國立交通大學

生醫工程研究所

碩 士 論 文

應用圖形處理器技術於電腦斷層掃描重建之研究

Study CT Reconstruction Method Using GPGPU

研 究 生：高志文

指導教授：荊宇泰　教授

中 華 民 國　九 十 八 年 九 月

應用圖形處理器技術於電腦斷層掃描重建之研究
Study CT Reconstruction Method Using GPGPU

研 究 生：高志文　　Student：Chih-Wen Kao

指導教授：荊宇泰　　Advisor：Yu-Tai Ching

國 立 交 通 大 學

生 醫 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Biomedical Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

September 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年九月

應用圖形處理器技術於電腦斷層掃描重建之研究

研究生：高志文　　指導教授：荊宇泰 博士

國 立 交 通 大 學

生 醫 工 程 研 究 所

摘要

電腦斷層掃描技術可以廣泛應用在許多領域，例如：醫學上的放射檢查、機場海關的檢查等等；在電腦斷層影像的重建技術中，以濾波反投影以及疊代式兩種演算方法最為著名。濾波反投影演算法的優點是計算時間快速，但是重建出來的影像會因為計算上的誤差而品質較差；疊代式演算法源自於數值分析，需要耗費大量的計算時間，但是重建出高品質的影像為其優勢。近年來，愈來愈多研究利用圖形處理器平行計算的特性，幫助我們處理大量資料的計算。因此，本篇論文中利用 NVIDIA 所開發的整合技術 CUDA，將疊代式重建法中可以平行處理的部份實作於圖形處理器中執行，在計算時間上比在 CPU 中執行快了數十倍之多；並且更應用三角函數和角公式的特性，讓演算法在圖形處理器中更有效率地執行。實驗結果中以各種模擬與實際資料，比較計算效能上的改善，以及兩種不同演算法所重建出來的影像品質。最後，本篇論文利用 QT 的開發平台，將各種 CT 重建方法整合成可以跨平台的使用者視窗介面。

Study CT Reconstruction Method Using GPGPU

Student: Chih-Wen Kao Advisor: Dr. Yu-Tai Ching

Institute of Biomedical Engineering
National Chiao Tung University

Abstract

Computerized Tomography can be applied in many fields, such as medical radiology and Customs check in airport. Both Filter Back Projection and Iterative Image Reconstruction is well-known algorithm in the scope of CT image reconstruction method. The advantage of Filter Back Projection is very fast, but the image reconstructed usually suffers from noise problem because of the inaccuracy in computing process. The Iterative Reconstruction originated in numerical methods, which can reconstruct high quality image but require heavy computation power. In recent years, more and more researches take advantage of parallel computing property in GPGPU, GPGPU help us to handle large number computation power problem. Therefore, we utilized the integrated technique named CUDA which is developed by NVIDIA Corp in this paper. We implemented the parallel computing of Iterative Reconstruction Method in GPGPU, the expense time is better than it is executed in CPU by a factor of tens. Furthermore, Sum and difference formulas in Trigonometric functions are applied in order to let algorithm execute more efficiently in GPGPU. As the experiment result, we compared the improvement of performance between CPU

and GPGPU and the quality of image between Filter Back Projection and Iterative Reconstruction Method. Finally, we utilize QT creator to integrate CT reconstruction methods in cross-platform user interface.

誌 謝

首先感謝我的指導教授荊宇泰博士，在這兩年多來的細心指導，讓我學習與成長非常多得以完成此篇碩士論文。並且感謝林志陽、謝昌煥與胡宇光三位口試委員提出許多建議與謬誤之處，讓這篇論文更加完整。

謝謝昌杰學長時常耐心地講解，在論文實作上給予我非常大的幫助。謝謝 704 實驗室的所有同學與畢業學長姐們，帶來許多歡笑與傾聽，讓我得以分擔平常時候的課業壓力，阿 Ken、秉璋、宗澤、彥淳、柟旭、千千、政宇、冠緯、崇誠、書豪、懷元、嘉樺與筱莉，謝謝你們大家。謝謝樂觀的宜庭，讓我在焦慮與失落的時候給予我無比的信心。謝謝所有關心我的朋友，謝謝大家。

最後特別感謝我的父母與家人，讓我可以無後顧之憂的完成學業，並且給我非常多的鼓勵，一直是我往前邁進的動力。

目錄

圖表目錄

第一章 緒論

1.1 簡介

電腦斷層掃描(Computerized Tomography)技術，又稱為「計算機斷層掃描」，簡稱 CT。可以廣泛應用在許多領域，例如醫學應用以及非破壞性的偵測，機場的行李檢查，海關的貨櫃檢查等等。CT 影像重建技術上使用的演算法，以濾波反投影(Filter Back Projection，簡稱 FBP)，和疊代式重建方法(Iterative Reconstruction Method)最為著名。濾波反投影重建法優點是計算時間快速，但重建出的影像會因為計算上的誤差而品質較差；而疊代式重建法須要非常大量的計算量和時間，重建出高品質的影像為其優勢。

近幾年非常熱門的圖形處理器技術(GPGPU)，本篇論文中利用該技術適合大量資料計算以及具高度平行處理能力，加速上述演算法的計算時間與效能。並且從 GPGPU 與 CPU 不同實作的方式，來探討濾波反投影和疊代式兩種重建方法的運算時間，以及電腦斷層掃描的影像品質。另外在疊代式的重建方法中，我們應用三角函數中和角性質於圖形處理器中的平行化演算法，節省計算過程所需花費的時間。

1.2 論文架構

本篇論文分為五個章節。首先第一章為序論，可以了解相關研究的發展以及我們研究的目標。第二章介紹研究相關背景以及理論基礎的文獻回顧。而第三章介紹實作方式以及實驗流程。第四章為實驗設計和實驗結果。第五章為結論以及未來展望。

第二章 背景與理論基礎

2.1 前言

電腦斷層掃描技術是一種利用 X 射線在身體不同角度進行照射；在每個 θ 角度下發射許多條 X 射線，在人體中經過不同組織，使得 X 射線的吸收率(或稱阻射率)有所不同；最後經由接收器收集該條 X 射線的衰減值，如圖 2-1[1]。將各個不同角度的衰減值訊號輸入電腦；影像重建技術(Image Reconstruction) ，即可將這些收集到的訊號，重建為應用於醫學影像診斷上的電腦斷層影像。

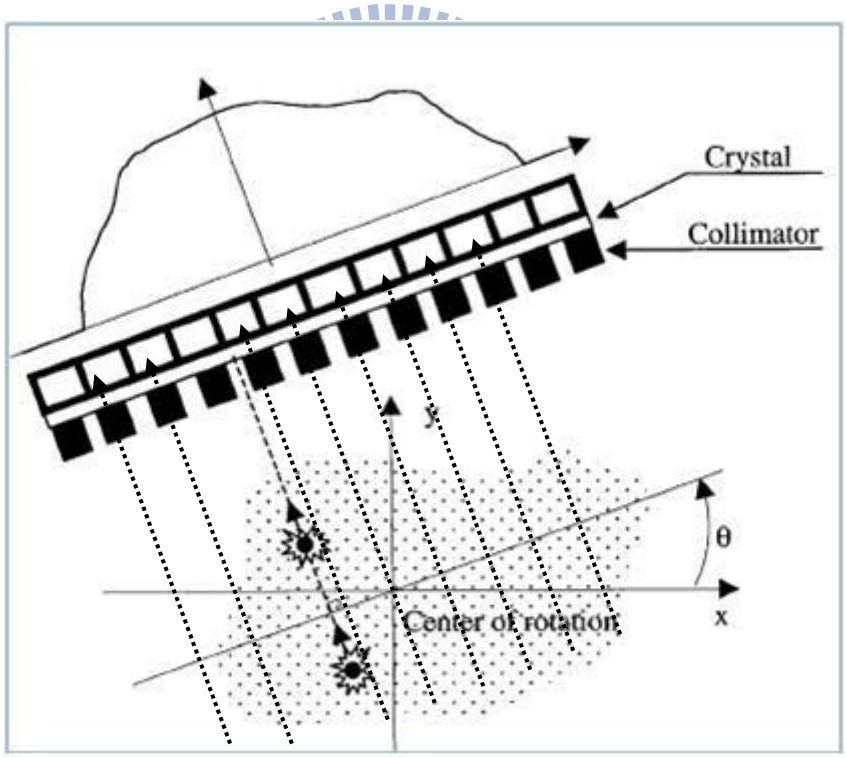


圖 2-1：電腦斷層掃描技術原理

2.2 CT(Computerized Tomography)醫學背景

自從 1895 年德國物理學家 Roentgenz 發現 X 射線以來，其在醫學影像領域的應用即漸漸開始受到世人的關注；然而受到電腦技術的侷限，真正的臨床應用則是到了二十世紀中之後才被實現。第一個商業化應用於臨床的電腦斷層系統於 1972 年正式發表，增強了人類觀察體內內部結構的能力。

數學家 Radon 早在 1917 年和 1919 年分別提出 Radon Transformation 及 Inverse Radon Transformation 公式，表現在影像處理上意義即為後來的投影變化(投影算子)。二十世紀 50 年代初期，美國神經外科醫生 Oldendorf 為了克服普通 X-Ray 成像組織結構重疊僞影(artifact)，發表了第一篇斷層掃描論文，此時才稱為電腦斷層掃描技術；1963 年，南非物理學家 Cormack 為了準確估計 X-Ray 在不同組織間的衰減率，第一次把非疊代的級數理論引入 CT 重建演算法中；1968~1972 年，英國工程師 Hounsfield 為了區別大腦的灰質和白質，製造了世界上第一台商用電腦斷層掃描機。Cormack 與 Hounsfield 在 1979 年獲得諾貝爾獎，並且在影像處理界，設置了 Hounsfield 獎。近 20 多年，CT 影像重建理論在演算法的研究一直處於極為活躍的地位，而且新進的成像方式，例如：MRI(Magnetic Resonance Image)及 PET(Positron Emission Tomography)等，在重建的計算方法上與電腦斷層掃描也很相似。

2.3 CT 影像重建技術理論基礎

在 2-1 節中提到，經由 CT scanner 接收器，可以得到所有不同攝影角度中，不同 X 射線的衰減值；我們將這些所有量測到的訊號，依照 0~π 角度排列，所得到的投影資料(Projection data)，稱為 Sinogram。CT Image Reconstruction 的目的是藉由 Sinogram 作為 Input data，重建出物體橫切面的影像。垂直方向為每一個角度所量測到訊號值；水平方向為該 θ 角度下每一條 X 射線，在接收器不同位置所量測的訊號值。圖 2-2 即為 CT Scanner 所量測出來的 Sinogram，尺寸為 256x256，即在 0~π 角度內進行照射投影 256 個角度，每次投影包含 256 條 X 射線。

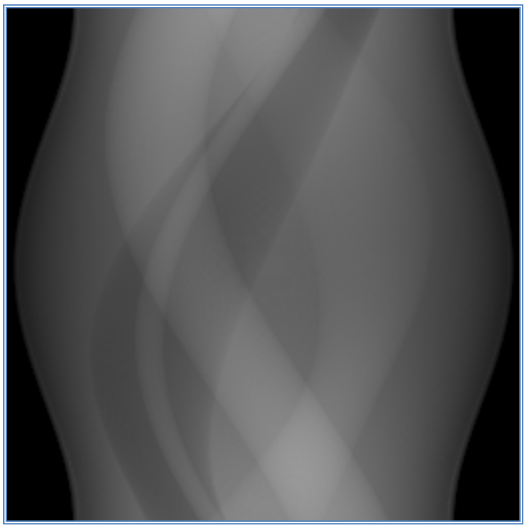


圖 2-2：Sinogram 資料

2.3.1 FBP (Filter Back Projection)

Filter Back Projection algorithm是由Bracewell和Riddle[2]所提出，為CT影像重建技術中常用的演算法。其主要的理論基礎包括Radon transform、Filtered Projection以及 Back Projection三個部分。

由圖2-3，我們利用數學來描述電腦斷層掃描系統，以二維函數$f(x,y)$來描述物體(object)。定義θ角度中，接收器(detector)上任一個位置s的方向向量為：

$$s = x\cos\theta + y\sin\theta \tag{1}$$

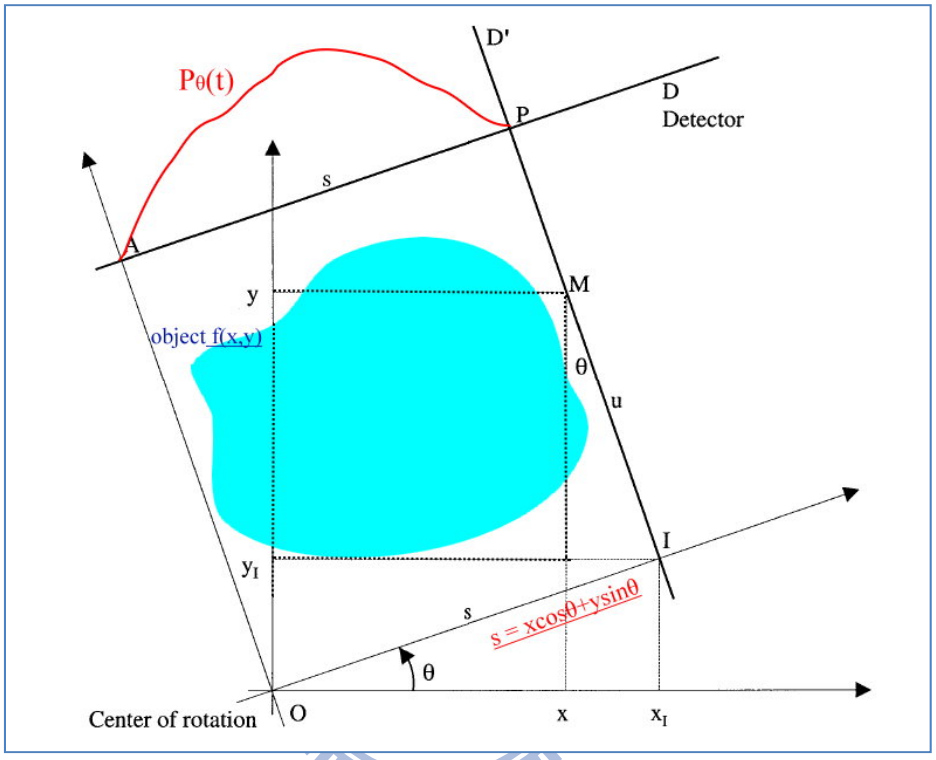


圖 2-3：以數學描述電腦斷層掃描系統

加上(θ,s)參數，則可以利用Radon transform定義投影算子(Projection operator)；以線性積分$P_\theta(s)$代表(θ,s)參數對應的X射線，對物體$f(x,y)$作投影所得到衰減係數的總合。

$$P_\theta(s) = \int_{(\theta,s)} f(x,y)ds \tag{2}$$

使用delta function，可以將公式(2)改寫如下：

$$P_\theta(s) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\delta(x\cos\theta + y\sin\theta - s)\,dxdy \tag{3}$$

