УДК 004.75

СИСТЕМА УДАЛЁННОЙ ВИЗУАЛИЗАЦИИ ДЛЯ ИНЖЕНЕРНЫХ И СУПЕРКОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЙ

М.О. Бахтерев, П.А. Васёв, А.Ю. Казанцев, Д.В. Манаков

SYSTEM OF DISTANT VISUALIZATION FOR ENGINEERING CALCULATIONS AND HPC

M.O. Bakhterev, P.A. Vasev, A.Y. Kazantsev, D.V. Manakov

Рассмотрены вопросы доступа к результатам вычислений. Выделены случаи, в которых выгодно применение удалённой визуализации, то есть такого распределения системы визуализации, когда изображения генерируются на одном компьютере, а просматриваются на другом. Предложена новая система удалённой визуализации, которая задействует преимущества веб-технологий.

Ключевые слова: визуализация, инженерные вычисления, распределённые системы, рендер-серверы, удалённая визуализация

The authors analyses issues of access to the calculation results. They marked the cases in which the usage of the distant visualization is rational, i.e. such distribution of the visualization system when the image data are generated in one computer and can be browsed on the other. The authors offer a new distant visualization system which will enable the web-technology advantages.

Keywords: visualization, engineering calculations, distributed systems, render-servers, distant visualization

1. Введение

Современная вычислительная техника позволяет осуществлять всё более сложные расчёты. Результаты таких вычислений могут представлять собой значительный объём данных. Очевидно, возникает проблема эффективного доступа к этим данным. Одним из решений такой проблемы является технология удалённой визуализации. Это подразумевает такую модификацию системы визуализации, когда изображения создаются на специальном компьютере и передаются по сети на компьютер пользователь. Пользователь может управлять визуализацией, влияя на процесс создания изображений, которые немедленно появляются на его компьютере.

Среди достоинств данного подхода можно отметить следующие. Во-первых, появляется возможность визуализации экспериментальных данных большого объёма без необходимости копирования этих данных на компьютер пользователя. Действительно, рендеринг может быть осуществлён на вычислительных мощностях, имеющих быстрый доступ к данным. При этом пользователю передаются лишь изображения, а не исходные данные. Во-вторых, анализ результатов эксперимента может быть начат без задержек, связанных с передачей большого объёма данных для визуализации. Далее, за счёт централизации программного

обеспечения появляется возможность сокращения лицензионных издержек на задействованное программное обеспечение. Также можно отметить упрощение процессов администрирования и поддержки системы визуализации и вычислительного комплекса в целом.

В Институте Математики и Механики УрО РАН разрабатывается технология поддержки удалённой визуализации. Эта технология позволяет создавать различные системы визуализации для суперкомпьютерных вычислений и инженерных расчётов через Интернет. Прежде чем перейти к описанию созданной технологии, предлагаем рассмотреть вопрос визуализации с точки зрения взаимного размещения данных, системы рендеринга и компьютера пользователя.

2. Расположение данных и системы рендеринга

С технической точки зрения визуализация включает в себя этапы загрузки данных, обработки, преобразования этих данных в визуальные сущности, их рендеринг и отображение на дисплее или другом устройстве [1]. Это относится как к онлайн-визуализации, так и к визуализации после счёта. В связи с наличием этих этапов для архитектуры системы визуализации существенными являются следующие вопросы.

1. Расположение данных – локальное или удалённое. Локальное расположение подразумевает наличие файлов на компьютере пользователя. Однако данные могут располагаться и во-вне – например, когда их невозможно или неэффективно целиком передать на машину пользователя для визуализации.

2. Расположение системы рендеринга – локальное или удалённое. Локальный рендеринг задействует мощности компьютера пользователя. Удалённое расположение системы рендеринга подразумевает её исполнение на отдельном компьютере (или даже на самом вычислителе). Это может быть оправдано в различных случаях, например, когда процесс рендеринга требует вычислительных мощностей, превышающих возможности машины пользователя.

Если рассмотреть комбинации расположения данных и рендеринга, то можно вплотную приблизиться к возможным архитектурным решениям для систем визуализации.

1. Данные и рендеринг – локальные. Этот случай можно считать классическим и самым распространённым. Подразумевается, что время чтения и визуализации данных приемлемо. Плюсом этого подхода является относительная простота реализации системы визуализации.

2. Данные удалённые, рендеринг локальный. Возникает необходимость создания технических средств для передачи данных или их части. Например, это может быть программа на компьютере с результатами вычислений, которая по команде считывает и передает часть данных на машину пользователя. Команды на чтение и передачу инициируются системой рендеринга в зависимости от текущего вида и параметров отображения. При этом локальный рендеринг обеспечивает оперативность взаимодействия с системой – например, поворот трёхмерной сцены с помощью мыши может выглядеть непосредственным и плавным [2].

