TECHNISCHE UNIVERSITÄT DRESDEN

FAKULTÄT INFORMATIK
INSTITUT FÜR TECHNISCHE INFORMATIK
PROFESSUR FÜR RECHNERARCHITEKTUR
PROF. DR. WOLFGANG E. NAGEL

Großer Beleg

Untersuchung der Parallelisierung des Feldkamp-Davis-Kress-Algorithmus mittels CUDA®

Jan Stephan
(Mat.-Nr.: 3755136)

Hochschullehrer: Prof. Dr. Wolfgang E. Nagel
Betreuer: Dr.-Ing. André Bieberle
Dr.-Ing. Guido Juckeland
Matthias Werner

Dresden, 21. März 2017

Inhaltsverzeichnis

Abkürzungsverzeichnis

CUDA® NVIDIA® CUDA®.

FDK-Algorithmus Feldkamp-Davis-Kress-Algorithmus.

FPGA *Field Programmable Gate Array*.

OpenCL™ *Open Computing Language*.

PARIS *Portable and Accelerated 3D Reconstruction tool for radiation based Imaging Systems*.

1 Einleitung

1.1 Die gesellschaftliche Relevanz der Computertomographie

- CT in ihrer Bedeutung nur mit Röntgen vergleichbar
- Nobelpreise für die CT-Erfinder unterstreichen das

1.2 Der Einsatz der Computertomographie am Helmholtz-Zentrum Dresden-Rossendorf

- Ausgangssituation am HZDR: Uraltes FDK-Programm braucht mehrere Tage für eine Rekonstruktion (schlecht)
- Gesamtziel: Dieses Programm soll durch ein schnelleres und möglichst gut optimiertes abgelöst werden

1.3 Aufgabenstellung

Der Feldkamp-Davis-Kress-Algorithmus (FDK-Algorithmus) ist ein weit verbreiteter Ansatz zur Rekonstruktion von kegelförmiger Computer-Tomographie. In diesem Beleg soll untersucht werden:

- Zusammenfassung des Forschungsstandes hinsichtlich der Parallelisierung / der Verwendung von NVIDIA® CUDA® (CUDA®)
- Gegenüberstellung verschiedener Optimierungsziele (Time-to-solution, Occupancy)
- Variantenvergleich verschiedener Implementierungsstrategien
- Implementierung und Analyse einer dieser Strategien

2 Grundlagen

2.1 Die Computertomographie

2.1.1 Die prinzipielle Funktionsweise der Computertomographie

Am Anfang der Computertomographie steht das Röntgenverfahren, das 1895 vom deutschen Physiker Wilhelm Conrad Röntgen entdeckt wurde [Rö95]. Mit Hilfe einer Strahlungsquelle wird ein Objekt durchleuchtet und auf einem Film bzw. einem Detektor abgebildet; der dreidimensionale Körper wird also auf eine zweidimensionale Fläche projiziert. Diesen Schritt bezeichnet man als *Vorwärtsprojektion*. Führt man die Vorwärtsprojektion genügend oft in aufeinanderfolgenden Winkelschritten aus, bis man (idealerweise) einen Vollkreis abgefahren hat, so lässt sich aus den dabei entstandenen *Projektionen* der ursprünglich durchleuchtete Körper, den wir in der Folge als *Volumen* bezeichnen, rekonstruieren. Für jeden Punkt im Volumen (*Voxel*) kann anhand der Informationen aus den Projektionen der Absorptionsgrad berechnet und dadurch die innere Struktur des Volumens bestimmt werden. Dieser Zusammenhang wurde in den 60er Jahren des 20. Jahrhunderts durch den südafrikanisch-amerikanischen Physiker Allan McLeod Cormack festgestellt, der ebenfalls die dazu notwendigen mathematischen Grundlagen entwickelte [Cor63] [Cor64]; ihm war allerdings unbekannt [Cor79], dass diese schon 1917 vom österreichischen Mathematiker Johann Radon gefunden wurden [Rad17]. Mathematisch ist der Vorgang der *Rückprojektion* eine Anwendung der nach Radon benannten *Radon-Transformation*.
Ein Problem der Vorwärtsprojektion ist der Informationsverlust, der durch die mangelnde Tiefe des Films bzw. Detektors entsteht; die Tiefeninformationen werden auf die zweidimensionale Fläche „verschmiert“. Bei der Rückprojektion lässt sich dieser Verlust durch die Wahl eines geeigneten Bildfilters wiederum kaschieren, weshalb man auch von der *gefilterten Rückprojektion* spricht.
Da die gefilterte Rückprojektion für jedes Voxel einzeln berechnet werden muss, ist sie für einen Menschen nicht in sinnvoller Zeit lösbar. Aus diesem Grund ist man für die Lösung des Gesamtproblems auf einen Computer angewiesen, woraus sich der Name des Verfahrens ableitet: *Computertomographie*. Die ersten bis zur Marktreife entwickelten Computertomographen wurden gegen Ende der 60er Jahre des 20. Jahrhunderts vom englischen Elektroingenieur Godfrey Hounsfield gebaut. Dieser entwickelte die für die Rückprojektion nötigen Algorithmen ebenfalls selbst, da ihm die Vorarbeiten von Cormack und Radon nicht bekannt waren [Kal00]. Für ihre voneinander unabhängigen Arbeiten erhielten Godfrey und Cormack 1979 den Nobelpreis für Physiologie oder Medizin, was die Bedeutung der Computertomographie insbesondere für die Medizin unterstreicht.