公式(3)即是物體$f(x,y)$的Radon transform(亦稱為Sinogram)，所以收集每個θ角度的$P_\theta(s)$值即為該角度所有射線的投影資料。

Filter Back Projection algorithm是利用傅立葉截面理論(Fourier Slice Theorem)與每一個θ角度的$P_\theta(s)$投影資料求出封閉解(closed form solution)；可以得到延著此θ角度方向的投影值，進而累加成為物體截面的影像。因此首先定義θ角度的投影資料及其傅立葉轉換(Fourier Transform)公式如下：

$$S_\theta(\omega) = \int_{-\infty}^{\infty} P_\theta(s)\, e^{-j2\pi\omega t} ds \tag{4}$$

其中ω是radians，經由公式推導後可以得到下列公式：

$$\begin{cases} \text{Filter Projection algorithm:} \quad Q_t(s) = \int_{-\infty}^{\infty} S_\theta(\omega)\,|\omega| e^{-j2\pi\omega t} d\omega & (5) \\ \text{Back Projection algorithm:} \quad f(x,y) = \int_{0}^{\infty} Q_\theta(x\cos\theta + y\sin\theta) d\theta & (6) \end{cases}$$

而在FBP重建演算法的傅立葉轉換步驟中，我們通常選擇由Cooley與Tukey所提出的快速傅立葉轉換[3] (Fast Fourier Transform, FFT)，是基於一種稱為連續加倍法(Successive doubling method)所推演出來的演算方法；將原本時間複雜度為$O(N^2)$降低為O(NlogN)。

2.3.2 Iterative Image Reconstruction

CT Reconstruction 另一種主要方法為疊代式影像重建法(Iterative Image Reconstruction)，於 1982 年由貝爾實驗室(Bell lab)的數學家謝普(L. A. Shepp)和瓦第 (Y. Vardi) 首先提出。此方法主要的原理源自於數值分析(Numerical Methods)，泛指一個固定運算公式的重複演算法則，演算過程會不斷重複直到特定的條件被滿足而停止。此重建方法在核子醫學的應用上為，假設物體截面是由一個未知的矩陣所組成；由投影資料(Sinogram)的數據建立未知向量的線性方程式，給予不同向量估計方法，經過多次疊代求出未知的矩陣即為物體的截面影像。

首先建立 Iterative Image Reconstruction 的代數數學模型[4]，如圖 2-4。圖中以網狀方格矩陣涵蓋所有區域，包括進行照射的物體(object)。我們令 x_j 是矩陣中第 j 個元素的 Intensity 值，j=0,…,N-1，N 為全部矩陣元素的個數。b_i 代表第 θ 角度中某一條 X 射線的投影資料，i=0,...,M-1，M 為全部射線個數的總和；等於 Sinogram 的像素個數。最後，令矩陣 $A = (a_{ij})$ 表示物體像素 x_j 對於投影資料 b_i 的映像(mapping)；其 a_{ij} 表示第 j 個像素對第 i 條射線積分的貢獻值(constribution)或稱為權重值(weight)。

因此，利用此數學模型，可以建立一個線性系統如下公式：

$$Ax = b \tag{7}$$

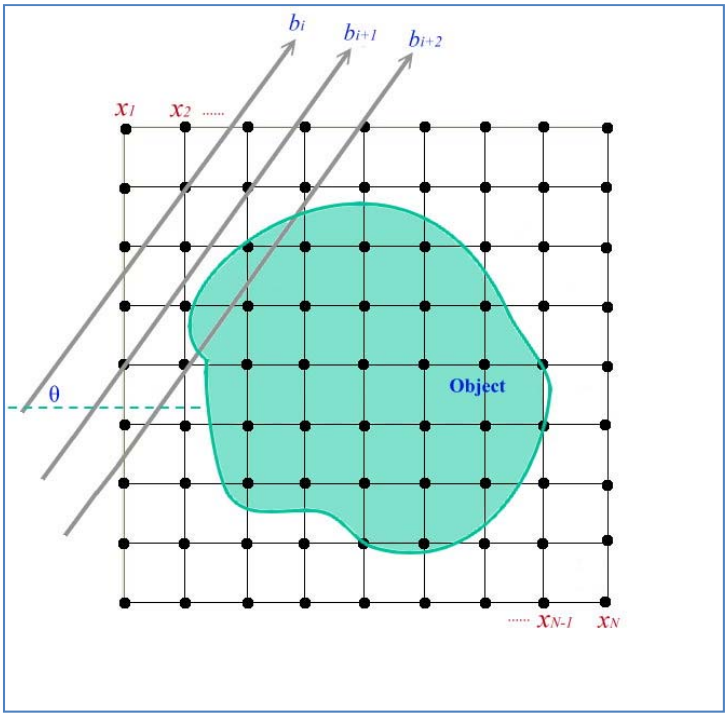


圖 2-4：Iterative Image Reconstruction 幾何數學模型

Iterative Image reconstruction 主要的原理是以公式(7)的線性系統，利用連續地估計逼近物體影像像素值 x_j 的解。首先對於未知像素值的影像作投影估計，所得到的投影資料稱之為估計投影資料(Re-projection data)；並且將 Re-Projection data 與 CT Scanner 測量到的實際投影資料(Measured projection data)相減，當作下一次疊代演算法中對物體影像像素值 x_j 的修正值(Correction term)。最後將影像像素與此修正值作一次反投影(Back projection)修正；反覆上述過程，直到物體截面影像被清楚呈現。

基於此Iterative Image reconstruction的基本原理；在代數系統$Ax = b$中，依照不同的權重矩陣(矩陣 A)的決策方式，發展出兩種不同的經典疊代演算法；ART(Algebraic Reconstruction Techniques) 與 SART(Simultaneous Algebraic Reconstruction Technique)。

ART 演算法中，矩陣 A 中每個元素 a_{ij} 的權重值，通常被簡單的以 0 或是 1 來決策；若 X 射線有經過物體影像的像素 x_j，則此像素對於此條射線權重值 a_{ij} 即被設為 1；反之，若 X 射線沒有經過物體影像的像素 x_j，則此像素對於此條射線權重值 a_{ij} 即被設為 0。以這種較簡單的決策方法，大幅降低矩陣 A 中元素 a_{ij} 的計算時間，推導出疊代計算之 ART 公式如下：

$$x_j^{(k+1)} = x_j^{(k)} + \frac{bi - \sum_{j=1}^{N} x_{ji}^{(k)}}{H_i} \quad ,j=1,...,N\text{-}1,\ i=1,...,M\text{-}1 \tag{8}$$

公式(8)中，分母 H_i 為$\sum_{k=1}^{N-1} a_{ik}^2$的簡化，意思是針對第 i 條 X 射線，累加所有影像像素點對此第 i 條射線的權重值 a_{ik}；又 ART 的權重值分配的決策方法如上述非 0 即 1，因此分母 H_i 意為第 i 條 X 射線經過幾個影像像素點。$x_j^{(k)}$ 與 $x_j^{(k+1)}$ 表示本次(k)與下一次(k+1)疊代物體影像的像素值。b_i 為第 i 條 X 射線所量測到的實際投影資料(Measured projection data)。$\sum_{j=1}^{N} x_{ji}^{(k)}$ 表示為估計投影資料，對物體影像作 Re-projection；即針對第 i 條 X 射線，累加所有此 X 射線經過影像的像素值 x_j，成為 Re-projection data。將 Measured projection data 與 Re-projection data 相減，成為疊代過程中影像像素需要的修正值(Correction term)。

由 ART 演算法重建出來的影像，往往有 salt and pepper 雜訊以及偽影；是因為對於權重值 a_{ij} 的決策方式簡化為 1 或是 0。當射線只有經過某像素的邊緣時，此像素點對射線的權重值若仍設為 1，則此像素點對於此條射線的貢獻就被放大了。為了抑制 ART 演算法重建後影像的雜訊，可以在修正值反投影(Back projection)於物體影像像素之前，先乘以一個 relatxation 係數 α，該係數 α 小於 1 並且隨著疊代次數增加而減少；利用這個方法可以使得疊代過程的收斂速度較快，進而得到品質較好的物體截面影像。

SART(Simultaneous Algebraic Reconstruction Technique)[5]由 Andersen 和 Kak 所提出，對於 ART 演算方法的影像品質有顯著的改善。SART 與 ART 演算法有三個最大的不同點如下。

a. 傳統 ART 演算法是基於 pixel-basis 的概念，即每一個影像像素對於射線的貢獻值為 1 或 0，造成影像會有不連續與偽影的情形。因此在 SART 演算法中，採用 bilinear element，一條射線所經過影像像素的網格區域，會被鄰近四個像素點所貢獻；而貢獻的比例是依射線與此四個像素點的距離成反比，當作 SART 演算法中影像像素對於射線的權重值 a_{ij}，即為雙線性內插法(bilinear interpolation)。因此，利用 blinear element 計算權重值比傳統 pixel-basis 還要精準許多，藉此改善傳統 ART 方法中影像不連續的問題。

b. 傳統 ART 演算法的疊代方式為 ray-by-ray，意為當一條射線的修正值被計算出來後，就更新影像像素值 $x_j^{(k)}$ 一次。而 SART 演算法中，同步(Simultaneous)意思是計算完所有射線後才將此修正值更新於影像像素值 $x_j^{(k)}$。

c. SART 演算法中，我們利用漢明視窗(longituidinal Hamming Window)，如圖 2-5，來改善重建後影像的品質。在反投影的步驟中我們將計算出的修正值再乘以 Hamming window 調整，藉此強調影像矩陣中間(實際物體部分)；相對於矩陣四周(空白部分)給予較多的修正值比例。

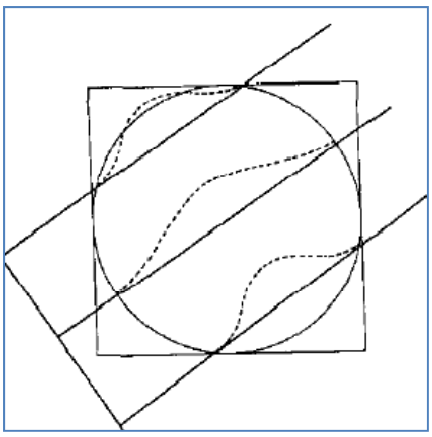


圖 2-5：漢明視窗 (longituidinal Hamming Window)

因此，經由上述改良傳統 ART 演算法機制，SART 演算方法公式如下：

$$x_j^{(k+1)} = x_j^{(k)} + \frac{\sum_{i=1}^{M} a_{ij} \frac{(b_i - \widehat{b_i})}{\sum_{j=1}^{N} a_{ij}}}{\sum_{i=1}^{M} a_{ij}} \quad , j=1,...,N\text{-}1,\ i=1,...,M\text{-}1 \qquad (9)$$

公式(9)中針對第 i 條射線，$\widehat{b_i}$是估計投影資料(Re-projection data)，b_i是實際投影資料(Measured projection data)，$(b_i - \widehat{b_i})$為修正值(Correction term)。

SART 演算方法中 blinear element 是利用每條不同射線，在網狀方格中建立取樣點(Sampling point)；並且以數學中雙線性內插法(bilinear interpolation)，計算矩陣 A 中每一個像素對於每一條射線的貢獻值a_{ij}，針對第 i 條射線，累加所有影像像素點對此條射線的貢獻值，成為$\sum_{j=1}^{N} a_{ij}$項；將修正值$(b_i - \widehat{b_i})$除以此項再乘以a_{ij}，表示修正值中反投影分配給影像像素 Intensity 值的比例。

由此我們可知，影像像素 x_j 的 Intensity 值來自所有射線分配的總和，得到$\sum_{i=1}^{M} a_{ij} \frac{(b_i - \widehat{b_i})}{\sum_{j=1}^{N} a_{ij}}$項，此即為 SART 演算法中同步地(Simultaneous)的概念。最後將影像像素 x_j 的 Intensity 總和，除以該像素給予所有射線 j 的貢獻值總和$\sum_{i=1}^{M} a_{ij}$，可以得到一次疊代(iteration)過程，所有影像像素的 Intensity 值。

2.3.3 OS (ordered subset)-SART

OS-SART 演算法[6]，1994 年由 H. Malcolm Hudson 和 Richard S. Larkin 所提出；最主要的目的是加快 SART 演算法的收斂(Converge)速度，使其疊代(iteration)次數減少，達到重建速度上增進的目的。OS-SART 演算法為在疊代運算之前，先行前處理 Sinogram 中所有角度的投影資料(Projection data)；前處理包括排列(Ordered)與分組(Subset)。

SART 演算法中，依 θ 角度由小到大順序，計算該 θ 角度的所有射線；而 OS-SART 演算法中的排列(Ordered)是將 Sinogram 中所有角度重新排列，一般最常用的方法為隨機排列(Random ordered)；再將隨機排列後所有角度的投影資料作分組，分組方法是利用 Nonoverlapping subsets 如下：

