Wall Distance Search Algorithm Using Rasterized Marching Spheres

Beatrice Roget [*] [1] and Jayanarayanan Sitaraman [†1]

[1] *Department of Mechanical Engineering, University of Wyoming, Laramie, WY*

1 Introduction, Objective and Methodology

Minimum distance to a solid wall is a primary parameter that is utilized in most turbulence models - e.g in the commonly used Spalart-Allmaras turbulence model, the turbulence destruction source term is inversely proportional to the square of the wall distance d. For a grid with N_f field points and N_b boundary faces, the direct exhaustive computation of wall distances are of $O(N_f \times N_b)$, which is quite expensive for grids utilized in the state-of-the-art RANS based CFD calculations (where N_b is of order hundred thousand and N_f is of order several million). Therefore, several efficiency improvement algorithms have been devised to compute the wall distance. They can be characterized in to three groups:

1. $k-d$ tree based search approaches [1, 2, 3], which is an adaptation of the classic nearest-neighbor search.
2. Differential equation based approaches [4, 5]
3. Advancing surface front type methods [6]

In a parallel computing environment, all of the above approaches suffer from scalability and accuracy issues. The $k-d$ tree approach constructs a digital tree of the boundary faces and follows a divide-and-conquer algorithm by eliminating regions that do not intersect with the sphere corresponding to the "current best" minimum distance. This is a quite efficient approach for points that lie close to the boundary surface. However as points are further away from the surface, lesser number of regions and hence branches of trees can be eliminated from comprehensive checking leading to poor scalability - because grid partitions lying further from the surface will incur more computations compared to those that lie closer to the surface. In contrast, the differential equation based approaches (Poisson, Eikonal) are quite scalable and much easier to implement as parallel algorithms. However, the accuracy of the wall distances are driven by the order of accuracy of the discrete approximation (of the governing PDE) as well as the level of convergence obtained. True minimum distance can be achieved only in the limit of zero grid spacing and machine-zero convergence. Advancing front methods have inherent defects in scalability as processes controlling partitions further away have to wait idle until the front approaches them after passing through closer partitions. In summary $k-d$ tree approaches can provide the true-estimate of the wall distance, but are not scalable. Differential equation based approaches are scalable, but only provide approximate estimate of the wall distance and advancing surface front type methods are neither scalable nor accurate.

The objective of this work is to explore a different algorithm, still based on searching rather than differential equations, that is capable of mitigating the scalability and accuracy issues mentioned above. In contrast to $k-d$ trees, we utilize a structured auxiliary mesh (SAM) for facilitating divide-and-conquer. We summarize the major steps of the algorithm below:

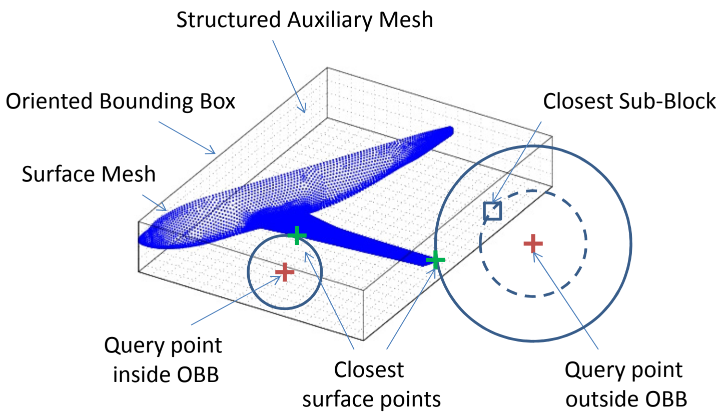
1. Preprocessing:
   (a) *Build SAM*: SAM can be built by just constructing an oriented bounding box of the boundary faces and equally dividing it in each of the coordinate direction using a suitable metric based on the statistic of the boundary face size - e.g. size of a cell in SAM can be equal to 10 times the length of the mean boundary face size

[*] Senior Research Scientist
[†] Asssistant Professor

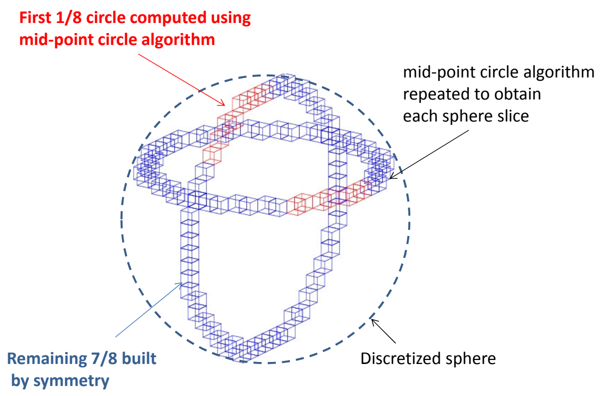
(b) *Populate SAM*: the boundary face list is traversed linearly tagging all the cells of the SAM that intersect them - this is a very simple computation owing to the Cartesian nature of the SAM. At the end of this step, each cell of the SAM will have information of all the boundary faces that have overlap with it.

2. *Search* : For any given field point, first locate the closest point in SAM and construct an expanding spherical front using a rasterized representation. The concept of rasterization is borrowed from computer graphics and is weakly equivalent to a 3-D extension of the Bresenham's circle drawing algorithm. It is worth noting that only integer arithmetic is needed for constructing rasterized spheres. Once the expanding spherical front touches cells of SAM that contain boundary faces, a comprehensive check is performed on all these boundary faces to achieve the true minimum distance.

