Defence Science Journal, Vol. 61, No. 4, July 2011, pp. 299-305, DOI: 10.14429/dsj.61.1115

REVIEW PAPER

Real-time Interactive Steerable Scientific Visualisation of Free Surface Flow in the Context of Synthetic Vision

G.M. Lingaraju[*#], P. Nithiarasu[**], M. Kusuma[***], and Rajaram M. Gowda[*]

[*]*M.S. Ramaiah Institute of Technology, Bengaluru-560 054*
[**]*Swansea University, Swansea SA2 8PP, UK*
[***] *Dayananda Sagar College of Engineering, Bengaluru-560 078*
[#]*E-mail: gmlraju@gmail.com*

ABSTRACT

The existing techniques in the context of fluid modelling and simulation have been reviewed. Generic framework that enables an easy integration of various modules has been developed, extending the work to real-time simulation and visualisation, and developed user interaction during run-time using the concept of computational steering. In the generic framework developed, a new class of visualisation technique that simplifies the visualisation tasks of a scientist has been identified. Investigating these techniques is important as the tools such as visualisation tool kit (VTK) and VTK designer are freely available (Open Source) and easy to integrate. A technique for interactive visualisation of the free surface flow, that introduces the concepts of geometrical steering and properties steering, has been developed. These techniques constitute computational steering. The concept of real-time interactive scientific visualisation using a surface flow application has been demonstrated. As a proof of concept, the behaviour of flow is simulated and visualised during training in a virtual environment on a desktop computer. It is expected that the generic framework, device, interface, and simulation engine, used in this work will not only have a significant impact in the area of free surface flow, but also in real-time applications such as synthetic vision in avionics. The underlying formulation/methodology involved in parametric approach is explained.

Keywords: Properties steering, geometrical steering, real-time simulation, interaction, computational steering, system design, synthetic vision, visualiser

1. INTRODUCTION

Conventional method of viewing the results is via post-processing the data (results) after completion of a simulation. With the desire for developing a rapid visualisation system, a method is to be developed during the run-time, instead of viewing the post-processed data. This can be achieved by implementing interactive visualisation techniques like changing the camera view, colour and other parameters related to computer graphics, that allows spontaneous control of the data visualisation displays and enhances the scientific visualisation. Scientists can interactively compute output data in the visualisation diagnostics and through this, steer the parameters of simulation code in real-time, called as Interactive scientific visualisation. Here is an attempt to do this using an application of free surface simulation. Since this work is interdisciplinary, literature review has been highlighted by keeping fluid simulation application model in different categories like visualisation, computational steering, and synthetic vision. As far as independent areas are concerned, there exist ample literature, but the integration of all these technologies is rarely available.

1.1 Visualisation

Overview of the early approaches in the field of ocean visualisation, issues related to plunged breaking waves modelling due to sinusoidal assumption in the parametric surface, as well as effective use of graphics hardware to achieve more realism can be found[1-3]. The most closely related work to the scientific visualisation of the current topic is solution of 2-D Navier-Strokes equation via a computational fluid dynamics (CFD) method mapping the surfaces into 3-D and achieve realistic real-time fluid surface behaviours by Jim X. Chen[4], *et al*. Interactive scientific visualization, as currently understood and practiced, is still a relatively new discipline and many unresolved problems of visualisation are discussed[5].

1.2 Synthetic Vision

The synthetic vision information systems is an emerging revolutionary technology, which will be an added value to cockpit displays that give pilots a clear ‘out-of-the-window’ view, regardless of fog (weather) or low ambient lighting conditions. Visual displays is often the most efficient method of presenting safety-critical data to pilots in real-time. Background of synthetic vision and human factor issues has been briefed in a panel discussion[6] from the perspective of government, academic, military and industry. D.C. Foyle[7], *et al.* have reviewed some of the research related to enhance synthetic vision systems. Out-of-the-window viewing in low visibility weather with limited display characteristics, that may result in degradation of the visual cues, is discuss. To

Received 23 March 2011, published online 25 July 2011

counteract such degradation, augmented displays are proposed to enhance the necessary visual cues back to the scene. The studies[8] demonstrate that such a visualisation cockpit decision-support system significantly improves the performance of helicopter pilots during landing under turbulent conditions. Their proposed plan focuses on a simulated-assisted reality system for aerial vehicles using 3-D to provide pilots with clear and intuitive means of understanding their flying environment. When the aircraft interacts with the fluids such as turbulent air, fog, clouds or water, a study regarding the reactions from these obstructions and the visualisation in real-time will be helpful. This proof of concept is modelled, simulated, and visualised in real-time. The technologies like real-time physics-based computation and synthetic vision in aircraft are proposed. Lawrence[9], *et al.* have given future directions of research towards design of added functionality display concept for accident prevention and increased operational capability.

