Measurement analysis of cacheability across multiple methods of grey-scaling

Niels de Waal (1698041), Jasper Smienk(1700502)

April 15, 2018

Contents

1 Target

This report will focus on the cacheability of 3 different methods of grey-scaling. These will be compared to each other and the default method for grey-scaling.

These results can help decide on which algorithm to use in a facial recognition project. More and more small systems are getting an on-cpu cache. This cache should be very useful in situations where a lot of data needs to be read and processed by the cpu. Which is the case for facial recognition, where every frame needs to be read from memory and processed by the cpu. In this report, there will be an investigation using multiple methods of grey-scaling and seeing how well these methods behave with respect to the cache.

2 Hypothesis

We can't say anything about how they will perform against the default implementation, as we don't know how it works.

We suspect there will be very little difference between the different methods when they will be examined on their own. It is unlikely that the results will change with or without the facial recognition enabled. We do expect to see a lot of data going to the L2 cache. This because the cpu pre-fetcher should see the sequential reads from the pixel buffer and try to cache data ahead of these reads.

3 Method

The binary will run one of the grey-scaling methods 10000 times, and we will run this binary 10 times, after which we will aggregate the results. We will use the *perf* tool for collecting the data. *Perf* is a linux tool often used for measuring kernel performance. In this test we will look at both L1-dcache and L2 cache. Seeing how a single pixel is quite small in terms of the size of the data (3 * 8 bits). It could be benificial to know where most of the data ends up.

The test will be run twice for each method, once with the facial recognition, and once without.

We will make sure the tests are run on the same system and keep an eye out on the temperature to make sure it does not thermal-throttle.

Specifications of the used laptop:

CPU	Intel Core i5 6600K [1]
RAM	16GB DDR4 2400MHz
OS	Linux 4.15.15-1-ARCH

Used compiler optimisations:

- -std=c++14
- -g3
- -ggdb
- -g
- -O1

Used compiler warnings:

- -Wall

- -Wextra
- -pedantic-errors
- -Wfatal-errors
- -Wcast-align
- -Wmissing-declarations
- -Wredundant-decls
- -Wuninitialized
- -Wno-unused-parameter
- -Wno-missing-field-initializers

Perf command used:

```
$ perf stat -r 10 -d -e
    l2_rqsts.demand_data_rd_hit,l2_rqsts.demand_data_rd_miss ./vissen}
```

4 Results

With facial recognition:

Method	L1 DCache hits	L1 DCache misses	L2 Data hits	L2 Data misses
Default	209359655680	6524946521	596465950	50420831
Averaging	204605775027	5838956725	537926381	40272104
Luma	207067812232	5790118160	496829390	35546606
Decomposition (Max)[1]	-	-	-	-
Decomposition (Min)	206050242309	5817410748	516984118	37468228