Set S_1={1,2,...,T}, S_2={T+1,...,2T},..., S_n={(n-1)T+1,...,nT}

因此 OS-SART 演算法可以表示如公式(10)，L 為 Subset 的個數：

$$x_j^{(k,t+1)} = x_j^{(k,t)} + \left(\frac{\sum_{l \in S_t} a_{ij} \frac{(b_i - \widehat{b_i})}{\sum_{j=1}^{N} a_{lj}}}{\sum_{l=1}^{M} a_{ij}} \right), \; t=1,...,L, \; j=1,...,N \qquad (10)$$

我們可以發現公式(10)與 SART 演算法公式(9)中不同在於，當執行完一個 Subset 中的所有射線後，即反投影分配影像像素 x_j 的 intensity 值。因此，當計算下一個 subset 的 Re-projection 過程中，所參考的影像像素 x_j 即與前一個 subset 所參考的不同。

2.4 CUDA 開發環境

現代的圖形處理器已經具有高度的可程式化能力，並且通常具有相當高的記憶體頻寬，以及大量的執行單元，例如：NVIDIA GeForce 8800GTX 的記憶體頻寬超過 50GB/s，與 128 個 stream processors，時脈(clock rate)為 1.35GHz。因此近幾年來，開始出現利用圖形處理器來幫助一些可以平行計算工作的想法。

Compute Unified Device Architecture(CUDA)[7]是 NVIDIA 的 GPGPU 模型，使用 C 語言為基礎，在 CUDA 的架構下，一個程式分為兩個部份：host 端與 device 端。Host 端為在 CPU 上執行的部份；而 device 端則是在圖形處理器(GPU)上執行的部份。Device 端的程式又稱為 kernel。通常在 host 端程式會將資料準備好後，複製到顯示卡的記憶體中，再由圖形處理器(GPU)執行 device 端程式，完成後再由 host 端程式將結果從顯示卡的記憶體中取回。

在 CUDA 的程式架構中，主要還是由 CPU 來執行，當遇到了資料可以平行化處理的部分，要把程式編譯成 device 能執行的程式，交由 device 執行，此程式叫做「kernel」。在實際在運作時，CUDA 會產生許多在 device 上執行的 thread，每一個 thread 擁有自己的索引(index)，我們可以利用不同的索引編號，去計算資料中不同的部份並執行 kernel 程式。

數個 thread 可以組成一個 block。一個 block 中的 thread 能存取同一塊共用的記憶體，而且可以快速進行同步的動作。執行相同程式的數個 block，可以組成 grid。在 CUDA 架構下要注意的是，不同的執行單位，允許可以存取與共用的記憶體皆不相同，registers 和 local memory 是以 thread 為單

位來使用的；而 shared memory 則是存在於 block，讓每一個 thread 共用。上面這三種，都是在 chip 上的記憶體，相較之下，速度會比屬於 DRAM 的記憶體來的快。Grid 中的 global, constant, texture 這 三種 memory 可以讓不同的 block 中的 thread 一起使用。此外，這三種記憶體則是屬於 DRAM 上的記憶體，可以在同一個程式的不同 kernel 中，持續的存在、使用；而他們之間的差異，在於最佳化的方式不同。像 constant 和 texture 因為對於 thread 是唯獨的，所以實際上會有快取的機制，可以用來加速讀取。，如圖 2-6。

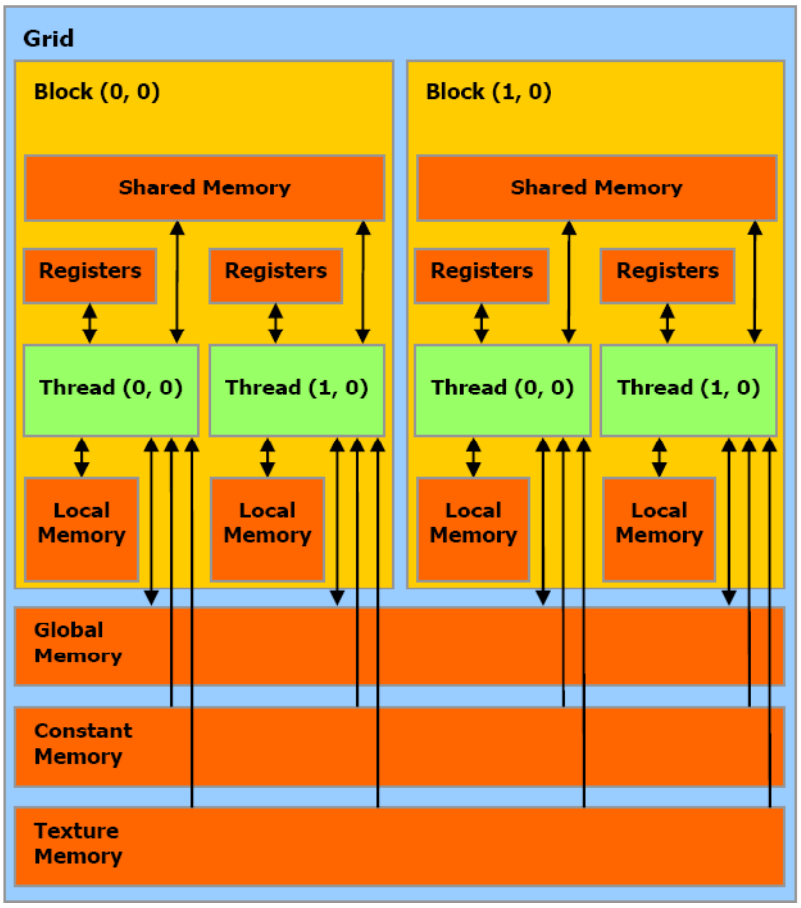


圖 2-6：CUDA 執行單位與記憶體關係之架構圖

2.5 QT 開發環境

QT[8]是一個跨平台的應用程式架構，由 Haavard Nord 和 Eirik Chambe-Engm 於 1994 年開發完成並創立 QT 軟體公司，為完全物件導向設計，易於擴充並且允許元件開發，可以開發一般應用程式以及使用者介面(GUI)，並且能夠很快速地部署於各種不同的桌面與嵌入式作業系統中，如圖 2-7，可以發現 QT 支援 C++與 Java，並且於跨平台支援包含以下作業系統：Microsoft Windows, Mac OS, Linux/X11, Embedded Linux 等。

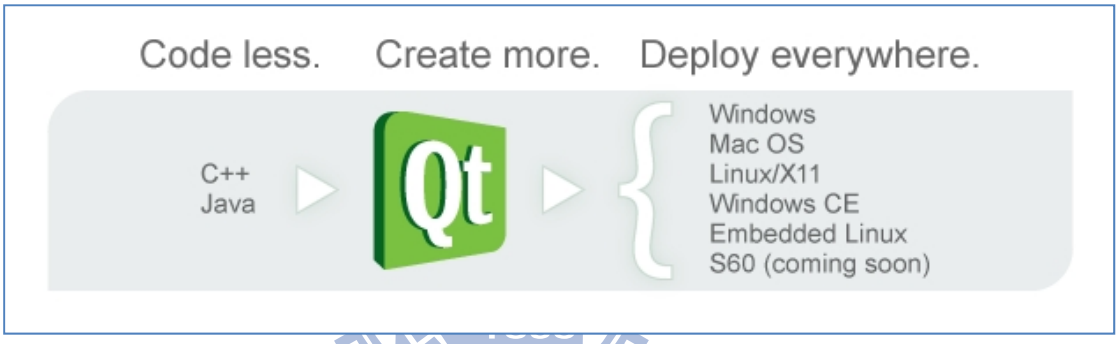


圖 2-7：QT 開發平台架構圖

在使用者介面(GUI)程式開發上，QT 利用訊號與槽(signal/slot)機制取代傳統的 callback 來連結物件之間的溝通。所以當操作事件發生時，物件會發出一個訊號，而槽為一個函式(function)即可以接受特定訊號並且執行本身設定的動作，而訊號與槽則透過靜態方式來連結(connect)。

第三章 實作方法

3.1 實驗簡介

在本章，我們利用第二章中所提到的理論基礎，實作不同電腦斷層掃描影像的重建方法，FBP、SART 與 OS-SART。並且利用 CUDA 的開發平台，實作於圖形處理器中執行。原本的影像重建演算法需要大量運算時間，我們從中探討演算法中可以平行處理的部分；設計實作於圖形處理器上的演算法，再藉由圖形處理器高度平行處理的能力加快演算法執行所需的時間。最後，我們利用 QT 開發平台，整合上述影像重建演算法，開發出可以跨平台與擁有使用者介面的電腦斷層掃描重建工具。

3.2 Iterative Reconstruction Method Implementation

Iterative Image Reconstruction 在實作上分為三大部份[9]，估計投影資料(Re-projection)、計算修正值(Correction term)與反投影(Back projection)。

由公式(9)，我們假設角度 0 ~ π 之間，CT Scanner 對物體影像進行 p 次投影，每個投影角度皆包含 q 條 X 射線；因此輸入資料(Sinogram)的尺寸為 $p \times q$，全部射線數目 $M = p \times q$，目標輸出資料的物體影像像素尺寸 $N = q \times q$。

3.2.1 SART

我們實作 SART 重建演算法如下步驟：

a. 初始化(Initization)：

設定物體影像矩陣中每元素 x_j 的 Intensity 值皆為 0，作為第一次的估計；並且將疊代索引(iteration index) k 設為 0，表示第零次疊代。

b. 估計投影資料(Re-projection)：

以 Ray_i 表示第 θ 角度中的某一條射線來說明，我們計算出此條射線與影像矩陣的進入點(Entry point)與單位向量(Uint vector)，並且每間隔單位長度 1 作一次取樣(Sampling)；如圖 3-1 中藍色點即為射線對影像矩陣的取樣點。

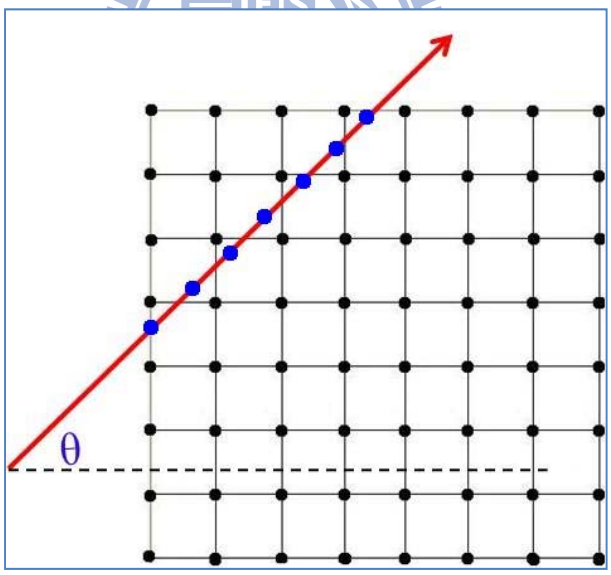


圖 3-1：SART 演算法 Re-projection 過程

對於每個取樣點(Sampling point)，利用數學上 bilinear interpolation 的計算方式，如圖 3-2；假設取樣點座標為(a,b)，則此取樣點取樣到的 Intensity 值如公式(11)：

$$\{P_1 \times (1-a)b + P_2 \times ab + P_3 \times (1-a)(1-b) + P_4 \times a(1-b)\} \quad (11)$$

按照此不同比例取得鄰近四個像素點的 Intensity 值；累加此條射線上所有取樣點的 Intensity 值的總和，成為公式(9)中的$\hat{b}_i$項為估計投影資料。

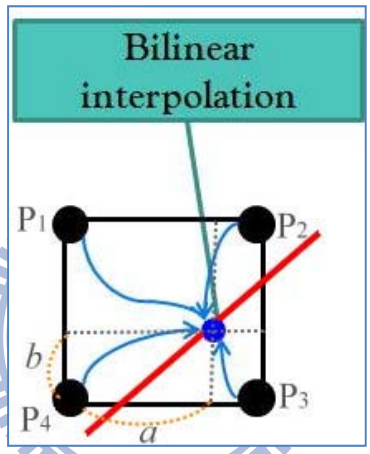


圖 3-2：bilinear interpolation in Re-projection

c. 比較(Comparsion)：

Sinogram 中第 i 條射線的實際投影資料(Measured projection data) b_i，與估計投影資料$\hat{b}_i$相減即可得到修正值(Correction term)。

d. 反投影(Back projection)：

反投影(Back projection)是將第 i 條射線計算出的修正值$(b_i - \widehat{b_i})$，反向分配回此條射線經過的像素點中，以 Ray_i 表示第 θ 角度中的某一條射線來說明；我們計算出此條射線與物體影像矩陣的進入點(Entry point)與單位向量(Uint vector)，並且以每間隔單位長度 1 作一次取樣(Sampling)；如圖 3-3 中藍色點即為射線對影像矩陣的取樣點(Sampling point)。

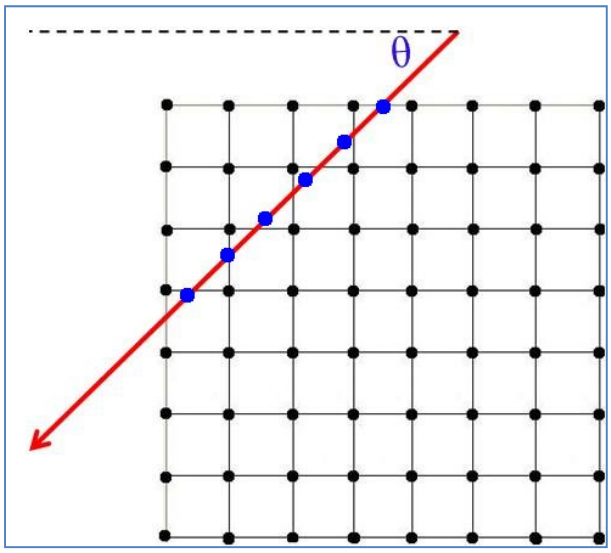


圖 3-3：SART 演算法 Back projection 過程

接下來，將修正值$(b_i - \widehat{b_i})$除以此條射線上所有取樣點的個數，可以得到每一個取樣點的部份修正值(sub-Correction)。對於每一個取樣點，我們必須將此 sub-Correction 利用數學上 bilinear interpolation 的計算方式，分配回影像中作為像素點的 Intensity 值。假設取樣點座標為(a,b)，分配 Intenstiy 給此取樣點鄰近的 bilinear element 計算方式如公式(12)：

$$P_1 = \text{sub\_Correction} \times (1-a)b$$

$$P_2 = \text{sub\_Correction} \times ab \qquad (12)$$

$$P_3 = \text{sub\_Correction} \times (1-a)(1-b)$$

$$P_4 = \text{sub\_Correction} \times a(1-b)$$

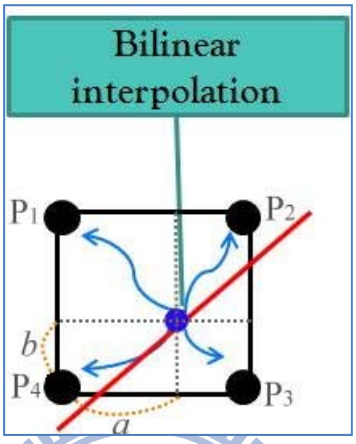


圖 3-4：bilinear interpolation in Back-projection

e. 更新參考矩陣(Update)：

在實作上，對於物體我們必須分為兩個影像矩陣，其一是 Re-projection 過程所參考的矩陣(Reference matrix)；另一個則是 Back projection 過程分配修正值的物體影像矩陣，以上述 b, c, d 三個演算方法反覆計算全部的射線之後，將物體的影像矩陣更新(Update)到 Re-projection 所參考的矩陣，如圖 3-5。