1.3 Computational Steering

With the advancement in computer processing power and networking, modelling and simulation forms a significant way of solving complex problems. Previously, the problems tackled in batch mode can now be visualised at run-time[10]. During run-time, one can interact and control the simulation by coupling a supercomputer with visualisation and steering interface and also alter parameters of the computation depending on the needs. This ability to monitor and change parameters of the computational process at run-time from anywhere without delay in visualising the result, is called computational steering, i.e., integration of computation, visualization, and control into one system.

Different tools like access grid, reality grid, steering library, etc. can be used to develop such a system. Pickles[11], *et al.* have steered many applications by enabling the steering library. The advances in computational steering and combination of real-time scientific visualisation will influence emerging trends in synthetic vision from research pilot projects to meaningful applications in avionics.

2. SIMULATIONS

An important property of a fluid is its dynamic behaviour. The author has introduced to computer graphics, the applications of Arbitrary Lagrangian Eulerian (ALE) formulation for free-surface flows using the characteristic-based split (CBS) scheme which they have developed in C language as numerical solution tool in simulation engine, to have a low-level control for interaction with the simulator. The target problem is formulated by discretising the model and further processing it through the finite method, which results in a large system of simultaneous ordinary differential equations. By solving these equations, the models which simulate the dynamics at each steps, have been designed. A standard broken dam simulation is taken as test case application. The details of the numerical formulations as well as their first levels of non-interactive scientific visualisations are found[12,13] and the extension of this work is presented.

2.1 Visualiser

Developers have lots of choices when they have to design and implement scientific visualisation applications. Several open source and commercial visualisation frameworks and tools, such as VTK, AVS, IRIS-Explorer, Khoros, and Labview, simplify the construction and configuration of a visualisation system. Some of these also provide mechanisms for connecting to external solvers, and visualise its results. Researchers who are interested in capitalising 3-D visualisation capabilities to view and better present the results of their computations can use such tools to quickly render and animate results. VTK designer provides researchers an open source visualisation system that offers the flexibility of a toolkit and ease of use of commercial graphical user interface (GUI) tools that allows researchers to just concentrate on the computation aspects of their solver and not be concerned about rendering, interaction, and user interface (UI).

2.1.1 Case Study

The value addition provided by the VTK designer used in solving the broken dam simulation problem is demonstrated. Flow visualisation system has the following entities:

- Input to the solver (in this case, it is input file of finite element method (FEM) engine).
- Flow visualisation solver (FVS)
- Output files containing the results of FVS
- 3-D renderer that displays the results of FVS
- UI that helps user to interact with the results and modify the inputs to FVS.

These entities are related to each other as shown in Fig. 1.

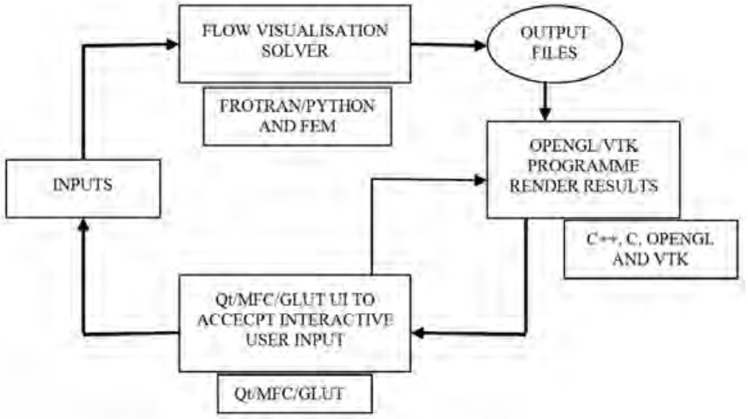


Figure 1. Relationship of entities relation to each other.

The flow visualisation solver accepts inputs in the form of input files and computes the flow. The computed results are written into output files as a set of (x,y) coordinates, where each output file describes the flow at a given instance of time. The output files are then treated as input to an OpenGL/VTK program, which then renders the flow in 3-D. In this case, simulation is in 2-D and third dimension is simply extruded to visualise the results in 3-D. User interaction with the 3-D view-port will have to be explicitly handled and programmed. Additionally, the user interface required for changing input parameters to the solver have to be programmed as well. To do all of the above, a researcher would have to know the following:

- Fortran and FEM for implementing the solver
- C++, OpenGL (or VTK) to implement the 3-D rendering system
- Qt/MFC/wx Widgets/GLUT to implement the user interface using which input parameters may be altered.

VTK designer to drastically simplify the visualisation process is used. Although it has its share of concepts and features to learn, these proved to be much simple when compared to C++, OpenGL, and VTK etc.

