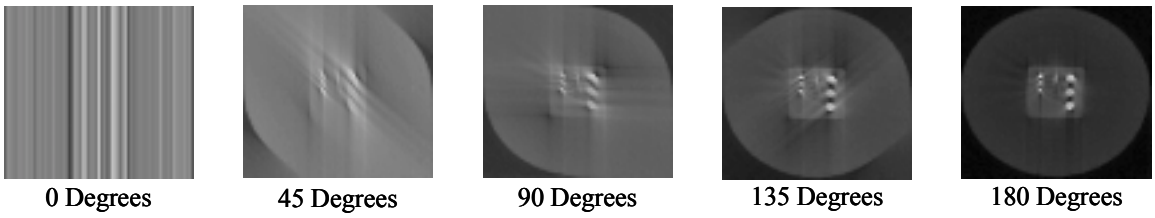

Figure 20: Backprojection as it progresses over all angles of a sinogram.

The run time for parallel back projection is bounded at $O(n^4)$ where n is the size of the input volume. Filtering is an *nlogn* process per row, which must be performed on the sinogram consisting of *n* rows. Backprojection is an n^3 process but is additive to filtering.

The process is repeated over n slices resulting in an asymptotic runtime of: $(n\log n * n + n^3) * n = O(n^4)$.

1.5.2 Fan Beam Reconstruction

Instead of a set of parallel rays, a fan beam spreads out substantially between the object and the detector. A fan beam is formed by a point source close to the object collimated such that only a thin plane of radiation escapes. Fan beams usually only apply in two dimensions such as with a linear detector CT system. An uncollimated fan forms a cone and becomes a different problem. Fan beam is employed because it has some benefits that make it useful in three-dimensional CT. Cone beam shares its asymptotic run time with fan and parallel beam, but contains a much larger hidden constant making fan beam considerably faster than cone beam. Even though a fan beam approach cannot be used to perform an exact three-dimensional reconstruction, it can closely approximate a cone beam in some circumstances. If the geometry contains only a slight cone angle then fan may be a good choice for reconstruction.

A fan beam reconstruction can be performed by modifying the geometry used in parallel backprojection to accommodate the fan, or by applying a simple pre-processing step on the sinogram to convert it into a parallel reconstruction[4]. The pre-processing method has the advantage of being able to re-use the backprojection algorithm created for parallel beam. Only minimal overhead is added so run times remain the same. The sinogram is made parallel by backing out the angular shift caused by the fan beam geometry. The de-shifted sinogram can be used directly as input for the parallel backprojection process described in section 1.5.1.

1.5.3 Cone Beam Reconstruction

Cone beam is the skeleton key of CT algorithms. Parallel and fan geometries are subsets of a cone, so a cone beam algorithm can handle all three. The unfortunate tradeoff is that cone is asymptotically more difficult than either parallel or fan beam reconstructions. Mathematically it is more difficult to understand and the algorithm can be much more difficult to implement. Feldkamp[5] published the definitive paper on cone beam CT in 1984. It is the basis for most cone beam implementations in use today and provides the fundamentals for the algorithm detailed here.

The key to creating a tractable cone beam implementation comes from viewing the problem differently from parallel or fan. The traditional means of implementing cone beam also involved backprojection. The entire set of sinograms is loaded into memory and backprojected one by one into a three-dimensional volume representing the final reconstruction. The concept is straight forward, but the implementation becomes unreasonable. Output volumes can exceed one billion pixels at 32 bits each: totaling four GBs of ram required to perform a reconstruction, not including the sinograms that must be loaded. While high-end PCs and supercomputers may be able to provide such blocks of memory, it is unreasonable to expect such an investment in hardware for CT reconstructions. An algorithm that can reconstruct on a per slice basis would eliminate this problem, however the brute force approach is currently unreasonable.

The pixel contribution algorithm solves the problem. Instead of backprojecting all the data into a large volume, each pixel in each slice is examined to determine what data in the filtered sinograms make a contribution to that reconstructed pixel. All data required by

the pixel is summed in creating a final reconstructed pixel. Repeated over the entire slice and then over all slices, a full three-dimensional cone beam reconstruction is created.

The process is similar to parallel or fan with a few modifications. Initially, the geometry is examined to determine how many sinograms are required to perform the reconstruction on a per slice basis. Cone beam reconstructions are not planar like parallel and fan so many sinograms may be required. Figure 21 illustrates that by using the extrema of the slice to be reconstructed, the number of sinograms needed is determined. The subset indicated is loaded into memory and filtered similarly to the parallel and fan algorithms.

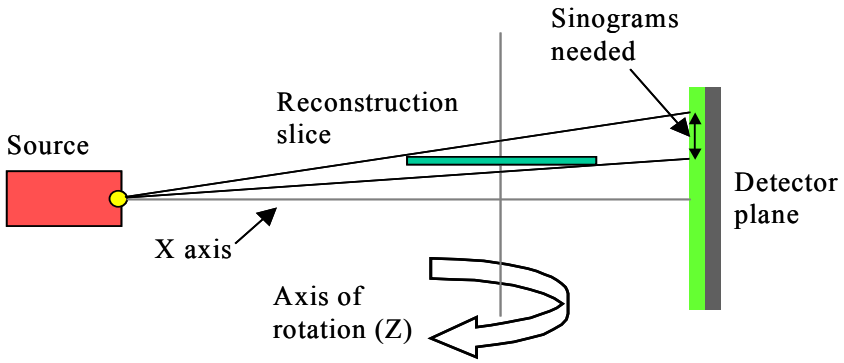

Figure 21: Finding the needed sinograms for cone beam reconstruction using the extrema of a plane.

Reconstruction is performed by constructing a geometry where a ray is cast from the point source through the pixel to be reconstructed. The ray's intersection with the detector plane is calculated and the point is used to interpolate a value for the pixel from the loaded filtered sinograms (Figure 22). The pixel is then rotated identically to the original part and a new intersection is found. The process is repeated for as many angles of data as were collected by the acquisition system; each time a new value being summed into the pixel. After all angles are completed, the result is a reconstructed pixel. The same process

is repeated for all pixels in the plane, resulting in a reconstructed slice. Once all slices are reconstructed, a three-dimensional CT volume is completed.

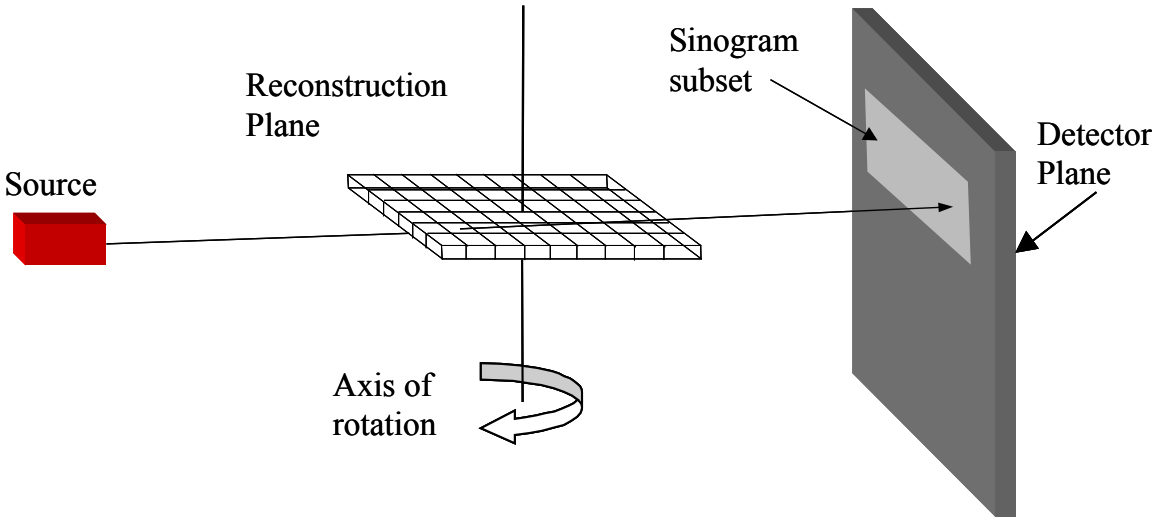

Figure 22: A ray is cast through a pixel to be reconstructed, with its value being interpolated from the sinogram subset.

The total asymptotic run time for cone beam is $O(n^4)$ where n is the size of the input volume. Again, filtering is nlogn per row of sinogram over n rows: n^2logn. The reconstruction is performed on n^2 pixels but each pixel must incorporate n rows of sinogram for a total of n^3 which is additive to the n^2logn. For an entire volume, n slices must be reconstructed for a grand total of $O(n^4)$. Additional trigonometric math is required in the inner loop of the cone beam algorithm that creates a large hidden constant. Long run times are expected for a cone beam algorithm, but cone beam is unavoidable for many CT scans.

The ultimate goal in further development of CT algorithms and methods is to develop a system, which incorporates high-resolution detectors with a cone beam design. The culmination of four years work is a full-volume industrial CT system named FlashCT.

1.6 FlashCT: A new idea in industrial computed tomography

FlashCT (Flat Panel Amorphous Silicon High-Resolution Computed Tomography) was created in response to the new demands of three-dimensional CT. It consists of a data acquisition system (FlashCT - DAS)[2], a data processing system (FlashCT - DPS), and several support utilities.

1.6.1 Data Acquisition System

The data acquisition system was designed to robust yet portable. Hardware consists of an A-Si detector, three-axis staging, and a personal computer running Microsoft Windows NT. The system (Figure 23) is compact and transportable by two people to any radiation source location desired. High-resolution corrected images are acquired in several seconds, and entire data sets can be captured in less than half an hour. The interface (Figure 24) is designed to be user friendly, and provides all the tools necessary to take a CT data set. The software performs image acquisition, correction, and storage suitable for use by the data processing system.

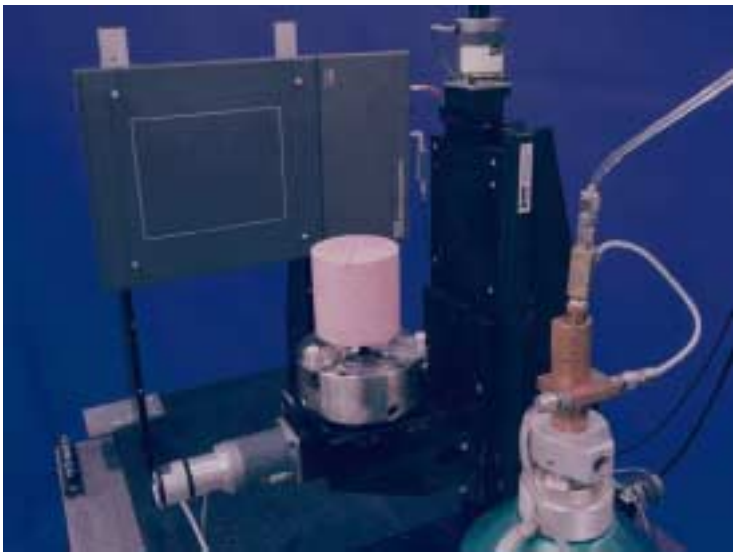

Figure 23: Data acquisition hardware. Back: flat panel detector; middle: three-axis staging; foreground: microfocus x-ray source.

