Resumen: "GPU maps for the space of computation in triangular domain problems"

Nancy Hitschfeld y Cristóbal Navarro.

There is a stage in the GPU computing pipeline where a grid of thread-blocks, or space of computation, is mapped to the problem domain. Normally, the space of computation is a k-dimensional bounding box (BB) that covers a k-dimensional problem. Threads that fall inside the problem domain perform computations and threads that fall outside are discarded, all happening at runtime. For problems with non-square geometry, this approach makes the space of computation larger than what is necessary, wasting many threads. Our case of interest are the two-dimensional triangular domain problems, alias td-problems, where almost half of the space of computation is unnecessary when using the BB approach. Problems such as the Euclidean distance map or collision detection are td-problems and they appear frequently as part of a larger computational problem. In this work, we study several mapping functions and their contribution to a better space of computation by reducing the number of unnecessary threads. We compare the performance of four existing mapping strategies; the bounding box (BB), the upper-triangular mapping (UTM), the rectangular box (RB) and the recursive partition (REC). In addition, we propose a map g(λ), that maps any λ block to a unique location (i, j) in the triangular domain.

The mapping is based on the properties of the lower triangular matrix and works in block space. The theoretical improvement I obtained from using g(λ) is upper bounded as $I < 2$ and the number of unnecessary blocks is reduced from $O(n^2)$ to $O(n)$. Experimental results using different Nvidia Kepler GPUs show that for computing the Euclidean distance matrix g(λ) achieves an improvement of up to 18% over the basic bounding box (BB) strategy, runs faster than UTM and REC strategies and it is almost as fast as RB. Performance results on shared memory 3D collision detection show that g(λ) is the fastest map of all, and the only one capable of surpassing the brute force (BB) approach by a margin of up to 7%. These results help us realize that one of the main advantages of g(λ) is the fact that it uses block space mapping, where coordinate values are small in magnitude and thread organization is not compromised, making the map stable in performance under different memory access patterns.