With VTK designer, user can directly import the output files into a pre-written easy-to-use GUI, view and interact with results by writing small snippets of JavaScript code. The JavaScript code only imports data from the output files into VTK designer and all the 3-D rendering (Fig. 2), interaction and OpenGL state-management are handled by VTK designer. Additionally, a plug-in is written for VTK designer to further simplify this.

The plug-in takes care of directly reading output from FVS and visualising it in VTK designer (Fig. 3). With this setup in place, they can just concentrate on the solver part and not be concerned about rendering and UI. Figure 4 is a screenshot of VTK designer with the FVS plug-in. The VTK function (vtkDamVisFileReader) reads data from output files generated by FVS and renders these in VTK designer using pre-written rendering modules.

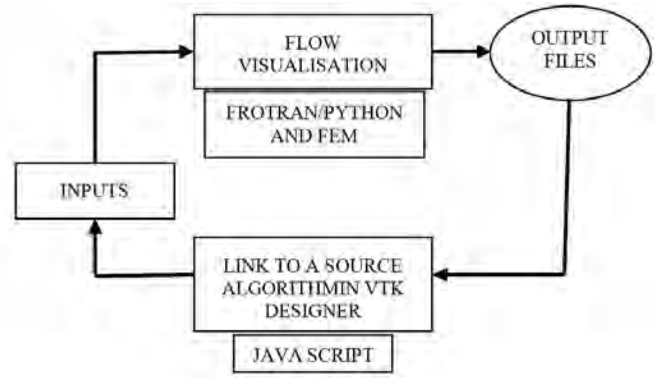


Figure 2. Without rendering modules.

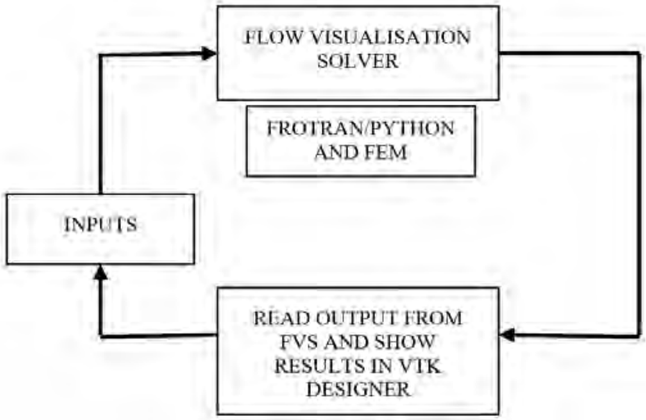
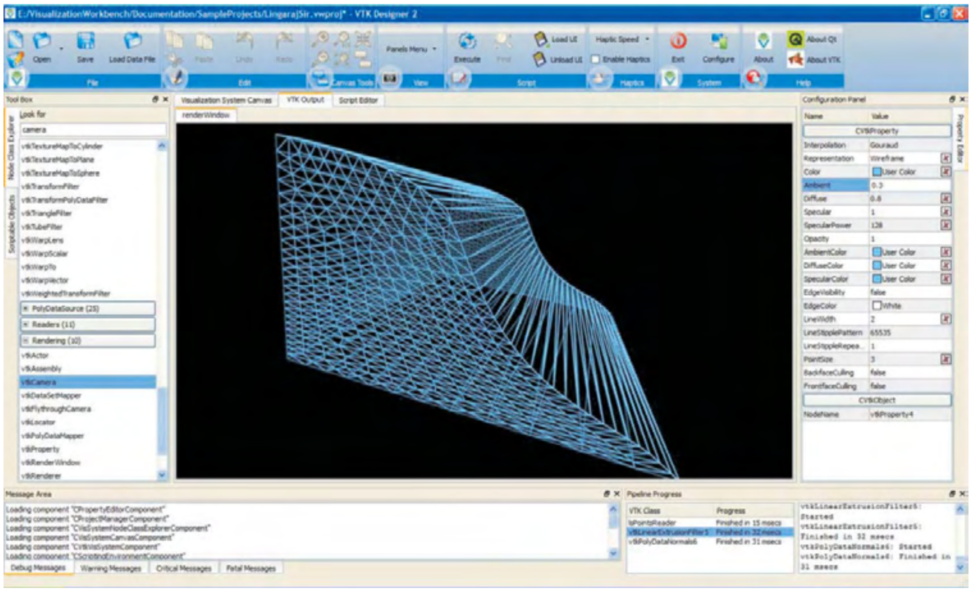


Figure 3. Direct linkage (without conventional output files).

