122nd ASEE Annual Conference & Exposition
June 14 - 17, 2015
Seattle, WA
Paper ID #11877

Viscous Fluid Dynamics App for Mobile Devices Using a Remote High Performance Cluster

Mr. Jared Rayleigh Wilson

Jared Wilson received his B.S. and M.S. in Mechanical Engineering from the University of Oklahoma in 2014. He is currently working as a mechanical engineer for Leidos Engineering within Oklahoma City designing commercial piping systems for both private and public sectors.

Dr. Kurt C. Gramoll, University of Oklahoma

Prof. Kurt Gramoll is currently the Hughes Centennial Professor of Engineering at the University of Oklahoma. He has previously taught at the University of Memphis and Georgia Tech. He graduated from Virginia Tech with a PhD in Engineering Science and Mechanics in 1988. His research includes development and implementation of educational technologies for engineering education and training that utilize simulations.

Viscous Fluid Dynamics App for Mobile Devices Using a Remote High Performance Cluster

Abstract

Classrooms and the learning process are becoming increasingly interactive as students shift toward mobile learning platforms, yet there is a distinct lack of engineering mobile apps. This research attempts to address this issue by developing and implementing an online, interactive mobile app for fluid flow, Flow HPC, which enhances the engineering student's access to basic fluid flow information. The tool models fluid flow around any two-dimensional cross-section, and allows students to interactively experience many fundamental aspects of fluid dynamics including viscid and inviscid flows, instantaneous drag and lift coefficients, and a visualization of velocity vectors, pressure distributions, and streamlines about a two-dimensional object.

By taking advantage of a remote, high performance cluster (HPC), the relatively low computational power of mobile devices was alleviated. Educationally, this allows the student to access a finite element fluid flow package outside of the class and without any additional cost. This permits anyone with an internet access to solve intensive engineering problems on a smartphone or tablet at their convenience. Since the solution is done at a remote cluster with dozens of central processing units or CPU (hundreds of cores), the local client CPU is not relevant other than minor drawing routines. The cluster can also accommodate hundreds of simultaneous users (estimated upper limit is 500 users).

Flow HPC has been developed to allow the user to interactively specify the shape of an object within a 2D flow field as well as the velocity, density, and dynamic viscosity of the flow. The current version of the tool includes cylindrical and elliptical shapes. The tool has been used by students in a basic fluid dynamics course to help them determine drag for objects in solving flow problems. This is used as a second source for drag coefficients, the first being engineering handbooks (or textbook appendices). It has not been used to teach computational fluid dynamics, CFD, or finite elements. Student's positive feedback on using the tool for classroom discussions and assignments in a traditional fluid dynamics class is presented. Flow HPC was constructed as a reference tool to help students solve standard fluid mechanic problems. The program can be downloaded at no cost from Google Play Store, Apple App Store, and Amazon App Store.

Technically, Flow HPC produces and solves a Galerkin formulation of the 2D primitive variable Navier-Stokes equations, i.e. velocity and pressure. The Galerkin formulation produces a set of nonlinear equations. After Picard linearization, a sparse linear equation solver, PARDISO[1] from the Intel Math Kernel Library (MKL), was wrapped inside a Picard iteration scheme to converge on the solution. Currently, turbulence is not modeled, and only low Reynolds Number (<50) are analyzed. Future plans are to include more shapes, unsteady flow, and turbulence.

Introduction

Classrooms and the learning process are becoming increasingly interactive as they shift toward more mobile and accessible platforms. Students and teachers are able to use relatively small devices to accomplish computationally powerful tasks. These tasks range from watching recorded lectures, reading online notes to looking up supplementary information such as data or relevant charts. Within engineering, this push towards digital, interactive media is also increasing, yet there is a distinct lack of engineering mobile apps. The following paper meets the needs of the emerging market for interactive engineering apps via smartphones and tablets as well as furthers the idea of creating free, easy, and straightforward access to relevant engineering data.

Mobile apps and tools related to engineering are an emerging market as the number of smartphones and tablets increase in the classroom setting. This paper presents a viscous fluid flow app, Flow HPC, which was developed to supplement an engineering student's understanding of fluid mechanics, and to act as a secondary source of drag coefficients around two-dimensional objects. Galerkin's method of weighted residuals was applied to the primitive variable finite element formulation to minimize the error in the approximate solution of the steady, two-dimensional Navier-Stokes equations.

Within the field of aerodynamics and fluid mechanics, students and engineers generally obtain drag coefficients through a process of searching for values in tables and books based on the cross-sectional area. With the rise in smartphones and tablets, many researchers, engineers, and students are already familiar with the basics of an app for their device(s). Therefore, it was desired to combine these two, and create a free, intuitive app that can be used to calculate the drag and lift coefficients of any two-dimensional cross-section. It was also desired to use this tool to enhance the average engineering student's understanding and exposure to the fields of aerodynamics and fluid mechanics. These goals were accomplished by integrating three main technologies: a high performance cluster, finite element method, and mobile applications.

The high performance cluster was utilized to handle the interaction between a graphical user interface and a finite element code while alleviating the burden of memory and processor requirements from the user's device. These clusters may also take advantage of parallel processing which has the potential to dramatically reduce the computation time, and allows multiple simultaneous users to access the app at once. The finite element code was written to interact between the cluster and the user input parameters. It also incorporates the Intel MKL to solve the sparse, linearized set of equations obtained from the formulation. These features were combined with a graphical user interface developed in Flash to form the final product of this paper, Flow HPC. Flow HPC is an interactive app which models two-dimensional, steady flow of Newtonian fluids around the cross-section of an object. It is completely free, available as a webpage[2], and a downloadable app at Google Play Store, Apple iTunes App Store and Amazon App Store.

