Optimierung, Robotik & Biomechanik (ORB)
Institut für Technische Informatik (ZITI)
Ruprecht–Karls–Universität Heidelberg

Prof. Dr. Katja Mombaur,
Anna Lena Kleesattel

Abgabe: 22.12.2017, 13 Uhr

Einführung in die praktische Informatik

WS 2017/18

Übungsblatt 8

Aufgabe 8.1: Aufgabe zur Versionskontrolle mit git (5 Punkte)

In der Vorlesung haben Sie das Versionskontrollsystem git kennengelernt. Zusätzlich finden Sie auf Moodle weiteres Material zu git, welches Sie als Referenz für diese Aufgabe heranziehen können. Sie können jederzeit mittels `git log` und `git status` den Status Ihres Repositorys überprüfen.

(a) Erstellen Sie für Aufgabe 8.2 einen neuen Unterordner mit dem Namen `aufgabe8_2` und arbeiten Sie in diesem Ordner.

(b) Erzeugen Sie ein neues git-Repository in diesem Ordner.

(c) Legen Sie Ihren Namen und Ihre Mailadresse fest.[1]

(d) Adden und committen Sie Ihre Änderungen jeweils nach der Bearbeitung der Teilaufgaben von Aufgabe 8.2.

(e) Erstellen Sie ein Archiv (.zip oder .tar.gz) Ihres Unterordners `aufgabe8_2` (inklusive des .git-Ordners) und geben Sie dieses als Lösung ab.

Aufgabe 8.2: Bildklasse (12 Punkte)

Lesen Sie sich die Aufgabenstellung von Aufgabe 8.1 vor der Bearbeitung dieser Aufgabe durch!

In dieser Übungsaufgabe sollen Sie eine Klasse `Image` vervollständigen, mit der zweidimensionale Bilder gespeichert und bearbeitet werden können. Auf Moodle finden Sie die Datei `image.h`, die einen Rumpf der Klassenimplementation enthält.

(a) Vervollständigen Sie den Code in `image.h` überall dort, wo "`//IHR CODE HIER`" steht. Die Funktionalität, die Sie jeweils implementieren sollen, wird in Kommentaren beschrieben. Eine Funktion ist bereits vollständig implementiert, lesen Sie sich dennoch die Beschreibung der Funktionalität durch und versuchen Sie zu verstehen, was die Funktion macht und wie das Vorgehen ist.

[1] Falls Sie Ihren Namen und Ihre Mailadresse nur für diesen Ordner festlegen möchten, verwenden Sie das Kommando ohne `--global`.

Den in der Vorlesung ausgeteilten Git Cheat Sheet haben wir auch als PDF auf Moodle hochgeladen.

Für die graphische Darstellung nutzen wir in der Klasse `Image` das einfache PGM-Format[2], wir verwenden speziell die Variante "Portable Graymap ASCII" mit der Kennung "P2".

Legen Sie für die folgenden Teilaufgaben eine Datei `image.cc` an, die `image.h` inkludiert. Kompilieren Sie Ihren Code mit dem Befehl für den C++-11-Standard, also beispielsweise

```
g++ -Wall -Wextra -pedantic -std=c++11 -o image image.cc  .
```

(b) Erzeugen Sie ein Bild mit der Größe `width=4` und `height=3`. Testen Sie die Ausgabe der Funktion `to_string(image)` mithilfe von `cout` in der Kommandozeile.

(c) Füllen Sie das Bild mit einem Schachbrettmuster, d.h. abwechselnd mit den Grauwerten '0' (schwarz) und '255' (weiß). Das erste Pixel in der linken oberen Ecke soll schwarz sein. Testen Sie das Ergebnis mittels der über `to_string(image)` erzeugten Ausgabe in der Kommandozeile.

(d) Exportieren Sie das Bild aus Aufgabenteil (c) mit der Funktion `writePGM()` in das File `board4x3.pgm`. Öffnen Sie dieses File in einem Editor und überprüfen Sie, dass der Inhalt den Erwartungen entspricht. Legen Sie in Ihrem Programm ein zweites Bild an, in das Sie das File `board4x3.pgm` mittels `readPGM()` wieder einlesen. Überprüfen Sie mit dem überladenen `==`-Operator, dass dieses Bild mit dem Originalbild aus Teilaufgabe (c) übereinstimmt.