Важной задачей, которую приходится решать в случае удалённых данных и локального рендеринга, является выбор правил передачи данных. Например, в широком спектре задач при визуализации расчётных сеток успешным является подход, когда данные огрубляются (с сохранением когнитивности восприятия) на удалённой машине, что позволяет передать их по сети и отобразить на машине пользователя [3].

3. Данные и рендеринг – удалённые. В этом случае изображения создаются на специальном компьютере и передаются по сети на компьютер пользователя. Пользователь может управлять визуализацией, влияя на процесс создания изображений, которые немед-

ленно появляются на его компьютере. Актуальной является ширина канала между ресурсом, содержащим данные, и системой рендеринга. Если этот канал обладает достаточной пропускной способностью, то система передачи данных может оказаться достаточно простой. Еще одним плюсом такого подхода является то, что система рендеринга может быть распределённой. Отметим также, что требования к ширине канала между системой рендеринга и машиной пользователя в общем случае являются фиксированной величиной, определяемой интенсивностью графического обмена для данной задачи.

Можно сказать, что каждый из перечисленных вариантов является предпочтительным в зависимости от ситуации. Конкретный выбор той или иной конфигурации определяется решаемой задачей и имеющимся системным окружением. В разработанной нами системе подразумевается использование третьего варианта, когда и результаты экспериментального моделирования, и программы рендеринга расположены на удалённых мощностях.

3. Классификация систем удалённой визуализации

В работах западных коллег [4] встречается следующая классификация систем удалённой визуализации, которая представляет на наш взгляд особый интерес.

1. Рендеринг на компьютере пользователя. В этом варианте программа визуализации исполняется на сервере, однако рендеринг происходит на машине пользователя. Ярким примером этого класса являются системы X-Windows и Chromium. Первая позволяет запускать программы на удалённом компьютере, но при этом они отображаются и ведут себя как локальные. Вторая система – Chromium – позволяет направлять поток OpenGL-команд от сервера к клиенту. Системы этого класса, помимо своих преимуществ, обладают также и недостатком – они не задействуют серверные мощности для выполнения рендеринга, оставляя эту задачу машине пользователя.

2. Рендеринг на стороне сервера для двумерных и административных задач. Наиболее распространённые представители этого класса – Microsoft Terminal Services, Citrix MetaFrame, а также технология VNC. Эти продукты используются, как уже отмечено, для выполнения задач по администрированию, а также для удалённого доступа к офисным и другим двумерным приложениям. Однако с трёхмерной графикой данные системы справляются слабо.

3. Рендеринг на стороне сервера для трехмерных приложений. В эту категорию попадают все системы, поддерживающие распределение «данные и рендеринг – удалённые», которое описано в предыдущем параграфе. Далее будет рассмотрен целый ряд таких систем.

Как уже отмечалось, в предлагаемой нами системе мы придерживаемся подхода, когда рендеринг осуществляется на стороне сервера. Обратимся теперь к краткому обзору имеющихся систем удалённой визуализации, уделив должное внимание техническим нюансам их применения.

4. Обзор существующих решений

Нами исследованы возможности систем Windows Terminal Services 2008, Citrix XenApp, решений VNC и XWindows, а также ряда менее известных систем. Анализ показывает, что ведущие разработчики этих решений только приступают к поддержке программ с аппаратной трёхмерной графикой. Так, в пресс-релизах компаний Microsoft и Citrix 2008 года заявлено, что следующие версии их продуктов будут непосредственно поддерживать приложения, использующие DirectX и OpenGL. Безусловно, данные заявления вселяют определённую уверенность и желание применить эти широко используемые технологии, однако в настоящий момент единственно возможный путь – рассмотрение альтернативных решений.

Существуют решения различных уровней. Например, на уровне библиотек и сервисов выделяются VNC и VirtualGL. А среди флагманов готовых продуктов можно отметить Paraview Enterprise Edition, Sun Visualization System, HP Scalable Visualization Array, Mental Images Reality Server и StreamMyGame. Подчеркнём, что большинство из этих решений ориентированы на пользовательский интерфейс, базирующийся на веб-технологиях. Среди представленных систем отдельно хотелось бы выделить следующие.

