COMMUNICATIONS IN COMPUTATIONAL PHYSICS
Vol. **4**, No. 5, pp. 1170-1193

Commun. Comput. Phys.
November 2008

Enabling Technologies in the Problem Solving Environment HEDP†

Lijun Xie, Yao Zheng*, Jianjun Chen and Jianfeng Zou

Center for Engineering and Scientific Computation and School of Aeronautics and Astronautics, Zhejiang University, Zhejiang 310027, China.

Received 4 April 2008; Accepted (in revised version) 15 July 2008

Available online 9 September 2008

Abstract. Enabling technologies are those technologies preparing input data, analyzing output data and facilitating the whole processes for numerical simulations. This paper outlines current enabling technologies for large-scale multidisciplinary simulations used in the High End Digital Prototyping (HEDP) system, a problem solving environment equipped with capability of mesh generation and large-scale visualization. A problem solving environment is a computer system that provides all the computational facilities necessary to solve a target class of problems. Mesh generation continues to be the pacing technology for a practical numerical analysis, which is essential to yielding an accurate and efficient solution. Large-scale visualization maps the massive data to some kinds of scenes interactively, which can be realized through a tiled display wall system with distributed visualization capability. HEDP is designed for large-scale and multidisciplinary simulations, and there are four categories of modules involved, namely pre-processing module, computing module, post-processing module, and platform control module. All these modules are coupled through a software bus, which makes the modules integrated seamlessly. Detailed design principles and applications of the HEDP environment are addressed in this paper.

AMS subject classifications: 65Mxx, 65Nxx, 68Uxx, 76Nxx, 76V05, 80A25

PACS: 02.30.Jr, 02.60.-x, 47.11.-j, 47.70.Fw, 47.70.Pq, 89.20.Ff

Key words: Enabling technology, large-scale simulation, mesh generation, problem solving environment, distributed visualization.

1 Introduction

With the development of computational methods and computing resources, large-scale multidisciplinary simulation is becoming an important area of computations in engineer-

†Dedicated to Professor Xiantu He on the occasion of his 70th birthday.

*Corresponding author. *Email addresses:* zdxlj@zju.edu.cn (L. J. Xie), yao.zheng@zju.edu.cn (Y. Zheng), chenjj@zju.edu.cn (J. J. Chen), zoujianfeng@zju.edu.cn (J. F. Zou)

ing and science. Technologies, such as problem solving environments, mesh generation, and large-scale visualization, enable us in fulfilling large-scale simulations.

For computational simulations of various physical phenomena and processes, there is a common demand to equip the users a user interactive platform with certain capabilities, and an effective method is to construct a Problem Solving Environment (PSE) [1–4] which is a computer system that provides all the computational facilities necessary to solve a target class of problems [5]. Typically, it can reduce the difficulty of physical simulations by utilizing user natural languages and application specific terminologies, and by automating many lower level computational tasks. We define a kind of PSEs in the following formula [5, 6]:

$$\begin{aligned}\texttt{PSE}=&\texttt{User interface}+\texttt{Enabling libraries and tools}\\&+\texttt{Problem solvers}+\texttt{Software bus}.\end{aligned}$$

Commonly, a PSE should have a friendly user interface such as nature language and graphical user interface that can help the user to use the system in a direct and efficient manner. Enabling libraries and tools are the most valuable parts of a PSE. They provide all the necessary assistant functions for a simulation, such as geometric modeling, mesh generation, and scientific visualization. Problem solvers are integrated into the computational module for various problem fields. Software bus is the method to integrate all the modules to work seamlessly and efficiently.

Mesh generation continues to be the pacing technology for a practical numerical analysis and is the area where significant payoff can be realized. Furthermore, high quality meshes for encompassing special regions are essential to yielding an accurate and efficient solution. Research on mesh generation technologies is challenging, while its importance is evident [7–14]. During the past decades, both structured and unstructured meshing techniques have been extensively developed and applied to solution of various engineering problems. To deal with situations in which complex geometry imposes considerable constraints and difficulties in generating meshes, composite structured mesh schemes [15, 16] and unstructured mesh schemes currently are the two mainstream approaches.

With the increase of scales of applied engineering problems to be solved, the requirement to visualization is increasing. For large-scale problems, visualization technology is demanded to meet the requirement of large-scale numerical simulations [19], and this includes special rendering techniques, parallel processing methods [20], and distributed and collaborative visualization approaches. Large-scale visualization can be realized through a tiled display wall system with distributed visualization capability. Large scale displays provide users experience totally different from common monitors or a single projector. The benefits include showing the whole view of a large scene, offering enough area to place lots of windows at the same time for group collaborating, providing much more details of objects, and giving users immersive feeling.

The HEDP (High End Digital Prototyping) system is a problem solving environment integrating these front-end enabling technologies for high end digital prototyping. In this

paper, we introduce the main enabling technologies used in the HEDP system. Section 2 describes the design principles and architecture of the problem solving environment, Section 3 addresses an unstructured mesh generation mechanism and its parallelization scheme, and Section 4 introduces the method of building a large scalable display wall for parallel visualization. An application running on HEDP is shown in Section 5, and finally some conclusions are drawn in Section 6.

2 Problem solving environment HEDP

The HEDP environment is a problem solving environment for multidisciplinary application simulations, and its predecessor is EEMAS (Enabling Environment for Multidisciplinary Application Simulations) [23]. Within HEDP, there are four categories of modules involved, namely pre-processing module, computing module, post-processing module, and platform control module. It is developed for complex and large-scale simulations to take advantage of powerful parallel and distributed computing technologies. All the modules are coupled through a software bus, which maintains the share memory and makes the modules integrated seamlessly.

The environment can reduce the time required for problem definition and for post-processing, whilst the unified environment is ideal for multidisciplinary design applications, as the data is handled in one consistent format. It hides many aspects of computational engineering that are not of prime interest or relevance to the engineer, e.g. the setting-up and subsequent execution of an application on a parallel platform. HEDP is designed mainly for large-scale simulations, and heavily depend on visual steering, parallel and distributed computation.

The HEDP framework is developed with C++, the kernel algorithms are implemented with C and Fortran, the graphical user interface is built with Qt, and visualization capabilities are developed on the top of OpenGL library. The system runs on both Linux/Unix and Windows platforms. Fig. 1 is a snapshot of a HEDP session running on an SGI Octane2 machine.

2.1 Design goals

The primary task of HEDP is to provide an efficient environment that enables scientists and engineers to create multidisciplinary application simulations, to develop new algorithms, and to couple existing algorithms with powerful enabling tools. The main design principles and goals that guide development in the HEDP project are as follows.

(1) Abundant Functions. HEDP contains generic modules that provide necessary functions needed in mesh-based simulations, such as geometric modeling, CAD repair, mesh generation, domain decomposition, scientific visualization, platform control, and numerical libraries.

(2) Scalability and Seamless Integration. As a PSE, the main aim of the HEDP project is not to provide concrete scientific computational functions, but to provide an efficient,

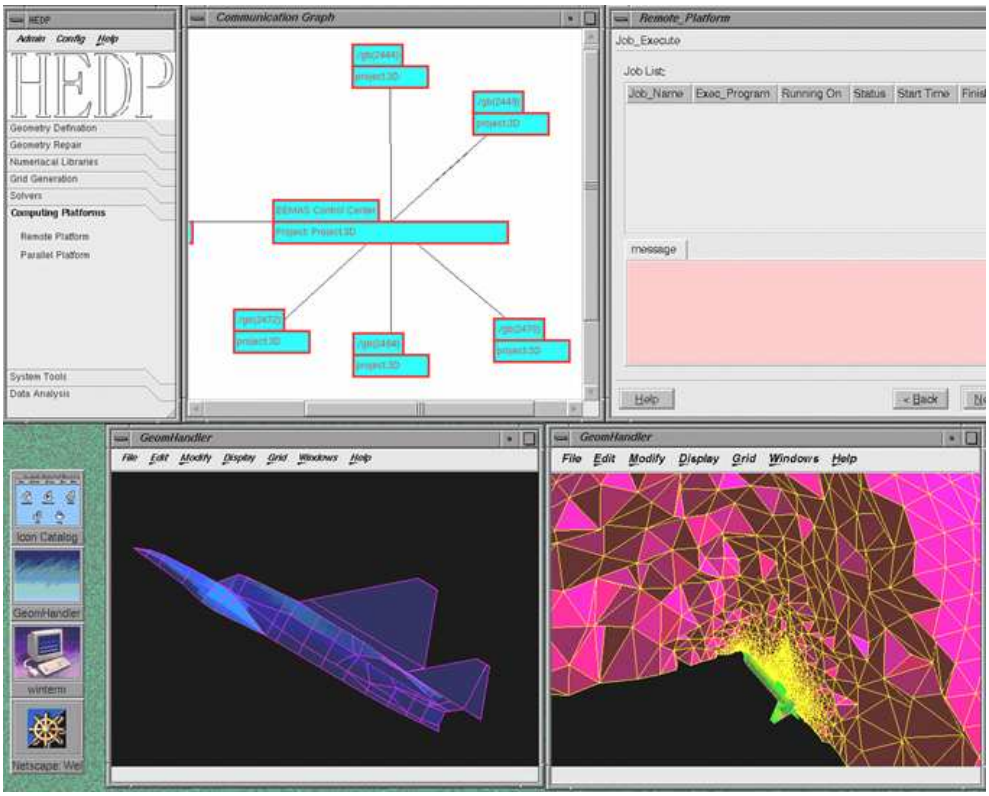


Figure 1: A snapshot of a HEDP session running on an SGI Octane2 machine.