(e) Schreiben Sie eine Funktion

```
Image chessboard(unsigned int width, unsigned int height,
                 unsigned int square_size)
```

die ein Schachbrett-Bild der Größe `width x height` erzeugt. Der Parameter `square_size` gibt dabei an, wie groß die einzelnen Schachfelder sein sollen. Das heißt, ein Quadrat der Größe `square_size*square_size` aus schwarzen Pixeln (Wert '0') soll sich oben links im Bild befinden, daneben und darunter entsprechende Quadrate aus weißen Pixeln (Wert '255') usw. Überprüfen Sie, dass der Aufruf `chessboard(4, 3, 1)` das Schachbrett aus Aufgabenteil (c) reproduziert (erneut mit `==`).

(f) Erzeugen Sie ein Schachbrett mit dem Aufruf `chessboard(400, 300, 20)` und geben Sie es in das File `board400x300.pgm` aus. Prüfen Sie mit einem Bildbetrachter, der das PGM-Format unterstützt (z.B. 'IrfanView' unter Windows, 'display' aus der imagemagick-Installation unter Linux, 'gimp' unter allen Betriebssystemen), dass der Inhalt des Files korrekt ist. Lesen Sie das File dann mit `readPGM()` wieder ein und testen Sie die Übereinstimmung mit dem Originalbild.

(g) Schreiben Sie eine Funktion `Image invert_image(Image const& image)`, die das gegebene Bild invertiert, d.h. jeden Pixelwert `p` durch den Pixelwert `255-p` ersetzt und das invertierte Bild zurückgibt. Wenden Sie diese Funktion auf das Schachbrett aus Aufgabenteil (d) an und exportieren Sie das Ergebnis als `board400x300-inverse.pgm`. Vergewissern Sie sich mit einem Bildbetrachter, dass die schwarzen und weißen Felder jetzt genau vertauscht sind.

[2]Erklärungen finden Sie in Wikipedia unter `https://de.wikipedia.org/wiki/Portable\_Anymap`.

(h) Auf Moodle finden Sie das Bild `christmas.pgm`. Lesen Sie dieses Bild ein und invertieren Sie es. Exportieren Sie das invertierte Bild mit der Funktion `writePGM()` als `christmas-inverse.pgm`.

Geben Sie die vervollständigte Datei `image.h` sowie die Datei `image.c` und die erzeugten Bilddateien ab.

Bemerkung: Wir verwenden hier den Pixeltyp `uint16_t`, obwohl `uint8_t` für die Aufgabe eigentlich ausreichen würde. Unsere Wahl sichert, dass alle Compiler die Pixelwerte als Zahlen behandeln. Der Typ `uint8_t` hingegen wird von manchen Compilern als Zeichentyp interpretiert, was zu Fehlern in `to_string()` und `readPGM()` führen würde. Natürlich könnte man das Problem durch relativ einfache Implementierungstricks in diesen Funktionen lösen, wir wollen diese zusätzlichen Schwierigkeiten aber in der Aufgabe umgehen.

Aufgabe 8.3: Kaleidoskop (8 Punkte)

In dieser Aufgabe sollen Sie den Umgang mit Klassen im Allgemeinen und der `Image`-Klasse im Besonderen üben, indem Sie Funktionen zum Spiegeln eines Bildes implementieren. Geben Sie die Lösung in einer Datei `kaleidoscope.cc` ab. Diese Datei soll die vervollständigte Datei `image.h` aus der vorigen Aufgabe inkludieren, um die bereits vorhandene Funktionalität der Bildklasse wiederzuverwenden.