Среда VirtualGL расширяет функциональность систем на базе технологий X-Windows, задействуя аппаратный рендеринг на стороне сервера. В X-Windows все команды OpenGL направляются по сети для исполнения на компьютере пользователя. После включения среды VirtualGL рендеринг осуществляется сразу на сервере, и по сети передаются готовые изображения. Плюсом этой среды, очевидно, является прозрачная интеграция с имеющимися приложениями. Минус – привязка к X-Windows.

Paraview Enterprise Edition – содержит веб-интерфейс к системе визуализации Paraview для организации удалённой визуализации. Paraview является распространенным инструментом для визуализации результатов различных расчётов. Среди плюсов программы – богатый набор видов отображения и различные варианты конфигурации распределённого рендеринга. Минус – необходимость конвертации данных в формат Paraview.

Программный комплекс RealityServer является платформой для создания 3D-веб-приложений. Система содержит инфраструктуру для рендеринга трёхмерных сцен, передачи изображений в веб-интерфейс, и приём команд от пользователя веб-интерфейса с последующей обработкой их на сервере. Существует возможность как настройки веб-интерфейса, так и программирования серверной логики. Среди плюсов программы можно отметить поддержку аппаратного рендеринга. Среди минусов – закрытость разработки. Кроме того, отдельной проблемой является необходимость конвертации данных в формат сцен «.mi», применяющихся в RealityServer.

По результатам исследования существующих решений нами было принято решение о создании новой системы удалённой визуализации. Это позволит в будущем глубже понять проблематику этой технической области, а также производить достаточно гибкие изменения, которые потребует практика.

5. Новая система удалённой визуализации

Система состоит из трёх уровней: (1) модулей визуализации, (2) сервиса-посредника и (3) интерфейсов пользователя. Каждый модуль визуализации (программа рендеринга) – это один или несколько процессов, ответственных за загрузку и графическое представление экспериментальных данных определённого типа. Сервис-посредник – это серверный процесс, он принимает команды от интерфейса пользователя и направляет их к программам рендеринга. Интерфейсы пользователя – в настоящее время построенные на веб-технологиях – отображают построенные графические представления данных и взаимодействуют с пользователем. На рис. 1 показана общая схема работы системы.

Итак, пользователь взаимодействует с системой через веб-интерфейс, в котором представлены элементы управления модулем визуализации. В основе этого интерфейса – сгенерированное визуальное представление данных. Пользователь может оказывать воздействие на изображение, а также на элементы управления. При воздействии веб-интерфейс направляет соответствующие команды сервису-посреднику. Посредник, в свою очередь, транслирует команды конкретному процессу модуля визуализации. Это могут быть запросы на изменение тех или иных параметров либо на перерисовку изображения. В случае запроса на перерисовку в модуле визуализации вызывается специальная функция рендеринга. После отработки этой функции изображение сжимается в определённый формат и передаётся через сервис-посредник в браузер пользователя.

Рис. 1. Общая схема системы онлайн-визуализации

В настоящий момент в рамках предлагаемой технологии для каждой системы визуализации подразумевается «ручное» создание собственного веб-интерфейса пользователя. Задача разработки универсальных механизмов создания таких графических интерфейсов – это особый вопрос, который заслуживает внимания отдельной исследовательской работы.

Модули визуализации создаются и работают по следующей схеме. При запуске модуль регистрируется в системе, сообщая свое сетевое расположение. Также модуль сообщает перечень параметров, которые влияют на визуализацию и которые можно изменять через интерфейс пользователя. Далее модуль регистрирует функцию рендеринга и переходит в режим ожидания.

Как отмечено выше, когда возникает необходимость сгенерировать изображение, система вызывает зарегистрированную функцию рендеринга. Если пользователь инициирует изменение параметров, то происходит обновление значений соответствующих переменных в оперативной памяти модуля визуализации.

На рис. 2 показан пример исходного кода модуля визуализации. В данном случае модуль имеет два параметра визуализации – углы поворота сцены `alfa` и `beta`. Эти параметры публикуются в системе с помощью вызовов `iv_publish`. После публикации система может читать и записывать значения параметров в память программы. Доступ к памяти осуществляется асинхронно, в параллельном потоке.

В этом примере функция рендеринга создана простейшим образом – она передает в графическую сцену указанные параметры и вызывает захват изображения с окна приложения. Эта функция также исполняется в потоке, отличном от главного потока приложения. Гарантируется, что во время работы данной функции параметры визуализации остаются неизменными.