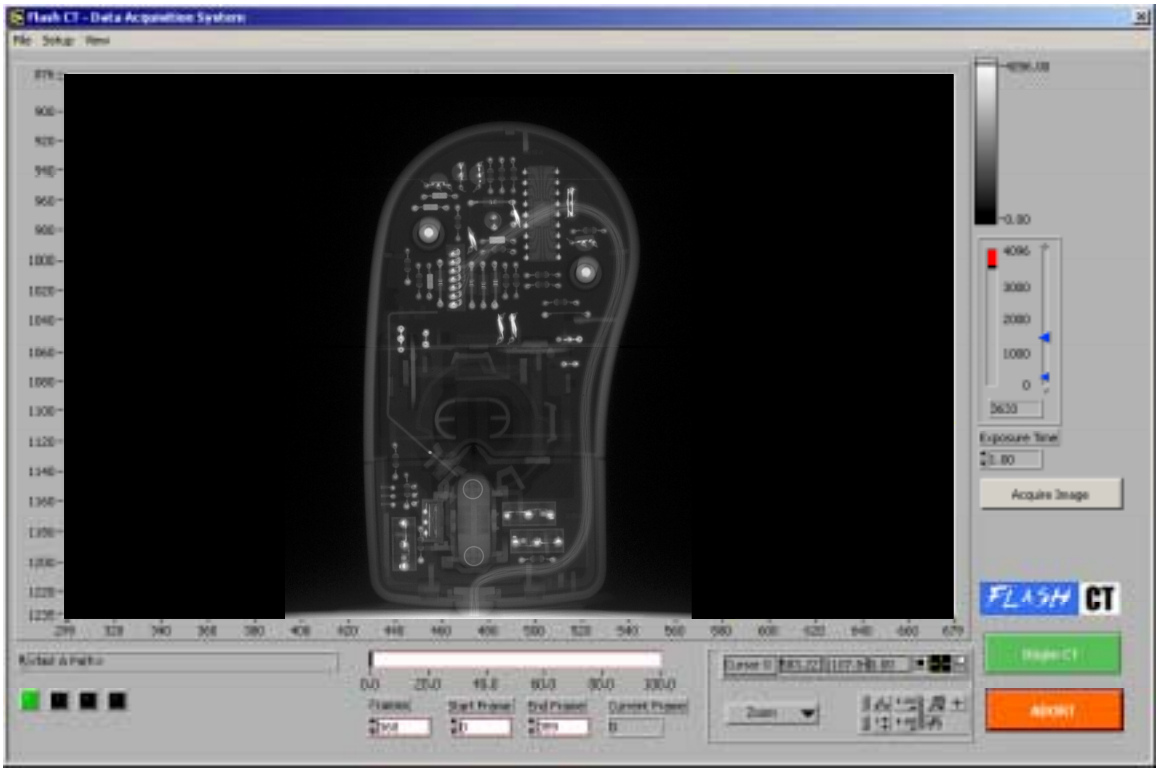

Figure 24: Data acquisition system software interface[2].

1.6.2 Data Processing System

The data processing system is a collection of data manipulation, reconstruction, and visualization tools designed specifically for three-dimensional CT. Many current CT systems use expensive legacy hardware and provide command line interfaces for acquisition and reconstruction. The goal was to create a system with all the necessary tools to perform three-dimensional CT, but to keep it easy to use and executable on popular architectures. FlashCT – DPS provides a GUI where all tools are clearly labeled. Images can be examined interactively, and processing is performed visually so problems can be identified quickly. It includes efficient algorithms for parallel, fan, and cone beam reconstructions and has been extended with multi-processor capabilities, allowing massively parallel reconstructions of CT data. Volume visualization tools are included supporting data slicing, surface rendering, and data analysis.

After undergoing several years of software design and testing, FlashCT – DPS, developed by M. Sheats and A. Davis, is a robust package for computed tomography data processing. It far surpasses earlier packages that required large amounts of memory and excessive compute time[6]. The design is highly modular and utilizes many commercially available packages, reducing development time and cost.

CHAPTER II
DESIGN OF THE DATA PROCESSING SYSTEM

The goal while designing the DPS was to overcome the limitations of other computed tomography systems. The software needed to run on industry standard hardware and software but remain as powerful as CT systems running on large expensive UNIX machines. A GUI must be provided to make the software easier to use and reduce turn around time on data processing. Finally, the software must be extensible to accommodate future developments in CT data processing.

2.1 Hardware, Operating System, and Programming Language Decisions

The obvious answer to the hardware and operating system question is a PC running Microsoft Windows. Flexible, inexpensive, and increasingly powerful, PCs have become the standard for industrial automation and data acquisition. Microsoft Windows is a popular development platform and provides many well-developed tools for creating large software applications. The PC/Windows machine solves the difficulty of users having to learn a new operating system to use the DPS and makes it inexpensive for users to incorporate it into their existing computing infrastructure.

Once a hardware platform has been chosen, the next task is to choose the programming language. The DPS performs extensive array processing, and would benefit from a language tuned for such a task like Fortran. Unfortunately, Fortran is becoming more difficult to maintain and would also be difficult to provide a nice GUI. During the evolutionary development of the DPS, much of the initial prototyping was done in a software package called Interactive Data Language (IDL) from Research Systems, Inc[7].

(RSI). IDL is an interactive data analysis and processing package that provides many of the tools needed to create the reconstruction algorithms. IDL was built on Fortran, so the array processing is fast. The algorithms developed in IDL provided excellent results, but even though IDL provides facilities for user interface creation, the GUI tools were not sufficient for the task.

During the development of the DPS in IDL, a piece of software was written in Microsoft Visual Basic (VB) for examining the contents of CT projects. It was a very simple program that displayed the project's geometry information, but it became obvious that VB could provide the user interface required for the DPS. Unfortunately, VB could never perform data processing like IDL, so moving the code into VB was out of the question.

The solution came when RSI released an Active-X control for IDL. The Active-X control allows Visual Basic to issue commands to IDL. The result was the ability to create excellent user interfaces in Visual Basic and have them utilize the data processing power of IDL. At that point, the first complete design of the DPS was created.

2.2 The Overall Design of The Data Processing System

The Active-X control created some interesting software design challenges. The Active-X control (also called a "widget") simply provides a conduit through which VB can issue commands to IDL (Figure 25). It would be possible to write no code at all in the IDL development environment and perform all the data processing required by sending command after command from VB. The problem comes during debugging and testing. The IDL development environment provides excellent tools for creating and testing its

software. Those tools would not be available if VB was serially issuing commands via the Active-X widget. The design calls for complete development of all CT routines in IDL with VB only making minimal commands via the Active-X widget. The problem remained how to keep the VB side from becoming disorganized with Active-X calls scattered throughout the software. The solution is the mirrored class design: A set of duplicate classes in both IDL and VB. The IDL classes contain all the code to do the data processing, while the VB classes only contain the commands necessary to execute the calls in IDL. The next task was to create a class system that allowed for further modification without causing complete redesign of the IDL and VB code.

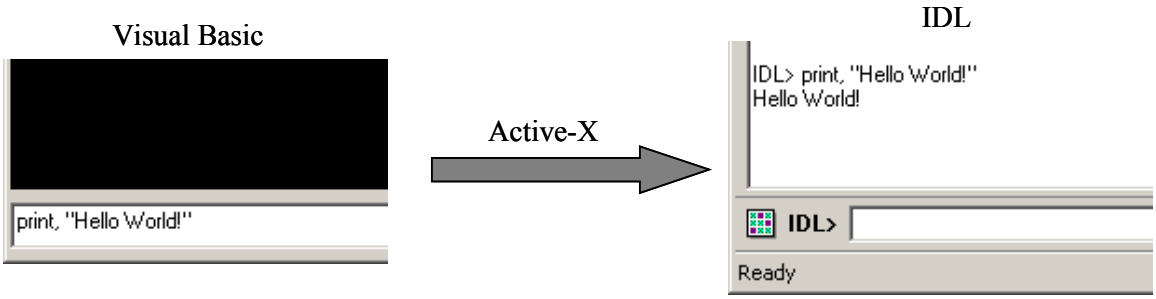

Figure 25: Active-X relationship between Visual Basic and IDL.

2.2.1 The Design of the IDL Classes

The IDL classes were created to follow the three major steps in CT data processing as presented in section 1.4. A central class called CTProject manages all project information, providing file names, geometries, and the necessary I/O routines. The main classes consist of: CTSinoProc for converting images to sinograms, CTCentering for centering the converted sinograms, CTParrRec, CTFanRec, and CTConeRec for performing parallel, fan, and cone beam reconstructions. CTParrRec and CTFanRec share a CTBackproject class that performs filtered backprojection. CTFanRec only differs from CTParrRec by the

pre-processing step necessary to convert the sinogram to a parallel reconstruction. CTConeRec contains a class called CTConeData, which is the custodian of sinogram management and selection. Figure 26 illustrates the main classes and their relationships.

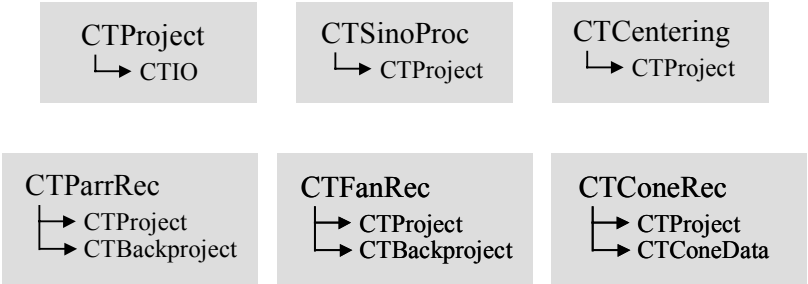

Figure 26: Main IDL classes. The arrows indicate a Contains-A relationship.

2.2.2 The Design of the Visual Basic Classes

The VB classes exactly mirror the IDL classes. Each class exists to provide a similar interface to the VB side of the program as exists on the IDL side. The VB program uses its own classes to setup and execute high level CT routines which are translated into the appropriate Active-X commands and passed along to the IDL classes. Figure 27 shows the mirrored relationship between the classes on the VB side and those on the IDL side.

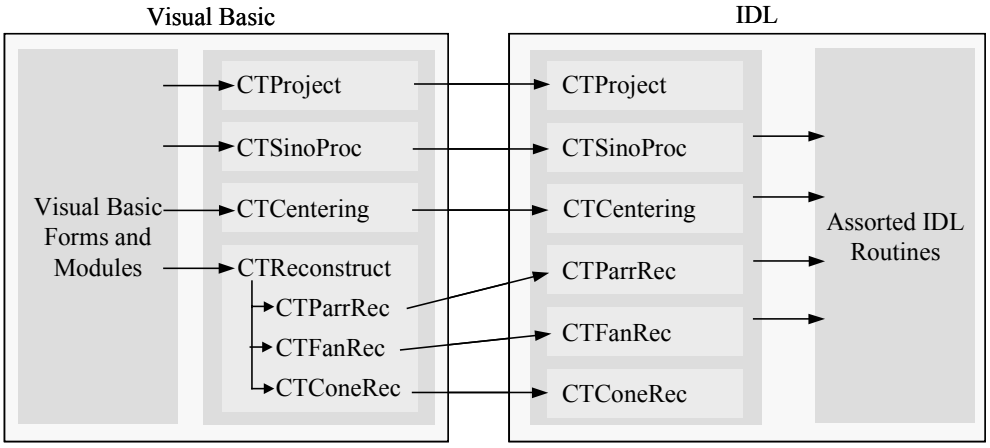

Figure 27: Relationship between Visual Basic classes and IDL classes.