(a) Implementieren Sie die Funktion `Image mirror_x (Image const& image)`, die das gegebene Bild am rechten Rand spiegelt und gemeinsam mit dem Originalbild in ein doppelt so breites Bild einfügt, das dann zurückgegeben wird:

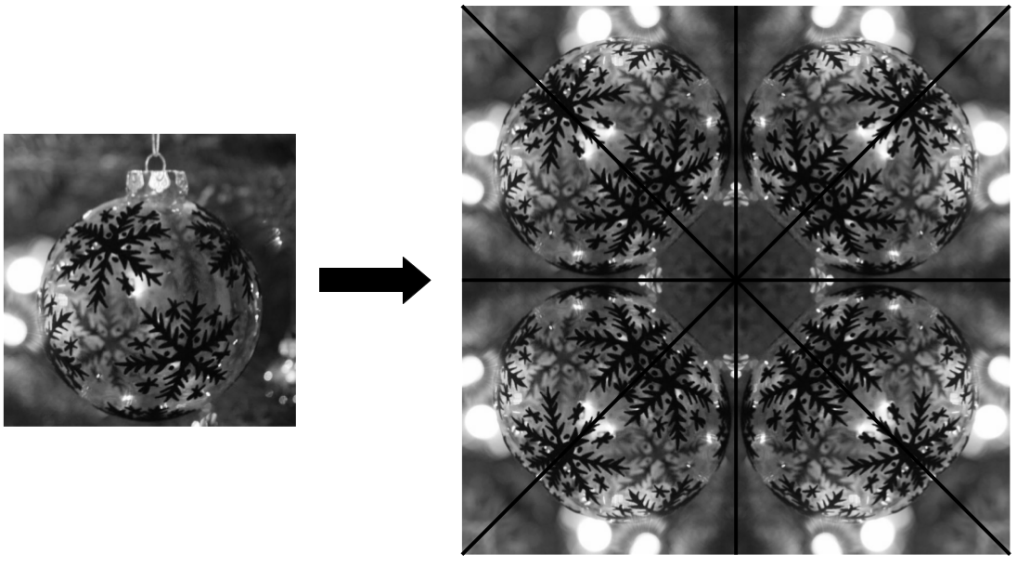
Original by David Singleton from London, UK (Christmas Bauble) [CC BY 2.0] via Wikimedia Commons

Implementieren Sie analog dazu die Funktion `mirror_y()`.

(b) Nutzen Sie die Bilddatei `christmas.pgm` aus der vorherigen Aufgabe. Lesen Sie diese Datei mittels `readPGM()` in ein Bild vom Typ `Image` ein und wenden Sie `mirror_x()` bzw. `mirror_y()` auf dieses Bild an. Exportieren Sie beide Bilder mittels `writePGM()` in die Dateien `christmas-mirror-x.pgm` und `christmas-mirror-y.pgm`.

(c) Implementieren Sie die Funktion `Image kaleidoscope4(Image const& image)` die das Bild mit Hilfe von `mirror_x()` und `mirror_y()` erst am rechten, dann am unteren Rand spiegelt und das Ergebnis zurückgibt.

(d) Implementieren Sie analog zu Aufgabenteil (c) die Funktion `kaleidoscope8()`, die ein Kaleidoskop mit 8-facher Spiegelung wie im folgenden Bild erzeugt (die schwarzen Linien sind nur als Hilfslinien gedacht und sollen nicht gezeichnet werden):



Original by David Singleton from London, UK (Christmas Bauble) [CC BY 2.0] via Wikimedia Commons

Spiegeln Sie dazu das Eingabebild zunächst von links unten nach rechts oben an der Diagonalen und benutzen Sie dann `kaleidoscope4()`. Das Resultat ist natürlich besonders wirkungsvoll, wenn das Eingabebild quadratisch war. Ihre Funktion soll aber so implementiert sein, dass Sie auch auf beliebige rechteckige Bilder angewendet werden kann.

(e) Benutzen Sie ein Bildbetrachtungsprogramm, das das PGM-Format unterstützt, um ein Bild Ihrer Wahl unter dem Namen `my_image.pgm` in dieses Format zu konvertieren. Wenden Sie `kaleidoscope8()` auf dieses Bild an und exportieren Sie das Ergebnis als `my_kaleidoscope.pgm`. Geben Sie die beiden Bilder ab. Wählen Sie ein Bild, das eine interessante Kaleidoskopdarstellung ergibt und das trotzdem nicht allzu groß ist (aufgrund der Abgabe per Mail).

Die Besprechung der Lösungen findet in der Woche ab dem 08.01.2018 statt.