После срабатывания функции рендеринга изображение сжимается и передаётся сервису-посреднику. Правилам сжатия изображений требуется уделить особое внимание. На текущий момент они упаковываются в формат PNG по следующей схеме. При первом запросе на кадр система высылает изображение, уменьшенное в 4 раза. При поступлении в браузер это изображение растягивается, и пользователь видит слегка размытую картину. В случае, если пользователь не проявляет какой-либо активности в течение нескольких секунд, посылается улучшенная копия изображения с коэффициентом уменьшения 2. Если в течение последующих секунд пользователь продолжает «молчать», то передаётся и показывается

```
double alfa=0,beta=0; // параметры визуализации

// функция рендеринга. Вызывается по требованию интерфейса.
void perform_render( char *name, int w, int h, fipWinImage*
theImage ) {
      scene.rot.z = -alfa;
      scene.Yaw( 0 );
      scene.Slide( -beta*100,0,0 );
      theImage->captureWindow( engine.render->hWnd );
}

...

iv_init( «gryffin»,0 );           // регистрация в системе
iv_publish(«alfa»,&alfa,1,0 ); // публикация параметра alfa
iv_publish( «beta»,&beta,1,0 ); // публикация параметра beta

// регистрация функции рендеринга с именем gryffinview
iv_remoteWindow( «gryffinview»,perform_render );

...\vspace{-0.5cm}
```

Рис. 2. Пример подключения модуля визуализации

исходное изображение. Этот подход позволяет уменьшить объём данных, передаваемых по сети. Действительно, если пользователь с помощью мыши вызывает интерактивное вращение трехмерной сцены, то с точки зрения восприятия важно показать результат быстро, пусть и огрублённым. Для одного конкретного приложения, которое будет приведено ниже, размер изображения PNG с разрешением 640x480 в среднем составляет 60 Кб, а уменьшенного в 4 раза – 5 Кб. Соответственно, при поворотах сцены передаётся 10 – 50 изображений суммарным объемом всего в 50 – 250 Кб. Это позволяет сократить время отклика системы.

Хотелось бы объяснить, почему используется формат PNG, а не JPEG. По наблюдениям авторов, использование формата PNG более предпочтительно из-за применения компрессии без потерь. При необходимости система проводит дополнительную компрессию, сжимая изображение по осям. При этом сжатое, а потом растянутое изображение визуально выглядит более приемлемо при динамических изменениях (например, во время вращения сцены), чем в случае сжатия с потерей качества JPEG. Последний формат в динамике воспринимается неаккуратным, хоть и предоставляет лучший коэффициент сжатия. Тем не менее, в мировой практике JPEG довольно часто применяется в работе систем удалённой визуализации. Более того, разрабатываются подсистемы компрессии с применением технологий CUDA [4]. Мы также надеемся со временем исследовать этот вопрос, что в результате может привести к еще большему сокращению времени отклика системы.

Что касается сервиса-посредника, то он выполняет три функции. Во-первых, посредник ведёт учёт активных модулей визуализации. Он хранит их сетевые адреса, а также список имен параметров визуализации для каждого модуля. Это и позволяет посреднику определять, куда именно необходимо маршрутизировать запросы от пользовательских интерфейсов. Прием таких запросов, их обработка и маршрутизация является второй функцией посредника. Наконец, посредник является веб-сервером, обслуживая файлы пользовательского веб-интерфейса.

Компоненты всех трёх уровней системы могут располагаться на разных компьютерах.

Так, например, интерфейс пользователя выполняется в браузере на компьютере пользователя, сервис-посредник может быть размещен на специальном сервере, а модуль визуализации может располагаться на вычислительном кластере. Взаимодействие посредника и модулей визуализации осуществляется с помощью протокола SOAP. Пользовательские интерфейсы взаимодействуют с посредником по протоколу HTPP, AJAX и, при необходимости, SOAP.

В завершение этого параграфа хотелось бы отметить, что в рамках разрабатываемой нами системы модуль визуализации не привязан к конечному типу оборудования – это может быть отдельное приложение на сервере визуализации или параллельное приложение, работающее на вычислителе. Более того, функции модуля визуализации может выполнять часть узлов вычислительной программы. Таким образом, система позволяет варьировать архитектуру для модуля визуализации и может применяться как для оффлайн-, так и для онлайн-визуализации.

6. Пример использования системы

На представленной иллюстрации (рис. 3) показан один из шагов расчёта процесса определённого воздействия на сечение трубы нефтепровода [5], проведённого в инженерном пакете DEFORM. Пользователь может нажать кнопку мыши на изображении и перемещать её, при этом изображение будет немедленно обновляться, показывая вращение объекта. Рендеринг осуществляется на удалённом сервере визуализации аппаратно специальной программой с применением технологии DirectX. В локальной сети 10 Мбит удалось достичь скорости работы 15 кадров в секунду. В эксперименте с визуализацией через Интернет по каналу Екатеринбург – Москва – Челябинск система показала скорость работы около 3 кадров в секунду.