2.2.3 The Complete Package

The finished product appears to the user as one software package. The VB part of the program hides the mechanics of the data processing techniques, allowing them to be upgraded without changing the way the user perceives the system. Via the Active-X widget, IDL can provide visual feedback in the form of data images and plots. The DPS combines the power of a complex image analysis tool with the ease of use of a VB application.

2.3 A Tour of the Data Processing System

The DPS is made up of several sections that provide the analysis, processing, and reconstruction of CT data. A quick tour of the system will provide an insight to its ease of use and how much complexity is hidden from the user.

2.3.1 The Main Menu

The main menu (Figure 28) is the starting point for all data processing. Here the user is able to load a project and choose the steps to perform. The main menu was designed to be minimally complex so the software does not immediately intimidate the user. Buttons and icons are provided to lead the user through the steps desired.

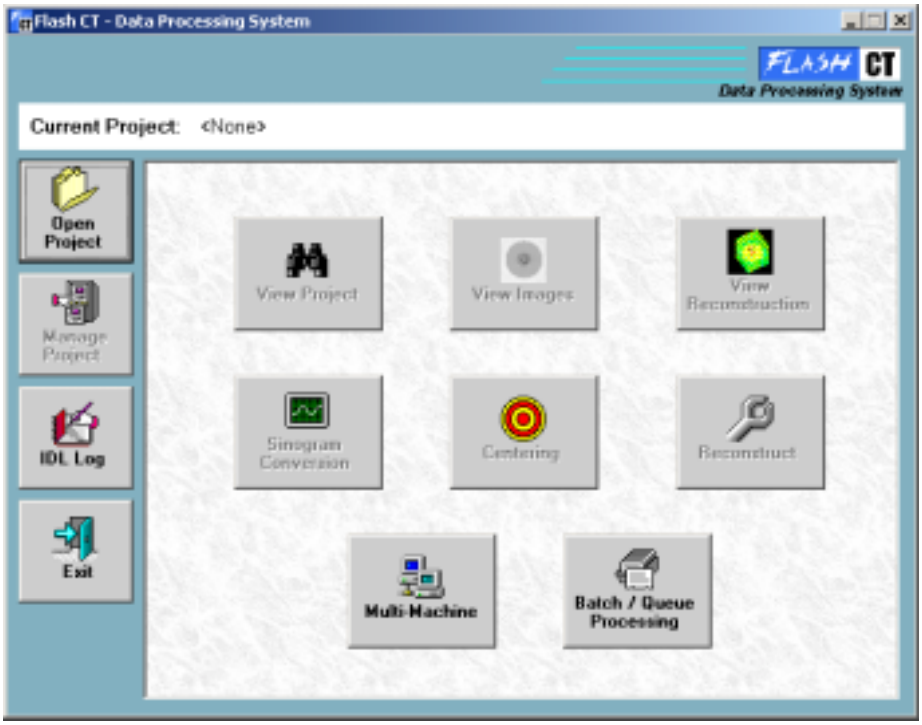

Figure 28: Main menu of FlashCT - DPS.

2.3.2 The Image Viewer

The image viewer is the primary tool for analyzing individual pictures from the CT data set (Figure 29). The bar on the left provides a listing of all images, sinograms, and reconstructions present in the currently loaded project. The large central pane displays the currently selected image, in this case, a radiograph from a Microsoft Mouse data set. Using the tools in the bottom left corner, the user can zoom the image or adjust the bit depth scaling. The graphs across the bottom provide horizontal and vertical profiles as well as a histogram. A color map editor is provided to adjust the appearance of the image or to accentuate desired features.

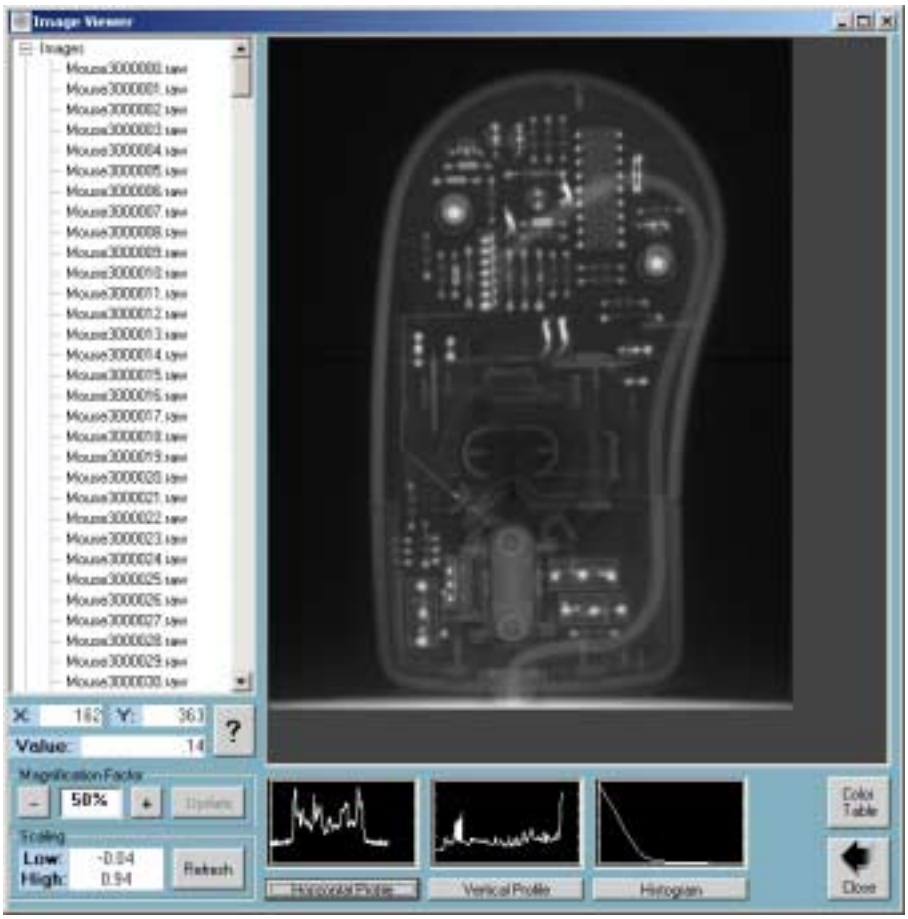

Figure 29: Image viewer from FlashCT - DPS. A radiograph of a Microsoft Mouse is displayed.

2.3.3 The Sinogram Converter

The sinogram converter gives the user the ability to convert the raw images into sinograms, or the sinograms back into raw images (Figure 30). Both abilities are provided because linear array systems directly capture sinograms, and it is useful to recreate the images represented by them. The central pane gives the user the ability to select a range of sinograms to be created by moving two horizontal bars across the image. The bars indicate the first and last rows to be used in all images when creating the sinograms. The user has the option of displaying debug statements as IDL processes the data. This is useful if an error is occurring, or if the user wishes to monitor the progression of the processing more closely. The batch processing option will be described further in Chapter 3.

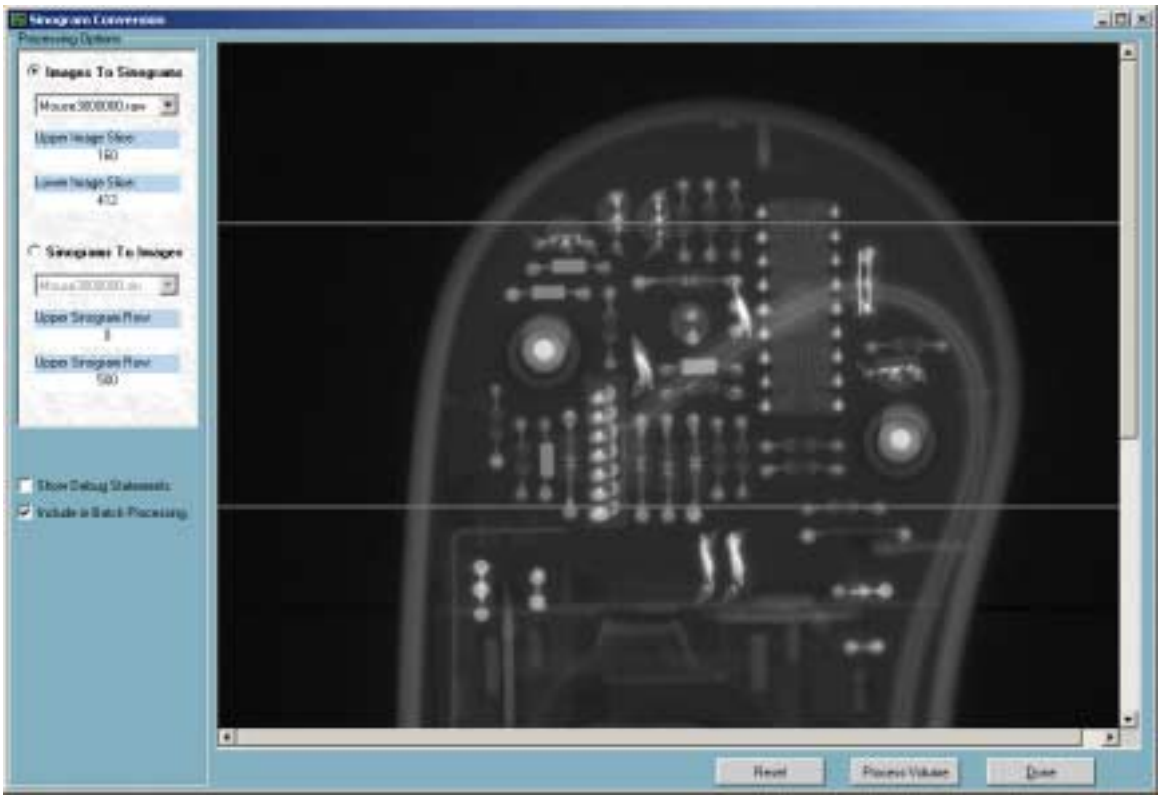

Figure 30: FlashCT - DPS sinogram processor.

2.3.4 The Centering Tool

The centering tool allows the user to determine how far off center the sinograms are and correct them either automatically or manually (Figure 31). The centering is done by determining the center of mass of the sinogram at every angle and fitting it to a sinusoid. The adjustment is done via a slope and intercept adjustment that describes how the center axis needs to be adjusted to line up with the correct center axis displayed on the center pane. These values can be calculated automatically, or if the user prefers, entered manually. Again, debug statements can be displayed during the process, and the centering can be included in batch processing.