flexible and yet consistent framework, that facilitates integration of domain solvers and enabling tools. For this purpose, much attention is paid to the scalability of the system, which is mainly implemented with consistent data format and flexible data transfer interface.

Three data transfer schemes are provided, that is, through pipes, sockets and temp files, respectively. For a module with its source code, users or developers are able to integrate it into the system by means of data transferring through pipes or sockets, of which the efficiency is quite high. If the source code is not available or the users do not want to spend much time on the integration, temporary files can be used for data transferring directly.

(3) Visual Steering and GUI. HEDP follows the philosophy of visual steering. Its graphical user interface guides users to utilize particular components without in-depth expertise on those components, and to control the computing in a straightforward way. This allows scientists to use visualization tools while focusing on computational algorithms, and allows programmers to create visualizations tools without implementing simulation modules either. Visualization and numerical feedback are used throughout the system.

(4) Parallel and Distributed Computing. HEDP is designed mainly for large-scale simulations. Therefore, majority of its modules utilize parallel and distributed computing, and can run on remote machines. Most of the modules, from mesh generation to problem solving, and to visualization, can run in the parallel mode. A module for parallel environment setup and control is also developed. Two parallel schemes are utilized. One is task parallelism that distributes different parts of a simulation to different processors. The other is data parallelism that is more widely used within a computationally intensive module.

2.2 Software architecture and features

Fig. 2 illustrates the HEDP architecture from users' perspective. As mentioned above, there are four categories of modules involved, named as pre-processing module, computing module, post-processing module and platform control module. The first three are common phases of a simulation whilst the last one serves for the entire process of a simulation. All the modules are coupled through a software bus, which maintains the share memory and makes the modules cooperate seamlessly.

(1) Pre-Processing Module. Pre-processing module deals with the problem definition and preparation for computing. This is the most time-consuming part in simulations, and many researchers are focusing on automating this work. In HEDP, the pre-processing module consists of a basic geometrical handling tool, several powerful mesh generators, and tools for general CAD format conversion, boundary condition definition and physical properties definition.

The geometrical handling tool in HEDP stores geometrical data by means of boundary representation. It is not as powerful as commercial CAD software, but it is adequate to process the common modeling work, especially with certain functionalities oriented to mesh generation. For complicated geometries, the user can directly model them with other CAD software, and then import them into HEDP.

Mesh generation is a crucial step in simulations that impacts both the calculating time and the accuracy. HEDP provides powerful serial and parallel mesh generators with the ability to generate 2D planar meshes, surface meshes of triangles, and volume meshes of tetrahedrons. This module is capable to generate high quality meshes by means of visual steering. Users can specify the mesh spacing in terms of a background mesh and mesh sources. The detailed meshing algorithms are to be addressed in Section 3.

(2) Computing Module. As mentioned above, the goal of designing the HEDP environment is to support multidisciplinary application simulations. In general, scientists and engineers could integrate various domain solvers into the environment to process particular simulations. To integrate into the environment, users should adapt their solvers with the HEDP data structure, or write an interface to convert formats. Data transfer involved can utilize pipes, sockets or temporary files. Meanwhile, We developed a set of data transfer tools to integrate our modules with broadly accepted file formats, such as CGNS (CFD General Notation System), and with widely used commercial soft-

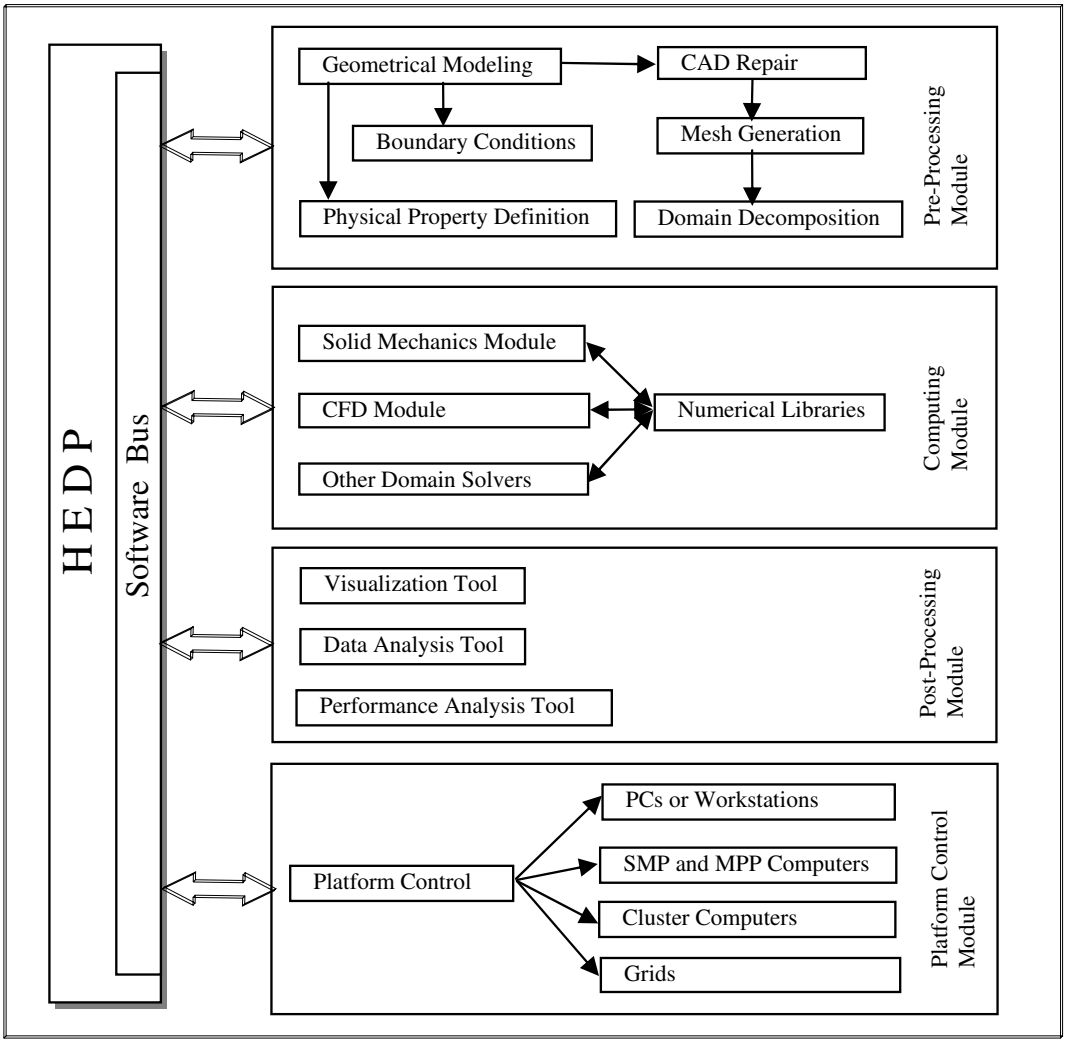


Figure 2: The HEDP architecture from users' perspective.

ware, such as ANYSYS, ABAQUS, and StarCD. Moreover, several solvers for CFD and solid structure analyses developed by ourselves have also been integrated.

(3) Post-Processing Module. Resulting data of complex and large-scale simulations are often difficult or even impossible to be understood without the assistance of visualization. A powerful visualization package, ParaView [24], has been integrated into HEDP.