2.1.2 Der Feldkamp-Davis-Kress-Algorithmus

Der 1984 entwickelte FDK-Algorithmus [FDK84] ist eine spezielle Ausprägung der gefilterten Rückprojektion für die Computertomographie mit Kegelstrahlen. Der Ausgangspunkt der Strahlung ist eine Quelle, die das Volumen mit einem *kegelförmigen* Strahl durchleuchtet und damit auf einem Detektor

abbildet. Die so aufgenommenen zweidimensionalen Projektionen werden dann pixelweise mit dem folgenden Wichtungsfaktor multipliziert:

$$w_{ij} = \frac{d_{det} - d_{src}}{\sqrt{(d_{det} - d_{src})^2 + h_j^2 + v_i^2}} \tag{2.1}$$

Zum Ausgleich der durch die Vorwärtsprojektion verloren gegangenen Tiefeninformationen werden die Projektionen im nächsten Schritt zeilenweise gefiltert. Zu diesem Zweck müssen die Projektionen und der Filter allerdings mittels der diskreten Fouriertransformation in den komplexen Raum transformiert werden; zum Einsatz kommt dabei das Verfahren der schnellen Fouriertransformation (*fast Fourier transform*, FFT) nach Cooley und Tukey [CT65]. Da dieses Verfahren nur mit einer Menge von Elementen funktioniert, die einer Zweierpotenz entspricht, müssen die Projektionszeilen und der Filter auf die nächste Zweierpotenz „aufgerundet“ werden. Dazu wird, ausgehend von der Länge einer Projektionszeile N_h, die Filterlänge N_{hFFT} berechnet:

$$N_{hFFT} = 2 \cdot 2^{\left\lceil \frac{\log N_h}{\log 2} \right\rceil} \tag{2.2}$$

Mit der so bestimmten Filterlänge lässt sich der Filter r erzeugen:

$$\begin{aligned} r(j) \text{ mit } j &\in \left[-\frac{N_{hFFT} - 2}{2}, \frac{N_{hFFT}}{2} \right] \\ r(j) &= \begin{cases} \frac{1}{8} \cdot \frac{1}{\tau^2} & \text{wenn } j = 0 \\ 0 & \text{wenn } j \text{ gerade} \\ -\frac{1}{2j^2\pi^2\tau^2} & \text{wenn } j \text{ ungerade} \end{cases} \end{aligned} \tag{2.3}$$

Nun wird die zu filternde Zeile so lange mit 0 aufgefüllt, bis die erweiterte Zeile N_{hFFT} Pixel umfasst:

$$\begin{aligned} p: &\text{ mit Nullen aufgefüllte Projektionszeile} \\ p(0 \ldots N_{h-1}) &= \text{det}(0 \ldots N_{h-1}) \\ p(N_h \ldots N_{hFFT}) &= 0 \end{aligned} \tag{2.4}$$

