Eurographics Conference on Visualization (EuroVis) (2013)
M. Hlawitschka and T. Weinkauf (Editors)

Toward Wall Function Consistent Interpolation of Flow Fields

Markus Üffinger[1], Filip Sadlo[1], Claus-Dieter Munz[2], and Thomas Ertl[1]

[1]Visualization Research Center University of Stuttgart (VISUS), Germany
[2] Institute of Aerodynamics and Gasdynamics (IAG), University of Stuttgart, Germany

Abstract

In this paper we provide a first step toward simulation-consistent visualization techniques. We focus on wall functions modeling near-wall flow in computational fluid dynamics (CFD) using the law of the wall. By integrating these functions, which are effective only in cells adjacent to solid boundaries, with traditional interpolation schemes used in the interior of the domain, we obtain results that account for the simulation model. We demonstrate the advantages of our scheme using flow visualization techniques on two three-dimensional CFD examples.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications Physical Sciences and Engineering—

1. Introduction

Many phenomena subject to scientific visualization are present in the form of continuous fields. But while major efforts are taken on the simulation side to accomplish appropriate models and to obtain accurate solutions thereof, it is still common practice to transfer the resulting data to the visualization side in terms of discrete values only. The fact that there the field continuum has to be reestablished from these values by means of interpolation is often ignored—the visualization side typically assumes tensor-product linear (bilinear in 2D and trilinear in 3D) interpolation for the analysis. This is no surprise since large parts of visualization techniques still rely on (and exploit the simplicity of) these interpolation schemes. Consequently, we propose that simulation results be accompanied by interpolation schemes that are consistent with the models that produced them.

In the context of computational fluid dynamics (CFD) the Navier-Stokes equations have to be solved [FP02]. Direct numerical simulation discretizes these equations directly. In the case of turbulent flow, however, this implies that all temporal and spatial scales have to be resolved, typically necessitating extremely fine grids and small time steps. In practice this is often not feasible, in particular for flows with high Reynolds numbers, which are found in many engineering applications. Such flows are strongly affected by the flow behavior near no-slip boundaries. The zero velocity condition on these surfaces causes strong shear forces, leading to the formation of a so-called *boundary layer*, a region where the velocity magnitude is smaller than 99% of the free-flow velocity further away from the boundary. The phenomena taking place within this layer are a major source of vorticity and the generation of turbulence [SG00]. In the cases where the small length scales of turbulence cannot be resolved with the grid, so-called subgrid-scale models are typically employed to account for the influence of the unresolved scales. For engineering applications Reynolds-averaged Navier-Stokes (RANS) solutions are commonly employed in combination with turbulence models, allowing for relatively coarse simulation grids. However, as the boundary layer is typically very thin and would require a comparably high resolution, often, high Reynolds number models are used in addition, harnessing analytic *wall functions*, like the *law of the wall*, to model the characteristic near-wall velocity distribution.

As near-wall flow is of great importance to the global characteristics of the flow field and development of turbulence, both accurate simulation and visualization of these regions is of particular importance. To this end, we try to help to close the gap between simulation techniques that employ subgrid-scale models and visualization techniques, which typically ignore these models. Specifically, we provide an improved interpolation scheme that applies to cells adjacent to no-slip boundaries where the simulation uses a wall function. Note that on the simulation side the wall functions are employed similar to boundary conditions, while on the visualization side we derive a respective interpolation scheme. The increased consistency with the simulation model not

DOI: 10.2312/PE.EuroVisShort.EuroVisShort2013.085-089

only improves the quality of the analysis, but should also help debugging of CFD solvers. While our approach works well for convex no-slip boundaries in unstructured grids, it is restricted to rectangular cells in concave no-slip boundary regions. Nevertheless, we see our approach as a first important step that shall trigger future work.