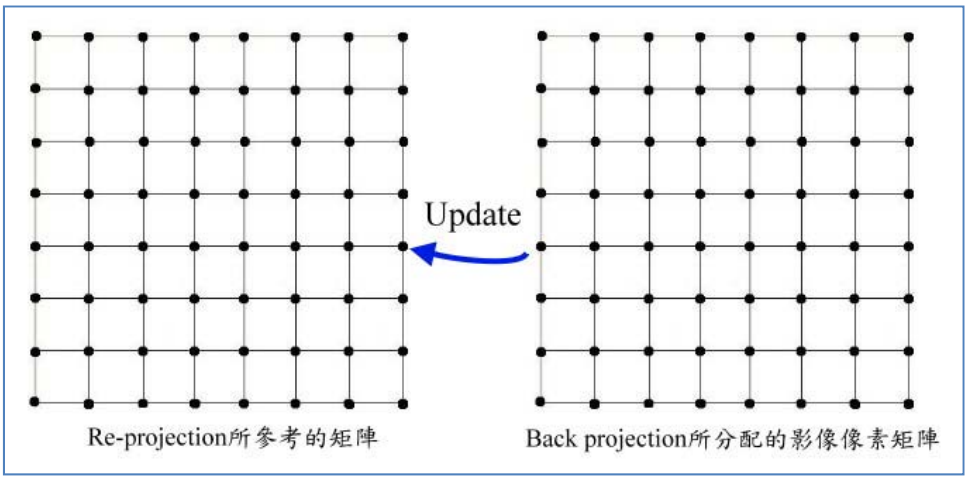


圖 3-5：更新參考矩陣過程(Update refernece matrix)

SART algorithm 流程圖如圖 3-6：

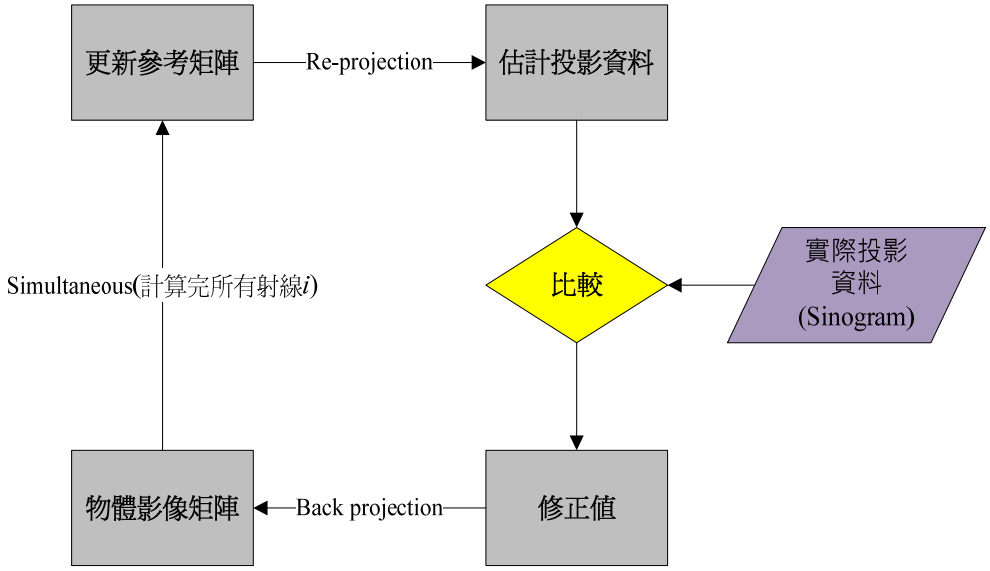


圖 3-6：SART 演算法流程圖

3.2.2 OS-SART

OS-SART 演算法目的在於增進疊代過程的收斂速度，我們將實際投影資料(Sinogram)依投影角度以隨機(Radomize)的方法作排列，再以 Nonoverlapping subsets 方法分成各個 subset。而 OS-SART 重建演算法同 3.2.1 SART 中 a.到 d.的步驟；稍有不同為 e.更新參考矩陣步驟中，OS-SART 演算法不是計算完所有射線才同步(Simultaneous)更新，而是計算完一組 subset 中所有射線後，即更新 Re-projection 步驟中所參考的矩陣。

我們實作 OS-SART 如圖 3-7 流程：

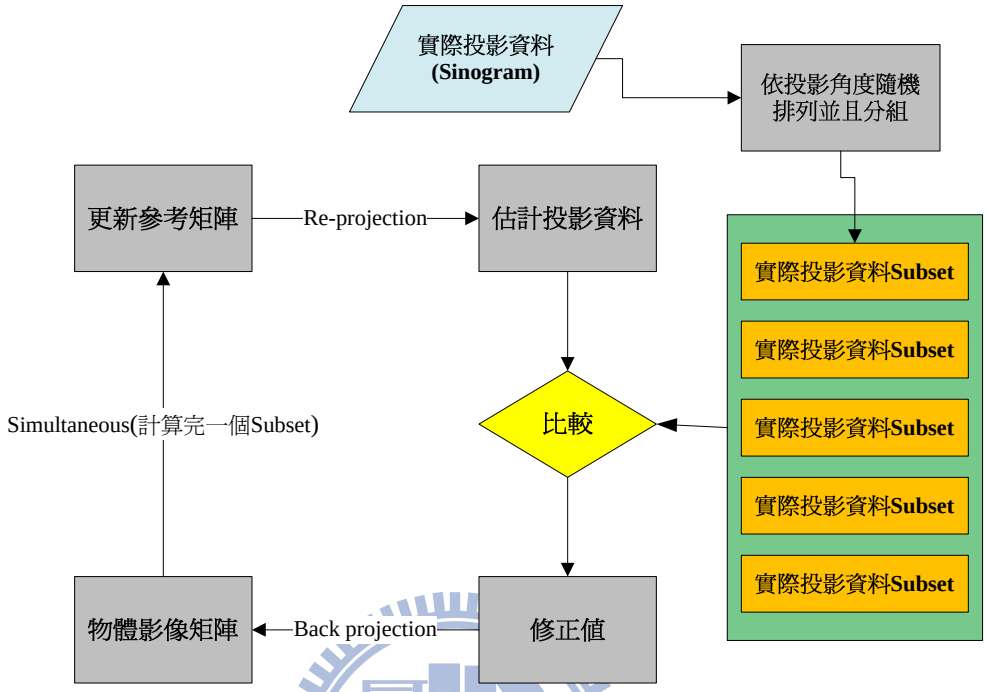


圖 3-7：OS-SART 演算法流程圖

3.3 FBP Reconstruction Method Implementation

Filter Back Projection演算法實作方法[10]流程如下：我們假設角度0～π之間，CT Scanner總共投影次數為p次不同角度的投影，我們可以得到θ_i角度的投影資料$P_{\theta_i}(t)$，利用公式(4)執行Filter Projection演算法共N次：

a. 將投影資料$P_{\theta_i}(t)$作一維快速傅立葉轉換到頻率域，成為$S_\theta(\omega)$。

b. 在頻率域下，利用Ramp Filter乘以$S_\theta(\omega)$進行濾波成為$S_\theta(\omega)|\omega|$，避免重建後剝蝕的情形。

c. 將$S_\theta(\omega)|\omega|$作一維反快速傅立葉轉換回空間域，成為$Q_\theta(t)$。

接下來利用公式(5)執行Back Projection演算法共N次：

a. 由物體(object)上座標(x,y)以及投影角度θ_i，計算出相對於θ_i角度座標系統的新位置，$t_j = x\cos\theta_i + y\sin\theta_i$，$i = 0,\ldots, p\text{-}1$。

b. 累加所有 $Q_{\theta_i}(t)$中，位置t_j上所得到的$Q_{\theta_i}(t_j)$值，物體中每個座標位置(x,y)，可以得到該θ_i角度反投影的線性衰減值。

FBP(Filter Back Projection)演算法完整的重建流程如圖3-8:

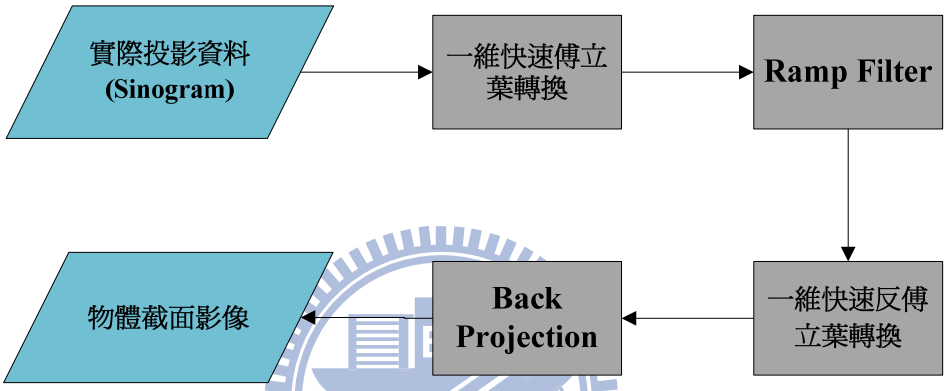


圖 3-8：Filter Back Projection 演算法流程圖

3.4 Reconstruction Method Using GPU (CUDA)

疊代式的影像重建演算法 SART 與 OS-SART 計算過程需要非常龐大的計算量，所以利用 CPU 在實作上非常耗時；而且當投影資料(Sinogram)角度和射線數目愈多，則需要花費的時間就愈龐大。

因此我們藉由圖形處理器(GPGPU)平行處理與適合大量資料的運算能力，來加快疊代重建過程需花費的時間。分析疊代重建演算法的步驟中，可以利用平行處理特性運算包含兩個部分[11]：估計投影資料(Re-projection)與反投影(Back projection)。

3.4.1 Parallel Re-projection Computing Method

在 Re-projection 步驟中，利用圖形處理器的架構下，我們平行計算θ_i角度中整排射線的取樣過程，可以同時得到該θ_i角度下全部射線的估計投影值，概念如圖 3-9。

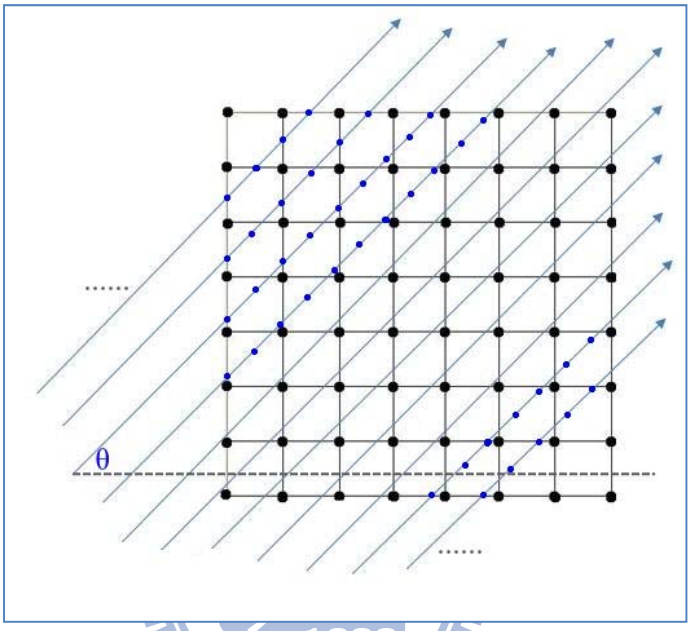


圖 3-9：平行計算應用於 Re-projection 概念圖

利用 CUDA 開發環境實作 Re-projection 平行化，必須將其參考矩陣、投影角度θ、單位向量(unit vector)與射線進入點座標由 CPU 端複製到圖形處理器端。我們配置一個執行緒(Thread)代表該角度的每一條射線，並且讓每個 Thread 執行 GPGPU 中「kernel」程式。於程式中計算每條射線的進入點(Entry point)，並且每間隔單位向量長度 1 同時對參考矩陣取樣；利用雙線性內插的計算方式，將每一個取樣點的估計投影值依序累加至$\widehat{b_i}$，即為每條射線所得到的估計投影值。存於一維陣列中，最後將此一維陣列的資料複製回 CPU 端進行運算。

由於每條射線在影像區域中必須取樣非常多次，因此在圖形處理器記憶體的管理中，我們使用 Texture Memory 來存放物體影像矩陣。CUDA 中的 Texture Memory 的特性為，限制只能讀取不能寫入、支援 Spatial Locality 存取方式下快速存取且頻寬較大、以及可以自動計算貼圖中座標位置的內插取值，相較於利用 Global Memory 來存放物體影像矩陣，改善讀取記憶體過程的時間延遲。