In order to support visualization for large datasets, such as gigabyte datasets visualization, parallel and distributed visualization technologies are employed. There are two modes of distributed visualization. One is the task parallel mode, in which different part of visualization pipes are processed by different processors. The other is data parallel

model, in which the data is broken into pieces to be processed by multiple processors. The second method is easier to be implemented, because many visualization algorithms need not much change when running in parallel. HEDP supports both distributed and local rendering, and their combination. This provides scalable rendering for large data sets without sacrificing performance when working with smaller data sets.

In HEDP, a large stereo display wall system named SimWall has also been developed, to support parallel visualization with scalable display resolution and parallel rendering capability. The design consideration and methods are detailed in Section 4.

(4) Platform Control Module. HEDP provides a platform control module to process the setup of parallel environments, computing source management, file transfer and so on. This hides lot of trivial details of platforms from users, and helps users utilize all kinds of computers from local PCs. The whole system only needs to be initialized once. At present, this module can be used to explore local or remote Unix/Linux systems running on personal computers, workstations, SMP and MPP supercomputers, and clusters. The functions to utilize grid resources in the Grid Computing concept are also under development [25].

3 Unstructured mesh generation and its parallelization

Currently, HEDP is equipped with a set of mesh generators for plane, surface, and volume geometries, where the surface one is mapping-based, and the plane and volume ones are DT (Delaunay Triangulation)-based. In order to overcome the serial bottlenecks of serial volume mesher in terms of time and memory (majorly in memory), a parallel version of this mesher is also developed and is being integrated into HEDP. In this section, we mainly introduce the serial 3D Delaunay triangulation mesher and its parallel version. Readers who feel interests in the surface mesher and other technique issues of mesh generation tools currently used in HEDP are recommended to read References [12–14].

3.1 Serial 3D Delaunay triangulation

Delaunay triangulation is one of major approaches for unstructured mesh generation, and has attracted immense attentions from researchers in this field for nearly thirty years. However, so far it has been still a research focus due to some unresolved issues, such as boundary recovery, mesh quality improvement and robustness.

(1) Boundary Recovery. The Delaunay criterion provides a good way to triangulate a given point set. However, the predefined point connectivity is not certainly preserved during the triangulation, and some boundary constraints may be lost in the resulting triangulation. Therefore, the recovery of missing boundaries becomes an important topic.

2D boundary recovery problem turns out to be much easier to resolve in theory and practice than its 3D counterpart. It has been shown that there are certain polyhedrons, e.g. the Schönhardt polyhedron, that cannot be triangulated without adding Steiner points. Moreover, Ruppert and Seidel [18] proved that it is an NP-complete problem to

judge whether a polyhedron could be triangulated without adding Steiner points. Consequently, almost all practically useful boundary recovery algorithms have to consider the problem of where and how to add Steiner points.

Boundary constraints could be recovered in two ways: conformal and constrained. In the conformal recovery, Steiner points are inserted on the constraints, and not removed in the resulting volume meshes; thus some of the missing constraints are recovered as concatenations of sub-constraints. In the constrained recovery, the constraints are exactly same as the prescribed ones, and no Steiner points are allowed to be left on them.

Weatherill and Hassan [10] firstly investigated a conformal boundary recovery algorithm by adding points directly at the intersection positions between missing boundaries and the current triangulation. Lewis and Zheng et al. [14] revisited the algorithm when implementing their 3D Delaunay mesh generator.

George et al. [26] proposed a constrained boundary recovery algorithm in the early 1990s based on local transformation operators in conjunction with heuristic rules for inserting Steiner points into the inside of the problem domain. However, it suffers from robustness issues [27]. Recently, George et al. [28] have presented an alternative constrained boundary recovery algorithm free of such problems. Interestingly, Du and Wang [29] independently proposed an algorithm based on almost the same idea as that of George's new algorithm.

The conformal boundary recovery algorithm, used in our present work, is a variation of those proposed in [10, 14]. We introduce some new concepts, operations and data structures to make its implementation rather routine. Moreover, to improve robustness of the algorithm, all intersection cases of missing entities and triangulations are examined systematically and their solutions are delivered [17].

Our present constrained boundary recovery algorithm is an indirect one, i.e. it needs a conformal algorithm as the preprocessor [30]. Points inserted on missing boundaries in conformal recovery procedure are split first, and then moved away from the boundaries. Edges and facets are recovered one part after another, by employing shell and constrained shell transformations, respectively. The algorithm is related closely with those proposed in [28, 29].

(2) Mesh Quality Improvement. Both field point creation and boundary recovery strategies affect the mesh quality. However, it is always not enough and also very difficult to guarantee mesh quality only by improving these strategies. In most cases, mesh postprocessing schemes are needed to improve mesh quality further.

Smoothing is a cost-effective tool for mesh postprocessing. Classical Laplacian smoothing could not prevent invalid elements appearing or mesh quality decreasing locally. Alternatively, constrained weighted Laplacian smoothing algorithm is currently preferred [14].

Freitag et al. [31] suggested to improve tetrahedral mesh quality using swapping and smoothing operations. For cases where skinny elements clustering in local regions or near boundaries, their strategies demonstrated having little benefit and needing further improvement. A more complex combinational swapping operations suggested by

Joe [32] are useful for alleviating such problems. Klingner and Shewchuk [33] recently devised a very aggressive tetrahedral mesh improvement technique. They declared that their software often improves a mesh so that all its dihedral angles are between 30 and 130 degrees. Another important work in tetrahedral mesh optimization was performed by Du and Wang [34], they suggested the CVT (Centroidal Voronoi Tessellation) method as a variant to improve tetrahedral mesh quality.

(3) Robustness. The Bowyer-Watson incremental point insertion procedure is numerically instable and very sensitive to round-off errors. The typical technique to resolve such a problem is to implement the kernel using the predicates free of round-off errors, e.g. those provided by Shewchuk based on adaptive precision floating-point arithmetic [35]. Implementing a robust point insertion kernel suggested by George [36] is also useful, since it removes the requirement for a robust in-sphere test predicator, which is the major source of numerical instability of the point insertion kernel. Another frequently used lightweight technique to alleviate the problem is to disturb problem points and postpone their point insertion operations [10].

In the boundary recovery stage, many intersection calculations have been involved. However, the precision of the calculation will affect the robustness of the boundary recovery, and that of the mesh generation finally. Therefore, more detail consideration has been taken to deal with this problem. Furthermore, a method has been adopted to deal with inconsistent geometrical judgment, which is based on Epsilon Geometry [14].

3.2 Parallel 3D Delaunay triangulation

With the emergence of the ever larger problems in areas such as Computational Fluid Dynamics (CFD) and Computational Electro Magnetics (CEM), a parallel simulation environment is required urgently, where the serial mesh generation process is a bottleneck in terms of both computing time and memory requirements. Therefore, parallel mesh generation has received intensive attentions since early 1990s [37, 38].

Previous works on the development of a parallel Delaunay mesher mainly adopt two methodologies, to parallelize the algorithm and to parallelize the problem.

One way to develop algorithm-parallel Delaunay meshers is to exploit geometrical locality of point insertion, which involves only a cavity consisting of triangulations not satisfying the empty circle rule. Therefore, insertion operations of multiple points could happen concurrently if the concerned cavities do not interface with each other, otherwise, these operations should be coordinated to avoid access conflict to shared data. Okusanya and Peraire first proposed a 2D algorithm-parallel Delaunay mesher [39], and soon extended it to a 3D version [40]. Chrisochoides and his group [41] improved the algorithm with overlapping computation techniques to decrease synchronization waiting costs, and finally accomplished a Latency Tolerant Bowyer-Watson (LTBW) kernel.