Im Anschluss werden sowohl der Filter r als auch die erweiterte Projektionszeile p in den komplexen Raum transformiert und dort miteinander multipliziert:

$$\begin{aligned} R &= \text{FFT}(r) \\ P &= \text{FFT}(p) \\ F &= P \cdot R \quad \text{sowohl für den reellen als auch den imaginären Teil} \end{aligned} \tag{2.5}$$

Die so gefilterte Projektionszeile F wird dann mit der inversen schnellen Fouriertransformation (IFFT) in den reellen Raum zurücktransformiert und von den „aufgefüllten“ Elementen bereinigt:

$$\begin{aligned} f &= \text{IFFT}(F) \\ \text{gefilterte Projektionszeile}: f(0 \ldots N_{h-1}) & \end{aligned} \tag{2.6}$$

Die auf diese Weise gefilterten Projektionen können nun wie folgt für die Rückprojektion verwendet werden:

Für jede Projektion p mit dem Drehwinkel α_p:

- berechne für jede Voxelkoordinate (x_k, y_l, z_m) deren rotierte Position (s, t, z):

$$\begin{aligned} s &= x_k \cos\alpha_p + y_l \sin\alpha_p \\ t &= -x_k \sin\alpha_p + y_l \cos\alpha_p \\ z &= z_m \end{aligned} \tag{2.7}$$

- projiziere die rotierte Voxelkoordinate (s, t, z) auf den Detektor:

$$\begin{aligned} h' = y' &= t \cdot \frac{d_{det} - d_{src}}{s - d_{src}} \\ v' = z' &= z \cdot \frac{d_{det} - d_{src}}{s - d_{src}} \end{aligned} \tag{2.8}$$

- interpoliere das Detektorsignal bei (h', v'):

$$det' = det(h', v') \tag{2.9}$$

- führe die Rückprojektion aus:

$$\begin{aligned} vol_{klm} &= vol_{klm} + 0,5 \cdot det' \cdot u^2 \\ \text{mit } u &= \frac{d_{src}}{s - d_{src}} \end{aligned} \tag{2.10}$$

Nach Abschluss der Rückprojektion erhält man ein Volumen, dessen Voxel Aufschluss über seine innere Struktur geben.
Aufgrund seiner geringen Komplexität und einfachen Implementierbarkeit ist der FDK-Algorithmus einer der beliebtesten Rückprojektionsalgorithmen für die Kegelstrahl-Computertomographie [XM04]. Der Vorteil des FDK-Algorithmus liegt außerdem darin, dass die gefilterte Rückprojektion für jedes Voxel individuell berechnet werden kann, das heißt ohne Abhängigkeiten zu anderen Voxeln. Dieser Umstand ermöglicht für die maschinelle Berechnung den maximalen Grad an Parallelität, der im englischen Sprachraum auch als *embarassingly parallel* bezeichnet wird, und macht den FDK-Algorithmus zu einem idealen Ziel für diverse Parallelisierungsansätze. Einige neuere Ansätze sollen im Folgenden vorgestellt werden.

2.1.3 Bisherige Parallelisierungsansätze

Seit seiner Einführung ist der FDK-Algorithmus ein beliebtes Untersuchungsobjekt diverser Forschungsgruppen, die sich mit seiner Beschleunigung bzw. Parallelisierung mittels einer großen Variation von Architekturen, Plattformen und Programmiermodellen beschäftigen.
Xu et al. untersuchten bereits 2004, inwieweit sich der FDK-Algorithmus durch den Einsatz handelsüblicher Grafikkarten (*commodity graphics hardware*) beschleunigen lässt [XM04]. Dabei wurden die Schritte *Wichtung* und *Filterung* aufgrund ihrer geringen Komplexität ($\mathcal{O}(n^2)$) auf der CPU ausgeführt, während man die komplexere *Rückprojektion* ($\mathcal{O}(n^4)$) auf der GPU berechnete. Die Rückprojektion fand

schichtweise statt, jeweils für eine Voxelebene entlang der vertikalen Volumenachse. In ihrem Fazit stellten die Autoren die Vermutung auf, dass der Abstand zwischen den Leistungen von CPU und GPU in der Zukunft zugunsten der GPUs immer weiter werden würde: *Since GPU performance has so far doubled every 6 months (i.e., triple of Moore's law), we expect that the gap between CPU and GPU approaches will widen even further in the near future.*

Li et al. beschäftigten sich 2005 damit, wie man den FDK-Algorithmus mit einem *Field Programmable Gate Array* (FPGA) implementieren könnte. Dazu teilten sie das Ausgabevolumen, also die Zieldaten der Rückprojektion, in mehrere Würfel (*bricks*) auf, um zu einer optimalen Cachenutzung zu kommen. Der verwendete deterministische Aufteilungsalgorithmus hatte zur Folge, dass bei der Berechnung auf dem FPGA kein Cache-Verfehlen (*cache miss*) mehr auftrat.

