Parallel Support Toolkit for AVS/Express

James Perrin
SVE Team, Manchester Computing, University of Manchester

Bringing multi-processor performance to a leading visualization application

Introduction

AVS/Express is a leading visualization and application development package. It provides a visual programming interface where the user connects together modules from libraries of data readers, filters and visualization techniques to create an AVS network (visualization pipeline). AVS/Express is a general visualization package though it is mainly aimed at scientific visualization for engineering, medicine, geology, simulations and other fields. Manchester Visualization Centre (MVC) has extensive experience with the package; using it for many projects and also hosting the International AVS Centre, an internet repository of user submitted modules.

AVS/Express is a well established piece of software, however it has always been single processor based. AVS took its first step towards HPC support with the development of Multi Pipe Express (MPE) which takes advantage of multiple graphics pipes as found on SGI Onyx hardware to render large datasets and enable use in multiple projector environments such as Caves and RealityCenters. MVC has designed and developed both MPE and the soon to be released Graphics Cluster Edition (GCE) for PC clusters, however, the visualization computation is still performed on a single CPU. Users obviously want to utilise the multi-processor systems that they are running MPE/GCE on, hence the conception of the Parallel Support Toolkit for AVS/Express between MVC, AVS, KGT (Kubota Graphics Technology Inc.) and JAERI (Japan Atomic Energy Research Institute).

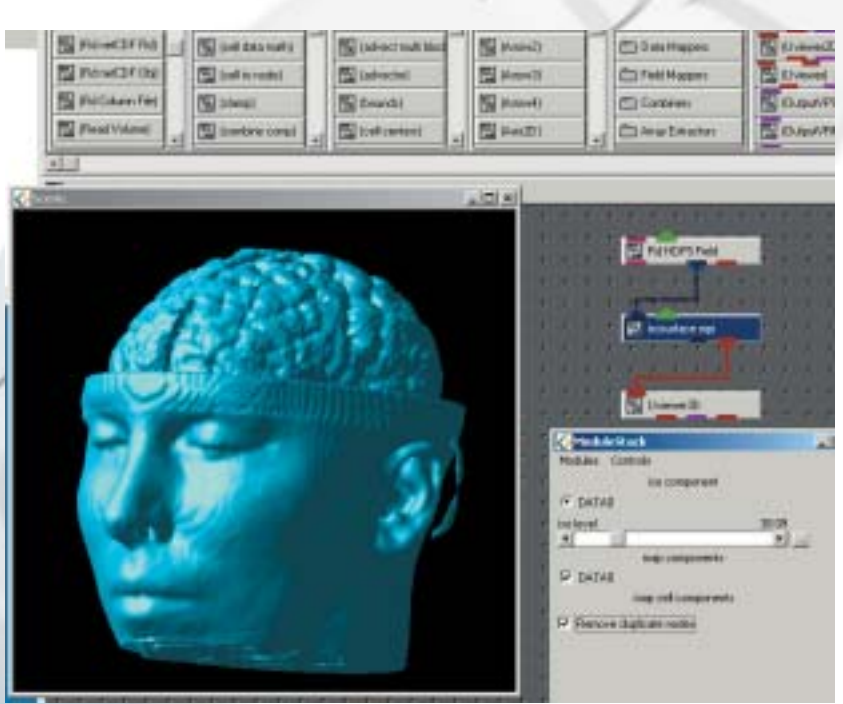

Figure 1: A simple AVS network exploiting the MPI version of isosurface from the feasibility phase

Overview

Unlike previous attempts to add parallel computation to AVS/Express, such as VIPAR, parallel modules will be aware of other parallel modules in the AVS network so the computation nodes can pass data between computation methods without the need to gather, recompose, decompose and distribute the data as each module is executed. Instead modules will pass tags (meta information) between themselves. These parallel modules can then be used with the standard serial modules to create new AVS networks or integrated into existing networks to enhance the performance of the visualization application.

PST is a multi phased project to run over a course of two years. It will provide developers with a toolkit (API and skeleton code) to enable them to build their own parallel modules for AVS/Express and after the first year a suite of parallel visualization modules will be created to empower the end user with the ability to create AVS networks and applications that harness the power of SMP and cluster based systems. The second year will extend the parallelism paradigms of AVS/Express, optimizing and integrating with current rendering methods and MPE.