3.4.2 Parallel Back Projection Computing Method

在原先 Back projection 的實作中，演算方法為將該射線的修正值反向投影回物體的影像矩陣中。而我們在圖形處理器的架構下，將此步驟的演算方法設計為適合平行處理的計算方式。

3.4.1 Re-projection 可以同時得到某θ角度中所有射線的修正值。因此我們可以設計平行處理演算方法為，以影像矩陣中的每個元素為出發點，反向對修正值進行內插取值，並填回矩陣元素中成為影像像素的 Intensity 值；平行計算概念如圖 3-10。

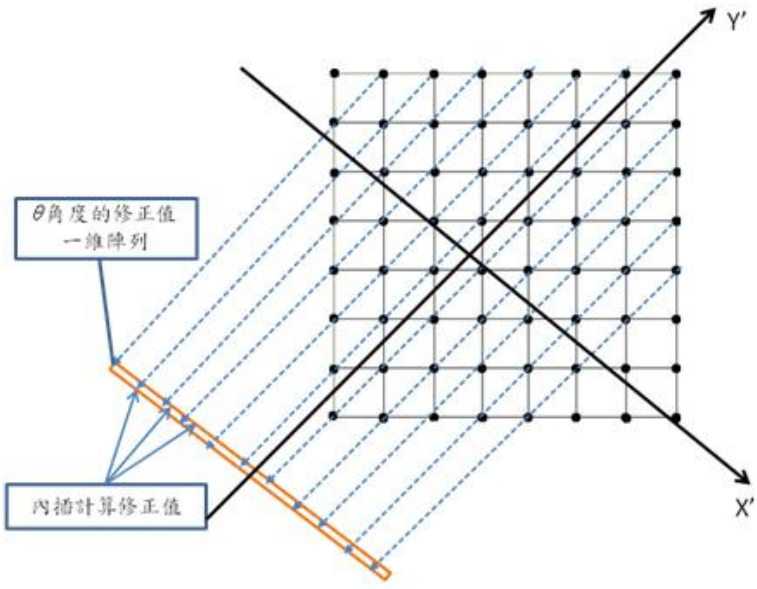


圖 3-10：平行計算應用於 Back projection 概念圖

利用 CUDA 開發環境實作 Back projection 的平行化，首先將修正值、影像矩陣與投影角度θ由 CPU 端複製到圖形處理器端。修正值矩陣的記憶體配置上，我們選擇存放於 Share Memory 中，讓同一個 block 中的所有 thread 快速的進行取值。

由於 CUDA 中 1 個 block 只能包含 512 條 thread，通常二維影像矩陣遠大於這個大小限制；因此我們必須依照影像的尺寸來配置 block 與 thread 的大小。例如：若影像矩陣尺寸為 256x256，則我們可以令二維 block 維度 16，即索引值為(0,0),(0,1),...,(16,16)，總個數為 256 個；而每個 block 裡面包含二維 thead 數目為 (256/16) × (256/16) 個，即索引為 (0,0),(0,1),...,((256/16),(256/16))。利用這個方式，我們可以計算 block 與 thread 的索引編號，來代表影像矩陣中的每個元素，概念如圖 3-11 所示。

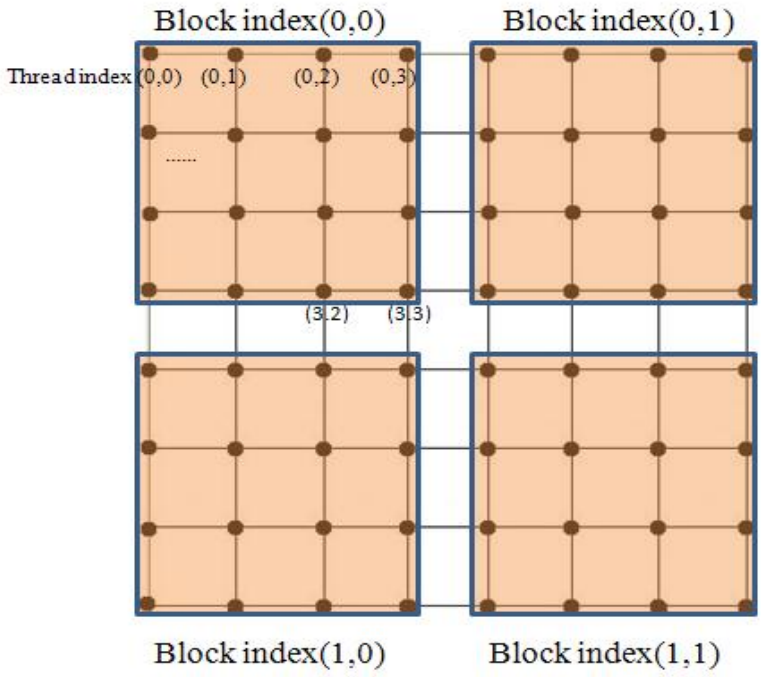


圖 3-11：CUDA 配置二維矩陣索引概念圖

GPGPU 中 Back projection 的「kernel」程式，程式中利用上述方法得到 (a,b)，用來代表影像矩陣中每個元素的索引值。將原先的座標系統依照投影角度θ旋轉，得到原座標(a,b)對於新座標系統的 x 座標，$x = a \times \cos(-\theta) + b \times \sin(-\theta)$。利用此 x 座標，即可對該角度的修正值進行內插取值。平行計算影像矩陣中所有元素(a,b)，同時得到該像素的 Intensity 值；再將此計算完畢的影像矩陣複製到 CPU 端，即完成該θ角度的 Back projection 步驟。

利用圖形處理器平行計算 Re-projection 與 Back projection，不用 ray-by-ray 對參考矩陣與影像矩陣作內插取值計算，大幅降低演算方法的時間複雜度。因此我們可以簡化 OS-SART 影像重建演算法如下：

OS-SART algorithm implementation:

假設角度 0 ~ π 之間，CT Scanner 對物體影像進行 p 次投影，且每個投影角度皆包含 q 條射線；subset_num 為 OS-SART 中 subset 的個數。

Input : sinogram[p][q];

Array : estimate_projection[q], correction_array[q], reference_image[q][q];

Output : CT_image[q][q];

for $i = 1$ to p

do estimate_projection[q] ← Re-projection using GPU(*reference_image);

for $j = 1$ to q

do correction_array[q] ← sinogram[i][j] - estimate_projection[j];

do CT_image← Back-projection using GPU(*correction_array);

if i = subset_num

then reference_image ← CT_image

3.5 Iterative Reconstruction Method Improvement

3.4 節 Iterative Image Reconstruction 演算法中 Back projection 步驟實作的過程，呼叫 GPGPU 的「kernel」程式執行一次，即平行計算該θ角度的修正值反分配;每次執行皆必須花時間將資料由CPU端複製到圖形處理器端。由此可知，提高「kernel」程式中的計算量，如果可以在一次呼叫完成較多筆資料的平行運算及減少「kernel」程式呼叫次數，可以改善 Iterative Image Reconstruction 執行於 GPGPU 中的效率。

為了達到上述目的，我們應用三角函數中和角性質於 Back projection 的過程中，推導出公式(14)如下過程：

三角函數中和角公式定義為：

$$\begin{cases}\sin(\theta+\Delta)=\sin\theta\cos\Delta+\cos\theta\sin\Delta\\ \cos(\theta+\Delta)=\cos\theta\cos\Delta-\sin\theta\sin\Delta\end{cases}$$

令X''與Y''為座標(x,y)對座標軸原點順時針旋轉$(\theta+\Delta)$度。

$$X''=x\cos(\theta+\Delta)+y\sin(\theta+\Delta)$$

$$Y''=-x\sin(\theta+\Delta)+y\cos(\theta+\Delta)$$

依照三角函數中和角公式定義展開公式並整理得到

$$X''=x(\cos\theta\cos\Delta-\sin\theta\sin\Delta)+y(\sin\theta\cos\Delta+\cos\theta\sin\Delta)$$

$$Y''=-x(\sin\theta\cos\Delta+\cos\theta\sin\Delta)+y(\cos\theta\cos\Delta-\sin\theta\sin\Delta)$$

=>

$$X''=(x\cos\theta+y\sin\theta)\cos\Delta+(-x\sin\theta+y\cos\theta)\sin\Delta$$

$$Y''=(-x\sin\theta+y\cos\theta)\cos\Delta-(x\cos\theta+y\sin\theta)\sin\Delta$$

令X'與Y'為座標(x,y)對座標軸原點順時針旋轉θ度後的座標。

$$\begin{cases} X'' = X'\cos\Delta + Y'\sin\Delta \\ Y'' = Y'\cos\Delta - X'\sin\Delta \end{cases} \tag{13}$$

當Δ角度等於π/4，代入公式(13)中可得：

$$\begin{cases} X'' = (X' + Y') \times 1/\sqrt{2} \\ Y'' = (Y' - X') \times 1/\sqrt{2} \end{cases} \tag{14}$$

座標系統上(a,b)點，順時針旋轉θ角度後(等同於原座標軸，逆時針旋轉θ角度)，成為新座標(a',b')；取Δ角度等於$\pi/4$，目的是以最精簡的運算元(Operation)，快速地得到原座標點(a,b)順時針旋轉$\theta+\pi/4$、$\theta+\pi/2$與$\theta+3\pi/4$角度後的座標(a'',b'')；如公式(14)。

我們運用三角函數和角性質於「kernel」程式，影像矩陣中的每個元素(a,b)，可以來自四個不同角度的修正值內插取值並加總；如圖 3-12 之概念。

因此，當呼叫一次 Back projection 的「kernel」程式，即不只有分配一個角度的修正值內插取值，而是分配四個角度修正值內插取值。成功地提高「kernel」程式的運算量，並且減少「kernel」程式的呼叫次數，節省圖形處理器中記憶體配置的時間，達成提昇 Back projection 此步驟計算效能的目的。