3. APPLICATIONS

Considering the scientific data, which is generated in this simulation of fluid flow, they animate the flow pattern using the VTK designer as a part of this research work. The snapshots of the dynamic fluid flow visualisation are shown in Figs 4, 5 and 6 as illustrations and the further animation details and abstract formulation of this free surface flow are discussed[13].

In the data flow diagram (DFD) shown in Fig. 7, the source and the sink are the simulator and the visualiser, respectively, and are represented using rectangles. Computational steering environment forms the process in the data flow diagram and hence is represented in form of ellipse. The researcher can initially give inputs through conventional file system (or through



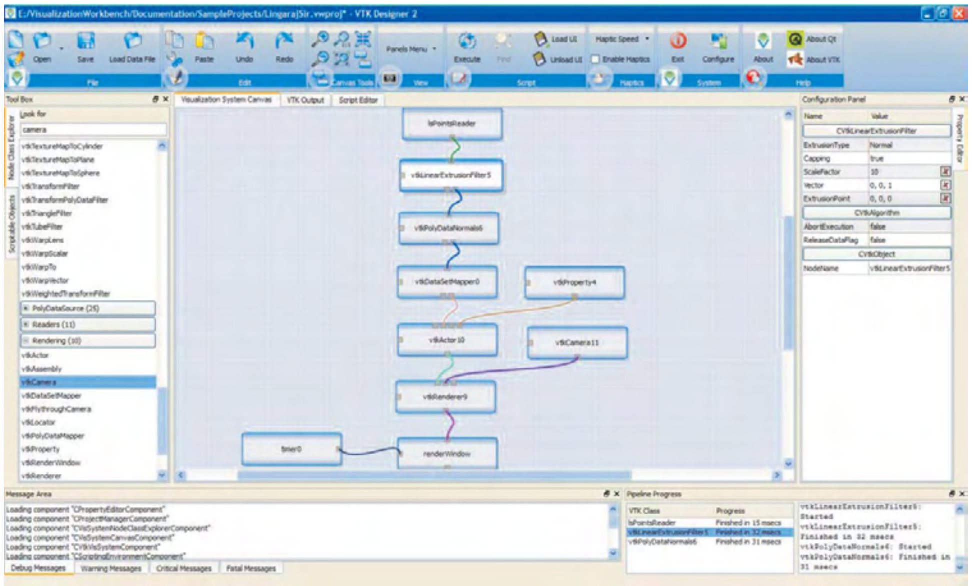
Name	Value
CVtkProperty	
Interpolation	Gouraud
Representation	Wireframe
Color	User Color
Ambient	0.3
Diffuse	0.8
Specular	1
SpecularPower	128
Opacity	1
AmbientColor	User Color
DiffuseColor	User Color
SpecularColor	User Color
EdgeVisibility	false
EdgeColor	White
LineWidth	2
LineStipplePattern	65535
LineStippleRepea...	1
PointSize	3
BackfaceCulling	false
FrontfaceCulling	false
CVtkObject	
NodeName	vtkProperty4

```
Loading component "CPropertyEditorComponent"
Loading component "CProjectManagerComponent"
Loading component "CVisSystemNodeClassExplorerComponent"
Loading component "CVisSystemCanvasComponent"
Loading component "CVtkVisSystemComponent"
Loading component "CScriptingEnvironmentComponent"
```

VTK Class	Progress
lsPointsReader	Finished in 15 msecs
vtkLinearExtrusionFilter5	Finished in 32 msecs
vtkPolyDataNormals6	Finished in 31 msecs

```
vtkLinearExtrusionFilter5:
Started
vtkLinearExtrusionFilter5:
Finished in 32 msecs
vtkPolyDataNormals6: Started
vtkPolyDataNormals6: Finished in
31 msecs
```

Figure 4. A snapshot of wire frame fluid flow model.



Configuration Panel

Name	Value
CVtkLinearExtrusionFilter	
ExtrusionType	Normal
Capping	true
ScaleFactor	10
Vector	0, 0, 1
ExtrusionPoint	0, 0, 0
CVtkAlgorithm	
AbortExecution	false
ReleaseDataFlag	false
CVtkObject	
NodeName	vtkLinearExtrusionFilter5

Message Area

```
Loading component "CPropertyEditorComponent"
Loading component "CProjectManagerComponent"
Loading component "CVisSystemNodeClassExplorerComponent"
Loading component "CVisSystemCanvasComponent"
Loading component "CVtkVisSystemComponent"
Loading component "CScriptingEnvironmentComponent"
```