Problem-parallel Delaunay mesh generation is more intensively investigated. The procedure proposed by Larwood et al. [42] bi-decomposes the surface model recursively by introducing separator planes. Boufflet et al. [43] also implemented such a domain

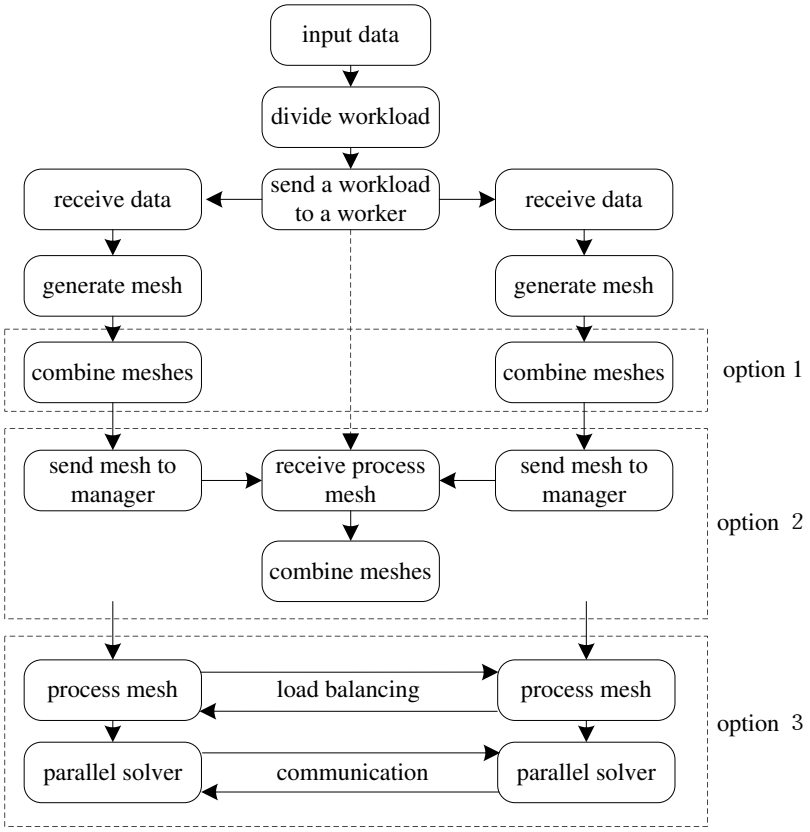


Figure 3: Structure of the manager/worker model.

decomposer for parallelizing a serial advancing front mesher. Invalidity of separator boundaries or meshes, and badly shaped elements near separator boundaries, are two main problems associated with the above algorithm. Galiter and George [44] strived to overcome the second problem with the aid of the Projective Delaunay Triangulation (PDT) theory. In their algorithm, the separator meshes are Delaunay admissible. Robust problems could arise for this algorithm when no qualified separator boundary could be found. Said et al. [45] decomposed the problem domain by dividing the coarse triangulations of boundary nodes, where trivial handling techniques are required to smooth sub-domain boundaries and coordinate interfacial mesh data.

We devised a parallel 3D unstructured Delaunay mesh generator based on the problem-parallel model. A recursive domain decomposition process resembling Larwood's [42] is redesigned, where coordination of separator meshes shared by neighboring volume sub-meshes is guaranteed by constrained boundary recovery. Sub-meshes distributed on the same processor are gathered together to enable further mesh post-processing operations, such as mesh smoothing or optimization, to be performed in a processor level or globally.

In our parallel scheme, surface models stored with various data formats are read into a domain decomposer and then divided into sub-domains. A manager/worker parallel framework (Fig. 3) is designed to distribute sub-domains on processors available for meshing, using a mesher configured with a constrained boundary recovery procedure. A mesh in the processor level is formed by merging the meshes of all sub-domains on the processor, on which some post-processing operations, such as mesh smoothing, could be performed. Distributed meshes are repartitioned and moved among processors for the conflicting goals of load balancing and minimization of communications. Finally, repartitioned meshes are sent to parallel solvers. Intermediate data are generated for the purpose of debugging and testing, and viewed via a visualization module.

4 Building a display wall for large-scale visualization

As a subsystem of HEDP for large-scale visualization, a user-friendly stereo tiled display wall system named SimWall is built [22]. SimWall is composed of 18 commodity projectors supported by a Linux graphics cluster. Collaborating together, these projectors work as a single logical display capable of providing a high-resolution, large-scale, and passive stereo scene. In order to avoid tedious system setup and maintenance, software-based automatic geometry and photometric calibration is used. The software calibration is integrated to the system seamlessly by an on-card transform method and is transparent to users. To end-users, SimWall works just as a common PC, but provides super computing, rendering and displaying capability. In addition, SimWall has a stereoscopic function that gives users a semi-immersive experience in polarized passive way. This section presents system architecture, implementation, and other technical issues such as hardware constraints, projectors alignment, geometry and photometric calibration, implementation of the passive stereo mode, and development of the overall software environment.

Display walls are effective at providing large-scale imagery to users, but its installation and operation is often a tedious undertaking. In recent research, some parts of the setup work have been automated, but because of design constraints, many problems still require solution. Nowadays, there are three main problems. One is the alignment of projectors. In a project array with more than eight projectors, manual alignment becomes extremely time consuming and even technically impossible. The second problem is how to eliminate the photometric variation between projectors. Unlike high-end projectors, there exists significant color and intensity variation between low-cost projectors. Although commodity projectors provide some functions for color adjustment, but the color space they can display are intrinsically different. So it is impossible to make all the projectors match photometrically by adjusting their properties. The third problem is how to provide a practical, user-friendly software environment. In order to support high resolution and complex scenes rendering, tiled display walls need a graphics cluster to drive them. The cluster introduces a totally different architecture which needs programmers to divide and synchronize tasks between nodes.

We built SimWall which utilizes some existed technologies in a novel way to automate the setup and maintenance process and to provide an easy-to-use environment. The projectors can be casually placed with a small area of overlap between neighbors. Using a single camera and computer vision techniques, the geometry and photometric calibration can be achieved automatically. The calibration is performed as a background daemon with an innovative on-card transform method. The different architecture and additional calibration work are transparent to end users. SimWall can show stereoscopic display to give users a semi-immersive experience in a polarized passive way. This character introduces many new design and implementation problems, such as screen material selection, polarized color offset, stereo image generation, and strict geometry alignment. Carefully treating these issues, SimWall provides a stereoscopic environment with very good effect. Notice that this system can be extended to a fully immersive CAVE easily in terms of technology [46].

4.1 Hardware issues

Fig. 4 shows the schematic representation of SimWall. The framework is similar to the display wall of Princeton University [21]. The significant character distinguishes ours from theirs is that each rendering machine in our system drives a pair of projectors projecting to the same area of screens. Mounting polarized filters before them, the users who wear 3D glasses which match the polarization of the projected images, will see stereo scenes. Fig. 5 illustrates the SimWall architecture located in the Center for Engineering and Scientific Computation, Zhejiang University. The display wall is comprised of a 2.5m$\times$2.2m rear projection screen and nine pairs of Epson EMP-74 LCD projectors (3$\times$3 arrays), where each pair is driven by a personal computer with dual 2.4 GHz Xeon CPUs and a NVIDIA Geforce 5200 graphics accelerator. The resulting image is with resolution of about 3000$\times$2300.

4.2 Geometry calibration

To achieve a large seamless uniform display, adjacent projectors must be aligned precisely to remove gap or overlap, and color variation between projectors must be eliminated or reduced perceptually. These two processes are called geometry calibration and photometric calibration correspondingly. In the past, the construction of the projected display wall was quite tedious, requiring precise projector alignment by hand. Some research groups designed six freedoms projector positioners to mechanically align the projectors [47]. But in practice, for a project array with more than eight projectors, manual alignment to achieve sub pixel alignment in each dimension becomes almost impossible.

Software calibration can eliminate the trivial hand work and does not need expensive positioners. Misalignment of projectors can be captured and measured by cameras with computer vision. The task to get the mapping relation, between each projector's image coordinates and the display's global coordinates, is called geometric registration. After

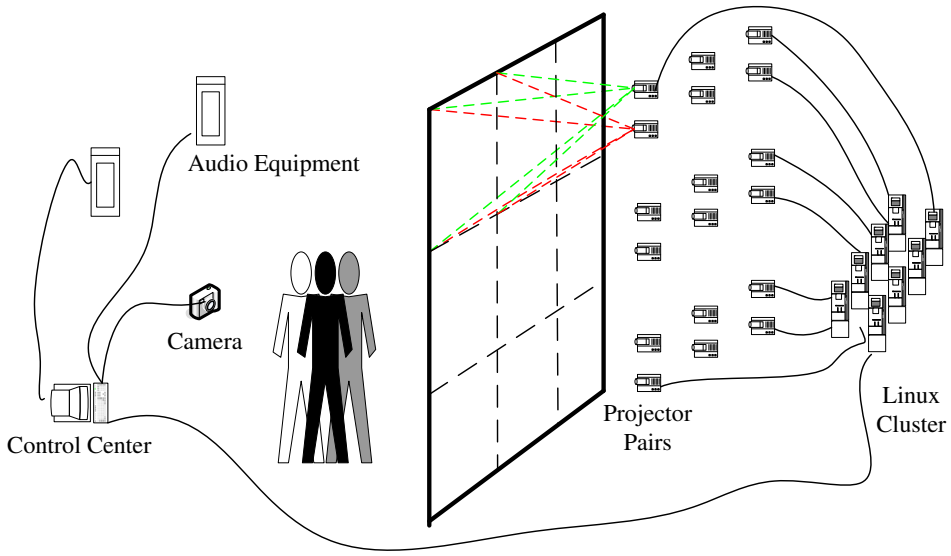


Figure 4: Architecture of the stereo display wall.

a) b)

Figure 5: A typical stereo display wall system supported by a PC cluster: a) A matrix of projectors; and b) A screen and projectors.

geometric registration, the origin projector's images can be pre-warped and projected just onto the desired areas. Several research laboratories implemented these ideas in different ways. Our method uses a digital camera which can capture the entire screen to process geometry register. Because the image distortion from projector to screen, and that from screen to camera, are both projective transformation, their coordinates can be transformed by a projective matrix. Then the transform matrix between the projector's image coordinate and the display's global coordinate, P, can be calculated as follows:

$$P = C^{-1}T, \tag{4.1}$$

where C is the projective matrix from the projector's image space to the camera's image space. T is the projective matrix from camera's coordinate to global display's coordinate. In practice, the camera lens and projectors will distort images in nonlinear ways. The nonlinear distortion of the camera is removed by a preprocess with OpenCV. The nonlinear radial distortions of projectors is approximated by dividing its image space into small rectangle areas.