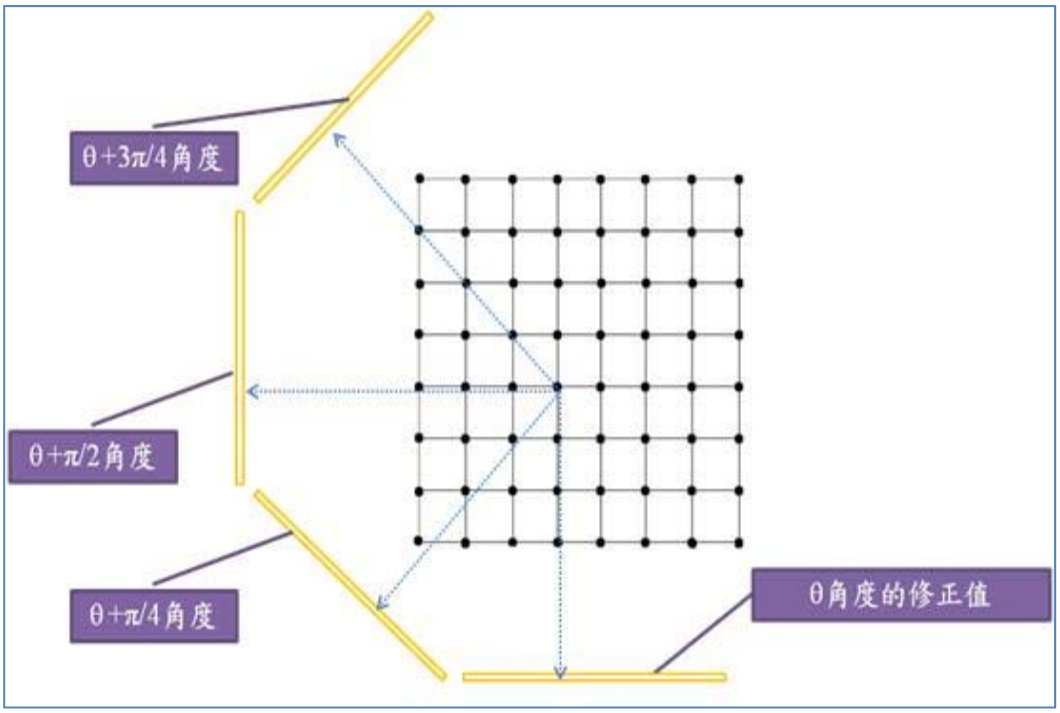


圖 3-12：Back projection 中增進運算效能概念圖

3.6 The Overiew of Reconstruction System

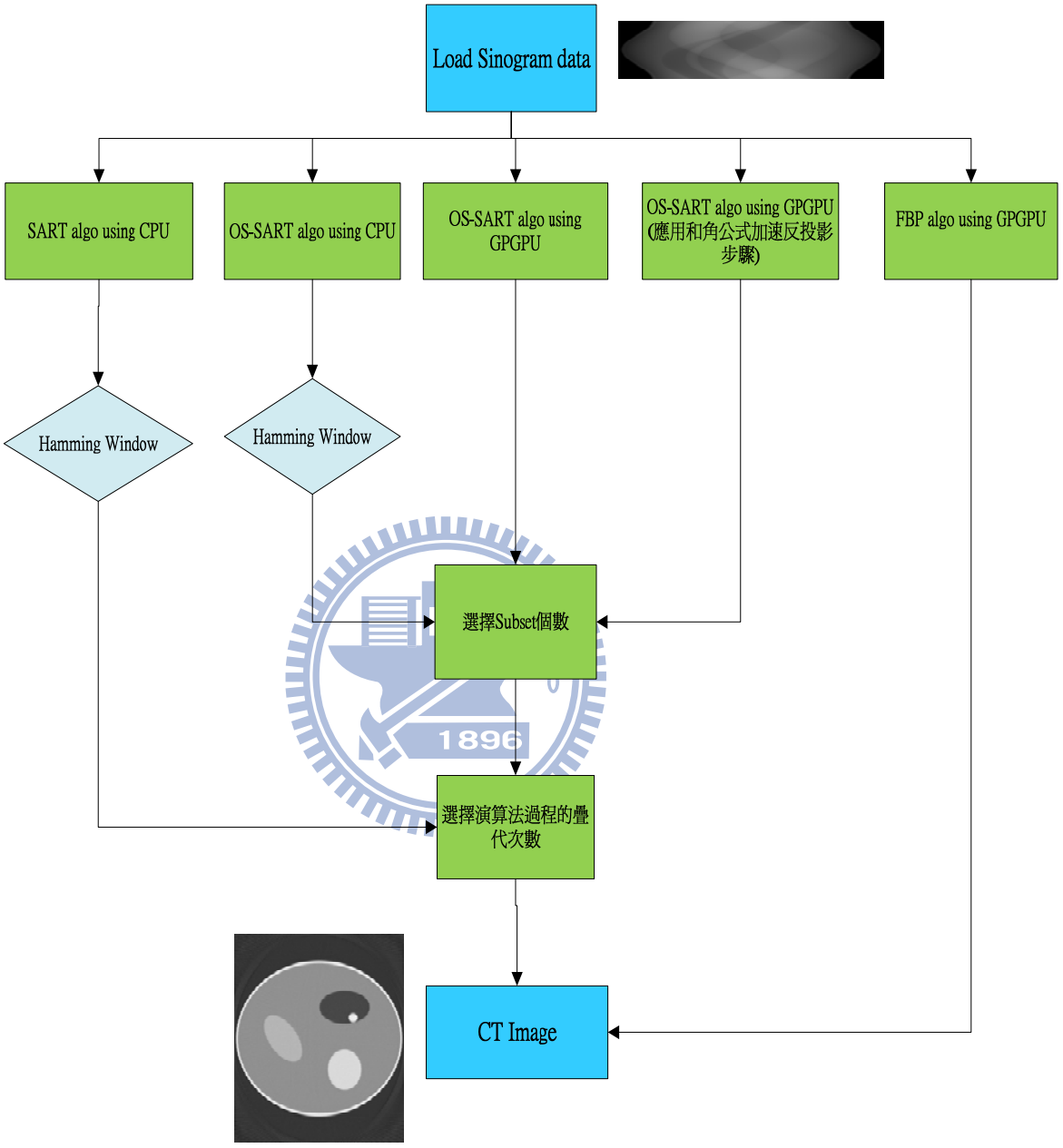


圖 3-13：CT 影像重建系統架構圖

第四章 實驗結果

4.1 實驗簡介

在本章節中，我們將第三章中各種不同 Image Reconstruction 方法，FBP、SART 與 OS-SART，以 CPU 與 GPGPU 不同的實作方式，討論重建完成後的斷層掃描影像品質以及評比其計算效能。

本論文中利用 PSNR(Peak signal-to-noise ratio)的計算方式，比較各種不同演算法所重建出的影像品質；PSNR 值愈大，表示重建後影像失真愈少。此為一種客觀上比較影像重建品質的數據。公式如下：

$$PSNR = 10 \times \log_{10}\left(\frac{255^2}{MSE}\right) \quad (15)$$

$$MSE = \frac{\sum_{i=1}^{FrameSize}(I_n - P_n)}{FrameSize} \quad (16)$$

本章節中用以實驗的資料來自於電腦模擬的模型(Phantom)與投影資料(Sinogram)，與國家同步輻射中心所取得的真實資料。

4.2 實驗環境

a. CPU：Intel Core(TM)2 CPU 4400 @ 2.00GHz

b. RAM：2GB

c. GPGPU：Nvidia GeForce 8800GTs

d. 作業系統：Microsoft Windows XP Professional Release mode

e. 開發環境與使用者介面平台：QT Creator

4.3 實驗結果

a. 模型(Phantom)影像重建實驗結果

1. 此模擬 Phantom 模型原始影像尺寸為1024×1024，如圖 4-1。並且對此 Phantom 影像進行 180 個角度模擬投影，每次投影包含 1024 條 X 射線，產生 Sinogram 資料尺寸為180×1024，如圖 4-2。

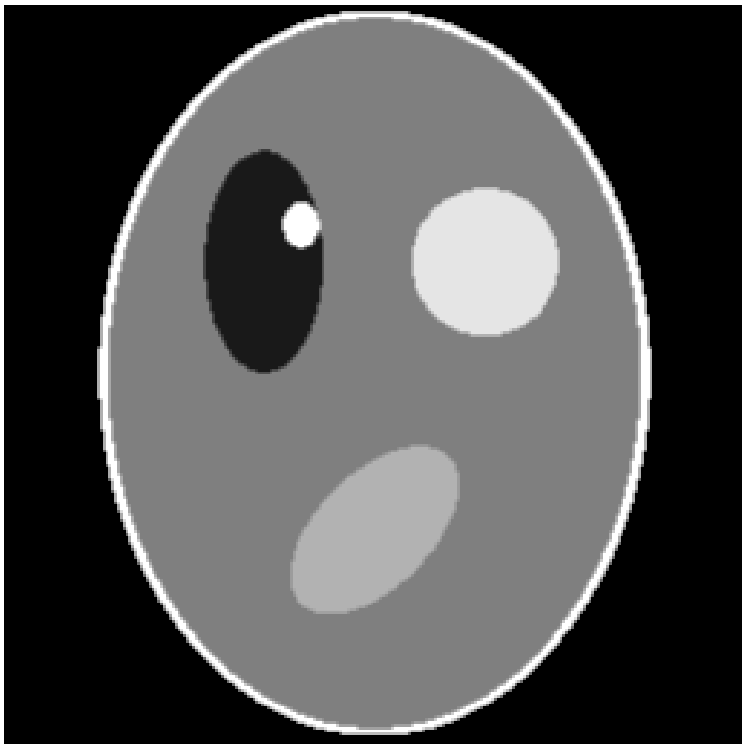


圖 4-1：Phantom 模型原始影像

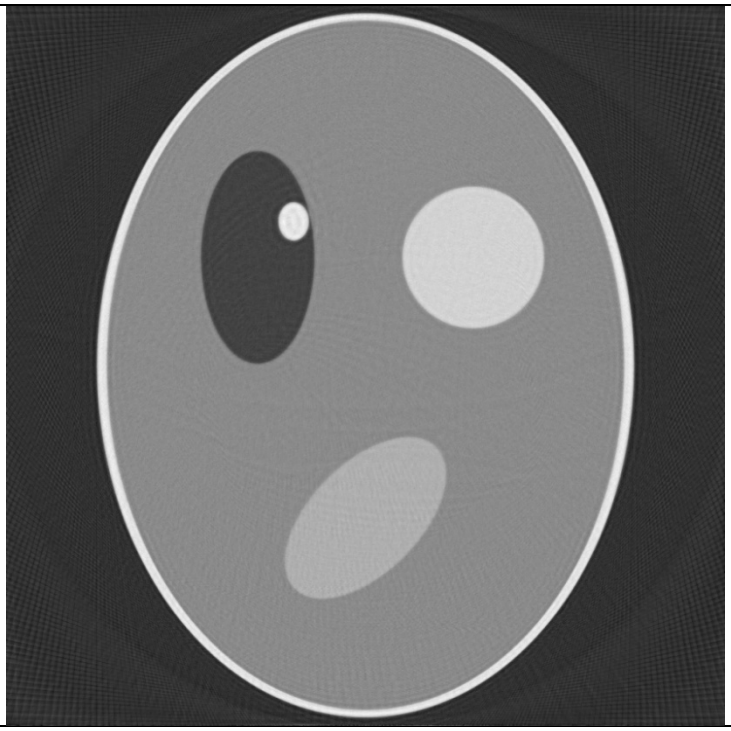
圖 4-2：Phantom 模型原始影像之 Sinogram 資料

各類影像重建演算法實驗結果以及花費時間與討論，首先是 SART 與 OS-SART 方法重建出的影像結果如圖 4-3 與 4-4。再來是 OS-SART 與 FBP 方法實作於晶片顯示卡中執行所產生的影像重建結果如圖 4-5 與 4-6。而應用三角函數和角性質於反投影的 OS-SART 結果如圖 4-7。

首先，我們比較 SART 與 OS-SART 兩種不同演算法以 CPU 實作，並且在反投影的步驟加上 Hamming window 濾波的結果；發現 OS-SART 的 15 次疊代，相較與 SART 30 次疊代，即可得到清晰 CT 影像。由此可以證明，OS-SART 演算方法的收斂速度比 SART 快上許多與更加節省多次疊代所花費時間。

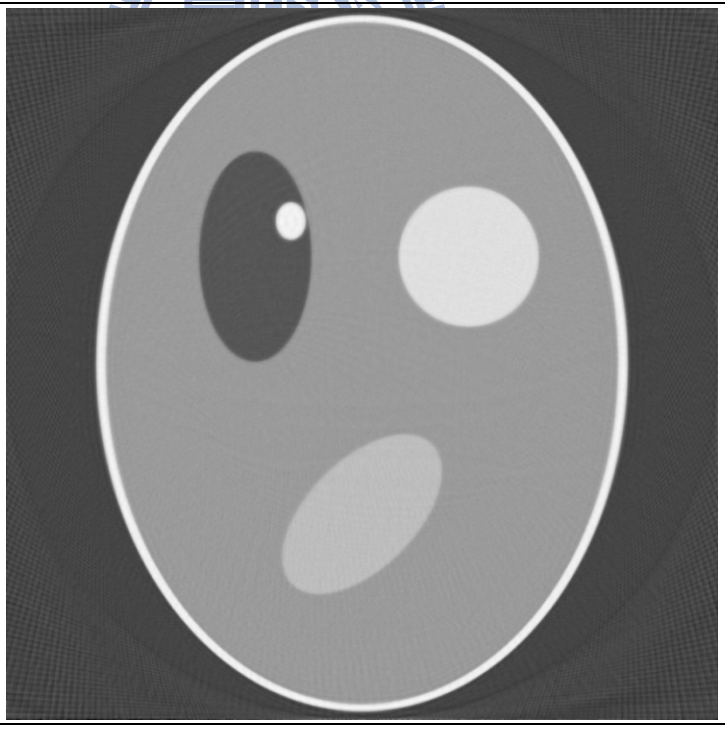
雖然 OS-SART 重建方法收斂快速，但整體演算法仍然需要耗費非常大量時間；應用圖形處理器處理大量資料的平行運算能力來加速執行速度。由實驗結果可以發現，OS-SART 一個疊代(iteration)在 CPU 中執行需要花費 131.047 秒；而應用於圖形處理器中計算執行，只需花費 3.016 秒。又我們在 3.5 章節中所提出的和角性質應用於反投影步驟，在圖形處理器中又可把執行時間加速為 1.875 秒；比起 CPU 中所花費的執行時間快了 70 倍之多。

另外，我們比較兩種不同方式的影像重建演算法，FBP 與 OS-SART。我們發現 FBP 重建演算法的優勢在於執行的時間相當快速，只需要 0.713 秒；但重建出來的影像，周圍有發散狀的雜訊和假影，以及背景顏色與原始 Phantom 影像有差異的問題，PSNR 值為 13.3658；這是由於 FBP 演算法中必須作傅立葉與反傅立葉轉換，造成資料會有誤差的情形發生。而 OS-SART 重建演算法的疊代演算方法可以改善這個問題，得到較高品質的電腦斷層影像，PSNR 值為 16.4373。



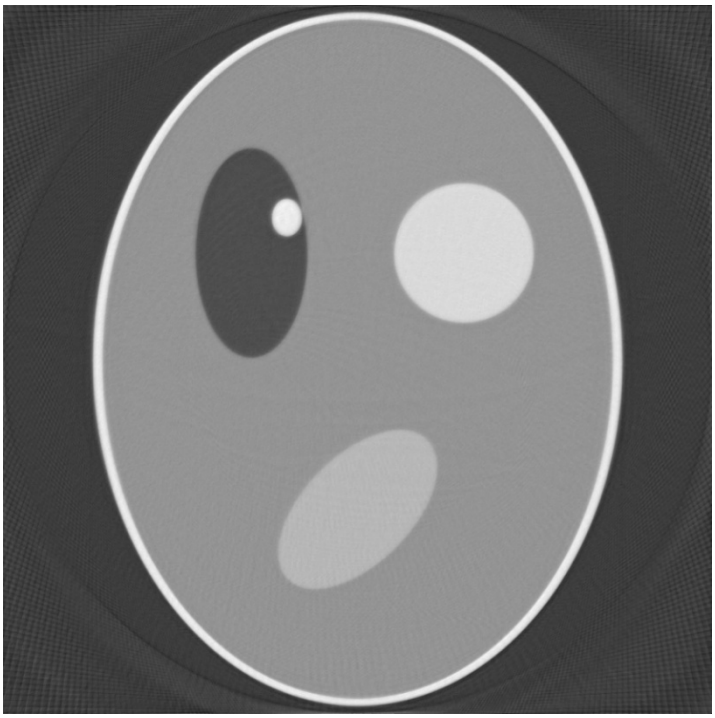
Iteration times: 60, and expand time: 10338.92sec, PSNR=16.91.

圖 4-3：SART 演算法之實驗結果(using CPU and Hamming window)



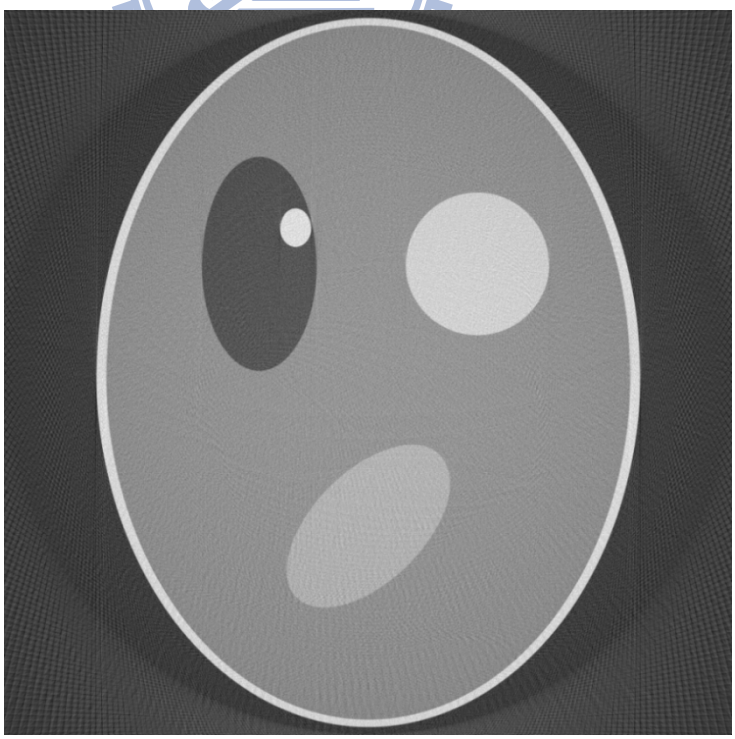
Iteration times:15, subset number: 10 and expand time: 2654.62sec, PSNR=15.107.