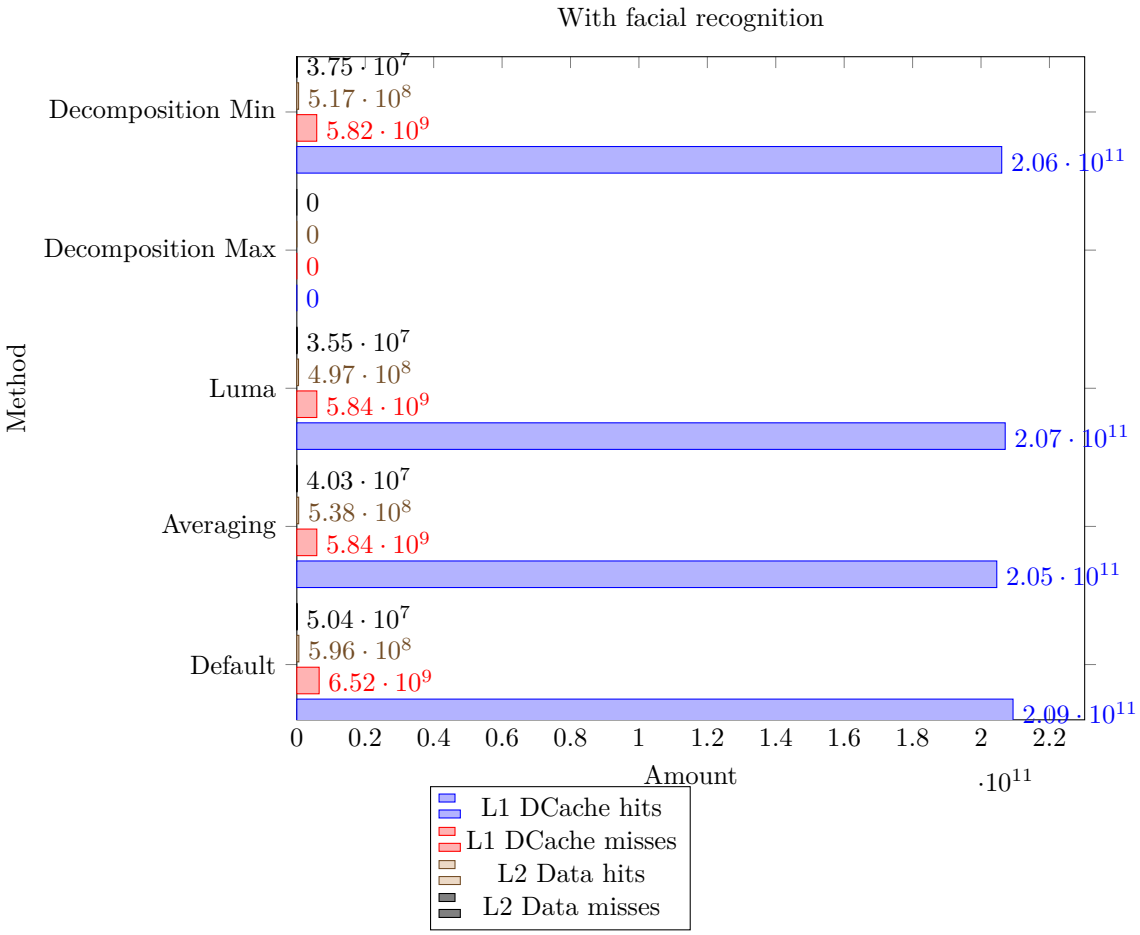
4.1 Raw data

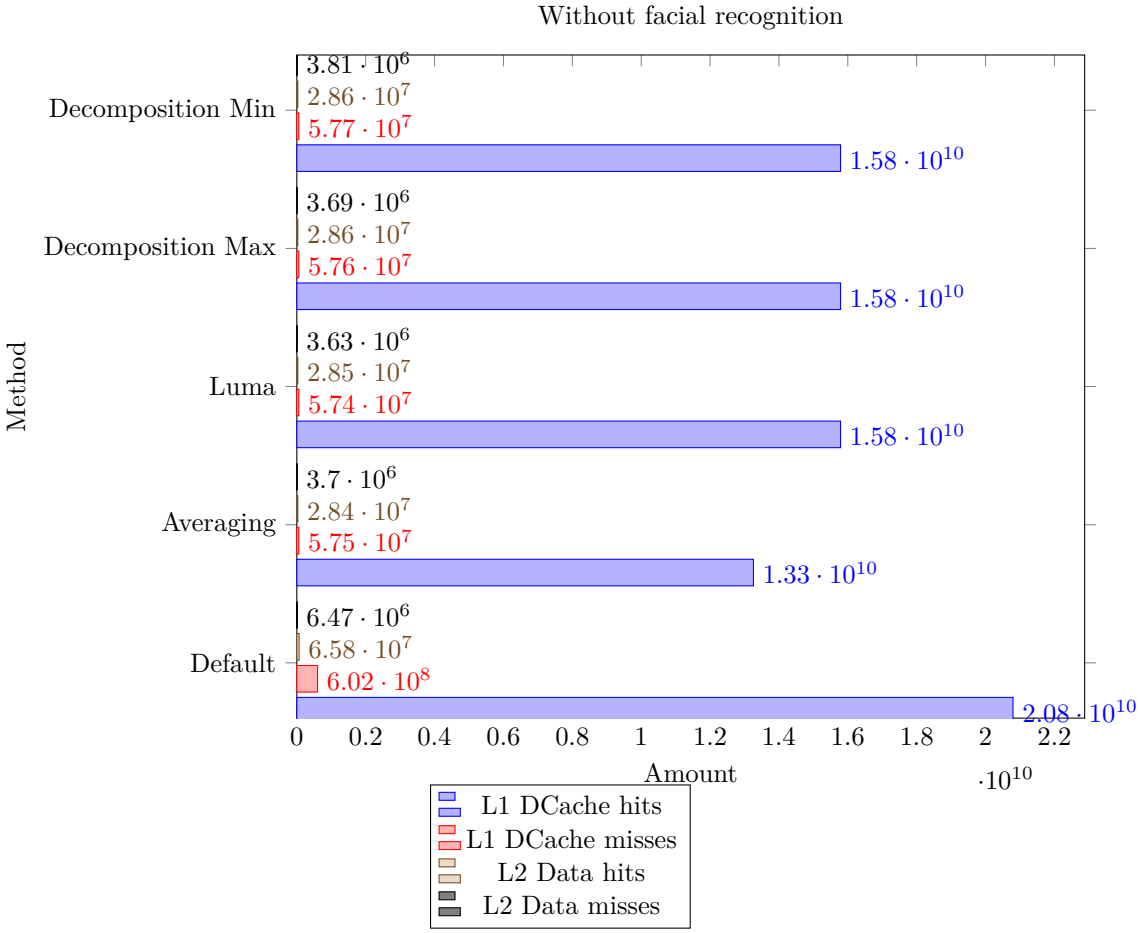
Without facial recognition:

Method	L1 DCache hits	L1 DCache misses	L2 Data hits	L2 Data misses
Default	20798177177	602197117	65785000	6469704
Averaging	13258521614	57533588	28390788	3701288
Luma	15789762340	57418201	28490336	3625812
Decomposition (Max)	15789802982	57631568	28642166	3686989
Decomposition (Min)	15789815633	57681407	28622821	3807790

[1] `Decomposistion (Max)` produced images so bad that these were unable to be used for facial recognition. This led to incorrect execution of the software, and therefore incorrect data.

4.2 Visualized





5 Processing

As predicted in our hypothesis the different methods are within margin of each other. Without facial recognition, it seems that *Default* has a bit more L1-DCache hits but also more L1-DCache misses.

According to our measurements it seems that both *Decomposition* and *Luma* perform almost identical.

6 Conclusion

From these measurements we can't come to a clear conclusion about a which is the better method. It seems that data from different aspects of these methods is needed to declare a winner of sorts. We can only really state that *Decomposition Max* is inadvisable as it produces bad images.

7 Evaluation

7.1 What went well

The project was a very cooperative experience for both parties involved. This cooperation went very well, everyone had their tasks and did them with equal amount of dedication. This made the project an overall pleasant experience.

7.2 What went wrong

We both had a vigorous start with the project, however it turned out to be very hard to keep up this speed. Eventually we started to have to divide our time to other projects that required our attention at the time. This made it so we had to do a lot of work near the deadline.

Also the original plan was to use OpenCL and SIMD for parallelism. Because of the state of the relevant tools, this turned out to be quite a bit harder than was originally thought.

7.3 What could be done differently

A better research on the more ambitious subjects should be done in order to avoid unnecessary delays.

References

[1] Intel Ark product specifications `https://ark.intel.com/products/88191/Intel-Core-i5-6600K`