The results of Flow HPC were then compared to various experimental results as well as the well-known CFD simulation software, ANSYS Fluent, for two-dimensional flow around a cylinder. In comparison to literature within the same range of Reynolds number, the resulting coefficients were reasonably accurate with discrepancies as the Reynolds number increased as was expected.

Comparisons of Flow HPC to ANSYS Fluent proved accurate as well with the drag coefficient accuracy ranging from 96% to 99% accurate.

Finite Element Approach

Currently, the three most popular methods of modeling fluid flow problems within the finite element method (FEM) are: primitive variables, vorticity-stream function, and various stabilization techniques of these formulations such as least squares method[3,4]. The primitive variable finite element formulation coupled with Galerkin's method provided a straightforward and efficient way of approximating the Navier-Stokes equations. One attractive reason for choosing the primitive variable formulation was the application of boundary conditions. These conditions required only a specification of either the velocity or pressure at the boundary in question. Additionally, the development of the governing set of equations can be easy to implement as well. Although the equations are nonlinear, there are two methods of linearization: Picard and Newton linearization[5]. Picard linearization was chosen for the sole fact that it was easy and straightforward. However, it does have a drawback in that it affects the convergence rate slightly. Preparation for Galerkin's method includes the full form of the steady, two dimensional Navier-Stokes equations, should be rewritten as

continuity

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0$$

x-momentum

$$\rho\left(u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)=-\frac{\partial P}{\partial x}+\ \mu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

y-momentum

$$\rho\left(u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)=-\frac{\partial P}{\partial y}+\ \mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)$$

Applying Galerkin's method yields the appropriate set of equations, again with Picard linearization to handle the nonlinear terms[4]. In order to satisfy a stability condition of the finite element method, Ladyzhenskaya-Babuska-Brezzi or LBB, the field variables were interpolated so that the velocity was approximated at least one order higher than the pressure nodes,

$$u=\sum_{i=1}^{n}N_{Vi}u_i$$

$$v=\sum_{i=1}^{n}N_{Vi}v_i$$

$$P = \sum_{i=1}^{n} N_{Pi} P_i$$

where the shape functions $N_{Pi} < N_{Vi}$ according to the LBB condition. There are a number of stable elements developed specifically to handle this condition, but the Taylor-Hood[6] quadratic triangle was chosen for this paper to satisfy the LBB condition[4].

The continuity, momentum equations, and resulting matrix formulation can be shown as,

x-momentum

$$\left\{\int_{\Omega}\left(2\mu\frac{dN_V^T}{dx}\frac{dN_V}{dx}+\mu\frac{dN_V^T}{dy}\frac{dN_V}{dy}+\rho\boldsymbol{u^e}N_V^T\frac{dN_V}{dx}+\rho\boldsymbol{v^e}N_V^T\frac{dN_V}{dy}\right)d\Omega\right\}\boldsymbol{u} + \left\{\int_{\Omega}\left(\mu\frac{dN_V^T}{dy}\frac{dN_V}{dx}\right)d\Omega\right\}\boldsymbol{v} + \left\{\int_{\Omega}\left(-\frac{dN_V^T}{dx}N_P\right)d\Omega\right\}\boldsymbol{p} = 0$$

y-momentum

$$\left\{\int_{\Omega}\left(\mu\frac{dN_V^T}{dy}\frac{dN_V}{dx}\right)d\Omega\right\}\boldsymbol{u} + \left\{\int_{\Omega}\left(2\mu\frac{dN_V^T}{dy}\frac{dN_V}{dy}+\mu\frac{dN_V^T}{dx}\frac{dN_V}{dx}+\rho\boldsymbol{u^e}N_V^T\frac{dN_V}{dx}+\rho\boldsymbol{v^e}N_V^T\frac{dN_V}{dy}\right)d\Omega\right\}\boldsymbol{v} + \left\{\int_{\Omega}\left(-\frac{dN_V^T}{dy}N_P\right)d\Omega\right\}\boldsymbol{p} = 0$$

continuity

$$\left\{\int_{\Omega}\left(-\frac{dN_V^T}{dx}N_P\right)d\Omega\right\}\boldsymbol{u} + \left\{\int_{\Omega}\left(-\frac{dN_V^T}{dy}N_P\right)d\Omega\right\}\boldsymbol{v} = 0$$

global stiffness matrix, K

$$\begin{bmatrix}\boldsymbol{K_{11}} & \boldsymbol{K_{12}} & \boldsymbol{K_{13}}\\ \boldsymbol{K_{21}} & \boldsymbol{K_{22}} & \boldsymbol{K_{23}}\\ \boldsymbol{K_{31}} & \boldsymbol{K_{32}} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\\ \boldsymbol{p}\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}$$

$$K_{11} = 2\mu\frac{dN_V^T}{dx}\frac{dN_V}{dx}+\mu\frac{dN_V^T}{dy}\frac{dN_V}{dy}+\rho\boldsymbol{u^e}N_V^T\frac{dN_V}{dx}+\rho\boldsymbol{v^e}N_V^T\frac{dN_V}{dy}$$

