Design Document

HW #3: Locality

By Jared Lieberman and Jordan Stone

Please estimate the expected cache hit rate for reads of each of the six operations in the table below. Assume that the images being rotated are much too large to fit in the cache.

	Row-major access (UArray2)	Column-major access(UArray2)	Blocked access (UArray2b)
90-degree rotation	4	4	2
180-degree rotation	1	6	2

Justifications:

- Assumptions:
  - an entire row can fit in the cache
  - an entire block can fit in the cache
- 90-degree rotation with row-major & col-major
  - We estimate that this hit rate will be 4 because there will be 1 miss per row loaded into the cache on the load, and there will be no hits when storing. This is the same as 90 degree rotation with column-major because both algorithms have the same ratio of hits to misses because in that case there will be no hits on the load and a single hit per row stored. The number of misses is quantified as the number of rows in the image plus the total number of cells in the image.
- 90-degree rotation with blocked access
  - We estimate that the hit rate will be 2 because there will be one miss per block for both loading and storing. Therefore the locality will be better than the row-major and column-major accesses for the 90-degree rotation. The number of misses is quantified as number of blocks*2.
- 180-degree rotation with row-major access
  - We estimate that this hit rate will be 1 because the algorithm applies well to this transformation. This is due to the fact that rows map to rows, and thus it is a quick process.
- 180-degree rotation with column-major

    - We estimate that this hit rate will be 6 because the algorithm does not apply well to this transformation. This is due to the fact that rows map to rows, and thus it is a very inefficient process mapping this transformation with column-major. The number of misses is quantified as number of cells times two.
- 180-degree rotation with blocked
    - We estimate that the hit rate will be 2 because there will be one miss per block for both loading and storing. Therefore the locality will be better than the column-major accesses for the 180-degree rotation. The number of misses is quantified as number of blocks*2.

Kind of rotation	adds/subs	multiplies	divs/mods	compares	loads	Hit rate	stores	Hit rate
180-degree row-major	5	1	0	5	1	(number of cells - number of rows)/(number of cells)	2	(number of cells - number of rows)/(number of cells)
180-degree col-major	5	1	0	5	1	0/(number of cells)	2	0/(number of cells)
180-degree block-major	10	7	5	9	1	(number of blocks * ((blocksize ^2) - 1))/(number of cells)	2	(number of blocks * ((blocksize ^2) - 1))/(number of cells)
90-degree row-major	3	1	0	5	1	(Number of cells - number of rows) / number of cells	2	0/(number of cells)
90-degree column major	3	1	0	5	1	0/(number of cells)	2	(number of cells- number of cols)/(number of cells)
90-degree block-major	8	7	5	9	1	(number of blocks * ((blocksize ^2) - 1))/(number of cells)	2	(number of blocks * ((blocksize ^2) - 1))/(number of cells)

Please estimate the expected speed of each of the six operations in the table below. Your speed estimate should include the cost of stores as well as the cost of loads.

	Row-major access	Column-major access	Blocked access
90-degree rotation	4	4	4
180-degree rotation	1	6	4

The blocked rotations are going to suffer in the performance ranking because of all of the necessary arithmetic and comparison operations which are time intensive. This ranking takes into account both the cache hits and the time spent doing math and other operations.