The matrices C and T are obtained by finding corresponding points in the three different coordinate spaces. A computer vision method is used to automatically find the relation points. By projecting some features and recognizing them, the relationship between image space and camera space can be obtained. We select structured light circles as features and seeking their center in the camera's image. By many experiments, we found out this is one of the most precise features a camera can recognize. Four green circular features are placed on the four corners of the screen for camera recognition and calculation of the projective matrix between screen and camera. After matrix P is obtained, a pre-warp process can be added to the source image.

4.3 Photometric calibration

To eliminate photometric invariance is much more complicated than geometry calibration. Our photometric calibration method is similar to that of Majumder [48]. We use an inexpensive camera to measure the ITF with High Dynamic Range (HDR) imaging method. This method uses differently exposed photographs to recover the response function of the imaging process. Although spectroradiometers and colorimeters are precise color measure devices, they can only test color character of one point each time. This makes it very difficult to measure the spatial large display of projectors by them. Moreover, they are quite expensive for common users. After the ITF is obtained, the luminance mapping table for each projector can be calculated out. We ignore the chrominance variance because it is less perceptually notable but very difficult to measure. The color mapping table is inserted before the frame buffer output in each render machine to get a uniform luminance response. Projector overlapped regions need to be incorporated otherwise they will cause noticeable bright areas. This process is called Edge blending. Edge blending can also blur the small position error on edge areas. We calculate the blending mask with the technique presented in Reference [49]. The frame buffer-sized alpha masks attenuate the pixel values of the corrected image accordingly.

4.4 Software environments

SimWall can run all kinds of GUI programs built on the top of X or OpenGL APIs. The programs with the capability of running on tiled displays, such as Paraview and EnSight DR, can run on SimWall directly. Other common programs need additional tools for distributed rendering. Currently in SimWall, this distributing task is performed by Chromium [50] and distributed multi-head X (DMX) [51].

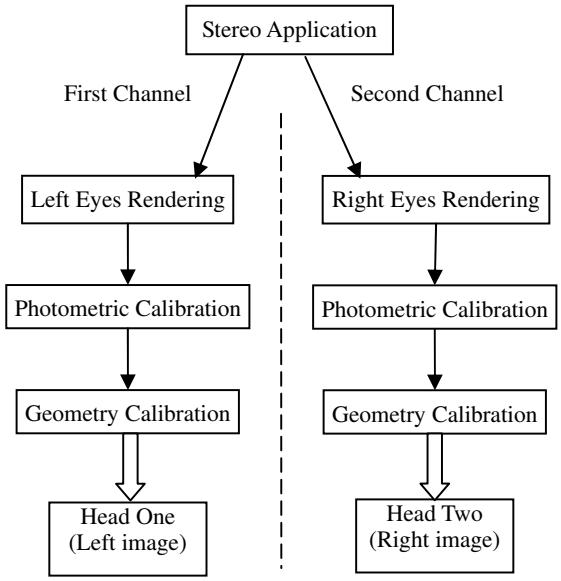


Figure 6: Render pipeline of SimWall in the stereo mode.

Chromium is a system for interactive rendering on clusters of workstations. It provides sort-first, sort-last and hybrid distributed rendering ways. Most OpenGL programs can run on Chromium without modification. DMX is a proxy X server that provides multi-head support for multiple displays attached to different machines, each of which is running a typical X server. The multiple displays on multiple machines are presented to the user as a single unified X desktop. Most other systems choose VNC as the desktop platform because it transfers the image of the desktop screen and so it is easy to apply image modification. DMX runs in a different mode, it distributes 2D primitives to clients instead of rendered image. This can decrease the load of both the server host and networks. By integration with Chromium, DMX can render all kinds of common GUI and OpenGL programs.

We develop two methods to integrate the geometry and photometric calibration to the applications. The two methods correspond to two running modes in SimWall. One is common mode for 2D GUI programs and common OpenGL programs. In this mode, the calibration process runs as a background daemon. Programs applying it will not notice its existence at all. The other mode is the stereo mode especially for OpenGL stereo programs. These two modes utilize the two channels of a graphic card in different ways. The common mode utilizes one channel to do normal rendering job, and the other for seamless geometry and photometric calibration. Only the output of the second channel will be projected and tiled as a large display. The stereo mode renders images for left and right eyes separately in the two channels, and projects both their output images. The flow chart of the stereo mode, and an example scene of the display wall working, are shown in Figs. 6 and 7, respectively.

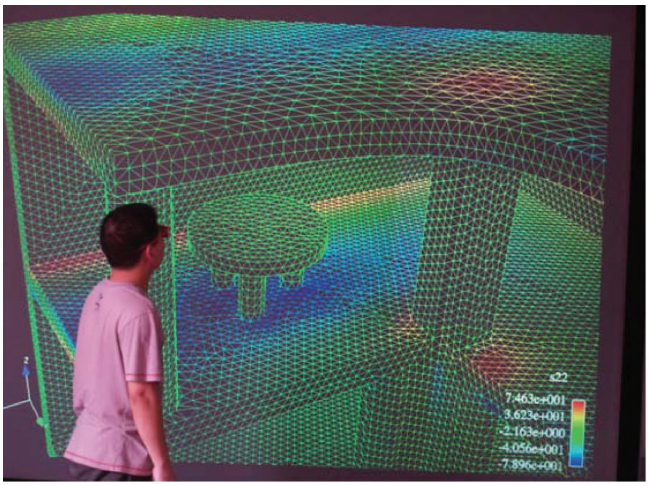


Figure 7: Illustration of the display wall working: Structure analysis of a house.

5 An application of the HEDP system

As an example, numerical simulation of turbulent combustion in the HyShot Scramjet is processed within HEDP. An increasing research effort has been made to investigate high speed turbulent combustion numerically, which is resulted from the interests in hypersonic air breathing propulsion. A serious issue about turbulent combustion is the simulation of turbulence with multi-scale feature. For the traditional Large Eddy Simulation (LES) or hybrid RANS/LES approaches [52, 54–56], spatially filtered equations are used to compute the resolved scales of turbulence and certain sub-mesh models for unresolved scales. There are several drawbacks associated with the spatially filtered equations and sub-mesh models, which are stated by Shih and Liu [57,58], i.e. the inconsistency between the filter function and sub-mesh models, the commutation error due to the nonuniform computational mesh, the mesh-dependent solution, and the effect of numerical dissipation introduced by higher-order schemes.

Shih and Liu proposed a new methodology of Partially Resolved Numerical Simulation (PRNS) based on temporal filtering, and demonstrated that, with the so-called "resolution control parameter", one can carry out a unified simulation from RANS towards LES or vise versa [57, 58]. This methodology does not involve spatial filtering, and there will be no issues about commutation errors, inconsistencies between the filter and sub-mesh scale models, etc. Shih and Liu have computed the turbulent flow for pipe flow and LM6000 combustor with the preliminary results being encouraging. Cai and Ladeinde provided their evaluation on the PRNS procedure for near-wall turbulence prediction [53].