2. Related Work

There are only comparably few visualization techniques that specifically target analysis of near-wall flow, e.g., flow separation [KHL99], topology [LGD*05], and generation of vortices [WTS*07, SPS06]. Petz et al. [PPG*08] presented a method that flattens boundary geometry in order to alleviate the analysis near curved walls. Nevertheless, to the best of our knowledge, there are so far no visualization techniques that account for subgrid-scale models such as wall functions. However, with higher-order simulation methods becoming more widespread, the interest in accurate visualization of cell-wise polynomial fields has increased in recent years [SUP*11]. Methods have been presented that are able to evaluate the nonlinear polynomial fields directly and accurately, e.g., for ray casting isosurfaces [NK06, PVS*11] or interactive direct volume rendering [UFE10]. While tensor-product linear interpolation schemes are successfully employed in tetrahedral and hexahedral grids, *mean value interpolation* [JSW05] has become popular for interpolation in more general polyhedral grids [MHDG11], in particular if complex field variations are observed on the faces of the grid. In our approach we make use of mean value interpolation for combining the wall function based scheme with the traditional interpolation used away from the boundaries. Thereby, mean value interpolation guarantees C^0 continuity.

3. Wall Function Consistent Interpolation

For a self-contained description and to motivate our interpolation scheme, we first describe how the data that we address are simulated (Section 3.1), and based on this we then describe our approach (Section 3.2).

3.1. Simulation Model

The law of the wall models the Reynolds-averaged wall-tangential velocity component $\bar{u}_t$ of a fluid flow near no-slip boundaries at high Reynolds numbers [SG00]. The relationship between $\bar{u}_t$ and the wall distance y is typically given by the respective dimensionless variables

$$u^+ = \frac{u_t}{u_\tau} \quad (1) \qquad \text{and} \qquad y^+ = \frac{\rho u_\tau y}{\mu} \quad (2)$$

with friction velocity $u_\tau = \sqrt{|\tau_w|/\rho}$, wall shear stress τ_w, density ρ, and dynamic viscosity μ. A linear relationship $u^+ = y^+$ can be observed in the direct proximity of the wall within $y^+ \leq 5$, where viscous effects dominate. This region is called the viscous sublayer. A transitional buffer layer ($5 \leq y^+ \leq 30$) connects the linear relationship with the logarithmic relationship

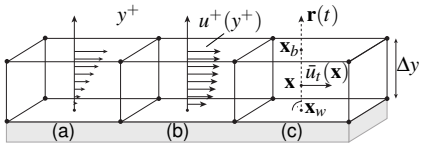

Figure 1: *Boundary cell layer with wall (gray) and bilinear face layer (faces opposite to wall). Trilinear interpolation (a) within the boundary layer is a poor approximation of (b), the velocity profile according to the law of the wall. (c) Evaluation of the wall function at a point* $\mathbf{x}$.

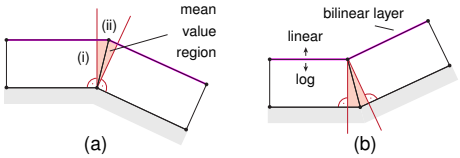

Figure 2: *Convex (a) and concave (b) boundaries with mean value regions (red) and the bilinear face layer (pink).*

$$u^+ = \frac{1}{\kappa} ln(y^+) + C \quad (3)$$

observed in the turbulent log-law layer ($30 \leq y^+ \leq 300$). The logarithmic relationship includes the von Kármán constant $\kappa = 0.41$, and a constant C that reflects the dimensionless thickness of the viscous sublayer, e.g., $C = 5.5$ for a smooth planar wall. Popular turbulence models, like the $k-\varepsilon$ model, are based on the turbulent kinetic energy k, which describes the kinetic energy of velocity fluctuations. Many solvers that use such models harness the relationship [LS74]

$$u_\tau = C_\mu^{1/4} \sqrt{k} \quad (4)$$

with constant $C_\mu = 0.09$ together with (3) to compute the friction velocity and then derive the wall shear stress from the field values k, $\bar{u}_t$, etc. at the first grid nodes off the no-slip boundary. Consistently, the simulation uses the wall function only within the layer of cells that is in contact with the no-slip boundary. In simulations using unstructured grids, near-wall regions are preferably discretized with several layers of hexahedra, as illustrated in Figure 1. If a wall function is employed, the first off-wall nodes are required to lie within the log layer, i.e., where $30 \leq y^+ \leq 300$.