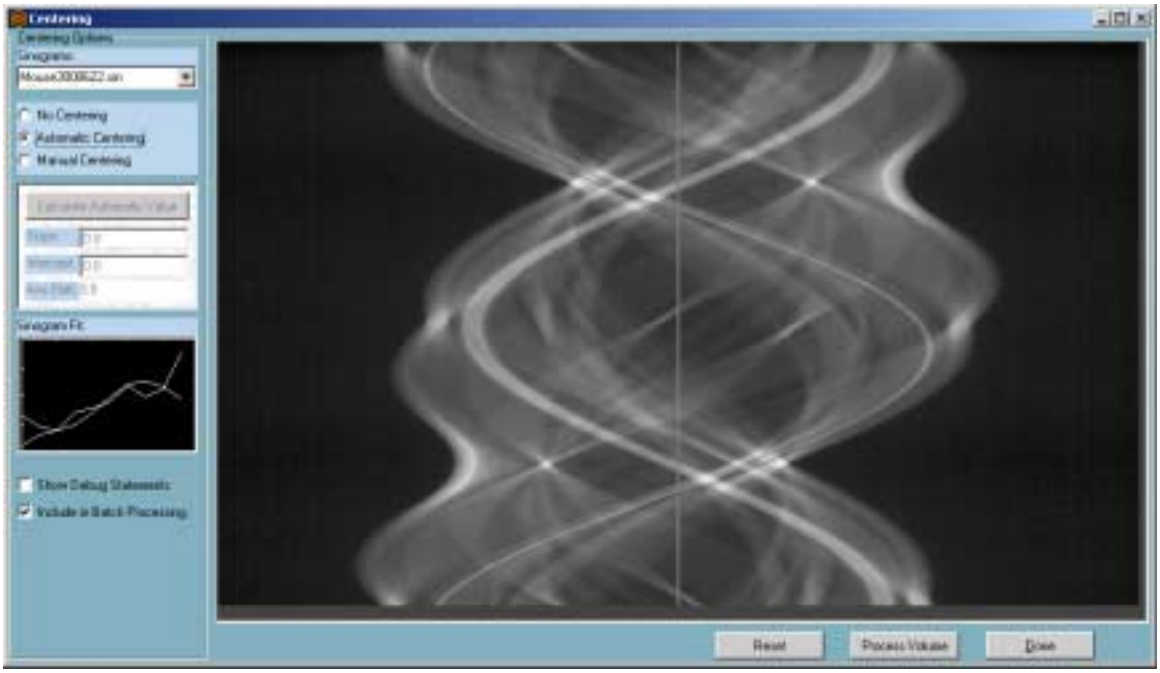

Figure 31: FlashCT - DPS centering tools displaying a sinogram from the Microsoft Mouse.

2.3.5 The Reconstruction Tool

The reconstruction tool provides all the resources for transforming sinograms into complete reconstructions (Figure 32). On the top left, the user can select the sinogram row to reconstruct, and the appropriate geometry. The user can than adjust the output size and resolution desired using the text boxes and slide bars. Several pre-processing tools are available to remove reconstruction artifacts. Ring removal performs filtering to suppress noise that causes rings in the reconstruction. Median filtering removes single pixel variations from the sinogram to minimize the salt and pepper noise in the reconstruction. The checkboxes on the bottom let the user select various other options. The “Skip if Newer Than Batch” button allows the user to resume a suspended reconstruction or overwrite an existing one. The “Show Slice Progression” button lets the user view the backprojection algorithm as it progresses as shown in Figure 20. “Show Slices as Processed” shows each finished reconstruction when a volume is being done. On the bottom right, the user

determines if a single slice or entire volume is to be reconstructed. A subset volume can be selected using another dialog box. Once all settings are complete, the user can include this step in the batch processing as well, or perform the processing using the "Process" button on the right.

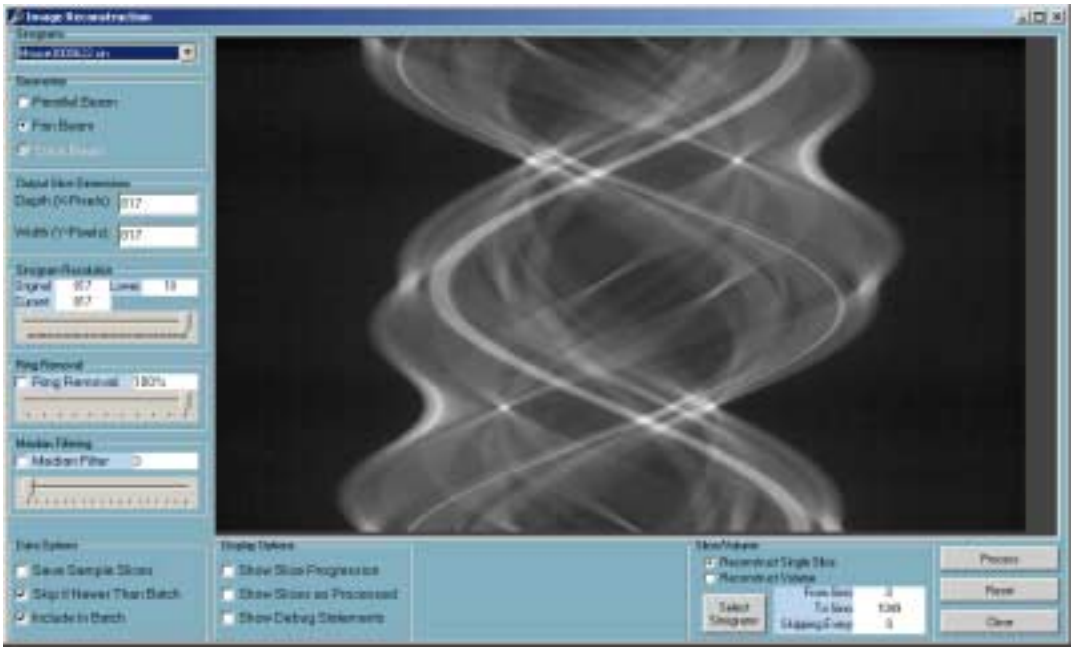

Figure 32: FlashCT - DPS reconstruction tool.

2.3.6 The Visualization Dialog

FlashCT – DPS uses IDL's built in visualization tool called Slicer. Often the data set is too large to load into memory, so a subset or resizing of the data is required. The visualization dialog informs the user of how much total memory is available and the amount the data set will use in its current configuration (Figure 33). The user can select a subset of slices to be loaded and can select how large they should be scaled while loading. Once the user is satisfied that the data will fit into memory, the "Load" button is pressed and the Slicer tool is given all the data the user wishes to see. The Slicer tool provides basic iso-surfacing and block cutting tools typical to most visualization packages.

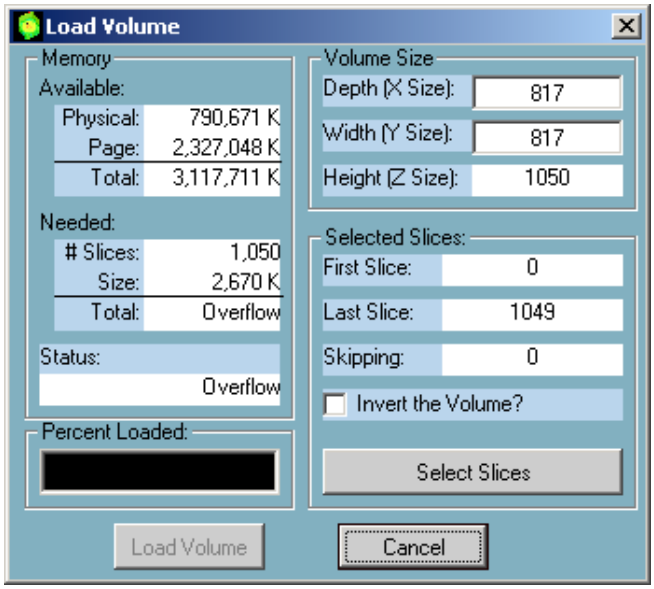

Figure 33: FlashCT - DPS Visualization dialog. The memory required for the Microsoft Mouse data set causes a math overflow.

2.3.7 File Management

While testing the system, it was often necessary to go into the project and delete or backup data files. Using the Windows file manager became cumbersome because of the number of files involved. So the FlashCT file manager was created to allow the backup, restore, or removal of all the files involved with the loaded CT project. It remains a useful tool for displaying the amount of disk storage used by the project, and for deleting sinograms and reconstructions that often do not need to be kept for archival. The backup and restore features are useful when the user wishes to try other reconstruction parameters without destroying the current set of reconstructions. Additionally, the manager can remove support files created during processing.

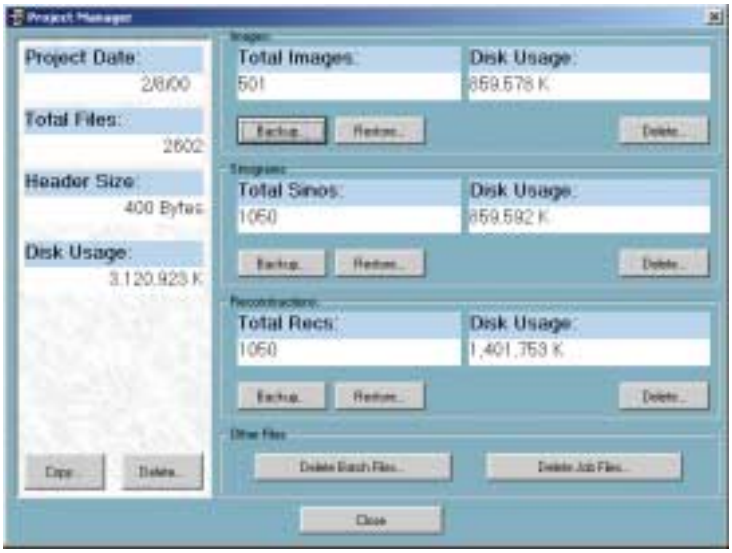

Figure 34: FlashCT - DPS file manager.

2.4 Additional Design Issues for the Data Processing System

The DPS meets all of the goals set at design time. It is user-friendly, runs on modern hardware and operating systems, and is easily extensible in the future. Unfortunately, there remains one inherent problem related to full volume CT. The reconstruction of large data sets remains slow. Examination of the reconstruction process reveals an important commonality of all three reconstruction algorithms: they are all slice based. A full volume reconstruction involves hundreds or thousands of slices. Each slice can be reconstructed independently, thereby implying massive parallelism. If a multi-processor version of the DPS could be created, the massive parallelism of the CT data sets could be exploited.

CHAPTER III
EXPLOITING THE MASSIVE PARALLELISM OF FULL VOLUME CT RECONSTRUCTION

The concept of a multi-processor version of the DPS came about late in the writing of the software. A final design was in place and most of the code was finished. Several possibilities were envisioned, such as writing an entirely new version for doing multi-processor reconstructions, or porting all of the code to a UNIX platform, where multi-processor toolkits such as MPI or PVM could be used.

Porting to UNIX goes against one of the main goals: keeping the system on popular hardware and operating systems. Also, a UNIX port would take considerable redesign time and research into new GUI toolkits and image processing libraries. Writing an entirely new version of the DPS using IDL and Visual Basic seemed like the best option, but after exploring the design for such a piece of software, it became obvious that it would share much of the same functionality as the existing DPS. The decision was made to extend the DPS to include multi-processor support as an option. The first problem encountered was how to automate a process that is intrinsically user interactive.

3.1 The Batch-Queue System

The DPS was originally designed to be wholly interactive with the user. The user would perform each step independently, setting up the options and pressing the process button. The process became problematic as the individual steps could take a considerable amount of time, meaning the user had to check in periodically to see how the DPS was progressing. Also, to implement a multi-processor version, all the settings would need to be known in

advance of task initiation. The batch-queue system was created to capture all settings needed to perform the data processing and call the routines necessary to advance through all the steps (Figure 35).

