

The Computational Steering Framework Steereo

Dipl.-Inf. Domenic Jenz, Dr.-Ing. Martin Bernreuther
High Performance Computing Center Stuttgart (HLRS)
June 8th 2010



Outline

Introduction

- Before Computational Steering
- Approaches
- Classification

Steereo - A steering framework

- Overview
- Parallel Steering

Batch Processing

The traditional approach is batch processing:

- ▶ Write simulation parameters into a config file or similar
- ▶ start the simulation
- ▶ wait and pray
- ▶ post process output
- ▶ visualize and maybe understand the results
- ▶ change parameters in config file and start all over again

A citation

Computer Graphics, Nov. 1987 von McCormick, DeFanti, Brown:

"Scientists not only want to analyze data that result from their supercomputations; they also want to interpret what is happening to the data during super-computations. Researchers want to steer calculations in close to realtime; they want to be able to change parameters, resolution or presentation, and see the effects. They want to drive the scientific discovery process; they want to interact with their data"

Approaches

There are two simple and one more complex possibility for Computational Steering:

- ▶ The simulation regularly writes data into a file, which can be read from a client
- ▶ The simulation can react to signals from the operating system, which can be specifically triggered by the client (e.g. with `kill -s signalNumber`)
- ▶ The client connects to the simulation (e.g. using sockets), which sends the needed data to it.

Classification I

Vetter and Schwan identified two types of computational steering in 1996 :

- ▶ human-interactive steering:
a person monitors and manipulates parameters of the computation while it is running
- ▶ algorithmic steering:
the computer makes decisions by monitoring runtime statistics and other information sources such as history files

Classification II

Another popular classification (e.g. by Steven Parker[SCIRun], Jurriaan Mulder[CSE]):

- ▶ application steering:
the computational process can be modified through parameter changes, mesh modifications or other changes
- ▶ algorithm refinement:
the underlying code can be modified or refined at runtime
- ▶ performance steering:
computational resources that affect the simulation performance such as load balancing, I/O or cache strategies

Steereo overview

- ▶ Is developed at the HLRS for the German projects SFB 716 and SKALB
- ▶ Is a (lightweight) steering FRAMEWORK, not a complete steering environment !
- ▶ main goal: simplify integrating and enabling steering capabilities to obtain a "reactive" numerical simulation server $\Rightarrow$ "Qt / wxWidgets/ ... for computational steering"
- ▶ support for steering clients
- ▶ Should be usable for any kind of computational steering
- ▶ Written in C++ with a Binding for C / Fortran 90+ simulations.
- ▶ Available at <http://steereo.hlrs.de>

Steereo connectivity

- ▶ add "perception" to the simulation server:
 - ▶ communication through POSIX sockets
 - ▶ handling signals
 - ▶ possibility to extend communication features (e.g. MPI)
- ▶ support of distributed, parallel systems
- ▶ simulation servers might accept connections of multiple clients, as well as clients might connect to multiple servers
- ▶ possibility to connect, disconnect and reconnect any time
- ▶ data transfer is based on byte array stream to store arbitrary data; with compression capabilities included

Steereo commands

By writing command classes, the steering capabilities of the simulation are defined

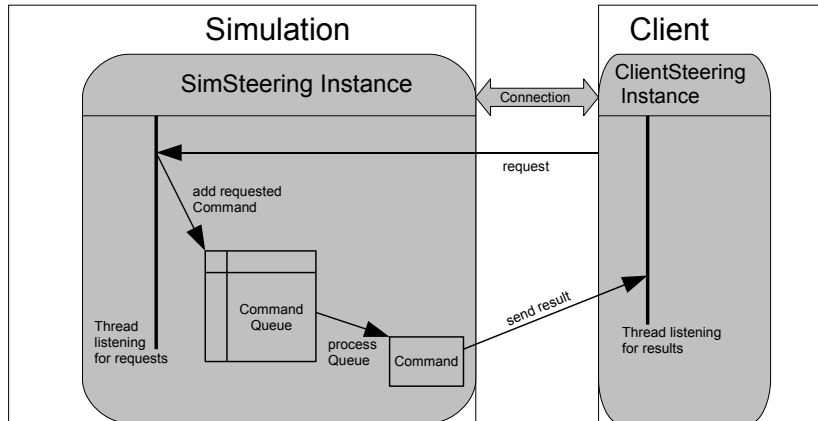
- ▶ on the simulation side commands
 - ▶ can be statically linked or sourced out to a dynamic library
 - ▶ are parametrized and can be executed conditionally
 - ▶ can be automatically (periodically) re-executed
 - ▶ may optionally implement undo-functionality
- ▶ there are predefined commands to query registered commands, to steer parameters...
- ▶ command execution is triggered by the simulation

Steering client

The steering client

- ▶ can be a standalone program or a module for an existing program (e.g. visualization tools)
- ▶ The steering client steers the simulation by
 - ▶ sending requests to the simulation.
Requests consist of the commando name on the simulation side and eventual parameters
 - ▶ receiving the results (if there are any) of the requests and processing them further, which may lead to new requests to the simulation.