3.2. Interpolation Scheme for Visualization

Figure 1 shows a common near-wall discretization with a layer of hexahedra. Consistent with the simulation, we evaluate the wall function only within the layer of boundary cells adjacent to no-slip boundaries, which we call the *boundary cell layer*. The figure illustrates that the velocity profile obtained with trilinear interpolation (a) is a poor approximation of the logarithmic profile observed near the wall in experiments (b). We define the *bilinear face layer* to consist of

all off-wall faces of the boundary cell layer. It connects the boundary cell layer with the interior of the domain, where the traditional interpolation scheme is employed (see Figure 2). In the following, we describe how the law of the wall (in particular Equation (3)) is evaluated at a point $\mathbf{x}$ within the boundary cell layer, i.e., how it is incorporated in the original interpolation scheme (cf. Figure 1(c)). First, the dimensionless distance from the wall $y^+(\mathbf{x})$ is obtained. The point $\mathbf{x}_w$ on the wall nearest to $\mathbf{x}$ is determined, and a ray $\mathbf{r}(t) = \mathbf{x}_w + t(\mathbf{x} - \mathbf{x}_w)$ is intersected with the bilinear face layer, resulting in the point $\mathbf{x}_b$. The traditional 2D interpolation scheme is then used within the "bilinear" cell face that contains $\mathbf{x}_b$ to interpolate the required field values at $\mathbf{x}_b$, including $\mathbf{u}$, y^+, and k. Common solvers, like CFX [ANS10a], provide all necessary quantities, such as k, μ, ρ, $y+$ etc., at the first off-wall nodes. Note that in our application the traditional 3D interpolation is trilinear and hence the respective 2D interpolation on a face is bilinear. However, other interpolation schemes would likewise fit into our approach. Evaluating (4) with the interpolated turbulent kinetic energy $k(\mathbf{x}_b)$ yields the friction velocity u_τ. With this, and the normal distance $y(\mathbf{x})$, the dimensionless distance $y^+(\mathbf{x})$ is computed according to (2). Then, the log law (3) can be evaluated to obtain $u^+(\mathbf{x})$. Special treatment is required if $\mathbf{x}$ lies within the viscous sublayer, where $y^+(\mathbf{x}) < y_v^+$. We chose a constant threshold of $y_v^+ = 11.06$, consistent with the used solver [ANS10b], and employ linear interpolation there:

$$u_v^+(\mathbf{x}) = \frac{y^+(\mathbf{x})}{y_v^+} u^+(y_v^+) \tag{5}$$

in order to bridge the buffer layer gap and ensure C^0 continuity at the boundary distance y_v^+. Transforming u^+ with (1) results in the tangential velocity $\bar{u}_t(\mathbf{x})$. While the simulation enforces zero velocity at the no-slip walls, the first off-wall nodes can exhibit a velocity component $\bar{u}_\perp(\mathbf{x})$ orthogonal to the wall. Since the law of the wall assumes planar walls and wall-tangential flow, the (simulation) model does not apply in such cases. Employing our approach only for $\bar{u}_t(\mathbf{x})$ and interpolating $\bar{u}_\perp(\mathbf{x})$ with the traditional 1D scheme, in our case linear, however, provided poor results: streamlines did not detach correctly at the step in Figure 3. Interpolating $\bar{u}_\perp(\mathbf{x})$ according to $\bar{u}_t(\mathbf{x})$ provided much better results. Hence, our interpolation approach is $\bar{\mathbf{u}}(\mathbf{x}) = \bar{u}_t(\mathbf{x}) \cdot \bar{\mathbf{u}}(\mathbf{x}_b)/\bar{u}_t(\mathbf{x}_b)$. A detailed examination is left as future work.