$$K_{12} = \mu \frac{dN_V^T}{dy} \frac{dN_V}{dx}$$

$$K_{13} = -\frac{dN_V^T}{dx} N_P$$

$$K_{21} = K_{12}$$

$$K_{22} = 2\mu \frac{dN_V^T}{dy} \frac{dN_V}{dy} + \mu \frac{dN_V^T}{dx} \frac{dN_V}{dx} + \rho \boldsymbol{u}^e N_V^T \frac{dN_V}{dx} + \rho \boldsymbol{v}^e N_V^T \frac{dN_V}{dy}$$

$$K_{23} = -\frac{dN_V^T}{dy} N_P$$

$$K_{31} = K_{13}$$

$$K_{32} = K_{23}$$

While this formulation does yield acceptable velocity and pressure fields, it complicates the programming by rearranging the overall structure of the stiffness matrix. On a local machine this would be a considerable problem due to the limited computational resources, but by utilizing the high-performance cluster's 48 gigabytes of memory and multiple cores, memory issues are rendered irrelevant. Nonetheless, this is the final formulation which was used to obtain the velocity and pressure fields necessary to calculate the lift and drag coefficients.

A structured mesh routine was developed[7] to discretize a flow domain into a number of six-node triangular elements up to a user-selected level of refinement. The refinement ranged from a coarse grid of 564 elements with 2,733 degrees of freedom to a fine grid of 2,548 elements with 11,821 degrees of freedom. Boundary conditions were then applied to model the cross-section of an object placed within a wind tunnel with the components of velocity being zero along the object boundaries due to the no-slip condition, and free-stream conditions applied at the transverse edges of the domain. A single value of pressure was applied at a single node in the middle of the downstream exit which is a common method to handle the pressure boundary condition[3,8,9].

A Picard iteration scheme was applied to the linear set of equations produced by the finite element formulation until convergence was within a predefined tolerance of the solution of the field variables. The results of the pressure field were then integrated along the object boundary using Simpson's rule in order to obtain the drag coefficient due to the imbalance of pressure forces acting in the flow direction. The same methodology, respectively, was used for the lift coefficients.

Computational Domain and Set up

As shown in Figure 1, a circular cylinder was used as a test two dimensional cross-section. It was placed at a location one third of the width, and half of the total height of the flow field. In order to limit the influence of the "outer walls" of the flow field and as a general rule for accurate drag coefficients, the flow field was given a total height of approximately 20D; D being the diameter of the cylinder. The flow field's width was chosen to be approximately 55D[10,11,12,13].

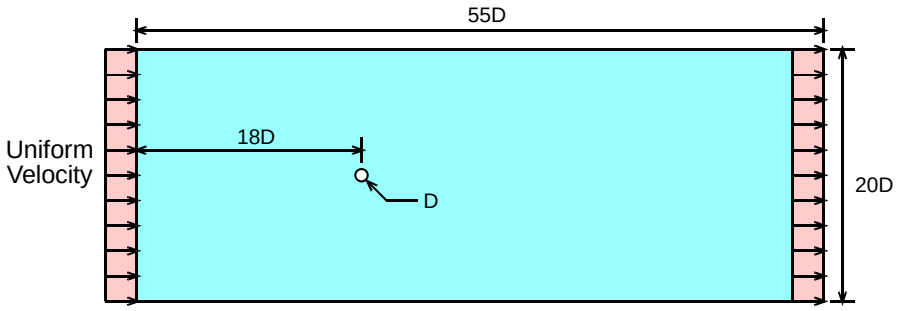

Figure 1: Flow Domain

These approximations tend to provide an adequate infinite flow field condition where the influences of the walls are negligible. Following these guidelines, the flow domain was discretized using a structured[7] meshing routine developed by the author that creates more triangular elements near the cylinder surface, and the concentration of elements decreases according to a logarithmic function as the grid is moved radially outward.

High Performance Cluster (HPC) Utilization

Flow HPC utilizes a 48 node, 576 cell cluster (Intel CPUs running under Windows 2008 HPC Server R2 system) at the University of Oklahoma. The cluster is for the exclusive use for engineering education. The Flow HPC program can be run from the eCourses.ou.edu web site or from an Android or iOS mobile device (tablet or smart phone). Both mobile versions are freely accessible for anyone at any institution, and since the bulk of the computation is done remotely, the user's device is not burdened by the simulation. Flow HPC takes advantage of the HPC in two main ways:

1. The HPC acts as a traffic director in a sense, establishing an organized and efficient order of the users who access the application. This in turn means multiple users may utilize Flow HPC at any given time and are not limited a "single file line" approach for calculations.
2. The HPC also acts as a remote calculation hub where the executable file is held containing the full finite element code and solver routines. It also creates the resulting image of the flow field, to some degree alleviating any dependency on the user's device.

In order to gauge the performance of Flow HPC and the high-performance cluster; it was desired to compare the required solve time of the MKL routine using a varying number of cores. The simulation was run for the finest mesh generated by the structured routine, resulting in 29,638 degrees of freedom. The number of cores varied from 1, 2, 4, 6, 8, and 12 cores. The MKL solve times and total simulation times are summarized in Table 1 and only the MKL solve times are visualized in Figure 2. The curve pictured in Figure 2 is typical of runtime improvement as the number of cores is increased. However, it was expected to see that the number of cores would have a more significant impact on the solve time. It was discovered that although the finest mesh is a large problem, the MKL routine PARDISO[1] would not demonstrate its full capability unless

the matrix was significantly larger than 30,000 by 30,000. While this is not as expected, it does show the power of Intel's routine, and benefit to future work as the problem sizes increase.