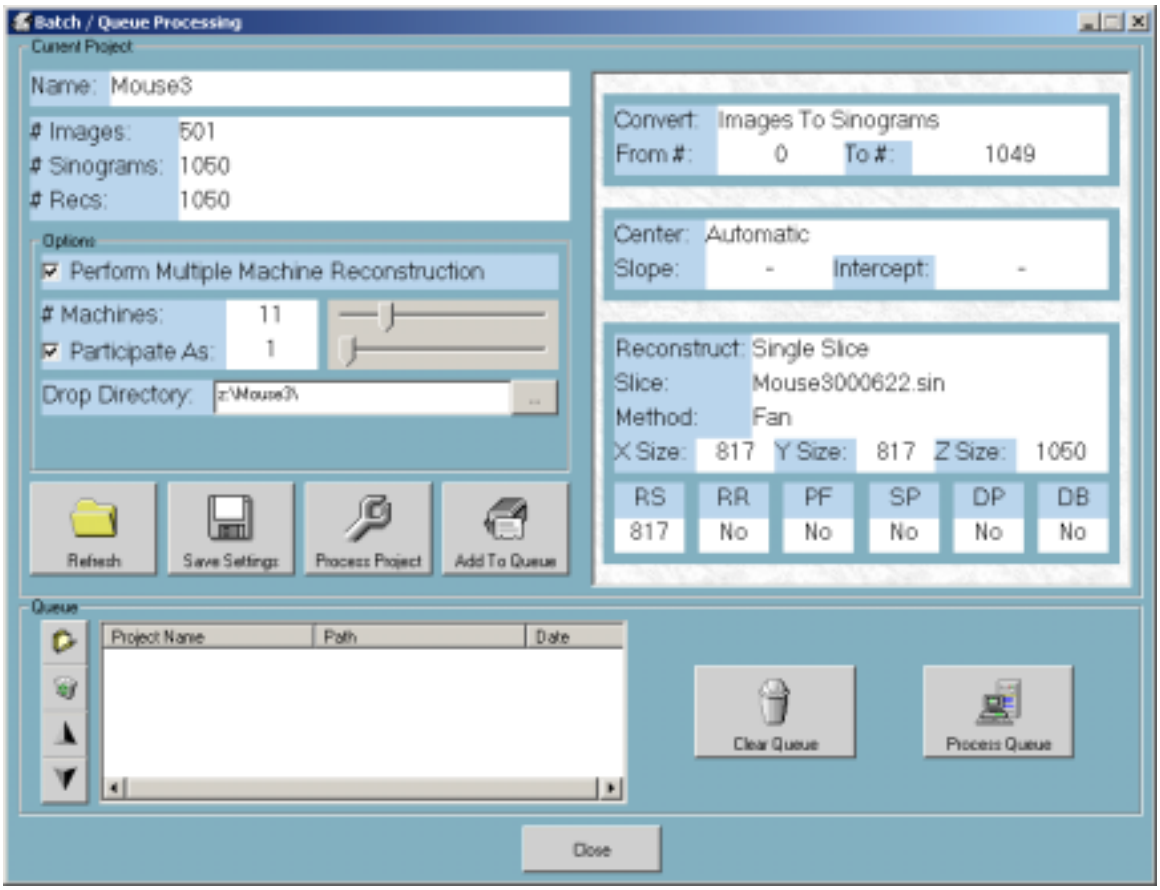

Figure 35: FlashCT - DPS Batch-Queue system.

The sinogram processing, centering, and reconstruction screens all contain the option to "Include in Batch Processing." If the check box is selected, all the settings from that step are saved for inclusion in the batch processing (Figure 36). The user goes to each of the three screens, selects all settings, returns to the batch-queue screen and selects "Process Project". The DPS takes over and performs all the steps specified (Figure 37).

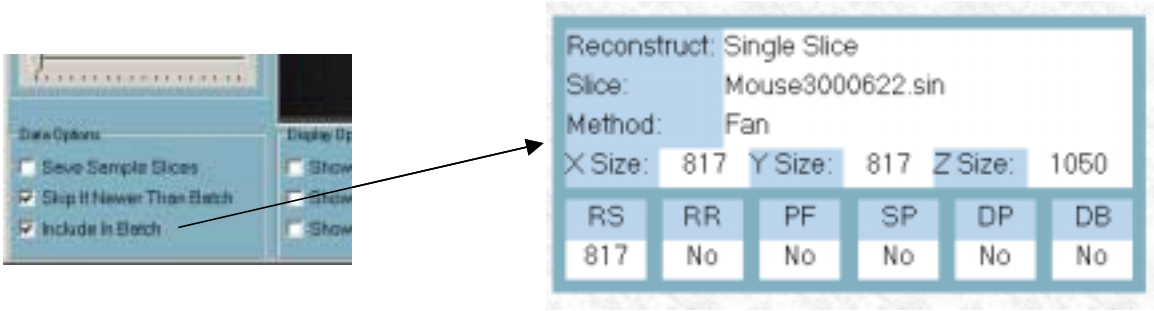

Figure 36: Selecting the "Include In Batch" checkbox includes the settings in the batch-queue process.

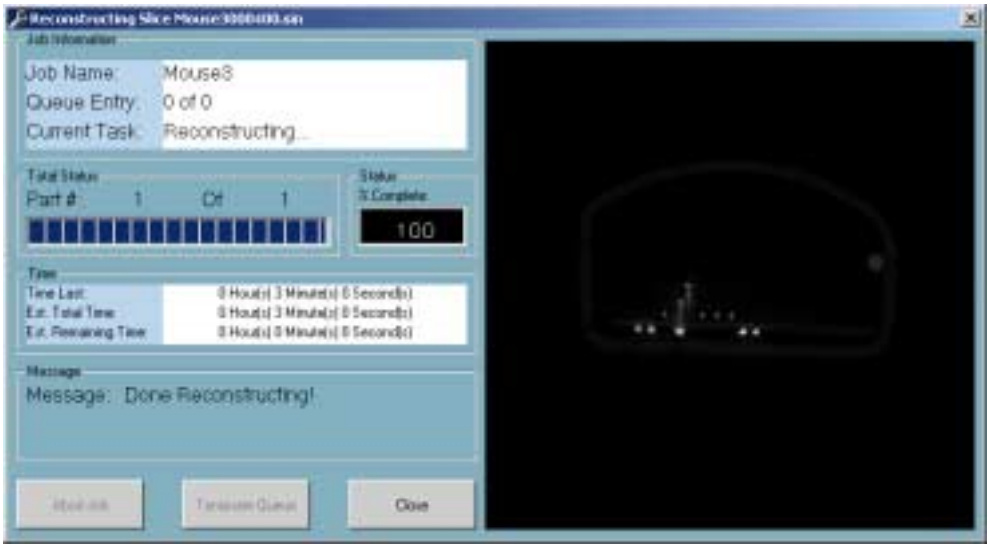

Figure 37: FlashCT - DPS status screen shown during data processing steps.

The queue part of the batch-queue system takes batch processing a step further. The user can not only setup all the properties for one project and have it processed, but can also setup multiple projects, add them each to the queue, and have the DPS process all projects in sequence (Figure 38). The queue feature was added to allow uninterrupted processing over nights and weekends. The user can queue all the jobs to be reconstructed and view the results later.

Figure 38: Queue where several projects can be issued for batch processing.

The batch-queue system provides a way to specify how a project is to be processed. Multi-processor reconstructions could be performed by having multiple machines load the batch information and process the project, but the batch information does not detail how a project would be broken up amongst multiple machines, and the mechanism for communication has not been established.

3.2 Communication and Job Determination

The next design issue was how to incorporate intra-machine communication into the DPS without disrupting the way it operates on a single processor. Sockets were explored as well as Microsoft Windows MPI (Message Passing Interface) implementations, but it was decided that both of these would dramatically change the way the DPS operates. Extensive communication was not required as the only message needed was: “process this project in this way”. If all the processors involved could monitor a central message space for that message, the problem would be solved.

Microsoft Windows provides a central message space via its file sharing capabilities. Multiple machines can mount a common file structure. By using a directory on that file structure, the message space is formed where individual files form the messages. The messages would contain the location of a project to be processed and provide instructions on how each processor should perform its task.

The implementation was named the “drop directory” method. The drop directory is a central location on a shared drive where machines that are to participate monitor for new messages. The DPS was given the capability to start in a multi-processor form which immediately sets it to monitoring on the specified drop directory (Figure 39). Each machine watches the directory for its own drop file that specifies the location of a project to be reconstructed. A machine identifies its drop file by a naming convention which includes the same processing number as the machine has been assigned (Figure 40).

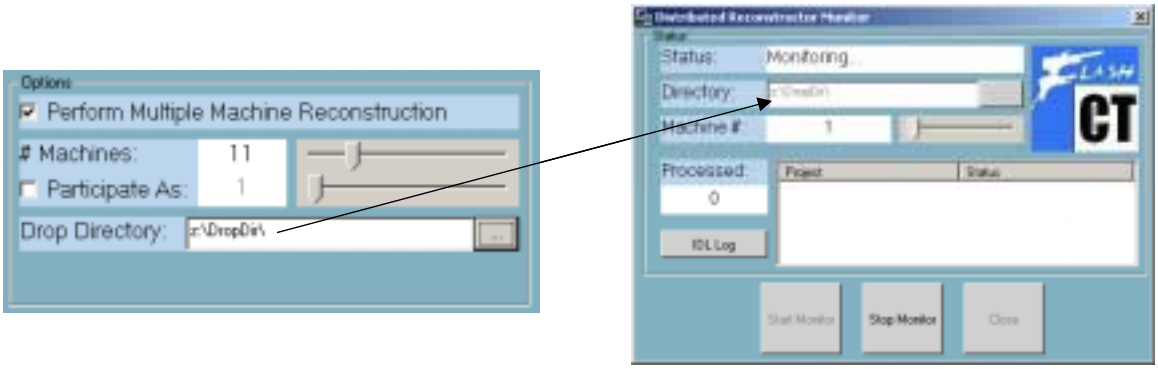

Figure 39: Drop directory is specified in the batch processing dialog and monitored on by the distributed reconstructor monitor.

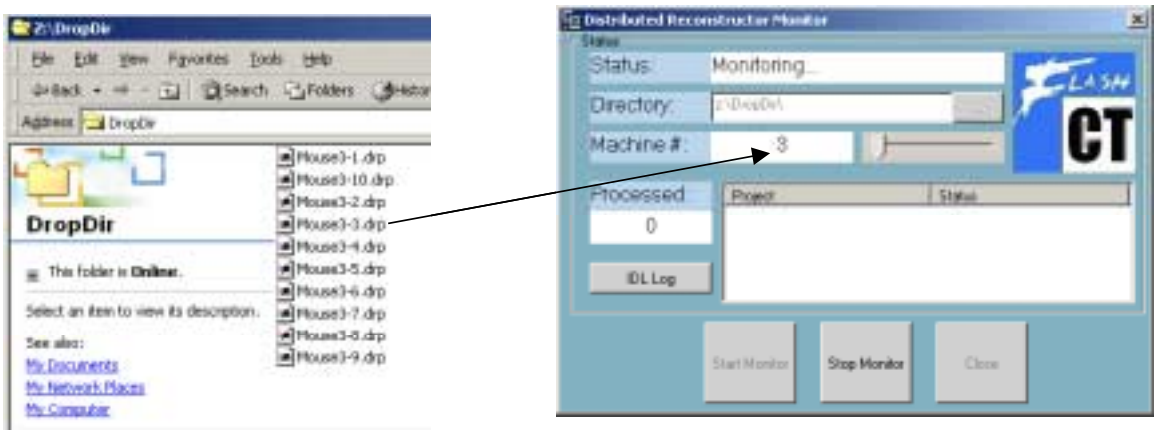

Figure 40: Each drop file is intended for a specific machine with the same processor identification.