The interpolation scheme as described so far can be only applied if the simulation mesh provides a well-defined wall normal at $\mathbf{x}_w$, i.e., it cannot handle the case if $\mathbf{x}_w$ lies on an edge or on a vertex of the wall faces and if these faces are not coplanar there, e.g., if the wall exhibits a convex configuration at the edge or vertex, see Figure 2(a). Another so far unsupported case are multiple points (candidates for $\mathbf{x}_w$) on the wall where the normal points to the sample point under consideration, see Figure 2(b). Both cases result in unsupported ambiguous regions (red in Figure 2), within which we use mean value interpolation to ensure C^0 continuity of

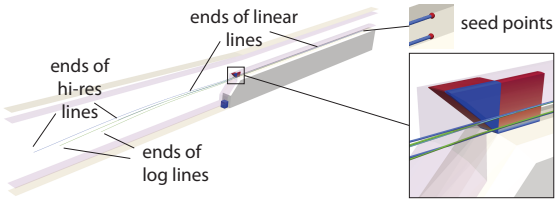

Figure 3: *Step with curved boundary and streamlines from hi-res field (green), log interpolation (blue), trilinear (light blue). The closeup box shows two mean value regions induced by convex edges (red) and one by a vertex (blue).*

our overall interpolation approach. Mean value interpolation obtains an interpolation within a volume from values on a triangular mesh that encloses the volume [JSW05]. These "red" regions are denoted as *mean value regions*, henceforth. Each "nonplanar" vertex or edge on the no-slip wall induces one mean value region, as illustrated with the curved channel example in Figure 3. The surface representation of each mean value region (the closed triangle mesh) is constructed in a preprocessing step, using the CGAL library [CGA] by cutting the involved cells with the planes indicated by the red lines in Figure 2 and subsequent merging of the "red" parts. The faces of the resulting polyhedron that were obtained by cutting with the aforementioned planes have to fit our wall function consistent interpolation scheme and thus have to be, due to its nonlinearity, subdivided and the values at the resulting vertices set by our new interpolation scheme. Triangles of the polyhedron lying within the bilinear layer are also subdivided, using the traditional scheme to compute the field values. Our overall interpolation approach is evaluated at runtime within a specific mean value region by accessing its precomputed surface mesh together with the values at its vertices and applying mean value interpolation.

A main reason for publishing our approach as a short paper are the difficulties in obtaining the mean value regions if concave boundary edges or vertices are involved. While it is straightforward to obtain the mean value regions for convex edge or vertex wall regions, it turned out to be intricate to obtain them for concave or concave/convex cases, as those mean value regions can intersect, be oblique, consist of several parts, or even intersect at remote regions in distorted grids. While our approach so far supports any unstructured grid consisting of hexahedra as long as the boundary is convex, we support concave wall regions in our current implementation only if the involved cells are rectangular with the following workaround. This workaround is better than traditional trilinear interpolation in these regions but does not employ the wall function to the full extent. To find such a formulation for generic configurations is, from our experience, nontrivial and we would like to trigger future research in this direction with the present paper. In our workaround we avoid the determination of the mean value regions involving concave wall edges or vertices and instead treat the whole cell that exhibits a concave edge or vertex as a mean

