Universität Karlsruhe (TH)

Institut für Innovatives Rechnen und Programmstrukturen (IPD)

Real Life Programming (Praktikum im SS 2005) http://www.info.uni-karlsruhe.de/
Dipl.-Inf. Michael Beck beck@ipd.info.uni-karlsruhe.de
Dipl.-Inform. Rubino Geiß rubino@ipd.info.uni-karlsruhe.de
Dipl.-Inform. Sebastian Hack hack@ipd.info.uni-karlsruhe.de

Übungsblatt 7 | Ausgabe: 21.06.2005 | Besprechung: 12.07.2005

Aufgabe 1: *Optimierung (6+10 Punkte)*

In dieser Aufgabe soll ein kleines Fragment aus einem Videocodec beschleunigt werden. Das Archiv `http://www.info.uni-karlsruhe.de/lehre/2005SS/rlp/material/aufgabe7.tgz` enthält die nötigen Quellen. Die zu beschleunigende Funktion befindet sich in der Datei `filter.c` und heißt `OBMC_FilterBlock()`. Ihr müßt nicht unbedingt verstehen, wozu diese Funktion dient. Für diejenigen, die es wissen wollen folgt im Anhang eine kurze Erklärung.

Die Datei `main.c` generiert sinnvolle Testdaten und läßt `OBMC_FilterBlock()` und `OBMC_FilterBlockX()` laufen, vergleicht die Resultate und die Laufzeiten. Weder `main.c` noch `filter.c` sollten geändert werden.

Ihr dürft eure Implementierung in `filterx.c` und weiteren Dateien falls nötig einfügen.

Einige Regeln:

- weder der Funktionstyp noch die Datenstruktur MV dürfen geändert werden
- die Funktion soll nicht in Assembler implementiert werden, wir lassen sie auf mehreren Architekturen und Compilern übersetzen
- wie schon erwähnt, darf natürlich die Implementierung von `OBMC_FilterBlock()` in `filter.c` nicht verändert werden

Natürlich muß eure Version ganau dasselbe machen, dies wird in `main.c` getestet.

Einige Hinweise:

- Analysiert zunächst, warum die Funktion so langsam läuft. Dazu kann es sinnvoll sein, sich den Assembleroutput anzuschauen.
- Es ist eher unwahrscheinlich, daß ihr einen Algorithmus einer besseren Komplexitätsklasse finden werden, aber Vorschläge sind willkommen.
- 6 Punkte gibt es für eine korrekte und zumindest etwas beschleunigte Version. Die restlichen Punkte vergeben wir nach Zieleinlauf, d.h. die schnellste Gruppe bekommt 10 usw.
- Für gcc, verwendet bitte die folgenden Übersetzerparameter, damit die Resultate vergleichbar bleiben: `-O3`
- Wir haben auf `i44pc23` mit unserer Implementierung und dem SuSE gcc 3.3.5 die folgenden Speedups erreicht:

```
i44pc23 (aufg7) $ ./aufg7
Standard Implementation = 1.63 sec
New      Implementation = 0.70 sec
Speedup                 = 2.33
```

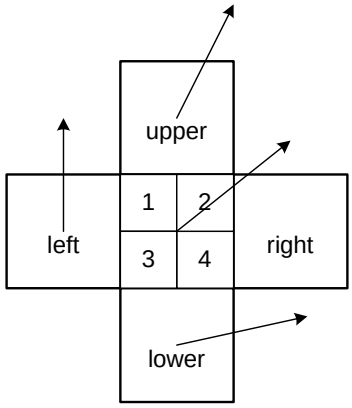

Abbildung 1: Overlapping Block Motion Compesation

Beschreibung Bei der Videokompression nach neueren Standards (H.263, MPEG-4) kommt eine Technik namens *Overlapping Block Motion Compesation* zum Einsatz. Dabei wird versucht, Blockartefakte durch eine Art „inteligentes“ Filtering zu unterdrücken. Blockartefakte entstehen dadurch, daß jedes Bild zunächst in 8x8 Blöcke zerlegt wird und diese danach kodiert werden. An den Blockgrenzen passen nun die Pixel möglicherweise nicht mehr ganz zusammen, was zu störenden „Blockgrenzen“ führt.

OBMC zerlegt nun jeden Block in vier kleine (4x4) Blöcke. Jeder dieser Blöcke hat vier Nachbarblöcke, im Bild `upper`, `lower`, `left` und `right` genannt. Jeweils 2 dieser 4x4 Nachbarblöcke stammen aus einem anderen 8x8 Block, z.B. hat Block 1 einen `upper` und einen `left` Nachbarblock, Block 3 einen `lower` und einen `left` Block.

Jeder dieser Nachbarblöcke hat einen eigenen Bewegungsvektor, der sich möglichgerweise vom Bewegungsvektor des aktuellen 4x4 Blocks unterscheidet. Jeder der 3 Bewegungsvektoren zeigt im Vorgängerbild auf einen 4x4 Block. Diese 3 Blöcke werden jetzt mittels 3 Matrizen zum Zielblock verschmolzen:

$$P(x,y) = (C1(x,y) * R(x,y) + C2(x,y) * S(x,y) + C3(x,y) * T(x,y) + 4)/8$$

R ist dabei der Block der vom aktuellen Bewegungsvektor stammt, S vom Bewegungsvektor aus dem oberen oder unteren Vektor und T aus dem linken oder rechten Vektor. $C1$, $C2$ und $C3$ sind dabei Konstante Matrizen, die natürliche Zahlen enthalten. Insbesondere gilt folgendes:

$$\forall x, y \in [0..3] : C1(x,y) + C2(x,y) + C3(x,y) = 8$$

Hinweise:

- Die Matrizen $C1$, $C2$ und $C3$ enthalten viele Einsen und Zweien.
- Oft ist auch der aktuelle Bewegungsvektor gleich einem der anderen Vektoren.