In this section, the PRNS is coupled with our own combustion code [59, 60], which is expected to present a more encouraging and accurate description for turbulence in the Scramjet engine. The "forebody fuel injection" scramjet engine was tested by the

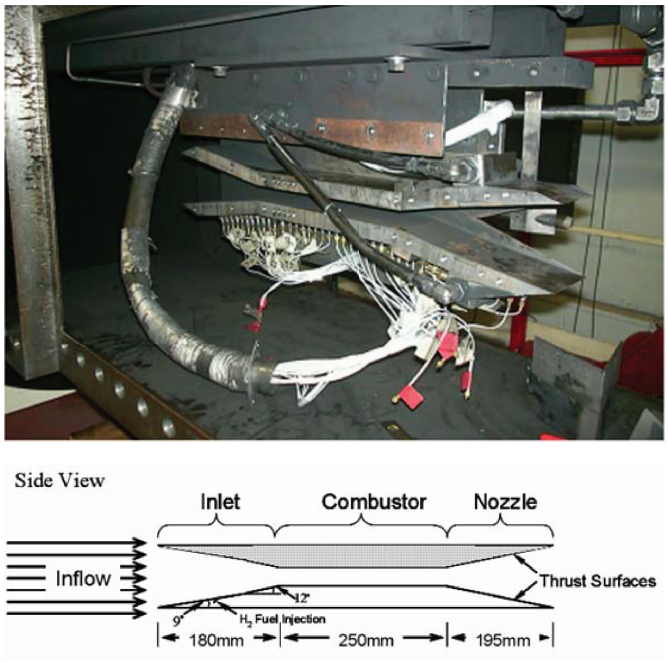


Figure 8: Scramjet model for the HyShot project.

"HyShot" project conducted in University of Queensland, Australia. The fuel can be made to premix with airflow in the inlet section so as to reduce the required length of the combustor and the skin friction drag [61].

(1) Geometry Model. The scramjet engine used by the HyShot project is presented in Fig. 8, which consists of inlet, combustor, thrust surface and etc. The total length of the scramjet is 625 mm and the combustor is 24 mm high. Four fuel injectors (diameter D=2 mm) are designed in the inlet section at an angle of 450 with the horizontal surface.

(2) Mesh Generation. The scramjet model was divided to 1597360 wedge elements (see Fig. 9(a)). Fig. 9(b) shows the meshing details near the fuel injectors. The mesh is decomposed into 20 domains and delivered to different processors for parallel execution as shown in Fig. 9(c).

Table 1: Boundary conditions for fuel off and on cases.

Cases	Fuel off	Fuel on	Cases	Fuel off	Fuel on
U_{air} (m/s)	2766	2612	U_{H_2} (m/s)	0	1321
P_{air} (Pa)	10230	8958	P_{H_2} (Pa)	0	64000
T_{air} (K)	487	412	T_{H_2} (K)	0	300
Ma_{air}	6.25	6.4	Ma_{H_2}	0	1

(3) Boundary Conditions. Non-reacting and reacting cases were simulated and the corresponding boundary settings are given in Table 1. The mass fraction of oxygen in

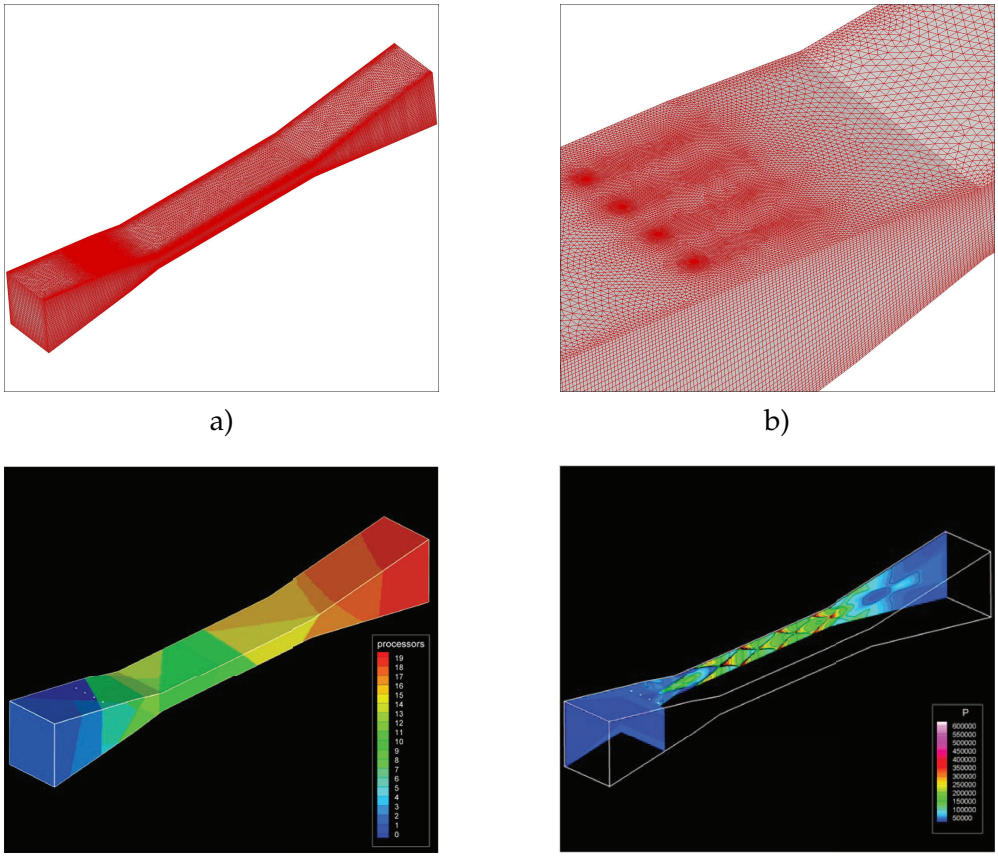


Figure 9: Computational domains and the corresponding results for the HyShot Scramjet engine: a) Mesh of the model; b) Details near the fuel injectors; c) Twenty domains of the mesh; and d) Pressure field of the simulation results.

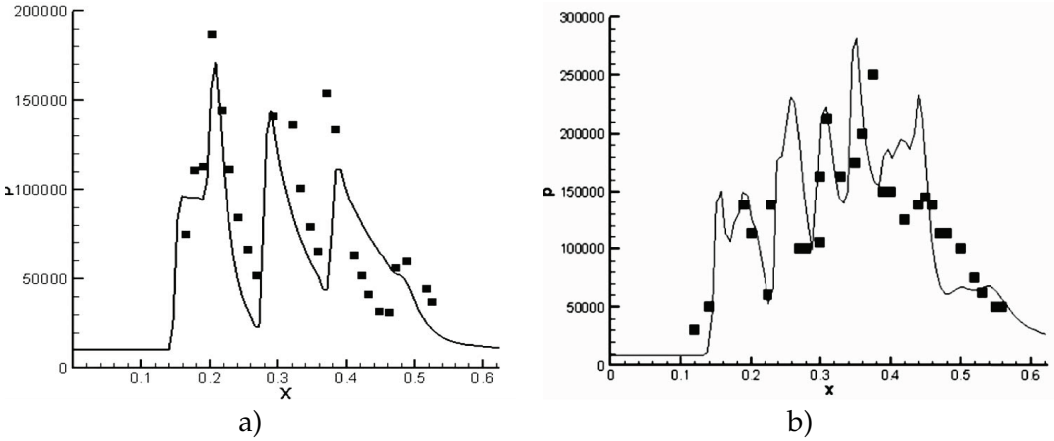
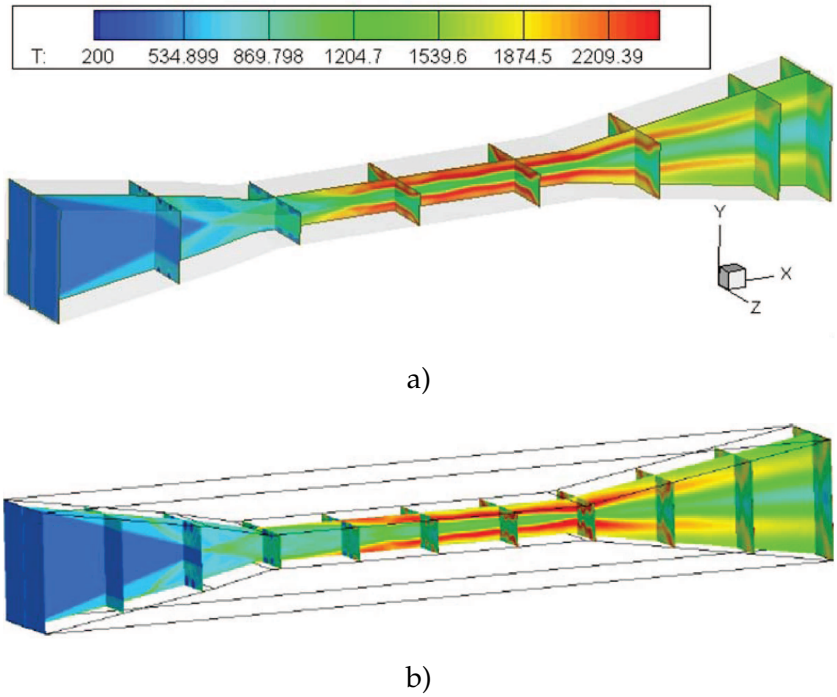


Figure 10: Pressure distributions along the center line for fuel off and on cases: a) Fuel-off; and b) Fuel-on.



a)

b)

Figure 11: Temperature distribution for the fuel-on case: a) Our result; and b) NASA result [62].

airflow inlet is set to 0.21. The two cases are used to investigate the combustion mechanism of the scramjet engine with Mach number of 6.4 and 30 km high. For "fuel-on" case, the fuel hydrogen injects with sonic speed and penetrating pressure of 0.64 MPa. The equivalence ratio is approximately 0.5.