Knaup et al. gingen 2007 der Frage nach, ob der FDK-Algorithmus durch die Eigenschaften der Cell-Architektur profitieren könne [KSBK07].

Scherl et al. unternahmen 2008 den Versuch, den FDK-Algorithmus mittels CUDA® zu beschleunigen [SKKH07]. Im Gegensatz zu der Gruppe um Xu et al. führten sie alle Schritte auf der GPU aus und führten die Rückprojektion projektionsweise durch, das heißt, dass jede Projektion einzeln in das Gesamtvolumen zurückprojiziert wurde. Diese Art der Datenverarbeitung ermöglichte es, die Schritte *Wichtung* und *Filterung* parallel zur Rückprojektion auszuführen. Zur Ausnutzung dieser Eigenschaft und zur besseren Kapselung bzw. Modularisierung der Teilschritte entwickelten die Autoren daher eine Pipeline-Struktur zur parallelen Abarbeitung des Algorithmus, basierend auf dem von Mattson et al. vorgestellten Entwurfsmuster [MSM04].

Balász et al. versuchten 2009 das Gleiche mit der *Open Computing Language* (OpenCL™) [BG09].

Hofmann et al. untersuchten eventuelle Vorteile durch den Einsatz der neuen Koprozessoren vom Typ Intel® Xeon Phi™ ‚Knights Corner' [HTHW14].

Zhao et al. verfolgten die Absicht, eine Beschleunigung durch Ausnutzung geometrischer Zusammenhänge zu erreichen [ZHZ09]. Sie setzten dabei auf die Tatsache, dass ein einmal bestimmtes, also auf den Detektor projiziertes, Voxel durch Rotation in 90°-Schritten die rotierten Voxel ebenfalls genau bestimmt. Ist also für ein Voxel im Projektionswinkel 0° die zugehörige Detektorkoordinate gefunden, so kann diese Detektorkoordinate für die Projektionswinkel 90°, 180° und 270° und die entsprechenden Voxel wiederverwendet werden.

2.2 Die NVIDIA®-CUDA®-Plattform

2.2.1 Grafikkarten und Wissenschaft

2.2.2 Das CUDA®-Programmiermodell

2.2.3 Alternativen zu CUDA®

3 Umsetzung

3.1 Variantenvergleich

3.1.1 Parallelisierungsstrategien und Hindernisse

3.1.2 Das Problem des GPU-Speichers

3.1.3 Heterogene GPU-Systeme und effiziente Arbeitsteilung

3.2 Das *PARIS*-Programm

Das Programm *Portable and Accelerated 3D Reconstruction tool for radiation based Imaging Systems* (PARIS)

3.2.1 Implementierungsziele

- Wartbarkeit
- Portabilität
- sinnvolle Geschwindigkeit
- nutzbar auf Laptop / Workstation / GPU-Cluster

3.2.2 Geometrische Berechnungen

- Berechnung der Volumengeometrie
- Aufteilung in Teilvolumen

3.2.3 Implementierung der Vorstufen

- Wichtung
- Filterung

3.2.4 Implementierung der gefilterten Rückprojektion

- welche Konstanten und Variablen gibt es
- welche Schwierigkeiten können auftreten

4 Optimierung und Leistungsanalyse

4.1 Optimierungsziele

Optimierungsziele vorstellen (Auslastung, Time-to-solution)

4.2 Leistungsmessungen

4.2.1 Übersicht

Messungen des Gesamtprogramms (Datentransfers, alle Stufen)
grobe Messungen der Teilstufen (Wichtung, Filterung, Rückprojektion)
Ergebnis: Rückprojektion braucht am längsten

4.2.2 Rückprojektion im Detail

detaillierte Messungen der Rückprojektion (Registerverbrauch, GPU-Auslastung, etc)

4.2.3 Vergleich mit der Literatur

5 Fazit

- Faktencheck: Wurden die in der Einleitung genannten Ziele erreicht?