Initially, increasing the number of cores from 1 to 2 produced a significant decrease in runtime, but this improvement becomes less pronounced as the number of cores continues to increase. In fact, there is almost a decrease in performance time as the number of cores is increased from 8 to 12. According to the results of the comparison, the number of cores was fixed to 4. Since the total available cores in the cluster are 560, theoretically, 140 users can simultaneously perform flow calculations. However, since all users will not be solving at the same time, it is estimated that up to 500 users could be using the app at the same time without any reduction in response. It was tested in class with 50 users solving simultaneously without any problems.

Table 1: Server solve times versus the number of cores

Number of Cores	Average Time per Iteration (secs)	
	Total	Intel PARDISO
1	7.597	0.766
2	7.176	0.530
4	7.371	0.430
6	7.353	0.428
8	7.693	0.394
12	7.759	0.420

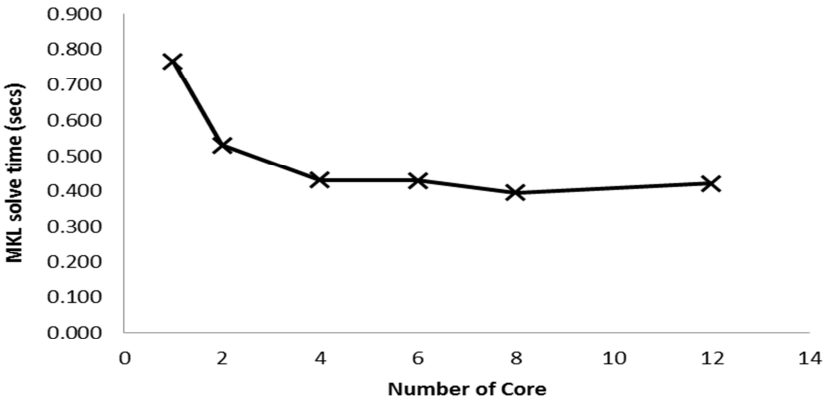

Figure 2: PARDISO solve time vs. number of cores

User Interface

One of the major objectives of this app was to build a tool capable of providing engineering students with drag coefficients for two dimensional objects. The user interface was important so that the students could use the app easily and without detailed instructions. It is to be a tool that students (or engineers) would find relatively intuitive to use after a basic fluid mechanics course. Students and engineers alike should be able to navigate throughout the program, and have a wealth of information about flow over two dimensional objects at a touch of a few buttons.

The Flow HPC app was developed using FlashDevelop which is an open-source environment for creating Adobe Air mobile applications. This set up allows the freedom of publishing the code as a desktop or mobile application which can be distributed onto both the Android and Apple app markets. One of the main concerns was ensuring an app that was easy and intuitive to use for engineering students. Flow HPC presents the user the choice of three predefined fluids as well as the ability to enter their own density and viscosity values, as seen in Figure 2, thereby determining the appropriate Reynolds number.

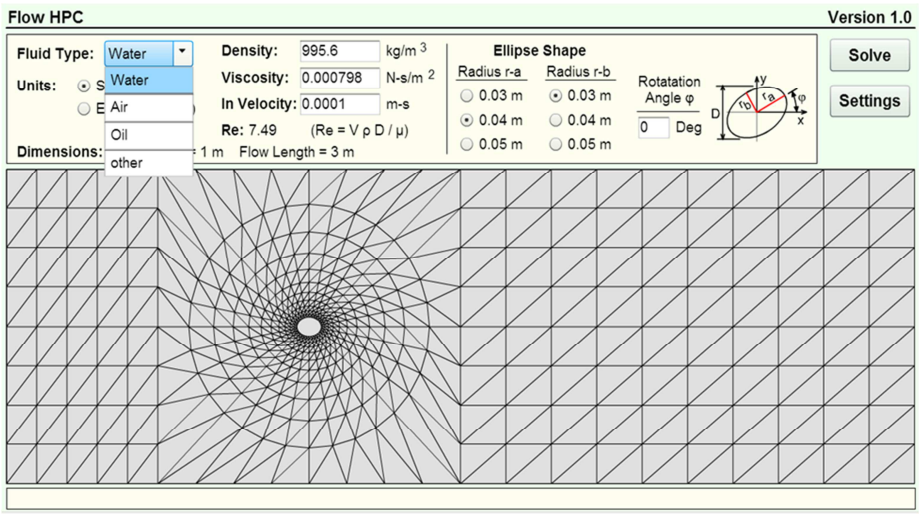

Figure 3: Predefined choice of fluids or user defined.

The freedom to choose from a variety of fluids as well as a user-defined set of properties gives the user the ability to customize the Reynolds's number while simultaneously seeing the affect different values may have on the solutions. Once the fluid has been chosen or values for density, viscosity, and free stream velocity have been updated; the user can decide whether or not the grid density needs to be adjusted based on the Reynolds's number. As the Reynolds's number increases the grid density also needs to increase to so that the solution can better converge. Otherwise, the solution can start to oscillate. This onset of oscillations is known as the critical Reynolds's number for the flow, and characterizes at what point the solution would start to oscillate. It has been established that for flow over a cylinder, this critical Reynolds's number is typically between 40 and 50. Figure 3 shows the pre-set grid densities and maximum

convergence interactions. If the program exceeds the number of maximum interactions, the HPC cluster will report back that it has not converged.

