4. Mittelwertfilter

Compilieren

Tipp zum Compilieren: Je mehr Klassen/Dateien compiliert werden müssen,

- umso unübersichtlicher wird der g++-Befehl und
- umso länger dauert das Compilieren.

Daher compilieren wir nun alle Dateien *einzeln* (mit der Compileroption -c) – aber nur, wenn sie wirklich verändert wurden: Also aus:

```
g++ image.cpp mean.cpp main.cpp -o output.exe
```

wird nun:

```
g++ -c image.cpp -o image.o
...
g++ -c main.cpp -o main.o
g++ image.o ... main.o -o output.exe
```

Mittelwertfilter

Implementieren Sie einen Mittelwertfilter für Grauwertbilder:

```
1 void FilterMean(Image& dst, const Image& src, int filterSize=3)
```

Es genügt zunächst eine einfache Implementierung ohne Randbehandlung. Setzen Sie z.B. alle Randpixel mit dem Befehl `memset` auf 0 oder übernehmen Sie die Originalpixel.

Laufzeit verbessern

Messen Sie nun die Laufzeit Ihres Filters und versuchen Sie, die Effizienz zu verbessern. Untersuchen Sie auch das zeitliche Verhalten bei großen Filtermasken, z.B. 21×21.
Mit folgender Definition

```
1 #include <chrono>    // time measuring features, namespace std::chrono
2
3 typedef std::chrono::high_resolution_clock::time_point Time;
4 Time timeNow() { return std::chrono::high_resolution_clock::now(); }
5 double timeMilli(std::chrono::nanoseconds d)
6 {return std::chrono::duration_cast<std::chrono::milliseconds>(d).count();}
```

lässt sich folgendermaßen eine Zeitspanne messen:

```
1 Time t1 = timeNow();
2 FilterMean(...);
3 Time t2 = timeNow();
4 double milli = timeMilli(t2-t1);
5 cout << milli << endl << endl;
```

Achtung: die Zeitliche Auflösung der Funktion `timeNow` ist begrenzt! Daher kann es notwendig sein, die zu untersuchende Funktion in einer Schleife mehrmals auszuführen und die Gesamtzeit dann durch die Anzahl der Schleifendurchläufe zu dividieren.

Optional

Implementieren Sie verschiedene Randbehandlungen:

```
1 void FilterMean(Image& dst, const Image& src, int filterSize=3,
2   int border=0);
```

0 Randpixel auf 0 setzen

1 Originalwerte übernehmen

2 Randpixel ignorieren (nichts tun, zufällige Werte)

3 Randpixel filtern