13th Australasian Fluid Mechanics Conference
Monash University, Melbourne, Australia
13-18 December 1998

DYNAMIC GRID REFINEMENT USING AN OBJECT-ORIENTED APPROACH

Johannes P. VAN DER WALT, Yos S. MORSI and Michael WALTERS

School of Engineering and Science
Swinburne University of Technology, Hawthorn, Victoria, AUSTRALIA

ABSTRACT

Object-oriented techniques are extremely well suited to the solution of Computational Fluid Dynamics (CFD) problems. Not only is the object paradigm a much more natural and comfortable system to work in, but it is also a much more powerful tool than procedural methods in the hands of the experienced engineer and scientist. This paper demonstrates the object-oriented implementation of a dynamic grid refinement routine in which the cells are refined in runtime in the areas of highest gradients. The implementation of grid refinement is known to be a very complex task when procedural languages are used, but relatively straight forward in the object-oriented paradigm.

INTRODUCTION

After more than two decades of intensive research in the field of numerical simulations of engineering problems in general and Computational Fluid Dynamics (CFD) in particular, the two parameters, computer speed and memory, are is still the limiting factors. Computer speed and memory roughly doubles every second year, implying an exponential growth in both speed and memory, and yet these remain the constraints in some numerical methods and especially CFD. As speed and memory capabilities increase, CFD software utilises these to the maximum, mainly to increase accuracy, which is accomplished by a finer computational domain as well as the implementation of newer and more complex technologies in computational methods. These new computational methods are mostly a direct result of the increased capability of computers and were therefore not possible before. Examples of this are non-rectangular and unstructured meshes and dynamic grid refinement. The result is often that faster computers, which mostly are equipped with larger memory, does not solve CFD problems faster. Instead, their greater capabilities are used to solve larger and more complex problems and simulation runs tend to take longer than before.

Whilst the interest and activity in virtual reality for engineering simulations are increasing rapidly, the already overly complicated engineering simulation codes of the day may become too complicated for the human brain to comprehend if traditional methods were to be used. In the computer sciences this limitation of traditional (structured) methods was already recognised more than a decade ago and the solution to the complexity issue was found in the creation of computer objects which exhibit comparable behaviour to their physical counterparts. The fact is that the human brain can deal with objects far easier than with abstract ideas and principles, since the user or programmer is mostly shielded from the internal complexities of each object. This in turn leads to a much better grasp of the overall strategy or algorithm, which will enable us to advance the frontiers of present engineering simulations. This paper is focussed on the development of an object-oriented dynamic grid refinement technique which alleviates two problems: it will be shown that a substantial saving in computer memory is possible, and furthermore it allows more complex algorithms to be implemented in a fashion much easier to comprehend.

TRADITIONAL CFD AND OBJECT-ORIENTED TECHNOLOGY

Object-oriented programming is relatively new if not completely foreign to most CFD research institutes and software houses where Fortran is almost exclusively used. This is probably due to the large base of scientific Fortran routines available and the level of computational efficiency achieved by "modern" Fortran compilers. Furthermore, and this may even be the primary reason, the fact that most if not all of the pioneers in scientific and engineering computations, and especially those in the CFD world started their research in an age when Fortran really was the only language available for engineers. Hence, it is logical that they continued their research and development in Fortran. Attempts were made with the release of Fortran 90 to address some of the deficiencies of Fortran 77, which is still the most-commonly used version. However, Fortran 90 mostly introduced features which were already available in C and Pascal for almost two decades, and most importantly, it does not provide Fortran with the benefits of object-oriented technology.

ADVANTAGES OF OBJECT-ORIENTED TECHNOLOGY

Object-oriented technology can reduce both memory and computing requirements by reducing the overall number of cells in the computing domain. This is achieved by building the entire calculation grid from cell objects. Each object contains all the data normally stored in several different vectors (or matrices) in traditional programming methods, which, in itself constitutes a significant reduction of vector accessing time. The cell objects are also much more intelligent than the ordinary data vectors in that they can carry information about neighbouring cells and also about any subdivided group of cells within itself.

However, the major advantage of object-oriented technology in CFD applications may be found in an object's ability to spawn other objects of the same kind, or of different types. This allows a calculation domain to initially consist of a very coarse grid which may be

solved within a few iterations, upon which the domain may be refined in areas of high gradients which requires a grid of finer resolution. The result is that the grid will be dynamically (automatically) refined only in areas where it is required (figure 1). In traditional methods the grid has to be refined over the full height or width of the grid, resulting in a significant percentage of "wasted" cells and hence an unnecessary demand on computers resources, both in memory and computational effort. Although more traditional methods exists to refine grids in localised areas, they are not flexible in that they have to be defined during the initial grid generation process and hence the grid cannot be refined at run time.