圖 4-4：OS-SART 演算法之實驗結果(using CPU and Hamming window)



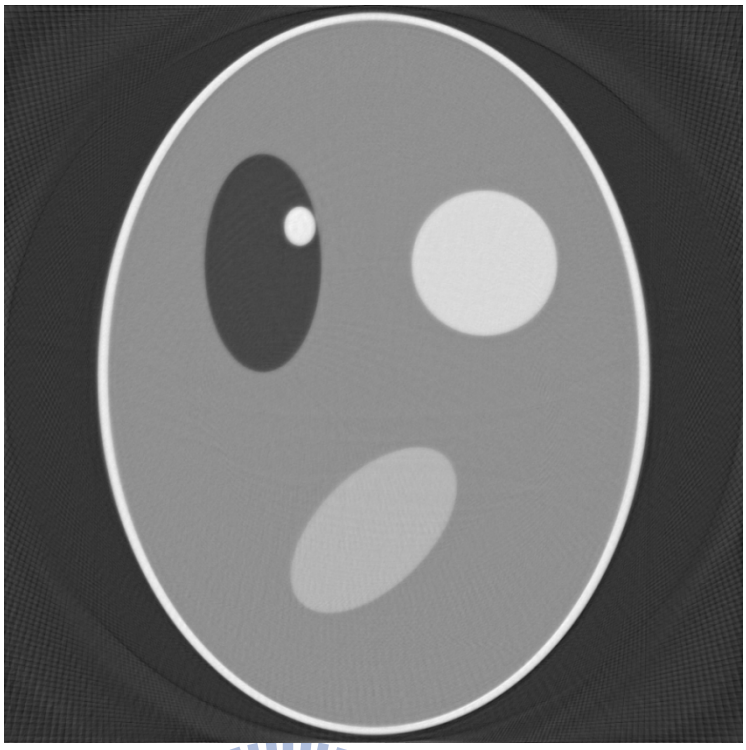
Iteration times: 15, subset number: 10 and expand time: 46.265sec, PSNR=14.1393

圖 4-5 ：OS-SART 演算法之實驗結果(using GPGPU)



Expand time: 0.713sec, PSNR=13.3658.

圖 4-6：FBP 演算法之實驗結果(using GPGPU)



Iteration times: 15, subset number: 10 and expand time: 18.312 sec, PSNR=16.4358.

圖 4-7：OS-SART 演算法之實驗結果(using GPGPU 與應用和角性質加速)

2. 第二組 Phantom 模型原始影像尺寸為512 × 512，並且對此 Phantom 模型在 180 度內進行 512 次投影，每次投影包含 512 條 X 射線；產生 Sinogram 資料尺寸為512 × 512，如圖 4-8。由這筆資料的實驗結果，OS-SART 的 PSNR 值為 21.4735，而 FBP 的 PSNR 值為 13.3028，由此我們可以知道，OS-SART 重建方法所產生的影像品質遠比 FBP 方法還要好。

圖 4-9 中由 FBP 方法所重建出的影像，三個點周圍都有明顯的發散狀雜訊問題。而圖 4-10 中由 OS-SART 方法所重建出的影像，三個點周圍沒有發散狀雜訊的情形，並且背景的顏色正確。

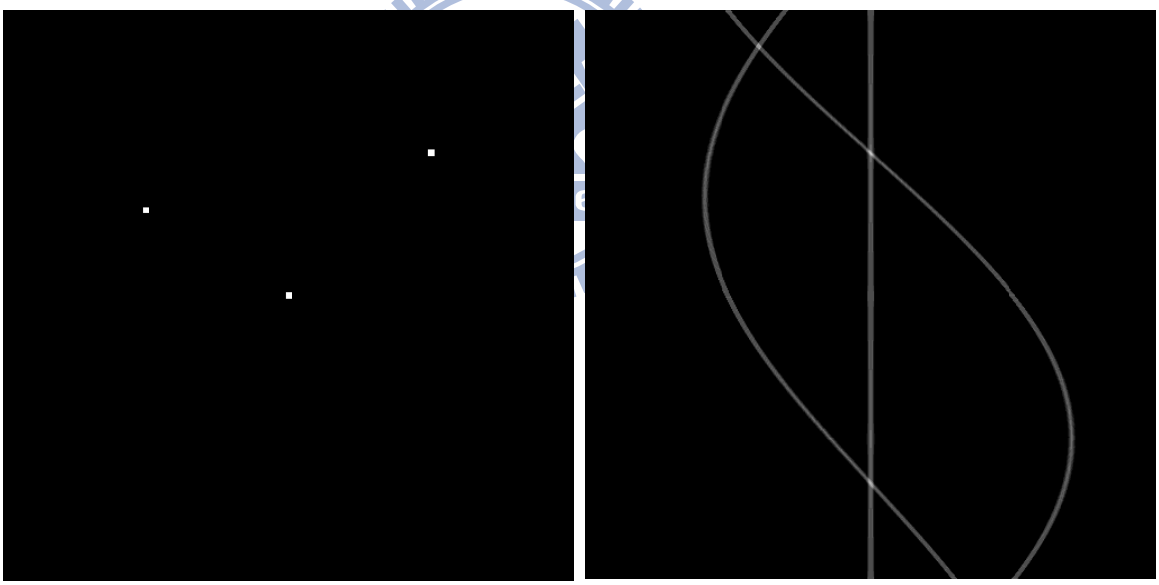


圖 4-8：Phantom 模型原始影像與其 Sinogram 資料

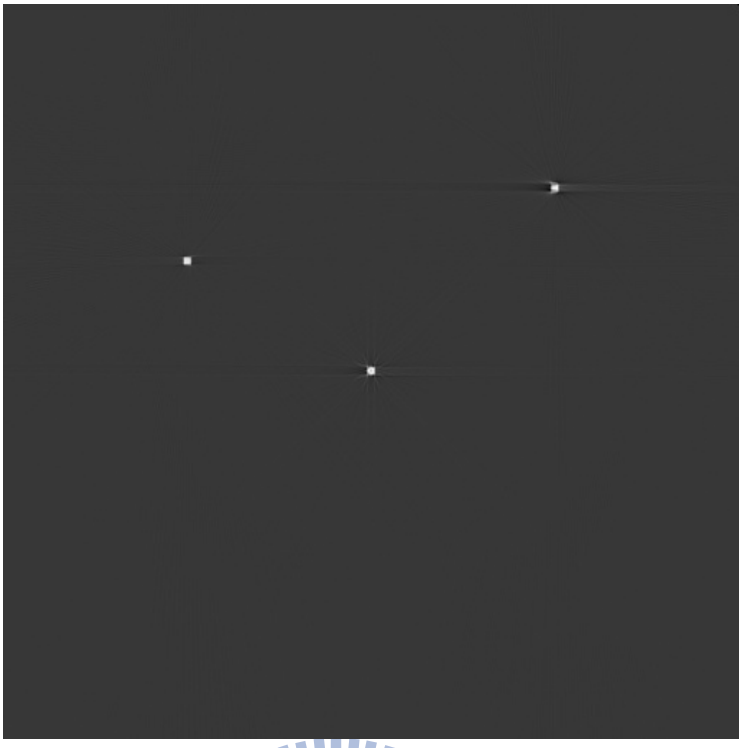


圖 4-9：FBP 演算法之實驗結果(using GPGPU), PSNR=13.3028.

圖 4-10：OS-SART 演算法之實驗結果(using GPGPU) , PSNR=21. 4735.

b. 投影資料角度不足情形下影像重建實驗結果

此實驗 Phantom 模型原始影像尺寸為1024 × 1024，如圖 4-1。對此 Phantom 影像在 180 度之間進行 180 次模擬投影，每次投影包含 1024 條 X 射線，產生 Sinogram 資料尺寸為180 × 1024。如圖 4-2。

我們實驗設計為將原本的 Sinogram，從中取部分角度的投影資料出來進行重建。間隔角度為 2 度、5 度與 10 度，而每個角度投影的射線數目沒有改變。比較 OS-SART 與 FBP 兩種重建方法的影像品質，如圖 4-11。

我們可以發現，當投影角度不足時，OS-SART 演算法重建的影像品質比 FBP 演算法還要好。當角度間隔只有兩度時，兩種方法差異不大。而當間隔到 10 度時，意為 Sinogram 原本包含 180 個角度，我們每間隔 10 度，只取 18 個角度進行影像重建。FBP 方法所重建的影像，對比度低、輪廓模糊，且有明顯發散狀雜訊問題；OS-SART 方法所重建的影像品質失真程度較低。由 PSNR 的數值觀察亦可發現 OS-SART 在缺少角度下重建品質較 FBP 好。因此 Sinogram 角度間隔大，或是投影資料不足時；我們可以採用 OS-SART 的影像重建方法來進行重建，可以得到品質較好的影像。

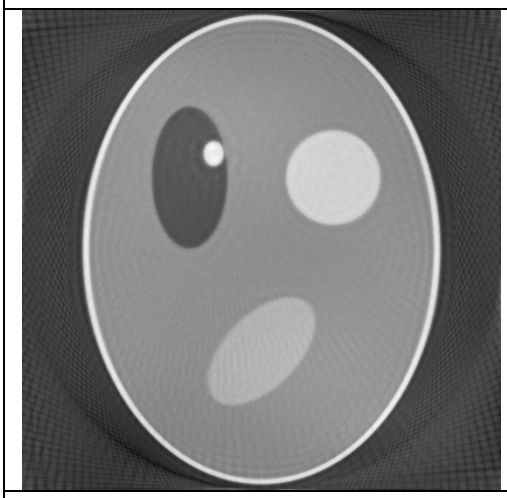
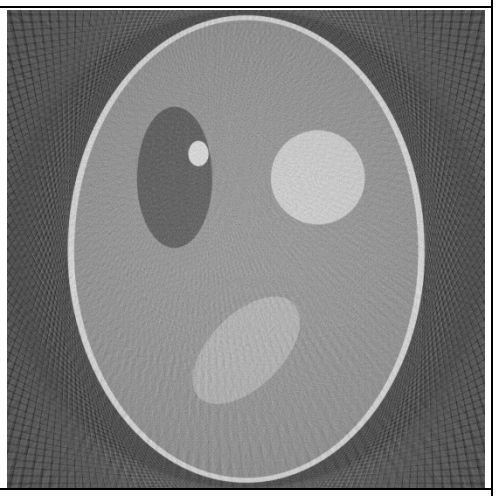
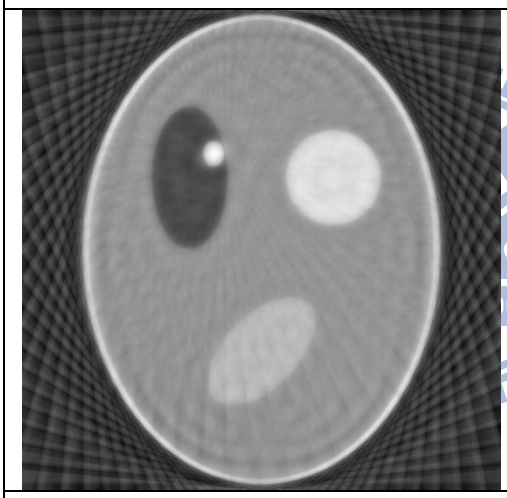
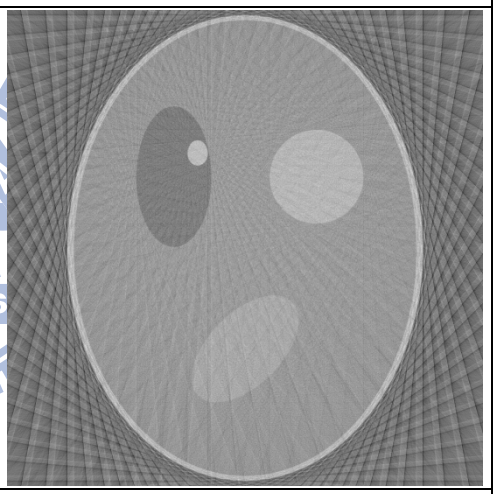
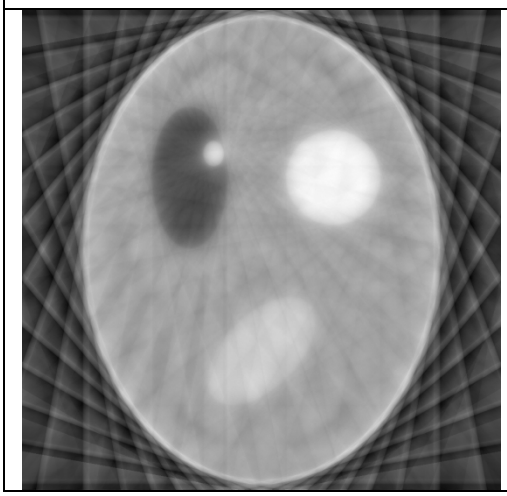
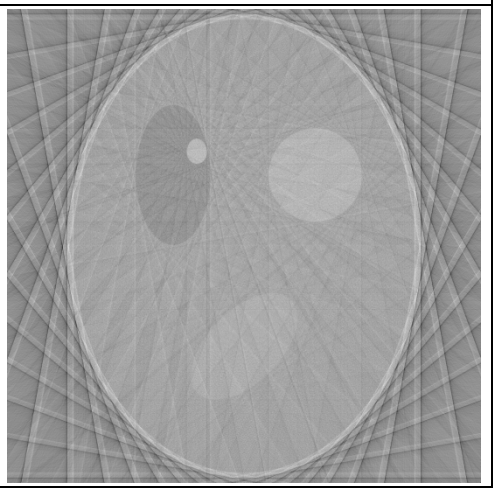
a. OS-SART(15 iterations)	b. FBP
投影資料角度間隔 2 度重建結果 PSNR: a=13.015, b=9.816	
	