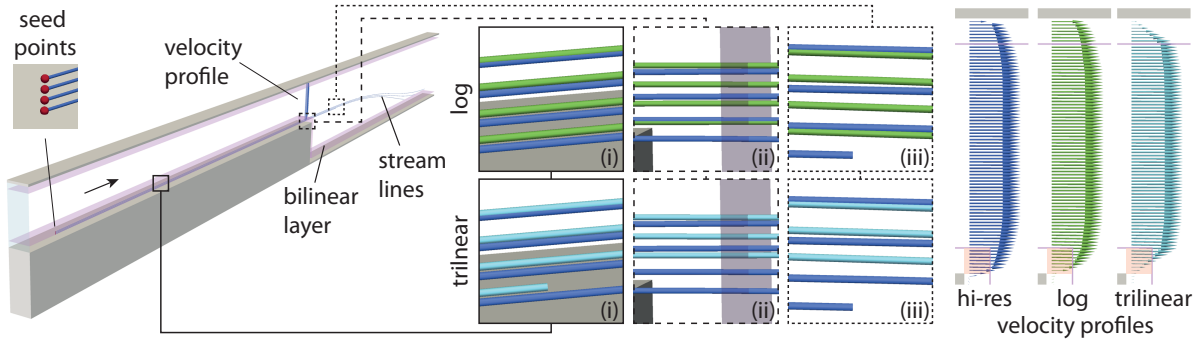

Figure 4: *Channel flow with a step. The bilinear layer (pink) separates the log region near the wall from the interior trilinear region. Closeups (i)–(iii) compare streamlines using trilinear (light blue) and log interpolation (green) with streamlines from* $4\times$ *hi-res field (blue). On the right a comparison of velocity profiles shows both, mean value region (red) and log region (top).*

value region (note that due to our restriction on rectangular grids in concave wall regions there is exactly one cell that is adjacent to a concave edge or vertex).

4. Results and Evaluation

We evaluate our approach at the example of two flows that were computed with a RANS solver in CFX. Both cases study the stationary solution of an incompressible fluid at high Reynolds numbers near no-slip walls. The $k-\varepsilon$ turbulence model was employed with a wall function. Details are given in [ANS10b]. Using local field evaluations and streamlines, our interpolation scheme (log) is compared to traditional trilinear interpolation on the same grid and to a reference simulation of four times higher resolution (hi-res).

The first example (Figure 4) is a channel flow with a step, discretized with a regular grid. The flow enters on the left and leaves the domain on the right. The bilinear face layer is represented by the transparent pink surface, demarking the interior from the region where the wall function is applied. Figure 4 (right) compares velocity profiles near the step. Trilinear interpolation shows an unnatural cutoff within the wall cells, compared to the profile obtained from the trilinearly interpolated simulation of four times higher resolution. Our approach (log), in contrast, reproduces the logarithmic flow behavior near the wall, even with the chosen coarse grid resolution. We also computed streamlines traversing the boundary region, all with equal integration time. The closeups in Figure 4 show that trilinear interpolation strongly underestimates the velocity near the wall. The wall-nearest line, e.g., already ends at (i). The underestimation of trilinear interpolation is also visible at the corresponding line in the hi-res field. It also ends early, before our lines (iii). Note that the vertical deviation to the "hi-res lines" is also caused by deviations between the hi-res and low-res simulations. Region (ii) demonstrates that our mean value interpolation approach does not introduce additional error. The second example (Figure 3) is also a channel but with a non-rectangular step and additional curvature along its edge. Again, the streamlines from our approach are more consistent to the lines from the hi-res field than those from trilinear interpolation.

Our single-threaded CPU implementation of the interpolation schemes take 4 ms (trilinear), 23 ms (log law), and 1594 ms (mean value), for 1000 evaluation points, on an Intel Xeon X5550. While the overhead of log interpolation is acceptable, the mean value interpolation takes much longer. It directly depends on the chosen resolution of the mean value mesh. Here, we chose a mean value region of the first data set consisting of 1162 vertices and 2052 triangles. To improve the timings, one could resample the mean value regions in a preprocessing step, using a fine rectilinear grid, and then use trilinear interpolation at runtime. The same could be done with the log regions, using refinement when approaching the wall. However, while resampling offers a more generic approach for improved visualization of logarithmic wall behavior with standard visualization software, it can only provide an approximation of the wall function.