Literaturverzeichnis

[BG09] BALÁZS, D. ; GÁBOR, J.: A programming model for GPU-based parallel Computing with scalability and abstraction. In: *SCCG '09 Proceedings of the 25th Spring Conference on Computer Graphics* Spring Conference on Computer Graphics, 2009, S. 103–111

[Cor63] CORMACK, A. M.: Representation of a Function by Its Line Integrals, with Some Radiological Applications. In: *Journal of Applied Physics* 34 (1963), S. 2722

[Cor64] CORMACK, A. M.: Representation of a Function by Its Line Integrals, with Some Radiological Applications. II. In: *Journal of Applied Physics* 35 (1964), S. 2908

[Cor79] CORMACK, A. M. *Early Two-Dimensional Reconstruction and Recent Topics Stemming from It (Nobel Lecture)*. Dezember 1979

[CT65] COOLEY, J. W. ; TUKEY, J. W.: An Algorithm for the Machine Calculation of Complex Fourier Series. In: *Mathematics of Computation* 19 (1965), April, Nr. 90, S. 297–301

[FDK84] FELDKAMP, L. A. ; DAVIS, L. C. ; KRESS, J. W.: Practical cone-beam algorithm. In: *Journal of the Optical Society of America A* 1 (1984), Februar, Nr. 6, S. 612–619

[HTHW14] HOFMANN, J. ; TREIBIG, J. ; HAGER, G. ; WELLEIN, G.: Performance Engineering for a Medical Imaging Application of the Intel Xeon Phi Accelerator. In: *2014 Workshop Proceedings* International Conference on Architecture of Computing Systems, 2014

[Kal00] KALENDER, W. A.: *Computertomographie: Grundlagen, Gerätetechnologie, Bildqualität, Anwendungen*. Publicis MCD Verlag, 2000. – ISBN 978–3–895–78082–0

[KSBK07] KNAUP, M. ; STECKMANN, S. ; BOCKENBACH, O. ; KACHELRIESS, M.: Tomographic image reconstruction using the cell broadband engine (CBE) general purpose hardware. In: *Proceedings Electronic Imaging, Computational Imaging V* Bd. 6498 SPIE, 2007, S. 1–10

[MSM04] MATTSON, T. G. ; SANDERS, B. ; MASSINGILL, B.: *Patterns for Parallel Programming*. Addison-Wesley Professional, September 2004. – ISBN 978–0–321–22811–6

[Rad17] RADON, J.: Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten. In: *Berichte über die Verhandlungen der Königlich Sächsischen Gesellschaft der Wissenschaften zu Leipzig, Mathematisch-physische Klasse* Bd. 69 Königlich Sächsische Gesellschaft der Wissenschaften zu Leipzig, 1917, S. 262–277

[Rö95] RÖNTGEN, W. C.: Über eine neue Art von Strahlen. Vorläufige Mittheilung. In: *Aus den Sitzungsberichten der Würzburger Physikalisch-medicinischen Gesellschaft 1895* Würzburger Physikalisch-medicinische Gesellschaft, 1895, S. 137–147

[SKKH07] SCHERL, H. ; KECK, B. ; KOWARSCHIK, M. ; HORNEGGER, J.: Fast GPU-Based CT Reconstruction using the Common Unified Device Architecture (CUDA). In: *IEEE Nuclear Science Symposium Conference Record* Institute of Electrical and Electronics Engineers, 2007, S. 4464–4466

[XM04] XU, F. ; MÜLLER, K.: Ultra-Fast 3D Filtered Backprojection on Commodity Graphics Hardware. In: *IEEE International Symposium on Biomedical Imaging: Nano to Macro* Institute of Electrical and Electronics Engineers, 2004, S. 571–574

[ZHZ09] ZHAO, X. ; HU, J. ; ZHANG, P.: GPU-based 3D cone-beam CT image reconstruction for large data volume. In: *Journal of Biomedical Imaging* 2009 (2009), Nr. 8

Danksagung

Für die fachliche Betreuung während der Erstellung dieser Arbeit bedanke ich mich recht herzlich bei Herrn Dr.-Ing. Stephan Boden.

Erklärungen zum Urheberrecht

Ich habe wirklich nichts geklaut