LEHRSTUHL FÜR RECHNERARCHITEKTUR UND PARALLELE SYSTEME

Praktikum Rechnerarchitektur

Moore-Kurven (A213)

Projektaufgabe – Aufgabenbereich Bildverarbeitung

1 Organisatorisches

Auf den folgenden Seiten finden Sie die Aufgabenstellung zu Ihrer Projektaufgabe für das Praktikum. Die Rahmenbedingungen für die Bearbeitung werden in der Praktikumsordnung festgesetzt, die Sie über die Praktikumshomepage[1] aufrufen können.

Wie in der Praktikumsordnung beschrieben, sind die Aufgaben relativ offen gestellt. Besprechen Sie diese innerhalb Ihrer Gruppe und konkretisieren Sie die Aufgabenstellung. Die Teile der Aufgabe, in denen Assembler-Code anzufertigen ist, sind für die 64-Bit x86-Architektur (x86-64) unter Verwendung der SSE-Erweiterungen zu schreiben.

Der **Abgabetermin** ist der **24. Juli 2020, 11:59 Uhr (MESZ)**. Die Abgabe erfolgt per Git in das für Ihre Gruppe eingerichtete Projektrepository. Bitte beachten Sie die in der `README.md` angegebene Liste von abzugebenden Dateien.

Die **Abschlusspräsentationen** finden in der Zeit vom **10.08.2020 – 28.08.2020** statt. Weitere Informationen werden noch bekannt gegeben. Beachten Sie, dass die Folien für die Präsentation am obigen Abgabetermin im PDF-Format abzugeben sind. Geben Sie *außerdem* eine Datei mit Notizen zum Vortrag ab, mit deren Hilfe die Folien auch ohne den gehaltenen Vortrag nachvollziehbar sind. Der Kurzvortrag entfällt ersatzlos.

Sofern die Rahmenbedingungen (Hygiene-, Abstandsregeln etc.) für eine Präsenzprüfung nicht vorliegen, kann gemäß §13a APSO die geplante Prüfungsform auf eine virtuelle Präsentation (Videokonferenz) oder eine kurze schriftliche Fernprüfung umgestellt werden. Die Entscheidung über diesen Wechsel wird möglichst zeitnah, spätestens jedoch 14 Tage vor dem Prüfungstermin nach Abstimmung mit dem zuständigen Prüfungsausschuss bekannt gegeben.

Bei Fragen/Unklarheiten in Bezug auf den Ablauf und die Aufgabenstellung wenden Sie sich bitte an Ihren Tutor.

Wir wünschen Ihnen viel Erfolg und Freude bei der Bearbeitung Ihrer Aufgabe!

Mit freundlichen Grüßen
Die Praktikumsleitung

PS: Vergessen Sie nicht, sich rechtzeitig in TUMonline zur Prüfung anzumelden. Dies ist Voraussetzung für eine erfolgreiche Teilnahme am Praktikum im laufenden Semester.

[1] `https://gepasp.in.tum.de/eraweb/`

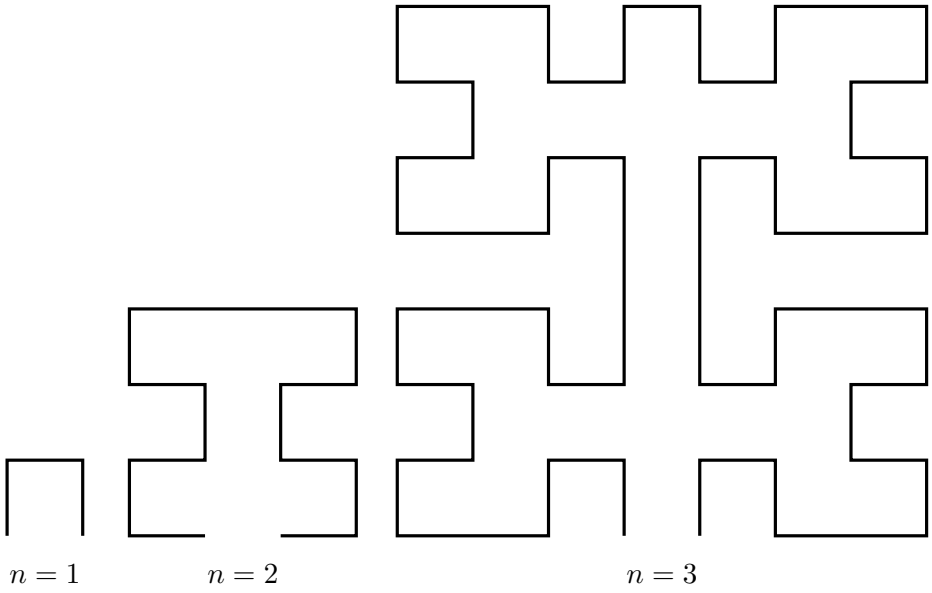
2 Moore-Kurven

2.1 Überblick

Im Zuge Ihrer Projektaufgabe werden Sie theoretisches Wissen aus der Mathematik im Anwendungszusammenhang verwenden, um einen Algorithmus in Assembler zu implementieren. Sie konzentrieren sich dabei auf das Feld des *Image Processing*, in welchem Pixelbilder, wie sie typischerweise Digitalkameras produzieren, als Eingabe für bestimmte Algorithmen verwendet werden und mathematische Überlegungen dadurch sichtbar gemacht werden.