Steereo workflow



Parameter Steering

- ▶ Parameter Steering is supported by a predefined command
- ▶ The variables or arrays, which should be steerable, just have to be registered at `SteerParameterCommand`
- ▶ It is also possible to register getter and setter methods.
- ▶ For the client there are also functions available which facilitate the requesting and setting of parameters

Parameter Steering example

The registration of the steered variable on the simulation side:

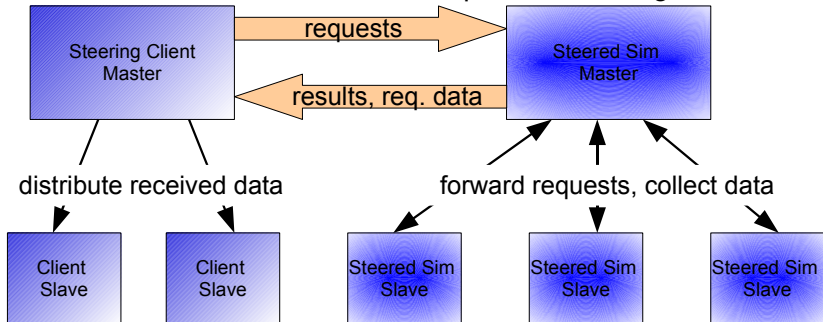
```
1 SteerParameterCommand::registerScalarParameter ("temp",
2   _domain, &Domain::getGlobalTemperature,
3   &Domain::setGlobalTemperature);
```

setting can be done this way on the client side:

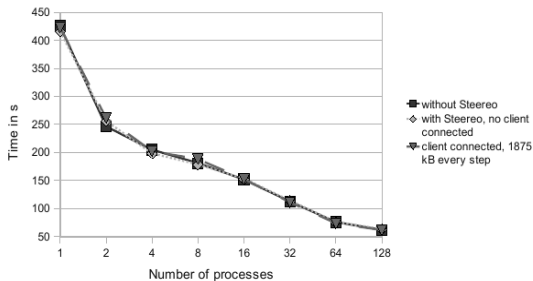
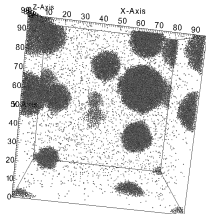
```
1 SteerParameterClient* spc = new SteerParameterClient();
2 spc->registerScalarParameter("temp", &myTemp);
3 spc->requestSetParameter ("temp", 0);
4 ...
5 // wait until the parameter is truly set
6 while (!spc->isParameterUpdated("temp"))
7 { }
```

Parallel Steering I

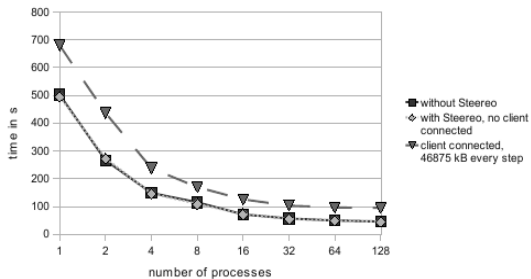
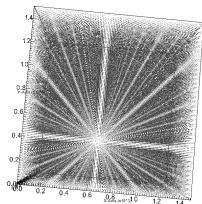
This is the conventional method for parallel steering:



Measurement I - 40000 Molecules



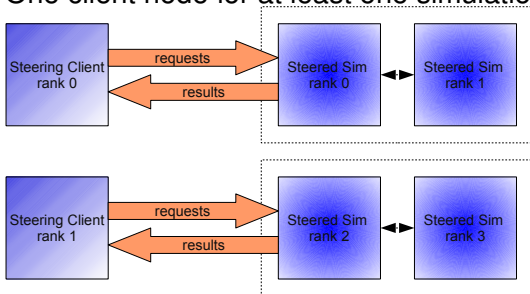
Measurement II - 1000000 Molecules



Parallel Steering II

Desirable is the following configuration:

- ▶ The simulation as well as the client run on multiple processors
- ▶ One client node for at least one simulation node



Acknowledgments and Conclusion

This work is part of

- ▶ the SFB 716 at Universität Stuttgart, funded by DFG,
- ▶ SKALB, funded by the BMBF

Conclusions:

- ▶ Steering functionality can be enhanced by writing own commands and communicators.
- ▶ For most cases the predefined SteerParameterCommand will suffice.
- ▶ Writing of client applications/modules is also supported
- ▶ Without connected steering client, the runtime of the simulation doesn't change noticeable.
- ▶ But with big parallel simulation runs a delay is quite recognizable.