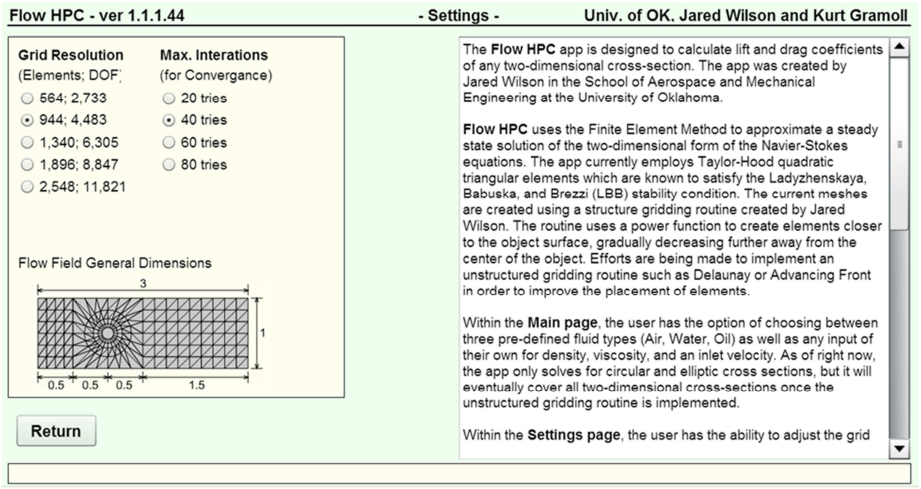

Figure 4: Settings page for grid resolution and iterations.

The following figures demonstrate some of the key features of Flow HPC. Figure 5 below is a circular cross-section tested at a Reynolds's number of approximately 10 with the calculated drag coefficient being 2. Figure 6 shows a higher Reynolds's number arbitrarily set at 37 yielding not only a smaller drag coefficient, but also shows the steady vortices which develop behind the object. The solve times are around 30s for both cases due to the size of the grid set up initially. Both of these examples are solving approximately 8,800 degrees of freedom. The remaining figures follow similarly, but rather than using a circle as the cross-section a rotated ellipse was used to focus on the lift coefficient as well as the drag coefficient.

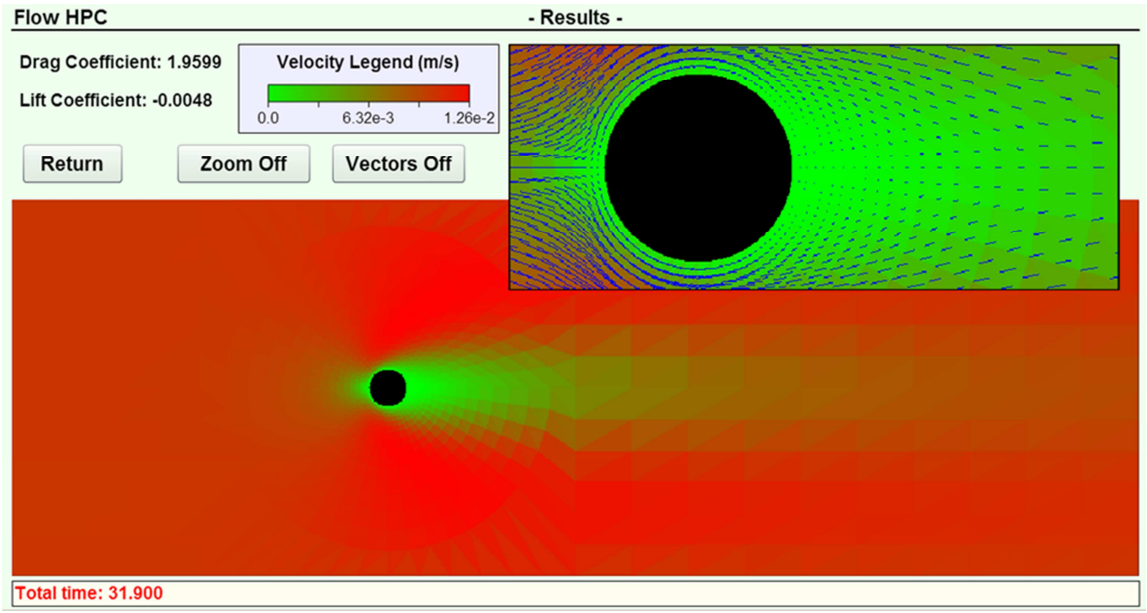

Figure 5: Flow around a cylinder Re = 10.