Once each machine receives its drop file, it loads the specified project and its batch information. A job file is available in the project's directory for each processor that will be operating on the project. The job file specifies the processor's job, which could include: sinogram processing, centering, or reconstructing a subset of the total reconstructions. Each processor performs its specified task and returns to monitor the drop directory upon completion. Drop and job files are not removed until the task is completed, so if the machine should crash or be rebooted, processing can continue once the machine re-runs the DPS.

File based communication and job submission allowed for rapid integration of multi-processor functions into the DPS. The system has proven robust through rigorous testing and though only run in configurations up through 26 processors, it is theoretically possible to use the system with hundreds of machines. The only bottlenecks are network bandwidth and file I/O. It may be possible using current technology to hierarchically organize a file sharing system where large groups of machines would each use a separate file server but operate on the same project. Separate subnets could be used to minimize network traffic. Using such a system it is conceivable to extend the multi-processor functions up to thousands of processors.

3.3 The FlashCT – DPS Multi-Processor Test Bed

Testing the multi-processor version of the DPS was originally done on several spare PCs available in the lab. However, it quickly became obvious that to use the multi-processor DPS as a valuable tool, a dedicated cluster needed to be built. The first cluster constructed was labeled the FlashCT Platinum Cluster (Figure 41). It consists of 10 Intel PIII-500 processors with 256 MB of RAM each. Their files are served from a 30 GB Quantum

Snap server (not pictured). All machines can be operated interactively via the keyboard/monitor switch. The cluster ran as expected and data could be processed 10 times faster than previously. This cluster is currently in service processing production CT data.

Figure 41: FlashCT Platinum Cluster. The first dedicated DPS reconstruction cluster.

With the success of the Platinum cluster, a second set of machines came under consideration. The Platinum cluster was built by hand from bulk parts and took a large block of time to construct. When trouble occurred, there was no means of tech support and more time had to be spent. Research was done into commercially available computing clusters, and two Cubix units were purchased. Each Cubix unit consists of four machines

with dual PIII-550s and 256 MB of memory, resulting in 16 processors total[8]. The entire system fits into a portable rack, taking up less space than the Platinum cluster (Figure 42). Additionally, the Cubix arrived pre-assembled, with the only task remaining: loading it into the rack and hooking up the power. The combined Cubix and Platinum clusters provided unparalleled reconstruction times, processing entire data sets in minutes rather than hours.

Figure 42: Second DPS cluster, built from commercially available Cubix computing units.

The clusters made it easy to perform large reconstructions and allowed the data processing to keep pace with the data acquisition schedule. Unfortunately, the clusters created a management nightmare. Whenever a reboot was necessary, each system had to be logged into and shutdown manually. Often, it was difficult to determine which

machines were operable and which had crashed. Some means of cluster management was needed to make administration of the systems simpler.

3.4 The FlashCT Cluster Manager

The cluster manager was created to meet several requirements. The manager should be able to monitor the cluster to show what job is running, show how it is progressing, and abort it if necessary. The manager should also be able to shutdown or reboot all the machines in the cluster when maintenance is required.

The manager monitors jobs by watching the drop directory (Figure 43). When a project is submitted, the monitor looks up the project and determines how many processors will be operating and what their tasks will be. The monitor is able to measure progress by watching as sinograms or reconstruction files are created. The monitor displays a graph of the overall progress of the reconstruction as well as the progress of the individual processors (Figure 44). As each job progresses, the monitor logs the steps so a record is kept of how the cluster is being used.

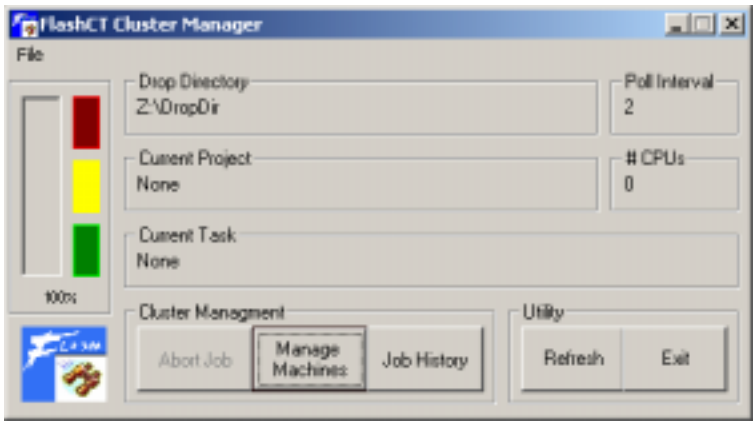

Figure 43: FlashCT Cluster Manger sitting idle, monitoring the drop directory for a new job.

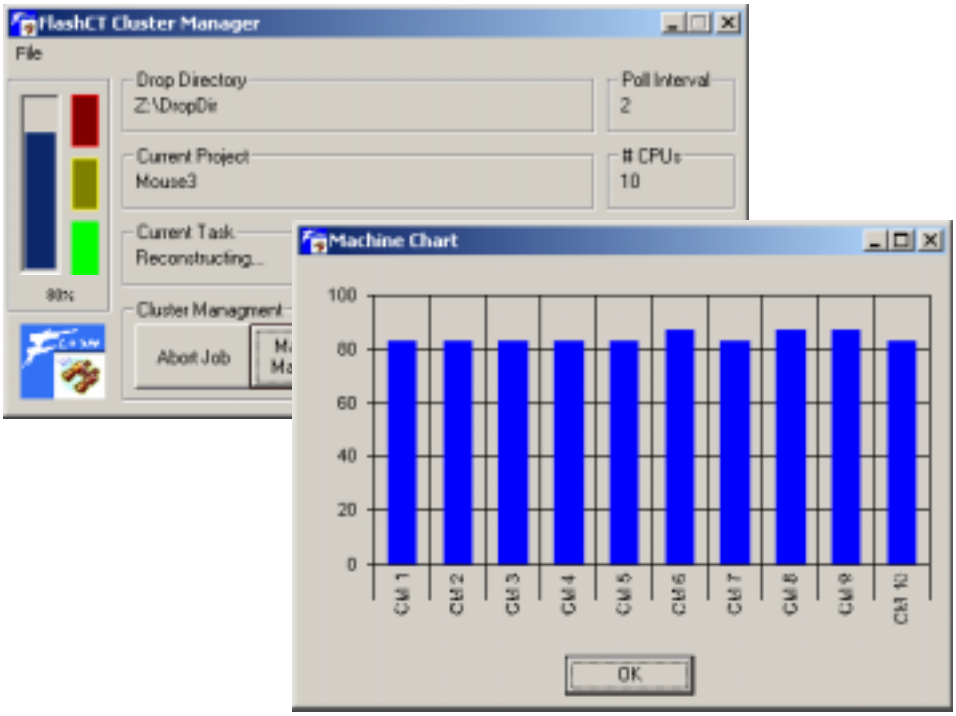

Figure 44: Cluster manager showing the progoress of a job. The machine chart illustrates the progress of each processor.

Hardware management is done via Microsoft Windows API calls. The API has routines to remotely shutdown and reboot workstations on a domain. The manager allows the user to specify the list of machines in the cluster or to discover them from an already processed data set. The manager pings the machines to determine if they are alive and provides a list of their status (Figure 45).

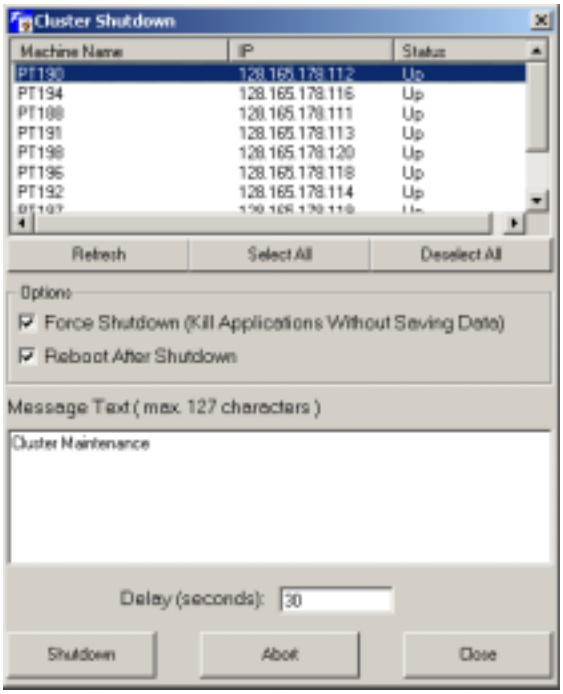

Figure 45: Cluster shutdown screen allows the user to see which machines are currently running and shutdown or reboot them as needed.

The cluster manager has made system maintenance trivial. New software can be installed easily and daily shutdowns are instantaneous. The monitoring facilities allow progress to be watched remotely. The manager is as extensible as the DPS and is able to handle jobs using hundreds of processors.

CHAPTER IV
DATA PROCESSING SYSTEM RESULTS

4.1 Reconstruction Algorithm Results

The DPS has undergone extensive testing using a variety of data sets. The best technique for finding problems in reconstruction algorithms is through the use of simulated data. Simulations remove the variable image quality of real CT data sets and allow for just the mechanics of the reconstruction algorithm to be examined. Figure 46 is a CAD model of a part simulated at LANL on the ASCI (Accelerated Strategic Computing Initiative) machine using MCNP, a radiography simulation package[9,10]. It is a stack of aluminum plates attached to a copper rod. The part was designed to be an excellent test piece for parallel, fan and cone beam reconstructions. A sample cone beam radiograph is provided in Figure 47.

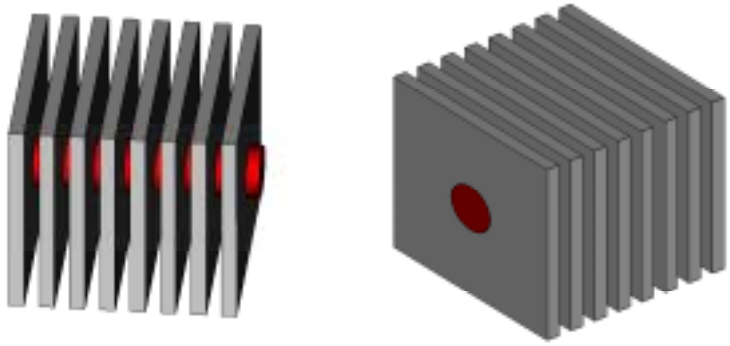

Figure 46: CAD representation of a test piece simulated using MCNP for diagnostics on the reconstruction algorithms[11].