(4) Numerical Results. Fig. 10 provides pressure distributions along the center line for both fuel off and on cases. Compared with the experimental pressure data of Judy Odam [61], good agreements have been achieved. Fig. 11 gives the temperature distributions of the present simulation and the data from NASA Langley's simulation [62]. Fig. 12 shows a user observing the simulation results on SimWall.

6 Conclusions

Environments such as HEDP can help reduce design complexity by creating 3D virtual prototypes, and optimizing and differentiating the designs through large-scale multidisciplinary simulations. It could avoid having to produce expensive physical prototypes. The present paper outlines the main enabling technologies used in HEDP, including mesh generation, problem solving environments, visual steering, and large-scale visualization. As an example of integrated systems, HEDP is a problem solving environment for multidisciplinary application simulations. It is a framework that can easily integrate arbitrary

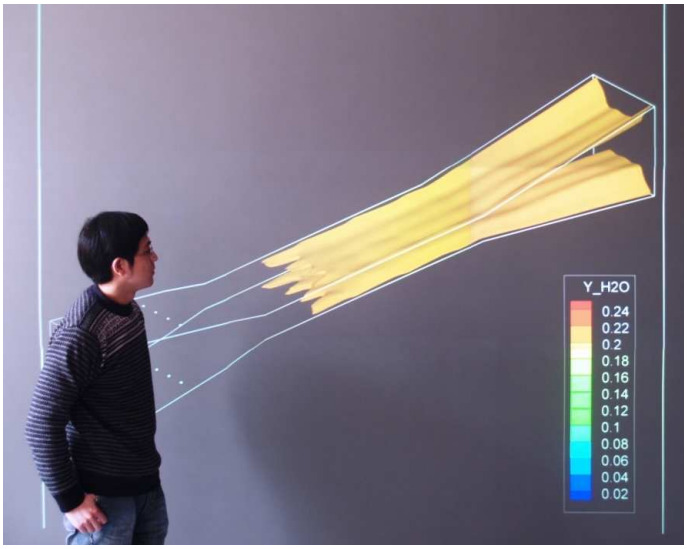


Figure 12: Observation on the HyShot scramjet combustion shown on the display wall.

modules, and it contains various enabling tools such as pre-processing, post-processing, and platform control.

Acknowledgments

The authors would like to thank the Center for Engineering and Scientific Computation, Zhejiang University, for its computational resources, with which the research project has been carried out. We should give special thanks to Dr. Jifa Zhang, Mr. Tingjun Yang, Dr. Guanghua Song, and Dr. Zhengge Huang, for their productive contributions.

References

[1] Baker A.J. The 21st century emergence of the multi-disciplinary problem solving environment, Chapter 4 in Enabling Technologies for Computational Science: Frameworks, Middleware and Environments, The Kluwer International Series in Engineering and Computer Science, Vol. 548, Houstis E.N., Rice J.R., Gallopoulos E., Bramley R., eds. Kluwer Academic Publishers, Boston, 2000; 45-52.

[2] Sastry L., Craig M. Scalable application visualisation services toolkit for problem solving environments. Proceedings of UK e-Science All Hands Meeting (Nottingham, UK, September, 2003), Cox S.J., ed., EPSRC, UK, 2003; 520-525.

[3] Zheng Y., Weatherill N.P., Turner-Smith E.A., Sotirakos M.I., Marchant M.J., Hassan O. Visual steering of grid generation in a parallel simulation user environment, Chapter 27 in Enabling Technologies for Computational Science: Frameworks, Middleware and Environments, The Kluwer International Series in Engineering and Computer Science, Vol.

548, Houstis E.N., Rice J.R., Gallopoulos E., Bramley R., eds. Kluwer Academic Publishers, Boston, 2000; 339-349.
[4] Zheng Y., Weatherill N.P., Turner-Smith E.A. An interactive geometry utility environment for multi-disciplinary computational engineering. International Journal for Numerical Methods in Engineering, 2002; 53: 1277-1299.
[5] Gallopoulos E., Houstis E., Rice J.R. Future research directions in problem solving environments for computational science. Report of a workshop on research directions in integrating numerical analysis, symbolic computing, computational geometry, and artificial intelligence for computational science. Technical Report 1259, Center for Supercomputing Research and Development, University of Illinois at Urbana-Champaign, 1992.
[6] Rice J.R., Boisvert R.F. From scientific software libraries to problem-solving environments, IEEE Computational Science and Engineering, 1996; 3(3): 44-53.
[7] Thompson J.F., Weatherill N.P. Structured and unstructured grid generation. Critical Reviews in Biomedical Engineering, 1992; 20: 73-120.
[8] George P.L. Automatic Mesh Generation: Application to Finite Element Methods. Masson, Paris, 1991.
[9] Schroeder W.J., Shephard MS. A combined octree delaunay method for fully automatic 3-D mesh generation. International Journal for Numerical Methods in Engineering, 1990; 29: 37-55.
[10] Weatherill N.P., Hassan O. Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. International Journal for Numerical Methods in Engineering, 1994; 37: 2005-2039.
[11] Weatherill N.P., Hassan O., Morgan K., Jones J.W., Larwood B. Towards fully parallel aerospace simulations on unstructured meshes. Engineering Computations, 2001; 18: 347-375.
[12] Zheng Y., Lewis R.W., Gethin D.T. Three-dimensional unstructured mesh generation: part 1. Fundamental aspects of triangulation and point creation. Computer Methods in Applied Mechanics and Engineering, 1996; 134: 249-268.
[13] Zheng Y., Lewis R.W., Gethin D.T. Three-dimensional unstructured mesh generation: part 2. Surface meshes, Computer Methods in Applied Mechanics and Engineering, 1996; 134: 269-284.
[14] Lewis R.W., Zheng Y., Gethin D.T. Three-dimensional unstructured mesh generation: part 3. Volume meshes. Computer Methods in Applied Mechanics and Engineering, 1996; 134: 285-310.
[15] Steger J.L., Benek J.A. On the use of composite grid schemes in computational aerodynamics. Computer Methods in Applied Mechanics and Engineering, 1987; 64: 301-320.
[16] Chesshire G., Henshaw W.D. Composite overlapping meshes for the solution of partial diffential equations. Journal of Computational Physics, 1990; 90: 1-64.
[17] Chen J.J., Zheng Y. Redesign of a conformal boundary recovery algorithm for 3D Delaunay triangulation. Journal of Zhejiang University Science A, 2006; 7: 2031-2042.
[18] Ruppert J., Seidel R. On the difficulty of triangulating three-dimensional nonconvex polyhedra. Discrete and Computational Geometry, 1992; 7(3): 227-254.
[19] Spivack M., Usher A., Yang X., Hayes M. Visualisation and grid applications of electromagnetic scattering from aircraft. Proceedings of UK e-Science All Hands Meeting (Nottingham, UK, September, 2003), Cox S.J., ed., EPSRC, UK, 2003; 672-678.
[20] Humphreys G., Hanrahan P. A Distributed Graphics System for Large Tiled Displays, Proceedings of IEEE Visualization '99 (San Francisco, California, 1999), 1999.