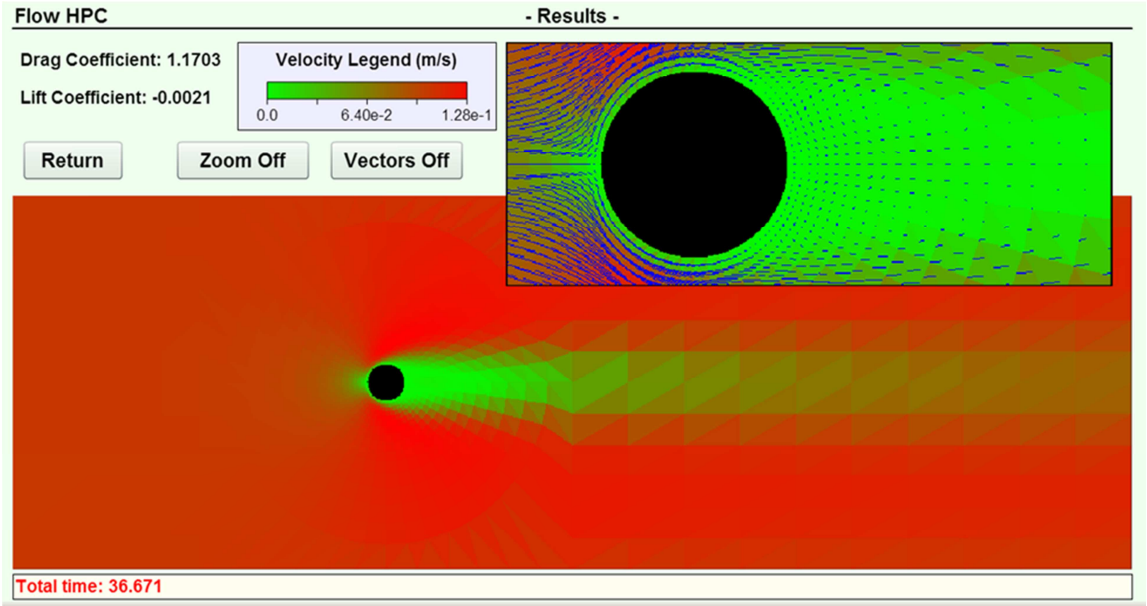

Figure 6: Flow around a cylinder Re = 40 with developed vortices

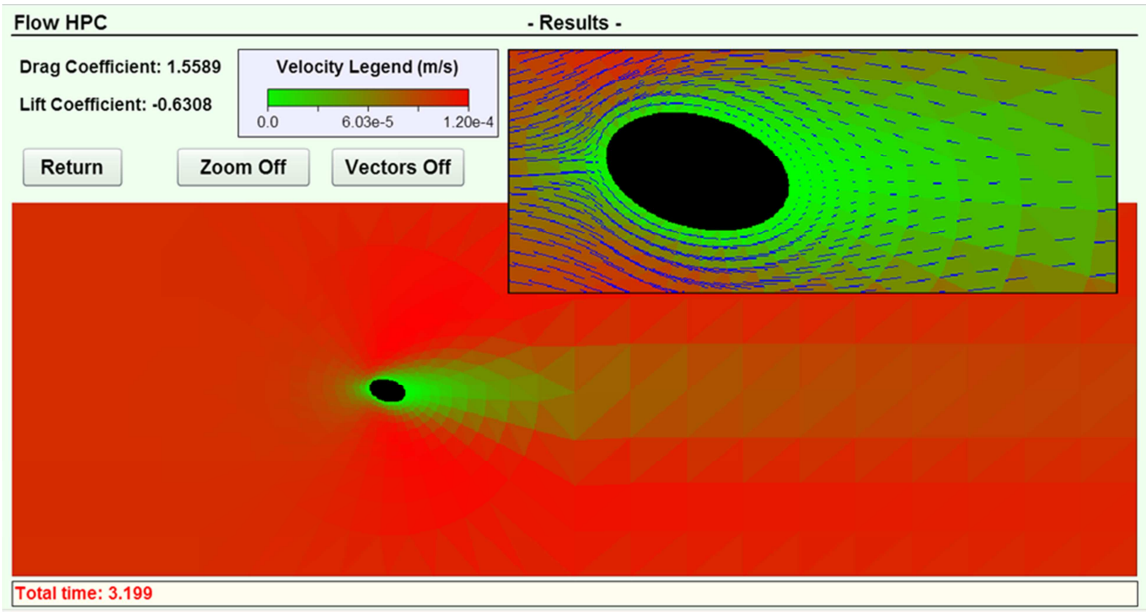

Figure 7: Flow around an ellipse Re = 10

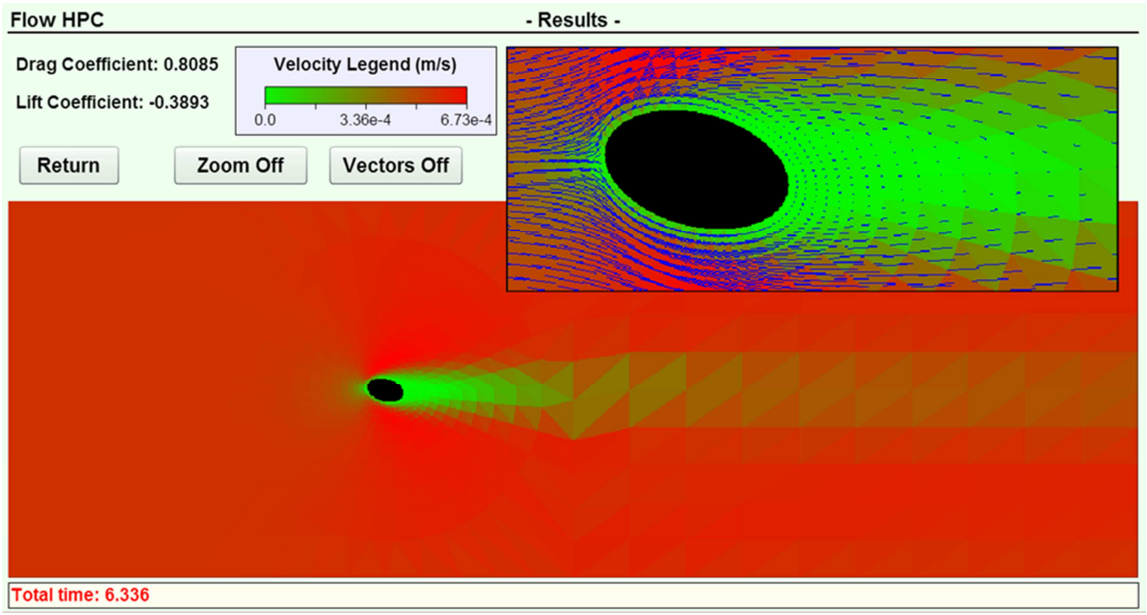

Figure 8: Flow around an ellipse Re = 37

The Flow HPC program was also checked with another standard numerical tool, ANSYS Fluent and generally accepted experimental results. When modeled under the same conditions and fluid properties within ANSYS Fluent, Flow HPC accuracy was within 99% - 96%. Table 2 compares the results of the drag coefficient due to the net pressure force obtained from ANSYS Fluent and Flow HPC. A range of Reynolds' numbers versus drag coefficients was also plotted in Fig. 9 to show not only the general trend of this plot, but as another means of comparing the accuracy to ANSYS Fluent.

Table 2: Drag coefficient comparison of Fluent to Flow HPC

Re	Fluent, C_D	Flow HPC, C_D	% error
1	6.134	6.048	1.424
5	2.206	2.131	3.542
10	1.607	1.545	3.954
25	1.169	1.204	2.895
35	1.067	1.087	1.846
40	1.033	1.047	1.314
45	1.005	1.013	0.835
50	0.985	0.985	0.039

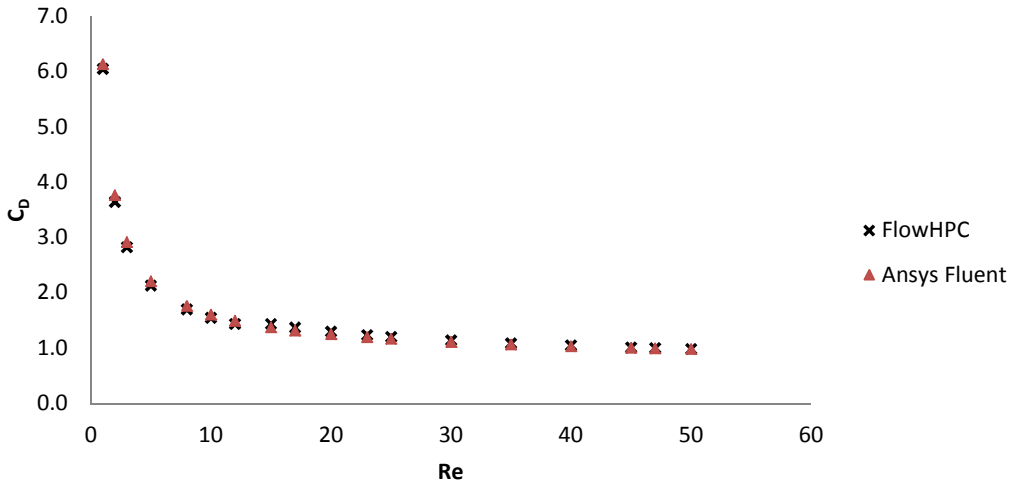

Figure 9: Drag Coefficient versus Reynolds number

Comparing Flow HPC with generally accepted experimental results[14], the app shows good correlation with the drag coefficient over the range the tool was designed. The app is currently limited to Reynolds number less than about 50. It is planned to extend the app to larger Reynolds number in the future, and to include unsteady flow and turbulent flow. Additional shapes are also planned along with an unstructured grid generating routine for user defined shapes.