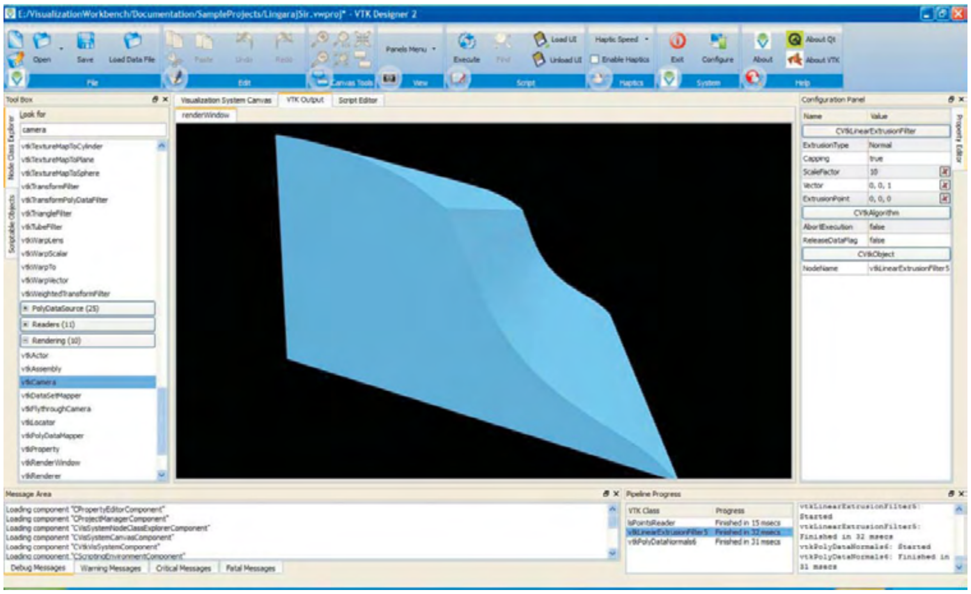
Debug Messages | Warning Messages | Critical Messages | Fatal Messages

Pipeline Progress

VTK Class	Progress
lsPointsReader	Finished in 15 msecs
vtkLinearExtrusionFilter5	Finished in 32 msecs
vtkPolyDataNormals6	Finished in 31 msecs

```
vtkLinearExtrusionFilter5:
Started
vtkLinearExtrusionFilter5:
Finished in 32 msecs
vtkPolyDataNormals6: Started
vtkPolyDataNormals6: Finished in
31 msecs
```

Figure 5. Pipeline execution for fluid animation.



Configuration Panel

Name	Value
CVtkLinearExtrusionFilter	
ExtrusionType	Normal
Capping	true
ScaleFactor	10
Vector	0, 0, 1
ExtrusionPoint	0, 0, 0
CVtkAlgorithm	
AbortExecution	false
ReleaseDataFlag	false
CVtkObject	
NodeName	vtkLinearExtrusionFilter5

Message Area

```
Loading component "CPropertyEditorComponent"
Loading component "CProjectManagerComponent"
Loading component "CVisSystemNodeClassExplorerComponent"
Loading component "CVisSystemCanvasComponent"
Loading component "CVtkVisSystemComponent"
Loading component "CScriptingEnvironmentComponent"
```

Debug Messages | Warning Messages | Critical Messages | Fatal Messages

Pipeline Progress

VTK Class	Progress
lsPointsReader	Finished in 15 msecs
vtkLinearExtrusionFilter5	Finished in 32 msecs
vtkPolyDataNormals6	Finished in 31 msecs

```
vtkLinearExtrusionFilter5:
Started
vtkLinearExtrusionFilter5:
Finished in 32 msecs
vtkPolyDataNormals6: Started
vtkPolyDataNormals6: Finished in
31 msecs
```

Figure 6. A snapshot of fluid flow after rendering.

GUI) to the computational environment, which are supplied as parameters to the simulation. Depending on the parameters, the result is calculated and sent back to the visualisation environment and the same is animated/rendered as output. Changes can be made at run-time, by directly manipulating the input data (supplied as state variable to the simulation) in GUI through computational steering environment, and the new results are calculated and rendered.

3.1 Steering

With the closed loop of computational steering, scientists can react to output image as these take place through interactively manipulating the input parameter. This technique enhances efficiency significantly by minimising the delay between changes to input parameters and the corresponding display of the results[14]. This technique is adopted for this work and is further classified and distinguished as geometrical steering and properties steering– a new classification introduced by the authors.