The project is currently in Phase 2, the design and implementation phase of the basic PST framework. This followed on from an initial feasibility stage that demonstrated the benefits of parallelization for the

standard isosurface module. PST is being developed for SGI and PC cluster systems and it was originally envisioned that both MPI and OpenMP (on SGI) would be utilised but there are compatibility issues between MPE and OpenMP. MPICH is being used for the PC cluster version.

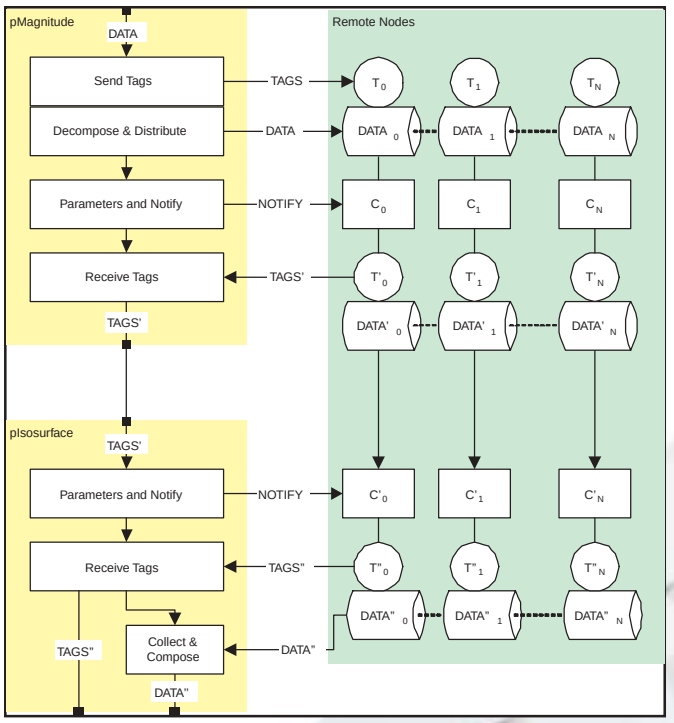

Figure 2: Passing tags rather than data between modules is an advantage over previous attempts to add parallel computing to visualization applications.

Parallelism, Lots of Parallelism

All the modules will be based on data parallelism. The problem is that different modules may require different decomposition methods; each module therefore has an associated schema which specifies its desired decomposition method (as well as other information about how the module is to be executed). These schema are then resolved to discern where re-composition and distribution is required e.g. a module that requires a domain border of N elements can pass data to a module that doesn't require border elements (assuming the data redundancy overhead being outweighed by not needing to recompose and distribute the data), however the reverse is not true and the schema resolution will force the first module to recompose its data before passing it to the second module. If a set of modules should execute on the same set of nodes, data will be cached on the nodes so that module parameter changes don't require further data distribution. Data only therefore needs to be distributed to and from the first and last modules.

Task parallelism will be implemented at an AVS network level where two or more modules share the same input. Normally these would execute in a serial manner controlled by the Object Manager (OM), but by getting the first module to distribute the data and then return control to the OM after initiating the computation, the next module can start its computation and so on.

Note that each AVS network still executes one cycle at a time, so after a module has executed it has to wait for the remaining downstream modules to complete and the scene to be rendered. By decoupling PST module execution from the OM, when a module has completed execution it can then re-execute if new data is available or parameters have changed. Decoupling modules from the OM enables this "pipeline parallelism" as well as further parallel features and optimization. These features will need to be balanced against the real world advantages that they can provide and the feasibility of adding these to a large and complex serial application within a reasonable timeframe.

MPE Integration

A driving force behind this work has been users of MPE, as well as increasing the computational performance of MPE. Later phases will integrate PST with the rendering methods of MPE, e.g. modules will be able to pass geometry data directly to the rendering pipes bypassing the re-composition and conversion stages that are used currently.

Springer Style Final Thoughts

Though there has been much work done in creating specific parallel techniques these have in general been applied to isolated methods such as isosurfacing and have been produced as research projects that only benefit a small number of users. This project aims to bring the fruits of these labours to a larger community and to build on them to produce an end user parallel visualization environment and to enhance the extensibility of AVS/Express with a standardized software toolkit.

Contacts

Paul Lever: paul.lever@man.ac.uk
James Perrin: james.perrin@man.ac.uk

http://www.sve.man.ac.uk/Research/PST/