[21] Li K., Chen H., Chen Y., Clark D.W., Cook P., Damianakis S., Essl G., Finkelstein A., Funkhouser T., Housel T., Klein A., Liu Z., Praun E., Samanta R., Shedd B., Singh J.P., Tzanetakis G., Zheng J. Early experiences and challenges in building and using a scalable display wall system. Computer Graphics and Applications, 2000; 20(4): 29-37.
[22] Xie L., Zheng Y., Yang T., Gao W., Pan N. SimWall: A practical user-friendly stereo tiled display wall system. Journal of Zhejiang University (Science), 2007; 8(4): 596-604.
[23] Xie L., Zheng Y., Zhang J., Huang X., Huang Z. EEMAS: An enabling environment for multidisciplinary application simulations, Proceedings of the GCC 2004 International Workshops, IGKG, SGT, GISS, AAC-GEVO, and VVS, (Wuhan, China, 2004), Grid and Cooperative Computing - GCC 2004 Workshops, Lecture Notes in Computer Science, Vol. 3252, (eds. Jin H., Pan Y., Xiao N., and Sun J.). Springer-Verlag, Berlin, Heidelberg, 2004; 681-688.
[24] Paraview Users Guide. URL: `http://www.paraview.org/HTML/Features.html` (Current March 1, 2008).
[25] Wei G., Zheng Y., Zhang J. Grid service-based parallel finite element analysis, Proceedings of the Second International Workshop on Grid and Cooperative Computing (GCC2003) (Shanghai, China, 2003), Grid and Cooperative Computing: Second International Workshop, GCC 2003, Part I, Lecture Notes in Computer Science, Vol. 3032, (Li M., et al., eds.). Springer-Verlag, Heidelberg, 2004; 123-130.
[26] George P.L., Hecht F., Saltel E. Automatic mesh generator with specified boundary. Computer Methods in Applied Mechanics and Engineering, 1991; 92: 269-288.
[27] Liu A., Baida M. How far flipping can go towards 3D conforming/constrained triangulation. Proceedings of the 9th International Meshing Roundtable, New Orleans, Louisiana, USA, 2000.
[28] George P.L., Borouchaki H., Saltel E. 'Ultimate' robustness in meshing an arbitrary polyhedron. International Journal for Numerical Methods in Engineering, 2003; 58: 1061-1089.
[29] Du Q., Wang D.S. Constrained boundary recovery for three dimensional Delaunay triangulations. International Journal for Numerical Methods in Engineering, 2004; 61: 1471-1500.
[30] Chen J.J. Unstructured Mesh Generation and Its Parallelization, Ph.D Thesis, College of Computer Science, Zhejiang University, 2006.
[31] Freitag L.A., Ollivier-Gooch C. Tetrahedral mesh improvement using swapping and smoothing. International Journal for Numerical Methods in Engineering, 1997; 40: 3979-4002.
[32] Joe B. Construction of three-dimensional improved quality triangulations using local transformations. SIAM Journal On Scientific Computating, 1995; 16: 1292-1307.
[33] Klingner B.W., Shewchuk J.R. Aggressive tetrahedral mesh improvement. Proceedings of the 16th International Meshing Roundtable, Seattle, Washington, US, 2007; 3-23.
[34] Du Q., Wang D.S. Tetrahedral mesh generation and optimization based on centroidal voronoi tessellation, International Journal for Numerical Methods in Engineering, 2003; 56(9):1355-1373.
[35] Shewchuk J.R., Delaunay Refinement Mesh Generation, Ph.D Thesis, School of Computer Science, Carneigie Mellon University: Pittsburgh, PA, USA, 1997.
[36] George P.L. Improvements on Delaunay-based three-dimensional automatic mesh generator. Finite Elements in Analysis and Design, 1997; 25: 297-317.
[37] de Cougny H.L., Shephard M. Parallel unstructured grid generation, in CRC Handbook of Grid Generation, Thompson J.F., Soni B.K., Weatherill N.P., eds. CRC Press, Inc.: Boca Raton. 1999; 24.1-24.18.
[38] Chrisochoides N., A survey of parallel mesh generation methods. Numerical Solutions

of Partial Differential Equations on Parallel Computers, 2005, http://www.cs.wm.edu/nikos/pmeshsurvey.pdf.
[39] Okusanya T., Peraire J. Parallel unstructured mesh generation. Proceedings of the 5th International Conference on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields, Mississippi State University, MS, USA, 1996.
[40] Okusanya T., Peraire J. 3D Parallel unstructured mesh generation. Proceedings of the 1997 Joint ASME/ASCE/SES Summer Meeting, Special Symposium on Trends in Unstructured Mesh Generation, Northwestern University, Evanston Illinois, USA, 1997.
[41] Chrisochoides N., Nave D. Parallel Delaunay mesh generation kernel. International Journal for Numerical Methods in Engineering, 2003; 58: 161-176.
[42] Larwood B.G., Weatherill N.P., Hassan O., Morgan K. Domain decomposition approach for parallel unstructured mesh generation. International Journal for Numerical Methods in Engineering, 2003; 58: 177-188.
[43] Boufflet J.P., Breitkopf P., Rassineux A., Villon P. A modular design for a parallel multifrontal mesh generator. Proceedings of the 8th International Euro-Par Conference on Parallel Processing, Paderborn, RFA, Germany, 2002.
[44] Galtier J., George P.L. Prepartitioning as a Way to Mesh Subdomains in Parallel. 5th International Meshing Roundtable, Pittsburgh, PA, USA, 1996.
[45] Said R., Weatherill N.P., Morgan K., Verhoeven N.A. Distributed parallel Delaunay mesh generation. Computer Methods in Applied Mechanics and Engineering, 1999; 177: 109-125.
[46] Cruz-Neira C., Sandin D.J., DeFanti T.A. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. Proceedings of the 20th annual conference on Computer graphics and interactive techniques, 1993: 135-142.
[47] Hereld M., Judson I.R., Stevens R.L. Introduction to building projection-based tiled display systems. IEEE Computer Graphics and Applications, 2000; 20(4): 22-28.
[48] Majumder A., Jones D., McCrory M., Papka M.E., Stevens R. Using a camera to capture and correct spatial photometric variation in multi-projector displays. IEEE International Workshop on Projector-Camera Systems, 2003.
[49] Raskar R., Brown M., Yang R., Chen W., Towles H., Seales B., Fuchs H. Multi-projector displays using camera-based registration. Proceedings of the Conference on Visualization'99: Celebrating Ten Years, 1999: 161-168.
[50] Humphreys G., Houston M., Ng R., Frank R., Ahern S., Kirchner P.D., Klosowski J.T. Chromium: a stream-processing framework for interactive rendering on clusters. ACM Transactions on Graphics, 2002; 21(3): 693-702.
[51] DMX, http://dmx.sourceforge.net/ (Current March 1, 2008).
[52] Bush R.H., Mani M. A two-equation large eddy stress model for high subgrid shear. AIAA Paper, 2001; No. 2001-2561.
[53] Cai X.D., Ladeinde F. An Evaluation of the partially-resolved numerical simulation procedure for near-wall turbulence prediction. AIAA Paper, 2006, No. 2006-115.
[54] Fan T.C., Tian M., Edwards J.R., Hassan H.A., Baurle R.A. Validation of a hybrid reynolds averaged/large eddy simulation method for simulating cavity flame-holder configuration. AIAA Paper, 2001; No. 2001-2929.
[55] Georgiadis N.J., Iwan J., Alexander D., Reshotko E. Development of a hybrid RANS/LES method for compressible mixing layer simulations. AIAA Paper, 2001; No. 2001-0289.
[56] Menter F.R., Kuntz M., Bender R. A Scale-adaptive simulation model for turbulent flow predictions. AIAA Paper, 2003; No. 2003-0767.
[57] Shih T.H., Liu N.S. Partially resolved numerical simulation. AIAA Paper, 2004; No. 2004-

0160.
[58] Shih T.H., Liu N.S. A unified strategy for numerical simulation of turbulent flows (Private Communication), 2005.
[59] Zheng Y., Zou J. Partially resolved numerical simulation for supersonic turbulent combustion. The 14th AIAA/AHI Space Planes and Hypersonic Systems and Technologies Conference (Canberra, Australia, November 2006), AIAA Paper, 2006; No. 2006-8040.
[60] Zou J., Zheng Y., Liu O. Simulation of turbulent combustion in DLR scramjet, Journal of Zhejiang University (Science) A, 2007; 8(7): 1053-1058.
[61] Odam J. Scramjet Experiments Using Radical Farming. Ph.D Thesis. Department of Mechanical Engineering, University of Queensland, 2004.
[62] Jason B.S., Edwards J.R., Michael K.S., Robert A.B. Numerical simulation of scramjet combustion in a shock tunnel. AIAA Paper, 2005; No. 2005-0428.