投影資料角度間隔 5 度重建結果 PSNR: a=12.061, b=8.821	
	
投影資料角度間隔 10 度重建結果 PSNR: a=11.834, b=7.637	
	

圖 4-11 投影角度不足情形下影像重建實驗結果

c. 真實投影資料重建實驗結果

我們實驗中真實的投影資料來自於同步輻射中心 NanoX 實驗室，為對螢火蟲內部進行攝影所產生出來的 Sinogram 資料。如圖 4-12。尺寸為1000×1000，而 FBP 與 OS-SART 20 次 iteration 兩種不同重建方法的執行結果如圖 4-13 與圖 4-14。

OS-SART 方法所重建產生的影像，背景顏色較為正確且影像中輪廓邊緣較為平滑；相較起 FBP 方法重建產生的影像在邊緣表現上，則是雜訊較多。以執行速度而言，FBP 重建方法比 OS-SART 重建方法快；以影像品質而言，則是 OS-SART 重建方法所重建的影像品質較好。

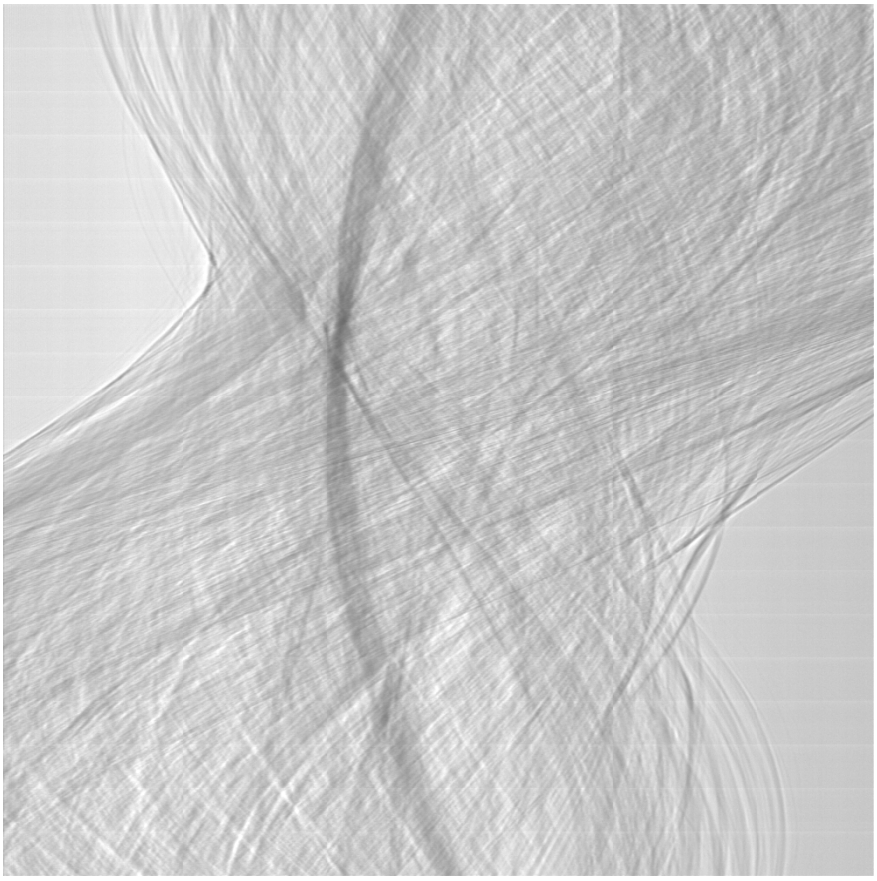


圖 4-12：實際對螢火蟲照影之投影資料

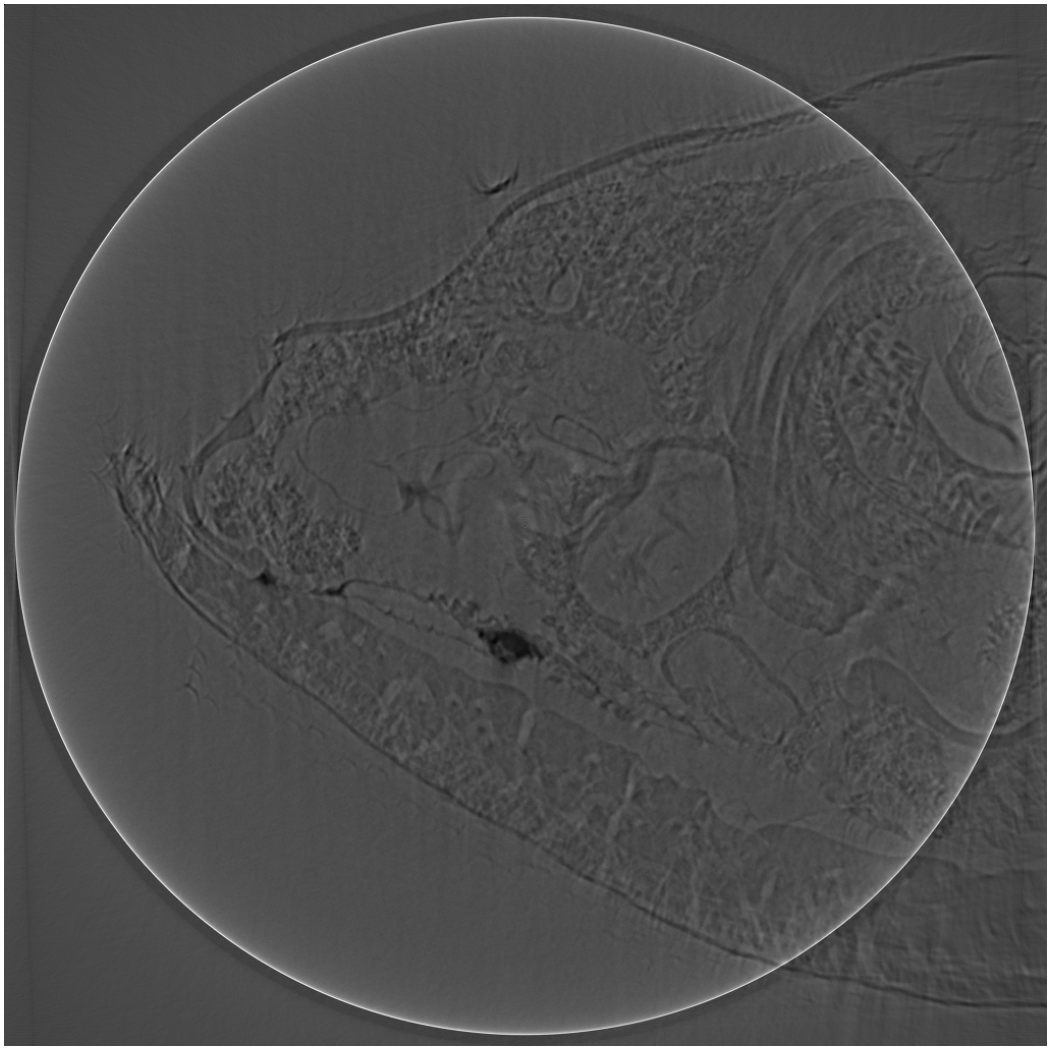


圖 4-13：FBP 演算法之實驗結果(using GPGPU)

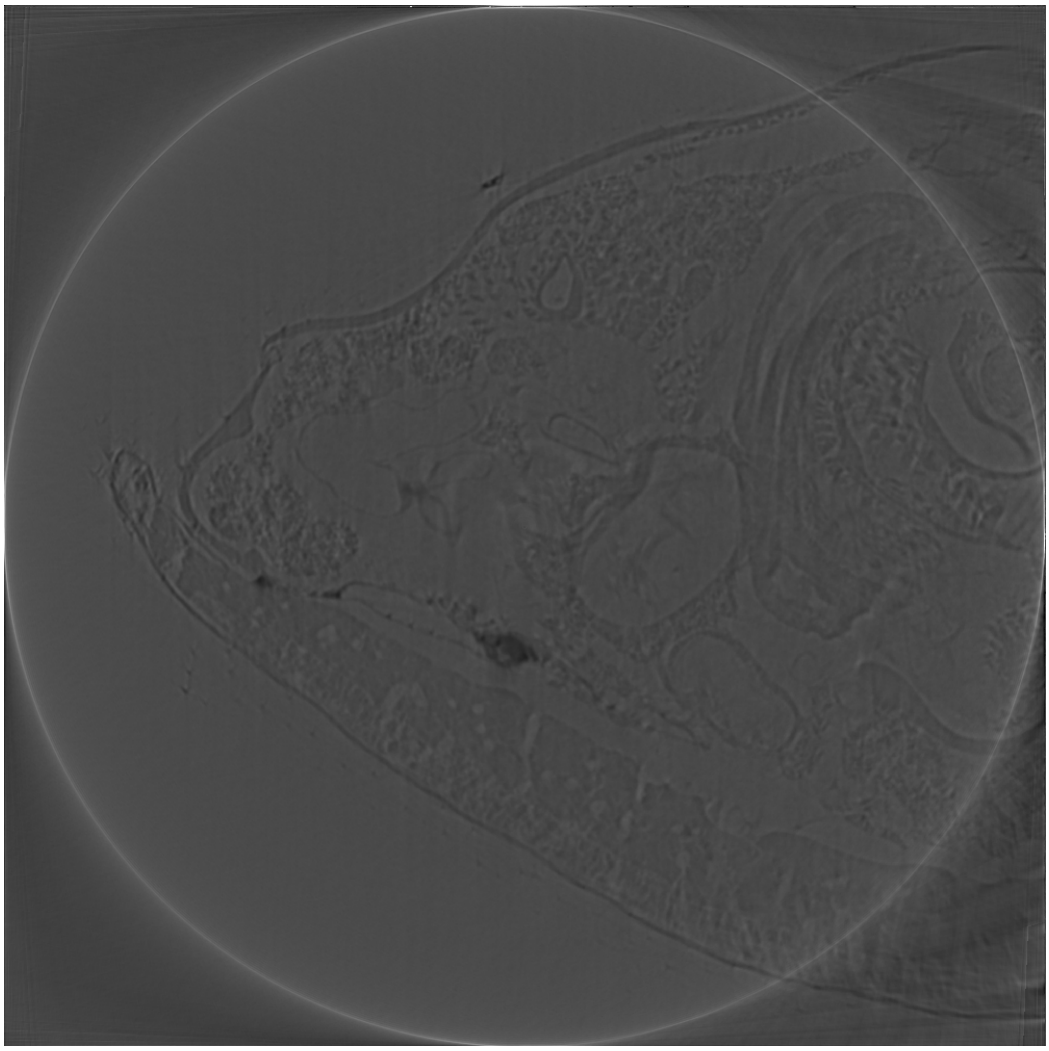


圖 4-14：OS-SART 演算法之實驗結果(using CPU)

第五章 結論與未來展望

本論文中，我們應用圖形處理器大量資料的平行處理能力於電腦斷層掃瞄重建，並且比較 FBP 與 OS-SART 兩種不同重建演算法的品質與時間，但是仍然有些在影像重建上的問題在本論文中尚未解決。

一般進行 CT 攝影時，照射物體中間部份，通常會因為該部分較厚，而使得 X 射線衰減過多，因此真實電腦斷層資料重建後，中間會比四周圍偏暗一些，此種雜訊問題稱為 hardening，藉由影像處理上濾波的技術，或是將衰減過多的投影資料作修正，可以解決此部份的問題，但是本論文中用以實驗的 Phantom 模擬 CT 攝影所產生的投影資料，並未模擬上述這些情形，以及討論解決的方式。

另外，當投影資料(Sinogram)中投影角度缺少某一大部分，例如：0~π度的投影資料，缺少了其中 30 度，在本篇論文中所提到的 OS-SART 以及 FBP 兩種演算方法都無法解決此問題，重建出的影像都會完全缺少該部分的輪廓以及投影值，而目前有相當多的研究文獻皆在討論該方面的問題。

參考文獻

[1] Philippe P. Bruyant, “Analytic and Iterative Reconstruction Algorithms in SPECT”, The Journal of Nuclear Medicine, Vol.43, no.10, October 2002.

[2] R.H. Bracewell and A.C. Riddle, “Inversion of fan beam scans in radio astronomy”, Astrophysics Journal, vol.150, pp.427-434, 1967.

[3] J. W. Cooley and J. W. Tukey, “An algorithm for machine calculation of complex Fourier Series”, Math, Comp.19, pp.297-301, 1965.

[4] Malcolm Slaney and Kak AC, “Principles of Computerized Tomographic Image”, chapter 7, pp.276-292, 1999.

[5] Andersen AH and Kak AC, “Simultaneous algebraic reconstruction technique (SART): A superior implementation of the art algorithm”, Ultrason Imaging, vol.6, pp. 81-94, Jan 1984.

[6] H. Malcolm Hudson and Richard S. Larkin, “Accelerated Image Reconstruction Using Ordered Subsets of Projection Data”, IEEE Transactions on Medical Imaging, vol.13, no.4, December 1994.

[7] “Compute Unified Device Architecture (CUDA) Programming Guile”, Nvidia Corporation, http://developer.nvidia.com/object/cuda.html, 2007.

[8] Jasmin Blanchette, Mark Summerfield, “C++ GUI Programming with Qt4 Second Edition”, Prentice Hall, 2008.

[9] Xiang Li, Ming Jiang and Ge Wang, “A numerical simulator in VC++ on PC for iterative image reconstruction”, Journal of X-ray Science and Technology, vol.11, pp.61-70, 2003.

[10] Raman Rao and Ronald Kriz, “Parallel Implementation of the Filtered Back Projection Algorithm for Tomographic Imaging”, April 1995.

[11] Fang Xu and Klaus Mueller, “On the Efficiency of Iterative Ordered Subset Reconstruction Algorithm for Acceleration on GPUs”, 2008.