A Distributed Computational Steering Environment for E-Learning Applications

Presented by:

Mostafa Herajy
SCU, Egypt, DAAD Scholarship Holder for Ph.D study in DSSZ Group of

Prof. Monika Heiner
Chair of Data Structures and Software Dependability BTU - Cottbus

Supervised by:

Prof. Essam Atta
Ain Shams Univ., Egypt
Agenda

• Introduction.
• Motivations.
• Objectives.
• STEEL (Steering Environment for Electronic Learning).
• Applications.
• Conclusions and future work.
Introduction

- Computational Steering:
  => the tight coupling of visualization and simulation.

  => Online and on the fly visualization.

  => Remote control of long running simulation.

  => Interactive simulation technique.
Haber and McNabb Visualization Reference Model
Introduction

Steering

Communication and Data Transfer

Data Source

User interface

Visualization

Computational steering model

Motivations

Objectives

STEEL

Applications

Conclusions
Computational Steering Tasks

- *Model exploration.*
- *Algorithm experimentation.*
- Performance optimization.
Model exploration

The user is primarily interested in the application’s input and output data. The main intention is to explore parameter spaces and simulation behavior to gain additional insight in the simulation.
Algorithm experimentation

Informs the user about the application’s program structure
Performance optimization

Is used to provide information about the application’s configuration and progress
Introduction

Computational Steering Approaches

- Program instrumentation.
- Direct Scientific Computation.
- Recasting Scientific Computation.
Introduction

Computational Steering Environments

- CSE
- DISCOVER
- POSSE
- RealityGrid
- SCIRun
- Progress and Magellan
- VASE
- Pablo
Motivations

- The ongoing efforts in modernizing learning.
- The need to change the learning from the traditional "schoolhouse" model into the networked virtual classroom.
• The need of using interactive techniques in electronic learning.
Objectives

- Designing collaborative and distributed computational steering environment for electronic learning.

- Developing a framework for integration of existing application into the developed environment.

- Applying the developed environment in learning abstract scientific concepts.
STEEL

STeering Environment for Electronic Learning.
STEEL Architecture

- GUI Editor.
- Steering and Monitoring Server.
- Runtime Utility.
- Steering API Library.
STEEL Architecture (Cont.)

**STEEL Steering and Monitoring Server**

- **Server Status**: Running
- **Number of Clients Connected**: 2
- **Number of Problems Running**: 1

Server log:

```
The Server started Successfully.
```

<table>
<thead>
<tr>
<th>Var. Name</th>
<th>Type</th>
<th>Data Type</th>
<th>Access</th>
<th>Var. Val</th>
</tr>
</thead>
<tbody>
<tr>
<td>AirO</td>
<td>STATIC...</td>
<td>Float</td>
<td>W</td>
<td>dims = 2</td>
</tr>
<tr>
<td>MINF</td>
<td>SIMPLE...</td>
<td>Float</td>
<td>R</td>
<td>0.000001</td>
</tr>
<tr>
<td>ALPHA</td>
<td>SIMPLE...</td>
<td>Float</td>
<td>R</td>
<td>0.000002</td>
</tr>
<tr>
<td>OMEGA</td>
<td>SIMPLE...</td>
<td>Float</td>
<td>R</td>
<td>dims = 3</td>
</tr>
<tr>
<td>RRHO</td>
<td>STATIC...</td>
<td>Float</td>
<td>W</td>
<td>dims = 4</td>
</tr>
<tr>
<td>GRD</td>
<td>STATIC...</td>
<td>Float</td>
<td>R/W</td>
<td>dims = 5</td>
</tr>
</tbody>
</table>
STEEL Architecture (Cont.)

- A library of API Calls.
- Injected into the original application code.
- Responsible for communicating the input and output of the application data.
- Can be called from Fortran, C, or C++.
STEEL Users

- Developer: develops the application using the GUI design and STEEL API.
- End users (students): use the developed application using the GUI runtime utility.
Visualization Using STEEL

- Introduction
- Motivations
- Objectives
- STEEL
- Applications
- Conclusions

STEEL Data Visualizer

Visualization Toolkit

Pipelines

Graphics Library
Advantages of The Developed Framework

- Can be used to implement distributed and collaborative learning environments.
- Can integrate existing codes easily.
- Can be used in complex simulation through its support for parallelism.
- Allow the rollback to previous stages
Applications

- The Interactive solution of the traveling salesman problem.
- Numerical flow simulation and visualization past an airfoil at transonic speeds.
- Image Analogies.
The traveling salesman problem using simulated annealing Algorithm

Random Number of Points Representing 100 Cities
The traveling salesman problem using simulated annealing Algorithm
The traveling salesman problem using simulated annealing Algorithm

Intermediate Path
The traveling salesman problem using simulated annealing Algorithm
The traveling salesman problem using simulated annealing Algorithm

Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

Movie Demonstrating STEEL While Working
The traveling salesman problem using simulated annealing Algorithm

Movie produced using STEEL
Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds
Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds

Movie Demonstrating STEEL While Working
Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds

Movie produced using STEEL
Image Analogies – Image Colorization

http://mrl.nyu.edu/projects/image-analogies/
Image Analogies - Texture by Numbers

Introduction

Motivations

Objectives

STEEL

Applications

Conclusions
Conclusions and Future Work

- We developed a framework for computational steering environments that can be used in electronic learning.
- We implemented this framework and the result is STEEL.
Conclusions and Future Work

- STEEL includes collaborative distributed interactive simulation, computational steering, and interactive visualization.
- Future extension of STEEL will focus on the addition of animation capabilities in virtual education environment.

Thank You
## Comparison with other systems

<table>
<thead>
<tr>
<th>Environment</th>
<th>Distributed</th>
<th>Allow Parallelism</th>
<th>Model Exploration</th>
<th>Performance Optimization</th>
<th>User Interface</th>
<th>Fault Tolerance</th>
<th>Backtracking</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSE</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
<td>Visualization and steering through PGO</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUMULVS</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>Visualization through AVS, textual steering</td>
<td>Fault tolerance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCIRun</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>Steering through tcl/tk, visualization module</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POSSE</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>Use existing package for visualization</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DISCOVER</td>
<td>YES</td>
<td>NC</td>
<td>YES</td>
<td>YES</td>
<td>Web based visualization and steering</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VASE</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>Visualization through existing packages, steering through textual inputs</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RealityGrid</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>Backtracking</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Progress &amp; Magellan</td>
<td>YES</td>
<td>NC</td>
<td>YES</td>
<td>NO</td>
<td>Visualization through existing packages, steering through command line and GUI</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEEL</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>Steering and visualization through dedicated client</td>
<td>Backtracking</td>
<td>E-learning</td>
<td></td>
</tr>
</tbody>
</table>
Total runtime as a function of the number of processors
Speedup as a function of the number of processors

![Graph showing speedup as a function of the number of processors. The x-axis represents the number of processors, ranging from 1 to 4. The y-axis represents the speedup, ranging from 0 to 4. There are multiple lines indicating speedup for different values of N: N = 100, N = 2700, N = 6400, N = 8000, and an ideal speedup line. Each line is labeled with the corresponding N value. The graph illustrates how speedup increases with the number of processors.]
Efficiency as a function of the number of processors
The performance analysis with fixed problem size