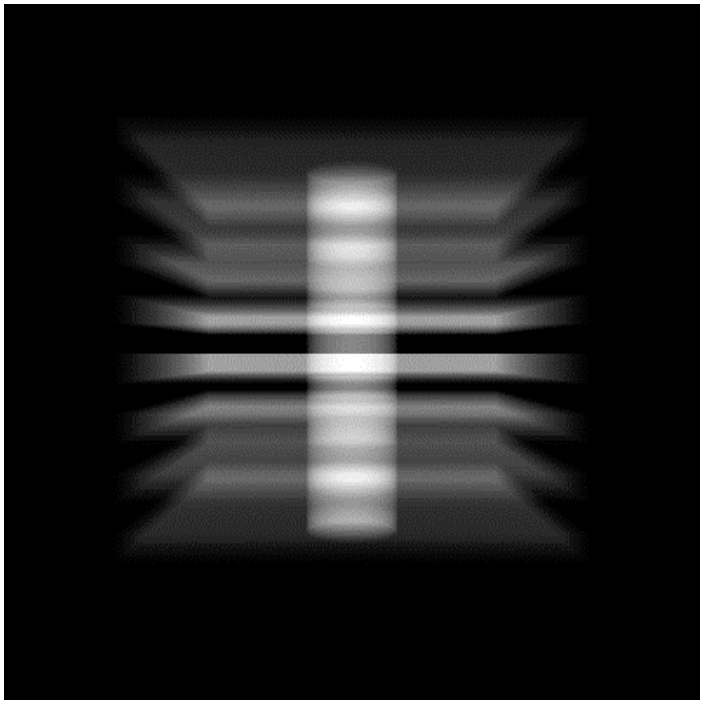

Figure 47: Simulated cone beam radiograph from the test piece.

4.1.1 Parallel Beam Reconstructions

The parallel beam reconstruction algorithm is tested by simulating the piece with perfect parallel beam geometry. To determine if the algorithm is functioning correctly, a central slice is reconstructed and examined for its accuracy in representing the original object. Sharp, straight edges are the best clue that the algorithm is functioning correctly. As shown in Figure 48, the DPS was able to reconstruct a slice that precisely matches the original simulation data.

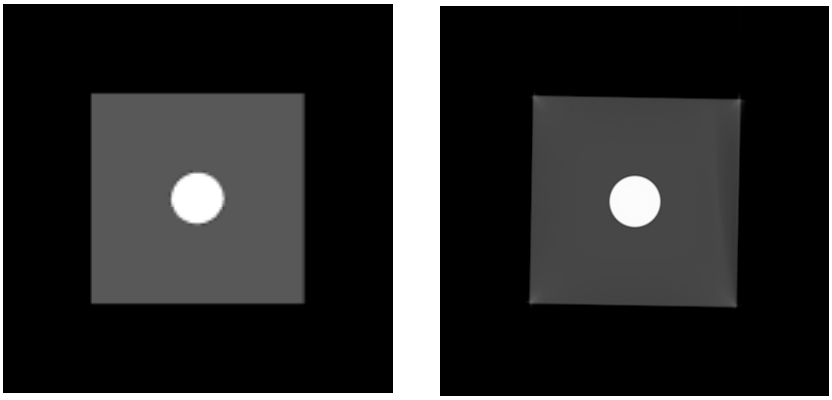

Figure 48: Slice from the test piece (left) and a parallel beam reconstruction created from the simulated radiographs (right).

4.1.2 Fan Beam Reconstructions

To simulate fan beam data, the geometry was modified into a cone configuration. The central slice of a cone beam data set is a perfect fan because the radiation has no vertical component. Again, the fan beam reconstruction is compared to the original simulation data, but also compared to a parallel beam reconstruction of the same slice. Figure 49 illustrates the massive distortion caused by a parallel beam reconstruction of a fan beam data set. The fan beam algorithm is able to reconstruct the data set to match the original model.

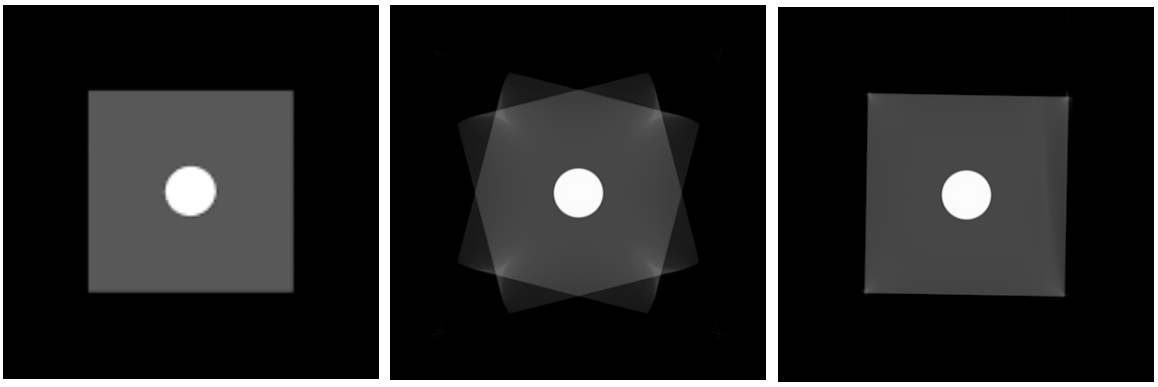

Figure 49: Slice from the test piece (left), a parallel beam reconstruction of a fan beam slice (center), and an accurate fan beam reconstruction (right).

4.1.3 Cone Beam Reconstructions

The same geometry as used for the fan is used for the cone beam reconstruction test. Cone beam artifacts are only seen when the reconstructions move away from the central slice. A slice is chosen high in the data set and reconstructed using both fan and cone beam algorithms. If the data set were not created using a severe cone, the fan would provide a close approximation to the cone algorithm, however Figure 50 shows that the fan is unable to recreate the structural precision the cone beam reconstructor generates.

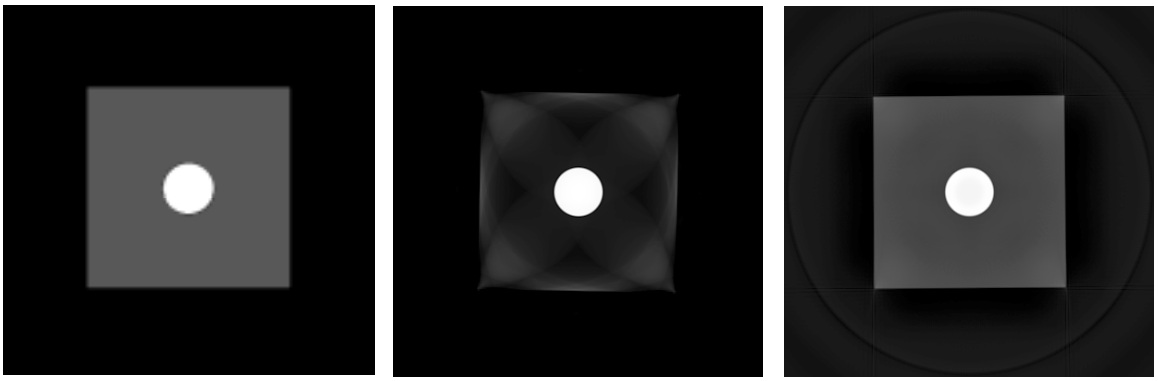

Figure 50: Slice from the test piece (left), A fan beam reconstruction of a cone beam slice (center), and an accurate cone beam reconstruction (right).

4.2 Data Sets of Interest

The simulated data shows that the three reconstruction algorithms perform correctly. However, it is only through real data that it can be determined how the DPS will function under normal circumstances. The following data sets were collected with a variety of sources and geometries.

4.2.1 Rock Core

The rock core is a core of concrete that was examined to determine the distribution of the aggregate[12]. The core dataset was acquired at the Los Alamos Neutron Science Center (LANCSE), using thermal neutrons in parallel beam geometry[13]. The core demonstrates

the variety of materials examinable with FlashCT and the structural detail that can be provided (Figure 51).

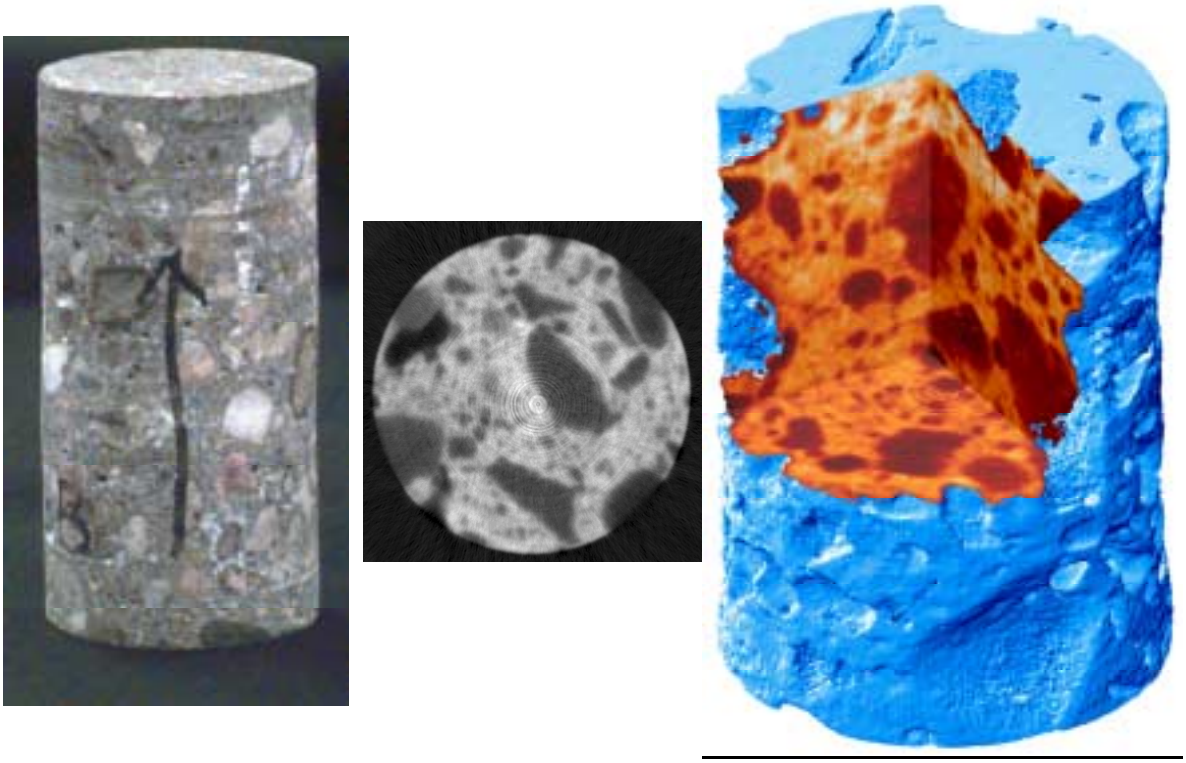

Figure 51: Concrete core data set reconstructd using FlashCT - DPS's parallel reconstrution algorithm. Photograph (left), a single slice (cneter), a volume (right) are displayed[12].

4.2.2 Coral

The growth patterns of coral can be difficult to determine. Any cut made to the coral damages the information needed. FlashCT was used to view the growth rings without having to make any cuts. Data was acquired using x-rays in slight cone beam geometry. The images shows were reconstructed using the fan beam algorithm (Figure 52).