7. Заключение

Система показала работоспособность и расширяемость; может работать под управлением Windows, Linux и переносима на другие ОС. Веб-интерфейс системы является независи-

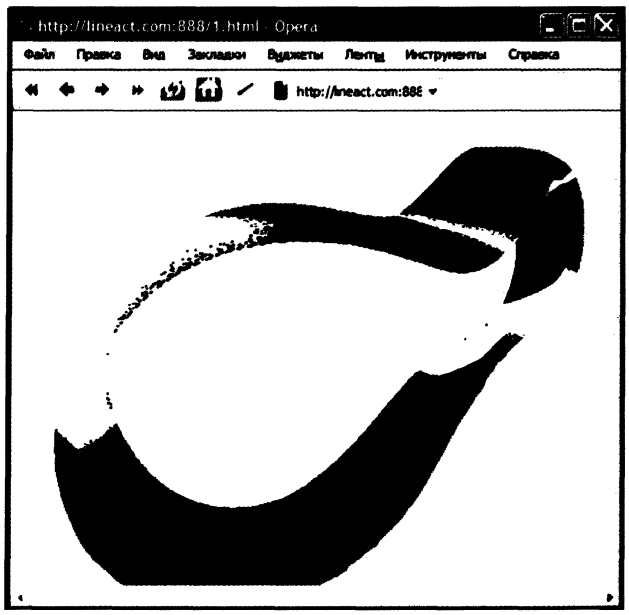

Рис. 3. Пример использования системы

мым от браузера. Технологии написания модулей визуализации могут быть самыми разнообразными. Теоретически, это любые программы, способные работать с сетевым протоколом SOAP. Модули визуализации могут исполняться в разных средах – будь то выделенные сервера визуализации или узлы вычислителя.

Созданная реализация системы показала достаточно высокое удобство использования и восприятия – до 15 кадров в секунду в локальной сети, что соизмеримо по скорости с визуализацией на компьютере пользователя.

Разработан модуль для визуализации определённого класса расчётных сеток. При этом необходимо подчеркнуть, что в общем случае в системе подразумевается создание индивидуального модуля визуализации для каждой конкретной задачи. Перспективным направлением, возможно, является создание набора универсальных модулей визуализации для определённых классов задач. Основу такого набора может составить какая-либо существующая система визуализации для научных исследований.

В заключении авторы хотели бы выразить искреннюю благодарность и признательность В.Л. Авербуху за научное руководство; С.В. Шарфу и М.В. Якобовскому за конструктивные обсуждения проблематики, а также уважаемым коллегам из Сарова (РФЯЦ-ВНИИЭФ) и Челябинска (ЮУрГУ) за поддержку и плодотворный обмен идеями.

Литература

1. Подходы к реализации средств on-line визуализации параллельных вычислений / В.Л. Авербух, Д.В. Манаков, П.А. Васёв и др. // Супервычисления и математическое моделирование: тезисы междунар. семинара, г. Саров, ВНИИЭФ-РФЯЦ. – Саров, 2003. – С. 14 – 16.
2. ScientificView – система параллельной постобработки результатов, полученных при численном моделировании физических процессов / А.Л. Потехин, В.И. Тарасов, С.А. Фирсов и др. // Труды XVIII Международной конференции по компьютерной графике и визуализации Graphicon, 2008. – М., 2008. – С. 65 – 69.
3. Якобовский, М.В. Решение сеточных задач на распределенных системах / М.В. Якобовский // Параллельные вычислительные технологии: тр. науч. конф. – Челябинск, 2007. – Т. 2. – С. 201 – 211.
4. Lietsch, S., A CUDA-Supported Approach to Remote Rendering / S. Lietsch, O. Marquardt // Proceedings of the International Symposium on Visual Computing. – Lake Tahoe, Nevada, United States, Springer. – 2007. – P. 724 – 733.
5. Технология создания виртуальных испытательных стендов в грид-средах / Г.И. Радченко, В.А. Дорохов, Р.С. Насибулина и др. // Вторая Международная научная конференция «Суперкомпьютерные системы и их применение» (SSA'2008): докл. конф. (27 – 29 октября 2008 года, Минск). – Минск, 2008. – С. 194 – 198.

Институт математики и механики Уральского отделения РАН,
Уральский государственный университет
mike.bakhterev@gmail.com, pavel.vasev@gmail.com, ajk@gmail.com, manakov@imm.uran.ru

Поступила в редакцию 6 февраля 2009 г.