3.1.1 Geometrical Steering

While running the simulation, user can introduce some obstruction in visualisation in the direction of flow path abruptly at arbitrary position and time. System will detect the collision of obstruction with the flow surface and the constraints will be updated in the variables of boundary conditions in the FEM engine. Accordingly, immediate effect of change in the flow direction can be observed, which is not pre-defined in the code, and since it deals with the geometry of simulation, it is called this as geometrical steering. To further illustrate the

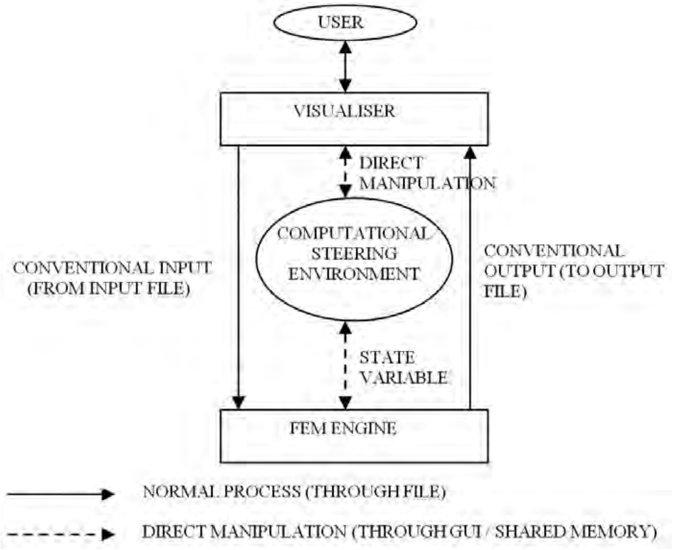


Figure 7. Steering environment.

concept of geometric steering, Fig. 6 shows the flow before obstruction and Fig. 8 shows the flow after obstruction is introduced through GUI.

3.1.2 Properties Steering

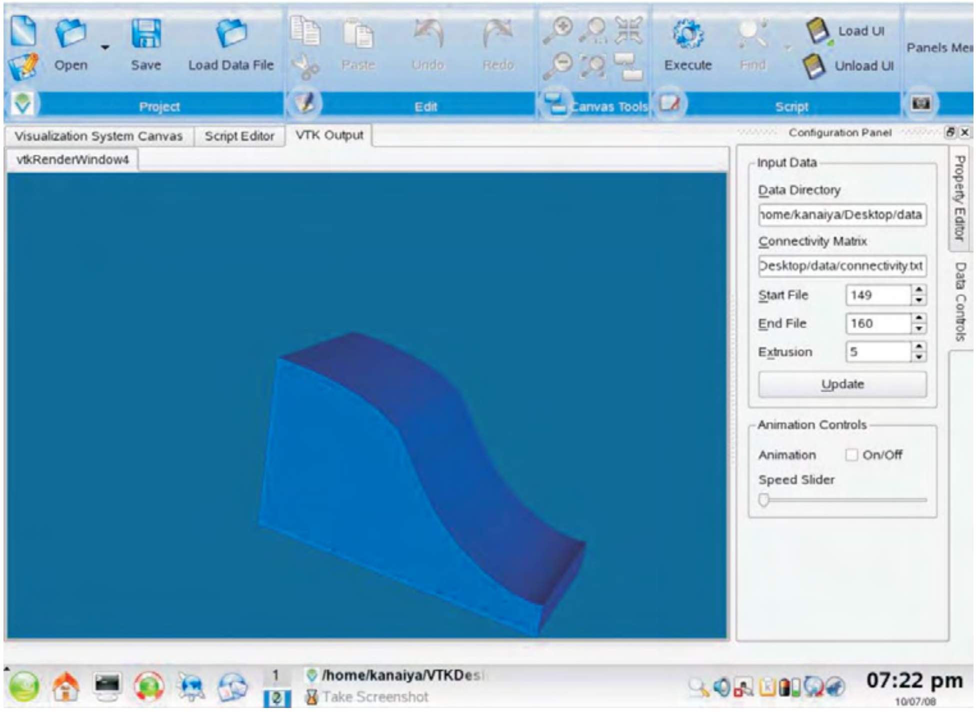
A simple sliding bar was created in the GUI. A variable *g* (gravitation) in the diffuse term of the FEM engine, as a test case was taken. This variable was directly linked to the sliding bar. Any modification in the sliding bar is the modification of the corresponding variable in the running program. Similarly, one can include other properties of flow by altering the position of sliding bar, hence, one can call it properties steering. The corresponding change in the rate of flow in the visualisation during interaction is observed.

The issue of providing human intervention in the timing loop of a real-time simulation (steering support) for variables, that are not visible in the visualisation, is addressed. Implementation of the same can be done in many ways. In this work, the values of the variable g (gravitation) can be altered through a sliding bar in the GUI (when the simulation is running) which does the steering. The abrupt changes in flow rate were observed in the output, for the corresponding changes made in the variables. This work provides a novel combination of simulation, visualization, and interaction. In addition to computational steering, a method of controlled execution was used to manipulate the communication structure during run-time, using shared memory.