A graphical description of the algorithm described above is shown in Figure 1.



(a)


First 1/8 circle computed using mid-point circle algorithm
mid-point circle algorithm repeated to obtain each sphere slice
Remaining 7/8 built by symmetry
Discretized sphere

(b)

Figure 1: (a) Oriented bounding box, structured auxiliary mesh and expanding sphere (b) Rasterized sphere generation

The key advantage of this algorithm compared to the $k-d$ tree approach lies in the amount of spatial elimination that can be achieved. Owing to the spherical marching, even for field points that are further away, the number of boundary faces that need to be comprehensively checked are comparable to those for field points that lie closer- e.g. in the limit of a field point lying infinitely far away, the $k-d$ tree degenerates to an exhaustive comprehensive check of all the boundary faces, while the spherical marching will still eliminate a significant group of bounding faces from being comprehensively checked. Furthermore, the algorithm described above is completely grid agnostic (i.e. does not require description of the underlying volume grid as in the case of differential equation based and advancing front approaches) and hence very modular.

2 Preliminary Results and Projection for Final Paper

In Figure 2 we show timing results of the wall distance calculation for a DLR-F6 geometry. Results are encouraging with about two orders of magnitude in speedup obtained when compared with the exhaustive search. For the final paper, we expect to demonstrate and report the following:

1. Complete description of all facets of the algorithm with underlying mathematical framework and data structuring. Extension of the approach to fully parallel computations.

2. Demonstration of parallel scalability of the algorithm - with the field points partitioned and the boundary surface maintained unsplit in all processes

3. Demonstration of parallel scalability of the algorithm - with both the field points and the boundary surface partitioned among processes.

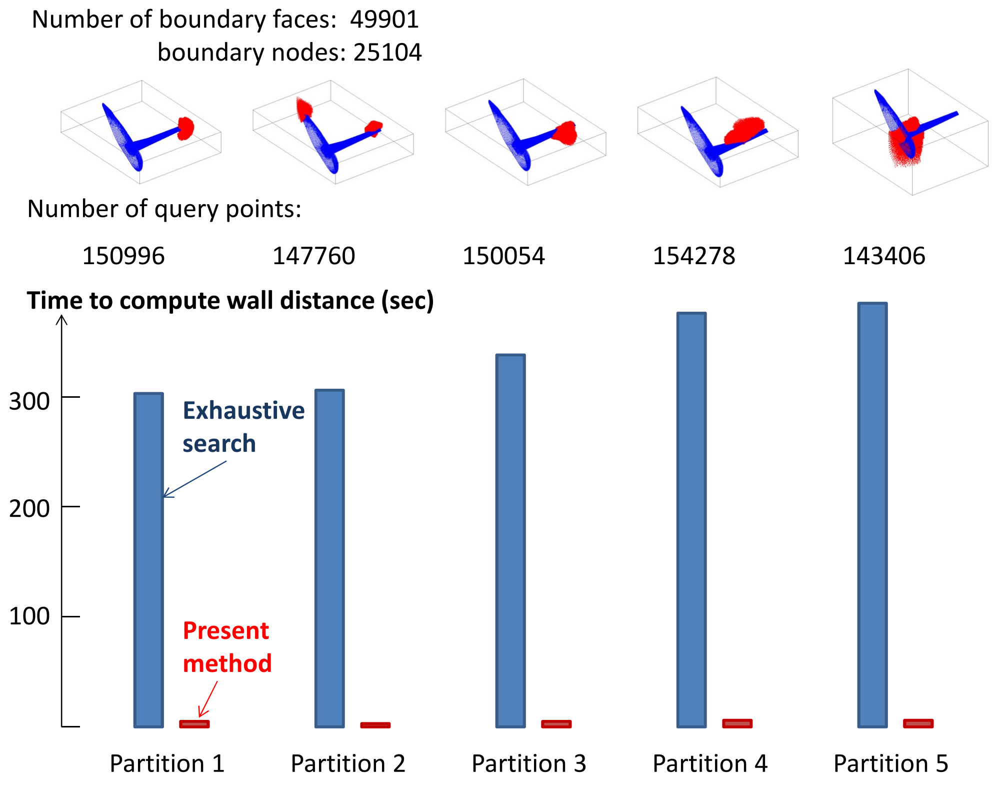


Figure 2: Comparison of exhaustive search and the spherical marching algorithm for 5 grid partitions of the DLR-F6 geometry.

References

[1] van der Weide, E., Kalitzin, G., Schluter, J., Alonso, J.J., "Unsteady Turbomachinery Computations Using Massively Parallel Platforms," 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA Paper 2006-0421, Reno, NV, January 2006.

[2] Boger, A., D. "Efficient Method For Calculating Wall Proximity," *AIAA Journal*, vol. 39, no. 12 (2404-2406), 2001.

[3] Sethian, J., A, "Fast Marching Methods", *SIAM review*, 41(2), 199-235 (1999)

[4] Tucker, P. G., "Differential Equation Based Wall Distance Computation For DES/RANS", *Journal of Computational Physics*, 190(2003), 229-248.

[5] Xu, J., Yan, C., Fan, J., "Computations of wall distances by solving a transport equation," *Appl. Math. Mech*, 32(2), 141-150 (2011).

[6] Lohner, R., Sharov, D., Luo, H. and Ramamurthi, R., "Overlapping Unstructured Grids," AIAA 2001-0439, Reno, NV, January 2001.