The same technology can also be applied to artificially create multi-grids, since the objects store all the information logically found within each computational cell, but without the high memory requirements of the traditional methods. This is possible since the same grid may be used to solve different parameters where the grid can easily present itself as a coarser or finer grid.

It also offers other advantages to CFD programs such as improved program structure and exceptional program extensibility which are major drawbacks of the still widely used Fortran language. It is common in CFD to write user-specified functions. The limitations of the traditional approach are that the user does not have any control over the parameters that are passed to the user-defined routine. Using object-oriented technology, the user can add any functionality to the code by deriving new objects from existing ones and add any new functionality to the new objects. In general, object-oriented languages are specifically designed with existing-code reuse in mind and this is an extremely important factor in CFD.

IMPLEMENTATION

A demonstration program solving for stream function values was coded in C++ using Microsoft Visual C++ 4.2 The program demonstrates the runtime grid refinement capability. This program is shown in figure 1.

CONCLUSION

Object-oriented techniques are ideally suited for many engineering applications, especially when complex simulation codes are involved. Computational Fluid Dynamics is rapidly becoming an important tool en engineering, and the demands on better and faster simulations are ever-increasing. The time is ripe for the CFD community to tap into the vast and rewarding world of object-oriented techniques.

REFERENCES

1. BUZZI-FERRARIS, G., "Scientific C++, Building Numerical Libraries the Object-Oriented Way", *Addison-Wesley*, 1993.
2. BARTON, J.J. and NACKMAN, L.R., "Scientific and Engineering C++", *Addison-Wesley*, 1994.
3. Faires, J.D. and Burden, R.L., "Numerical Methods in C++", PWS Publishing Company, Boston, 1993.
4. Budd, T.A., Classic Data Structures in C++, *Addison-Wesley*, 1994.
5. Sedgewick, R., "Algorithms in C++", *Addison-Wesley*, 1992.

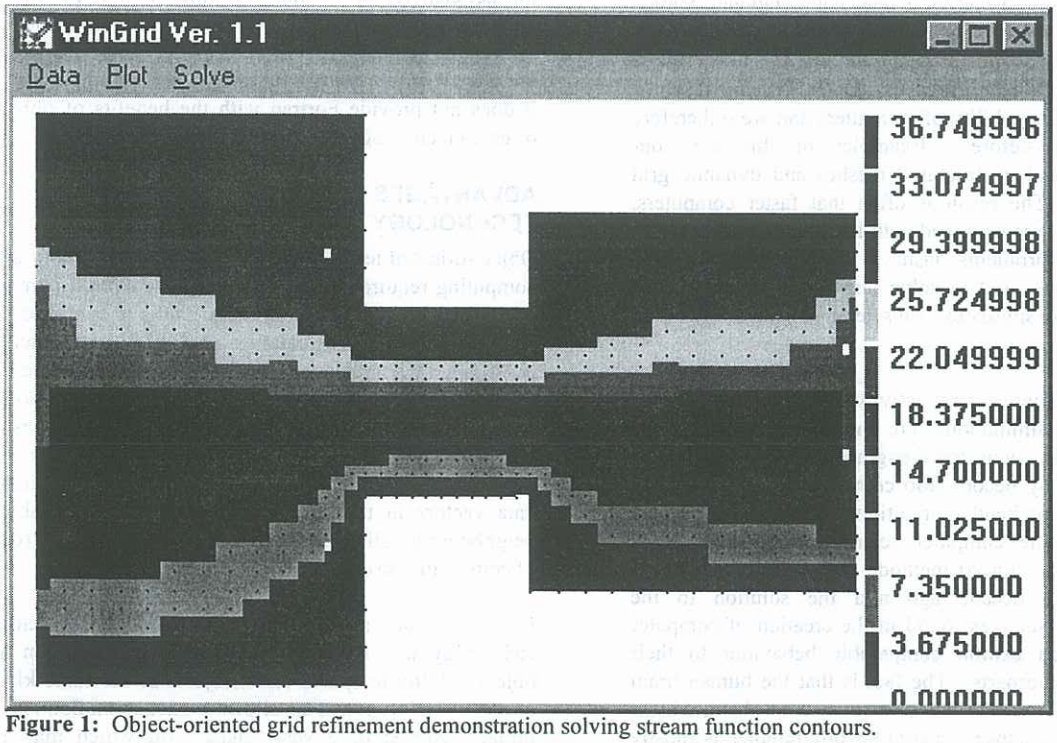

Figure 1: Object-oriented grid refinement demonstration solving stream function contours.