4. DYNAMIC GEOMETRY USING PARAMETRIC APPROACH

There are different ways of representing data[15] for geometrical models, such as polygon mesh surfaces, parametric surfaces, quadric surfaces and particle-based methods. The present work is an extension of the work from polygon mesh surfaces to parametric surfaces[16]. The details of the underlying formulation/methodology are explained, so that any researcher/user can use the technique for steering. In general, this will help to design similar type of interfaces to implement the computational steering.

Knowing the computed coordinates of the free surface (nodes of the FEM mesh, transforming this node to the geometric mesh) from the simulation at every time step. The control points of the free surface have been calculated, in the



Input Data
Data Directory: home/kanaiya/Desktop/data
Connectivity Matrix: Desktop/data/connectivity.txt
Start File: 149
End File: 160
Extrusion: 5

Animation Controls
Animation ☐ On/Off
Speed Slider

Figure 8. Geometric steering due to obstruction.

reverse direction using formulation of parametric curves/surfaces. Further parametric surface has been regenerated by taking the weighted summation of product of control points and blending functions (Hermite/Bezier formulation). Implementation of this adds to generate more realistic surfaces than through polygon approximation. This method will help to extend the work to breakable surfaces to achieve more realism in visualisation.

To make the simulation strictly real-time, instead of computing the numerical methods in a time loop, timer of the system have been connected to calculate the real-time steps in simulation, instead of fixed time step interval using conventional loop. With this, the authors initiated to consider the hardware issues/solutions to move it towards really real-time, instead of nearly real-time. They have also extended the work using C for graphics (CG) to program for hardware-related issues such as speeding up of rendering process and other display requirements for real-time visualisation issues in GPU level.

5. CONCLUSIONS

Lot of research is going on in the area of simulation, steering, and scientific visualisation as independent areas. But when integration is considered, extensive information fusion works need to be done to synchronise all the information. Real-time issues also arise to get physical realistic responses for the interactions in virtual environment.

Computational steering is the process of changing the data of the computer program during run-time and visualise the effect as the data changes. The use of computational steering enables the scientist to view their results dynamically during run-time, instead of waiting for the post-processed output. It also reduces the time to observe the response of the simulation for the changed parameter. The authors found this technique suitable in the area of avionics, especially when the aircraft in interacting with fluid and its responses in real-time can be simulated and visualised in synthetic vision.

Aragon[8] has carried out studies with and without visual hazard indicators for crash analysis and also demonstrates that simple and static visualisations yield improved performance on safety-critical tasks. To design a system, which can not only perform dynamic simulation and visualisation, but which can be mapped into real world, in real-time, and can also handle interaction with flow using computational steering technique, has been described.

With reference to our generic architecture, which allows information fusion for various human-machine interfaces, it focuses on visualisation and computational steering only in the context of synthetic vision. They have designed the system from scratch. Figures 1-5 show one of the preliminary steps to reach the generic system architecture. A comprehensive, yet extensible VTK designer, which handles most of the framework, was used. A case study has been described from the field of computational fluid dynamics to the area of computer graphics.

To provide a novel combination of simulation, visualization, and interaction, an attempt has been made to present human intervention in the timing loop of a real-time simulation (steering support) for variables that are not visible in the visualization. This was accomplished through a GUI sliding bar which changes the gravitation values to interact with simulation, by making changes in the variables abrupt changes of the flow rate in the output are observed. To do this computational steering, shared memory is used as communication structure to manipulate the data during run-time.

Application of this technology to synthetic vision information systems of aircraft has been proposed. In this technology, even though there is no vision at the outset, the pilot is still operating the aircraft through a synthetic vision display. This system provides situational awareness to the pilot using hydrological databases obtained using GPS. Another alternative, such as simulation engines, along with the data available from the databases, is proposed. With this, not only projection of animated data on the display systems along the flight path (sky high way) is displayed, but pilot can also visualise the physics-based reactions of the data along the flight path.