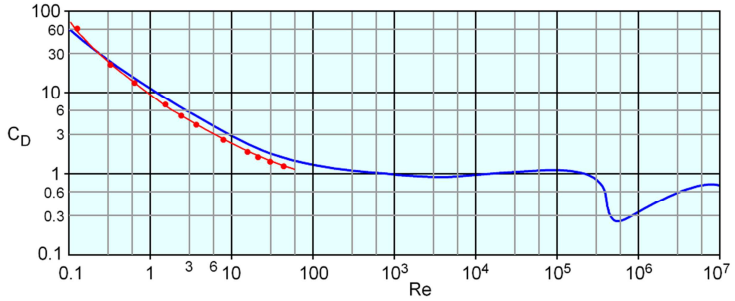

Figure 10: Drag coefficient comparison of experimental data (blue line) to Flow HPC numerical calculations (red dots and line)

Classroom Use

One of the core incentives for developing the Flow HPC application was to provide an additional resource to undergraduate students to determine drag coefficients when solving fluid problems. Most undergraduate texts have only a limited number of shapes in an appendix table where drag coefficients are provided. Even then, the Reynolds number usually very broad. Ultimately, it was the goal of the authors to have most major shapes available for a wide range of Reynolds numbers. As an initial version, a cylinder and ellipse was provided and used in the classroom in two ways. First, to App was provided to the students on a mid-range university-owned tablet that was distributed during the class. The students worked in groups to experimentally visualize what happens to the flow field at low Reynolds. In particular, the angle of the ellipse was shown to change both the drag and lift. The flow field pattern was also discussed. This visualization was something that wasn't available in the standard text. The students showed great interest in being able to change the few parameters in the app to see what happens. They expressed appreciation at the opportunity to control input and immediately see what happens. This same in-class exercise could have also been done with a full CFD program like Fluent, but then ease of use would have been lost. By having a simple, self-contained app, students were able to immediate grasp the cause and effect.