2.2 Funktionsweise

In Ihrer Projektaufgabe befassen Sie sich mit sogenannten Moore-Kurven. Dieser Text gibt Ihnen eine kurze Einführung, jedoch ist zum genaueren Verständnis ein Nachlesen in der Literatur erforderlich.
Die Moore-Kurve ist eine planare Kurve die im einfachsten Fall ($n = 1$) einem Einheitsquadrat einbeschrieben wird. Man nennt diese Kurve "raumfüllend", da sie jede Ecke des Einheitsquadrats passiert. Durch einen rekursiven Prozess lassen sich aus der Basiskurve die Moore-Kurven höheren Grades konstruieren. (Im Bild sind die Kurven für $n = 2$ und $n = 3$ dargestellt.)



Die Konstruktion der Moore-Kurve für $n = 1, 2, \cdots$ ist Thema Ihrer Projektaufgabe.

2.3 Aufgabenstellungen

Ihre Aufgaben lassen sich in die Bereiche Konzeption (theoretisch) und Implementierung (praktisch) aufteilen. Sie können (müssen aber nicht) dies bei der Verteilung

der Aufgaben innerhalb Ihrer Arbeitsgruppe ausnutzen. Antworten auf konzeptionelle Fragen sollten an den passenden Stellen in Ihrer Ausarbeitung in angemessenem Umfang erscheinen. Entscheiden Sie nach eigenem Ermessen, ob Sie im Rahmen Ihres Abschlussvortrags auch auf konzeptionelle Fragen eingehen. Die Antworten auf die Implementierungsaufgaben werden durch Ihrem Code reflektiert.

2.3.1 Theoretischer Teil

- Entwickeln Sie einen Algorithmus, welcher für eine Eingabe n eine geordnete Liste an (ganzzahligen) Koordinaten ausgibt, welche die Abfolge der Knotenpunkte der Moore-Kurve n-ten Grades repräsentiert. Bevorzugen Sie iterative Lösungsansätze und vermeiden Sie (nach Möglichkeit) Rekursion. Können Sie einen Punkt der Moore-Kurve in konstanter Zeit (d.h. unabhängig vom Grad der Kurve) berechnen?
- Beschreiben Sie ein Beispiel, wofür die Moore-Kurve eingesetzt wird.

2.3.2 Praktischer Teil

- Implementieren Sie in der Datei mit dem Assemblercode eine Funktion

  ```
  void moore(uint64_t degree, uint64_t *x, uint64_t *y)
  ```

  welche den Grad der zu konstruierenden Moore-Kurve `degree` übergeben bekommt und eine geordnete Liste von x- und y-Koordinaten in die Speicherbereiche `x` und `y` schreibt.
- Wie Sie wissen, dürfen Sie alle I/O-Operationen im Rahmenprogramm in C durchführen. Wir schreiben Ihnen kein Ausgabeformat vor, legen Ihnen aber nahe, die von der `moore`-Funktion zurückgegebenen, mit Linien verbundenen Koordinaten in Form einer SVG-Datei auszugeben. (SVG-Bild-Dateien werden durch Text beschrieben, ziehen Sie, wenn nötig, geeignete Literatur heran, um sich über das SVG-Format zu informieren.)

2.4 Allgemeine Bewertungshinweise

Die folgende Liste soll Ihnen als Gedächtnisstütze beim Bearbeiten der Aufgaben dienen. Beachten Sie ebenfalls die in der Praktikumsordnung angegebenen Hinweise.

- Stellen Sie unbedingt sicher, dass Ihre Abgabe auf der Referenzplattform des Praktikums (`lxhalle`) kompiliert und funktionsfähig ist.
- Fügen Sie Ihrem Projekt ein funktionierendes `Makefile` hinzu, welches durch den Aufruf von `make` Ihr Projekt kompiliert.
- Verwenden Sie keinen Inline-Assembler.

- Verwenden Sie SIMD-Befehle, wenn möglich.
- Verwenden Sie keine x87-FPU- oder MMX-Instruktionen. Sie dürfen alle SSE-Erweiterungen bis SSE4.2 benutzen. AVX-Instruktionen dürfen Sie benutzen, sofern Ihre Implementierung auch auf Prozessoren ohne AVX-Erweiterungen lauffähig ist.
- Sie dürfen die Signatur der in Assembler zu implementierenden Funktion nur dann ändern, wenn Sie dies (in Ihrer Ausarbeitung) rechtfertigen können.
- I/O-Operationen dürfen grundsätzlich in C implementiert werden.
- Denken Sie daran, das Laufzeitverhalten Ihres Codes zu testen (Sichere Programmierung, Performanz) und behandeln Sie alle möglichen Eingaben, auch Randfälle. Ziehen Sie ggf. alternative Implementierungen als Vergleich heran.
- Eingabedateien, welche Sie generieren, um Ihre Implementierungen zu testen, sollten mit abgegeben werden.
- Verwenden Sie für die Ausarbeitung die bereitgestellte TEX-Vorlage und legen Sie sowohl die PDF-Datei als auch sämtliche TEX-Quellen in das Repository.
- Stellen Sie Performanz-Ergebnisse nach Möglichkeit grafisch dar.
- Vermeiden Sie unscharfe Grafiken und Screenshots von Code.
- Geben Sie die Folien für Ihre Abschlusspräsentation im PDF-Format ab. Achten Sie auf hinreichenden Kontrast (schwarzer Text auf weißem Grund!) und eine angemessene Schriftgröße. Verwenden Sie 4:3 als Folien-Format.
- Zusatzaufgaben (sofern vorhanden) müssen nicht implementiert werden. Es gibt keine Bonuspunkte.