CT on GPU

Giovanni Di Domenico

Università di Ferrara

GAP Meeting 2014

Circular Cone-beam Tomography

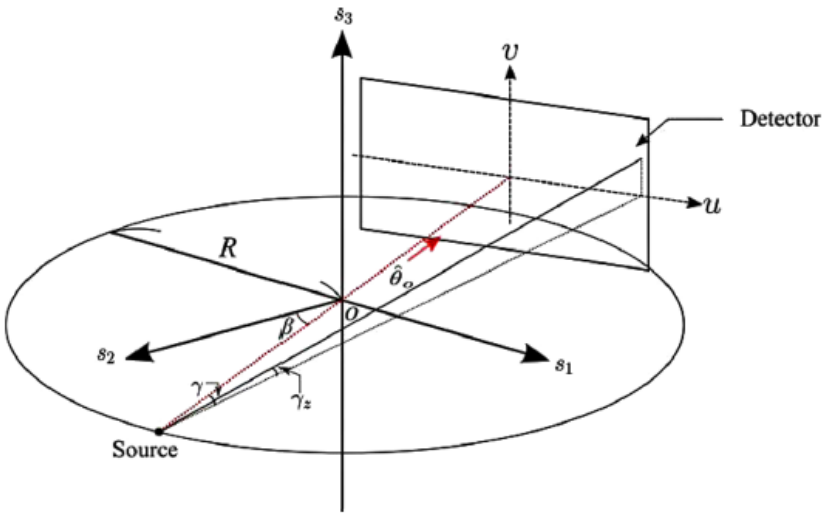

FIG. 1. A schematic of the imaging system geometry.

Il sistema misura l'integrale di linea della funzione $\mu(\mathbf{r})$

$$g_\beta(u,v) = \int_0^\infty \mu(\mathbf{r}_o(\beta) + \alpha\hat{\theta})d\alpha$$

dove

- $\mathbf{r}_o(\beta)$: è la posizione della sorgente
- $\hat{\theta}$: direzione del fascio di raggi-X

FDK algorithm - 1

L'algoritmo di Feldkamp-Davis-Kress è comunemente impiegato per ottenere una soluzione approssimata del problema.

$$\hat{f}(s_1, s_2, s_3) = \frac{1}{2} \int_0^{2\pi} d\beta \frac{1}{U^2} \int_{-u_m}^{u_m} du \frac{D}{\sqrt{D^2 + u^2 + v^2}} \cdot g_\beta(u, v) h(u - u') \tag{1}$$

dove

- $\frac{D}{\sqrt{D^2+u^2+v^2}}$ è il fattore peso legato alla variazione della lunghezza delle linee
- $U = \frac{R+s_1 \sin\beta - s_2 \cos\beta}{R}$ è il fattore peso nella retroproiezione

FDK algorithm - 2

In pratica, l'implementazione dell'algoritmo FDK prevede:

1. moltiplicare la funzione $g_\beta(u,v)$ per il fattore peso $\frac{D}{\sqrt{D^2+u^2+v^2}}$ ottenendo $g'_\beta(u,v)$
2. Convoluzione di $g'_\beta(u,v)$ con il filtro a rampa $h(u)/2$
3. retroproiezione dei dati opportunamente pesati con la funzione $\frac{1}{U^2}$

Le dimensioni tipiche del problema sono dell'ordine di 256^3, 512^3, 1024^3.

Implementazione per GPU

La scelta implementativa è stata quella di mantenere gli step precedenti e di parallelizzare i singoli passi con l'ausilio di CUDA iterando sulle proiezioni angolari.

1. weighting step: 1 thread Cuda per ogni pixel di una proiezione
2. filtering step: 1 thread Cuda per ogni riga di una proiezione
3. back projection step: 1 thread Cuda per ogni voxel

Test e risultati - 1

Il codice sviluppato è stato valutato su 3 tipi di schede NVIDIA in confronto con un codice parallelizzato con OpenMP. Al momento sono stati valutati 2 set di dati il primo del tipo 256^3 ed il secondo del tipo 512^3:

1. GTS 450, architettura Fermi 2 GByte di RAM,
2. GTX 680, architettura Kepler 4 GByte di RAM,
3. GTX Titan, architettura Kepler 5 GByte di RAM.

Test e risultati - 2

I risultati per il problema 256^3 sono:

step	GTS 450	GTX 680	Titan
weighting	0.32	1.5	0.39
filtering	49.17	15.01	13.57
backlproj	17.45	4.37	3.08
Total	66.94	20.88	17.04

I risultati per il problema 512^3 sono:

step	GTS 450	GTX 680	Titan
weighting	0.96	3.93	0.46
filtering	173.4	51.23	51.73
backlproj	42.2	8.38	8.03
Total	216.56	63.43	60.22

Lo step di weighting per la scheda GTX680 non è ottimizzato.

Confronto CUDA-OpenMP

Il confronto con OpenMP e CUDA per il problema 256^3 sono:

step	**OpenMP**	**Titan**	**Perf. ratio**
weighting	0.215	0.39	**0.55**
filtering	1.26	13.57	**0.092**
backlproj	80.85	3.08	**26.3**
Total	82.325	17.04	**4.83**

Risultati simili si ottengono per il problema 512^3. L'errore relativo tra CUDA e OpenMP è in media di $\approx 10^{-5}$.

Lavoro in progress

1. Ottimizzazione dello step filtering: tempi dell'ordine di 1-2 secondi.
2. Ottimizzazione dello step di retroproiezione: portare la retroproiezione a tempi di 1-2 secondi.
3. Confronto con codice sviluppato per OpenACC.
4. Completare lo sviluppo della versione 0.1 del sistema proiettore/retroproiettore.