The second use of the app in the class was an open-ended homework assignment. The students were asked determine the best shape and angle of an ellipse in a field to maximize lift and minimize drag. They also had to determine if the results varied with Reynolds number. While there is no single solution to this exercise, it did require the students to try different configuration. The app was provided to the students that had their own Android-based mobile device (beta version required side loading). The app was also made available on the class eBook web page (www.ecourses.ou.edu). Since then, both Android and Apple iOS versions have been released on Google Play Store, Amazon App Store, and Apple iTunes App Store.

The app was also used on one exam as part of a mid-term exam. The exam was tablet-based with multiple choice questions[15]. The tablets were university provided for the exam, and had the Flow HPC app installed. The question on the exam required the use of the drag and lift coefficient for an ellipse at 45 degree angle to the flow. This information was then used to calculate the force required to keep the ellipse in equilibrium. This use of the app was in place of the textbook or handbook tables. Students need to become comfortable in retrieving information from electronic sources in addition to print material.

It should be noted, that while the app was used in on Fluids Mechanics course, there was no requirement or comparison done between students that may or may not use the app. Use of the app was only required for homework and one test. All students used it. There was not an opportunity to do a formal assessment on its effectiveness, other than the homework and test question could not be answered without it. The authors hope to do more assessment in the future as the app capabilities are improved (i.e. larger range of Reynolds number flow, unsteady flow, turbulence, more shapes, and better visualization graphics). However, it is the firm belief of the authors that the app is useful, and lack of formal assessment is not a valid reason to stop from using it or publishing its use.

Further development of Flow HPC will include expanding the shapes and Reynolds number range of the app to allow more use in a basic fluid mechanics class. The current version is limited but does allow students better understanding of low Reynolds number flow fields. The authors felt best outcome of using the app in the fluids course was the ability of students to start experimenting with different configuration with no learning curve. Eventually, all engineering students interested in fluid flow problems will have to learn CFD and/or a major program like Fluent. However, for an undergraduate class that is not feasible, and thus Flow HPC can be used as an initial introduction to flow visualization.

Conclusion

It has been shown that the gap between advancing engineering simulations and education is quickly narrowing due to the emerging smartphone and tablet app markets. In an effort to merge the two areas, Flow HPC was developed for enhancing engineering education in the area of fluid mechanics, the finite element method, and to act as a secondary database for drag and lift coefficients for two-dimensional objects. While the current work only addresses low Reynolds number and two shapes, cylinder and ellipse, it was shown to be beneficial to students in better visualizing flow fields.

The Flow HPC is available for either Android or iOS based mobile device. It has also been made available to Flash-enabled browser at the eCourses.ou.edu web site.

An additional benefit of developing the Flow HPC application for mobile devices is to demonstrate that it is possible to perform highly computational intensive on mobile devices when connected to a HPC cluster. The trend to mobile devices, both tablets and smart phones, for students (and practicing engineers) has been accelerating over the past few years. It is only logical that students and engineers will need high-end apps to perform them work, both while out of the office. One of the purposes of developing Flow HPC was a proof of concept that this is now feasible. The authors realize the current version is only a proof-of-concept and there are many additional features that are needed. In particular, the app needs more shapes, wider range

of Reynolds number, user controlled shapes, and better grid generation. While turbulent flow modeling is still a few years off, it now seems feasible that it is possible on mobile devices.

Bibliography

[1] "PARDISO* - Parallel Direct Sparse Solver Interface." https://software.intel.com/en-us/node/470282 15 March 2015

[2] Gramoll, Kurt, "eCluster at OU" http://www.ecluster.ou.edu/ 11 Mar. 2015.

[3] Huebner, Kenneth H., and Earl A. Thornton. The Finite Element Method for Engineers. New York: Wiley, 1982.

[4] Reddy, J. N. An Introduction to Nonlinear Finite Element Analysis. Oxford: Oxford UP, 2004.

[5] Reddy, J. N. An Introduction to the Finite Element Method. New York: McGraw-Hill, 1984.

[6] Taylor, C., and P. Hood. "A Numerical Solution of the Navier-Stokes Equations Using the Finite Element Technique." Computers and Fluids 1 (1973): 73-100.

[7] Tannehill, John C., Anderson, Dale A., and Richard H. Pletcher. "Grid Generation." *Computational Fluid Mechanics and Heat Transfer*. By John C. Tannehill. 679.

[8] Hutton, David V. Fundamentals of Finite Element Analysis. Boston: McGraw-Hill, 2004.

[9] Stasa, Frank L. Applied Finite Element Analysis for Engineers. New York: Holt, Rinehart, and Winston, 1985.

[10] Connor, J. J., and C. A. Brebbia. Finite Element Techniques for Fluid Flow. London,: Newnes-Butterworths, 1978.

[11] De Vries, G., and D. H. Norrie. "The Application of the Finite-Element Technique to Potential Flow Problems." Journal of Applied Mechanics 38.4 (1971): 798.

[12] White, Frank M. Viscous Fluid Flow. New York: McGraw-Hill, 1974.

[13] Apelt, C.J. England. Ministry of Aviation. Steady Flow of a Viscous Fluid Past a Circular Cylinder at Reynolds Numbers 40 and 44. London: Her Majesty's Stationery Office, 1961.

[14] "Cylinder Drag" http://scienceworld.wolfram.com/physics/cylinderDrag.html, Web, 15 March 2015.

[15] Gramoll, K.C., " Development and Implementation of a Tablet-based Exam App for Engineering Course," ASEE Annual Conf. Proc., Seattle, WA, June 15-17, 2015