5. Conclusion

The presented interpolation technique accounts for wall functions modeling the law of the wall, which is harnessed by many CFD solvers. We demonstrated that our approach leads to results that are more consistent with the solver model than traditional trilinear interpolation near walls and that this allows for more consistent analysis. While the $k-\varepsilon$ turbulence model was used in our examples, the technique is in principle applicable to all simulations based on wall functions. This is, e.g., often the case for $k-\omega$ or Reynolds stress models. Our work demonstrates the potential of making visualization techniques more consistent with the respective simulation models. It is clear that this work is only a first step toward wall function consistent interpolation and we hope to trigger future work and awareness on the topic.

This work was supported in part by the Cluster of Excellence in Simulation Technology (EXC 310/1) and the Collaborative Research Centre SFB-TRR 75 at the University of Stuttgart.

References

[ANS10a] ANSYS: *ANSYS CFX-solver modeling guide*, 13 ed. ANSYS Inc., 2010. 3

[ANS10b] ANSYS: *ANSYS CFX-solver theory guide*, 13 ed. ANSYS Inc., 2010. 3, 4

[CGA] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org, last visited April 19, 2013. 3

[FP02] FERZIGER J. H., PERIC M.: *Computational methods for fluid dynamics*, 3 ed. Springer, 2002. 1

[JSW05] JU T., SCHAEFER S., WARREN J.: Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005) 24*, 3 (2005), 561–566. 2, 3

[KHL99] KENWRIGHT D., HENZE C., LEVIT C.: Feature extraction of separation and attachment lines. *IEEE Transactions on Visualization and Computer Graphics 5*, 2 (1999), 135–144. 2

[LGD*05] LARAMEE R., GARTH C., DOLEISCH H., SCHNEIDER J., HAUSER H., HAGEN H.: Visual analysis and exploration of fluid flow in a cooling jacket. In *IEEE Visualization 2005* (2005), pp. 623–630. 2

[LS74] LAUNDER B., SPALDING D.: The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering 3*, 2 (1974), 269–289. 2

[MHDG11] MUIGG P., HADWIGER M., DOLEISCH H., GRÖLLER E.: Interactive volume visualization of general polyhedral grids. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2011) 17*, 12 (2011), 2115–2124. 2

[NK06] NELSON B., KIRBY R. M.: Ray-tracing polymorphic multidomain spectral/hp elements for isosurface rendering. *IEEE Transactions on Visualization and Computer Graphics 12*, 1 (2006), 114–125. 2

[PPG*08] PETZ C., PROHASKA S., GOUBERGRITS L., KERTZSCHER U., HEGE H.-C.: Near-wall flow visualization in flattened surface neighborhoods. In *Simulation and Visualization (SimVis 2008)* (2008), pp. 93–106. 2

[PVS*11] PAGOT C., VOLLRATH J., SADLO F., WEISKOPF D., ERTL T., COMBA J.: Interactive isocontouring of high-order surfaces. In *Dagstuhl Follow-Ups 2* (2011), pp. 276–291. 2

[SG00] SCHLICHTING H., GERSTEN K.: *Boundary-layer theory*, 8th ed. Springer, 2000. 1, 2

[SPS06] SADLO F., PEIKERT R., SICK M.: Visualization tools for vorticity transport analysis in incompressible flow. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2006) 12*, 5 (2006), 949–956. 2

[SUP*11] SADLO F., ÜFFINGER M., PAGOT C., OSMARI D., COMBA J. L. D., ERTL T., MUNZ C.-D., WEISKOPF D.: Visualization of cell-based higher-order fields. *Computing in Science and Engineering 13*, 3 (2011), 84–91. 2

[UFE10] ÜFFINGER M., FREY S., ERTL T.: Interactive high-quality visualization of higher-order finite elements. *Computer Graphics Forum 29*, 2 (2010), 115–136. 2

[WTS*07] WIEBEL A., TRICOCHE X., SCHNEIDER D., JAENICKE H., SCHEUERMANN G.: Generalized streak lines: analysis and visualization of boundary induced vortices. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2007) 13*, 6 (2007), 1735–1742. 2