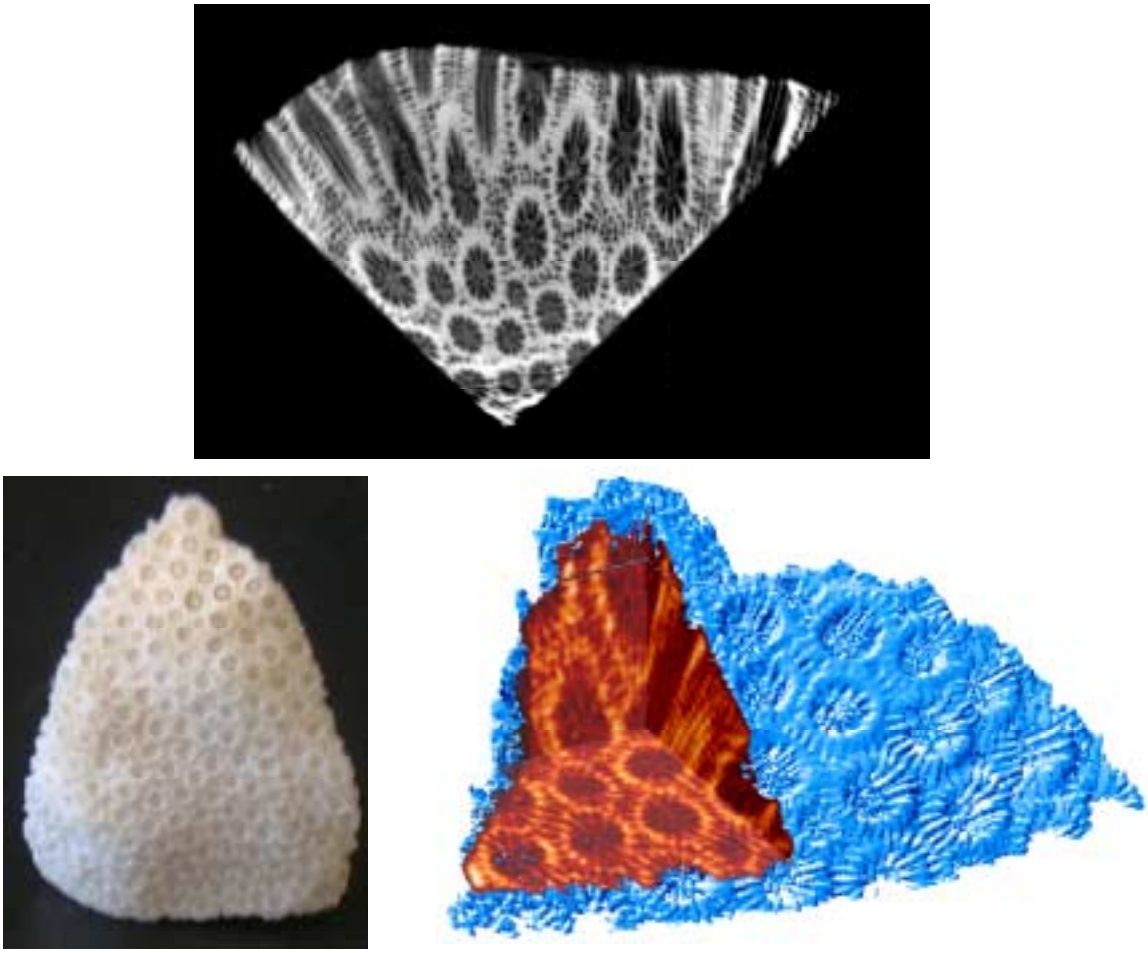

Figure 52: Coral data set reconstructed using FlashCT - DPS's fan beam reconstruction algorithm. A photograph (right), single reconstructed slice (top), and volume rendering (right) are displayed.

4.2.3 Mag Light

The mag light was created to be a small, sample data set for use in training. Figure 53 shows the images created from the data set. The detail discernable on the reconstructed slice demonstrates the power of the FlashCT system. The filaments of the light bulbs are visible, the cracking in the battery is obvious, and the threads on the end cap are measurable. The data set was collected with x-rays in slight cone beam geometry, but reconstructed using the fan beam algorithm.

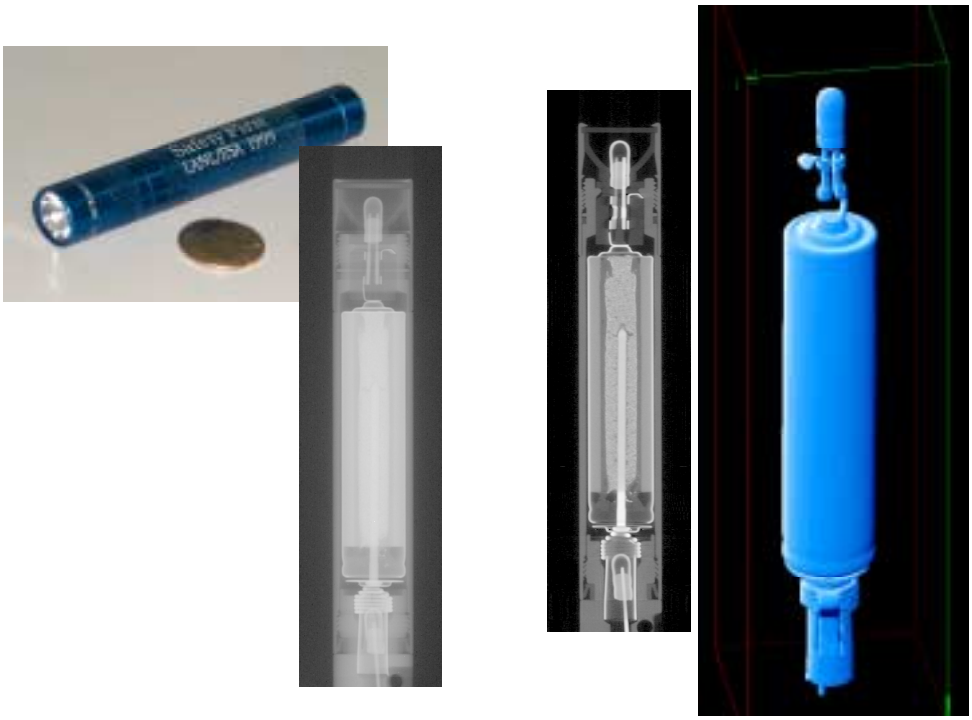

Figure 53: Mag light data set. A picture and radiograph (left), and a slice from the reconstruction with an isosurface rendering (right).

4.2.4 Microsoft Mouse

The Microsoft mouse has been the defining object during the development of FlashCT. The mouse is large relative to most test objects making it difficult to simulate fan or parallel beam geometries. It contains many small parts and fine features which make it an excellent resolution test. The latest data set of the mouse is computationally difficult containing over one thousand slices of data. The results prove the abilities of the DPS. The data set was acquired using cone beam geometry and was reconstructed using the cone beam algorithm (Figure 54).

Figure 54: Microsoft mouse data set. Reconstructed by FlashCT – DPS's cone beam reconstruction algorithm. A picture (top left), a reconstructed slice (center), and a volume rendering (right) are displayed.

CHAPTER V
CONCLUSIONS

FlashCT provides an excellent solution to full-volume industrial computed tomography. The DAS is able to acquire data faster than traditional linear array CT systems and the DPS is able to process the increased volume of data using multi-processor capabilities. The FlashCT system is easy to employ and is becoming a new standard for industrial CT.

5.1 Algorithm Effectiveness

The algorithms have been proven accurate using both real and simulated data. They require a minimum amount of resources making them operable on inexpensive hardware and software. The algorithms are easily extensible allowing for future modifications without affecting the overall function of the DPS. Finally, by being slice based, they avoid the pitfalls of previous reconstruction algorithms and allow for massively parallel operation.

5.2 System Speed

Using modern PC hardware, the algorithms are able to reconstruct entire data sets in hours where previous systems took days. A cluster of PC hardware provides a near linear speed increase as the number of processors grow. The interface allows users to quickly setup multiple projects for serial reconstruction or the ability to analyze and visualize existing data in minutes. The system has reconstructed objects with datasets up to resolutions of 2000 x 2000 x 1000.

5.3 System Cost

PC hardware makes the operational overhead of the DPS minimal. No expensive architectures or operating systems are required. The DPS does not require special hardware to perform multi-processor reconstructions, so a cluster of PCs can be built inexpensively using bulk parts.

5.4 Overall Functionality

The overall functionality of the DPS has been proven through its continued use as a non-destructive testing tool at LANL. It is used for material analysis[12], defect detection, reverse engineering[14], and geometry examination. Several dedicated FlashCT systems have been installed at LANL and more are planned. The sample data sets show the range of data the software is capable of processing and only hints at the possibilities that lie ahead.

5.5 Future Improvements

Several long-range improvements are planned for the DPS. A dedicated set of C++ libraries may be created for performing the data processing, replacing IDL. The advantage would be faster processing time and less overhead. A C++ library would also make possible a port of the DPS to super computer platforms such as LANL's existing ASCI grad super computer Blue Mountain or the newly contracted 30 TerraOP, 11,968 processor ASCI machine, Q. Because the reconstructions algorithms are near linearly scalable, thousands of processors on a supercomputer could be harnessed for performing reconstructions of large datasets. Refinements of the existing algorithms are expected to take place over time as new techniques evolve. The FlashCT system hardware and software is being commercialized and is available as an off-the-shelf tool for performing full-volume industrial CT[15].

REFERENCES

[1] R.R Hubert, and J.M. Beach, "Computed Tomography (CT) Standards", *Federal Manufacturing & Technologies, Topical Report on 708024*, December 1999.

[2] A.W. Davis, M.J. Sheats, T.N. Claytor, "High Speed Data Acquisition System for Three-Dimensional X-Ray and Neutron Computed Tomography", *44th Annual SPIE Meeting*, Denver, Co. July 1999.

[3] dpiX, "Flashscan 20 Specifications," dpiX Incorporated, www.dpix.com, 1998.

[4] A.C Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*, IEEE Press, New York, 1988

[5] L.A. Feldkamp, L.C. Davis, and J.W. Kress, "Practical cone-beam algorithm," *J. Optical Societ*y, A1, 1984, P. 612-619.

[6] M.J. Sheats, D.M. Stupin, "CoBRA: Cone Beam Computed Tomography (CT) Reconstruction Code in Interactive Data Language (IDL)," *QNDE Proceedings Volume 17A*, 1997

[7] Research Systems Incorporated, *Interactive Data Language version 5.2*, 1998.

[8] Cubix, "Dual Processor Pentium II System", Cubix Corporation, www.cubix.com, 1999

[9] Brockohoff, R.,C, Estes,G, P., Hills, C. R., ,DeMarco J., C., and Solberg T, D., "Application of MCNP to Computed Tomography in Medicine", LA-UR-96-135, 1996

[10] Hills, C. R., Brockhoff, R, C, and Estes, G. P, "CT Scan Simulation with the MCNP Monte Carlo Code"*, Medical Physics, Vol. 22, No. 9*, p. 1534, 1995.

[11] C. Hills, modeled and simulated at request, Los Alamos National Laboratory, 2000

[12] Loren W. Byers, James A. TenCate, and Paul A. Johnson, "Application of nonlinear elastic resonance spectroscopy for damage detection in concrete," *Journal of Cement and Concrete Research*, 1999.

[13] T.N. Claytor, W.L. Sievers, D.M. Stupin, M.J. Sheats, D.A. Summa, T.E. McDonald, E.H. Farnum, M.J. Schwab, "Thermal and Cold Neutron Computed Tomography at the Los Alamos Neutron Scattering Center Using an Amorphous Silicon Detector Array," *QNDE Meeting*, July 1998

[14] J. Hefele, C. Tsai, M. Sheats, T. Kelley, D. Stupin, "Accident Response – X-Ray to virtual environment," *Swiss conference on CAD/CAM*, February 22, 1999

[15] Hytec Inc., Industrial partner and commercial vendor of FlashCT, www.hytecinc.com, 2000