

ANALYSIS OF PARALLEL SURFACE MESH SMOOTHING METHODS

Q. Feng^a, Z. Wu^a, J.B. Weaver^{a, b} and K.D. Paulsen^a

^aThayer School of Engineering
Dartmouth College
Hanover, NH 03755-8000

^bDepartment of Radiology
Dartmouth-Hitchcock Medical Center
Lebanon, NH 03756
Ziji.Wu@Dartmouth.EDU

Surface mesh smoothing is necessary in many simulation and visualization studies in order to remove high frequency noise and to obtain a more realistic model appearance. Intensive computation and efficient memory allocation are required to process massive surface data sets. In order to achieve high quality results efficiently, the performance of parallelized smoothing methods have been evaluated by comparing to their sequential counterparts in terms of the execution time and the memory size required for computation and communication. In this article, three parallel surface smoothing methods are presented. The first two are tailored to shared memory systems (SMS), while the third is designed for distributed memory systems (DMS) and mixed architectures. All were derived from the conventional Laplacian algorithm, which is the most popular surface smoothing method. The parallel schemes obtained virtually identical smoothing effects as their sequential counterparts while exhibiting significant speedup and minimal memory overhead.

In SMS, all processors have equal access to all memory and there are no external communication issues involved. If one processor modifies a mesh point, all of the remaining processors will receive the updated value instantly. Two strategies were implemented. In the first scheme, all processors operate on the entire dataset simultaneously. In the second approach, the full dataset is distributed evenly among the processors. No additional memory is needed in either approach and the execution speedup is linearly proportional to the number of processors used. Because the entire data set must be available at all times, large memory is required in the SMS configuration. In addition, due to the high cost of SMS architectures, the number of processors is usually limited.

Relative to SMS each processor in a DMS maintains its own local memory. The entire surface dataset is distributed across the system. If a mesh point is modified by one processor, it will be updated in the other processors only through explicit communication. Workload distribution and efficient communication tactics were investigated in the DMS setting of parallelized surface mesh smoothing. The approach developed strives to improve performance by limiting communication costs.

The parallel methods presented in this paper were evaluated numerically against their sequential equivalents with two criteria, $\lambda = \frac{T_s}{\sum_i T_{P_i}}$ and $\kappa = \frac{M_s}{\sum_i M_{P_i}}$, where T_s and M_s are the run time and memory

consumption of the sequential code, T_{P_i} and M_{P_i} are the measurements on each processor in the parallel versions, and N is the number of processors. Apparently, both criteria are in the range of (0, 1], with the optimal strategy producing values of unity, which indicates N times of speedup without any memory overhead. Experiments with the parallel surface smoothing methods developed in this article at different workload show that the SMS schemes are optimal while the DMS method is nearly optimal.