REFERENCES

1. Tessendorf, J. SIGGRAPH 2001 Course 1. 47: Simulating nature: Realistic and interactive techniques. ACM SIGGRAPH, 2001.
2. Jensen, Lasse Staff & Golias, Robert. Deep-water animation and rendering. 26 September 2001. http://www.gamasutra.com/gdce/2001/jensen/jensen_01.htm
3. Schneider, J. & Westermann, R. Towards real-time simulations of water surfaces. *In* Proceedings of the Vision Modelling and Visualisation Conference. Stuttgart, Germany, November 21-23, 2001. pp. 211-18.
4. Chen, Jim X.; Hughes & Moshell, J. Michael. Real-time fluid simulation in a dynamic virtual environment. *IEEE Comp. Graph. Appl.*, 1997, **17**(3), 52-61.
5. Chris, R. Johnson. Top scientific visualisation research problems. *IEEE Comp. Graph. Appl.*, 2004, **24**(4), 13-17.
6. Prinzel, Lawrence J.; Comstock, J. Raymond; Etherington, Tim; French, Guy A.; Snow, Michael P.; Endsley, Mica R.; Wickens, Christopher D. & Corker, Kevin M. Human factors issues in synthetic vision displays: Government, academic, military, and industry perspectives. *In* Proceedings of the Human Factors and Ergonomics Society 48th Annual Meeting 2004. Aerospace Systems, New Orleans, Louisiana. pp. 75-79.
7. Foyle, D.C.; Ahumada, A.J.; Larimer, J. & Sweet, B.T. Enhanced/synthetic vision systems: Human factors

research and implications for future systems. *SAE Transaction: Journal Aerospace*, 1992, **101**, 1734-741.

8. Aragon, Cecilia R. Using visualisation in cockpit decision support systems. Lawrence Berkeley National Laboratory, University of California. IEEE SMC. Hawaii 2005. Report No. LBNL-58375
9. Prinzel, Lawrence J.; Kramer, Lynda J.; Arthur, Jarvis J. & Bailey, Randall E. Multi-dimensionality of synthetic vision cockpit displays: Prevention of controlled-flight-into-terrain. *In* 50th Annual Meeting Proceedings Human Factors and Ergonomics Society, NASA Langley Research Center. 2006. pp. 2056-060.
10. Chatzinikos, F. & Wright, H. Computational steering by direct image manipulation. *In* Proceedings of Vision Modelling and Visualization, 2001, Stuttgart, Germany. pp. 455-62.
11. Pickles, S.M.; Haines, R.; Pinning, R.L. & Porter, A.R. Practical toolkit for computational steering. *Philosophical Trans., Series A, Math. Phys. Eng. Sci.,* 2005, **363**(1833), 1843-53.
12. Lingaraju, G.M.; Bagewadi, C.S. & Nithiarasu, P. An arbitrary Lagrangian Eulerian (ALE) formulation for free surface flows using the characteristic-based split (CBS) scheme. *Int. J. Numer. Meth. Fluids*, 2005, **48**, 1415-428.
13. Lingaraju, G.M.; Bagewadi, C.S. & Nithiarasu, P. Real-time simulation of free surface flow for computer graphics applications. *Int. J. Syst., Cyber. Inform.*, 2009, **1**(2), 69-71.
14. Mulder, Jurriaan D.; van Wijk, Jarke J. & Liere, Robert van. A survey of computational steering environments. *J. Future Gen. Comp. Syst.,* 1999, **15**(1), 119-29. DOI: 10.1016/S0167-739X(98)00047-8
15. Adabala, Neeharika & Swami, Manohar. Techniques for realistic visualisation of fluids: A survey. *Comp. Graph. Forum*, 2002, **21**(1), 65–82.
16. Mulder, Jurriaan D. & van Wijk, Jarke J. 3-D computational steering with parametrised geometric objects. *In* Proceedings of IEEE Conference on Visualisation, 1995, **466**. pp. 304-11.

Contributors

Dr G.M. Lingaraju received PhD from Kuvempu University, Karnataka, in 2009. He is currently working as Professor, in the Department of Information Science and Engineering, MS Ramaiah Institute of Technology, Bengaluru. His Research Interests include Computer graphics, visualisation, CFD, synthetic vision, virtual reality, haptic technology, brain computer interface, and robotics.

Dr Perumal Nithiarasu received PhD from IIT Madras, Chennai, in 1997 and awarded DSc by Swansea University, UK in 2007. He is currently working as Professor at the College of Engineering, Swansea University, United Kingdom. He currently holds an Advanced Fellowship from the Engineering and Physical Sciences Research Council, UK. His research interests include: Computational methods and computational bioengineering. He is a fellow of IMA (UK) and IMechE (UK) and a senior member of AIAA.

Ms M. Kusuma received MTech (Information Technology) in 2005 and currently perusing PhD from Visvesvaraya Technological University (VTU), Belgaum. She is working as Senior Faculty in Department of Computer Science & Engineering at Dayananda Sagar College of Engineering, Bengaluru. Her research interests include: Advanced EEG signal processing, speech and control, brain computer interface, computational fluid dynamics and visualisation.

Mr Rajaram M Gowda is currently pursuing his PhD from VTU, Belgaum. Presently, he is working as Associate Professor at M.S. Ramaiah Institute of Technology, Bengaluru. His research interests include: Geometric modelling and image processing. He is a member of ISTE and NTS–India.