

# 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



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- Computational Steering :
  - => the tight coupling of visualization and simulation.
  - => Online and on the fly visualization .
  - => Remote control of long running simulation.
  - => Interactive simulation technique.



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

Data

Filtering

Mapping

Rendering

Final  
image

Haber and McNabb Visualization Reference  
Model



# Introduction



Introduction

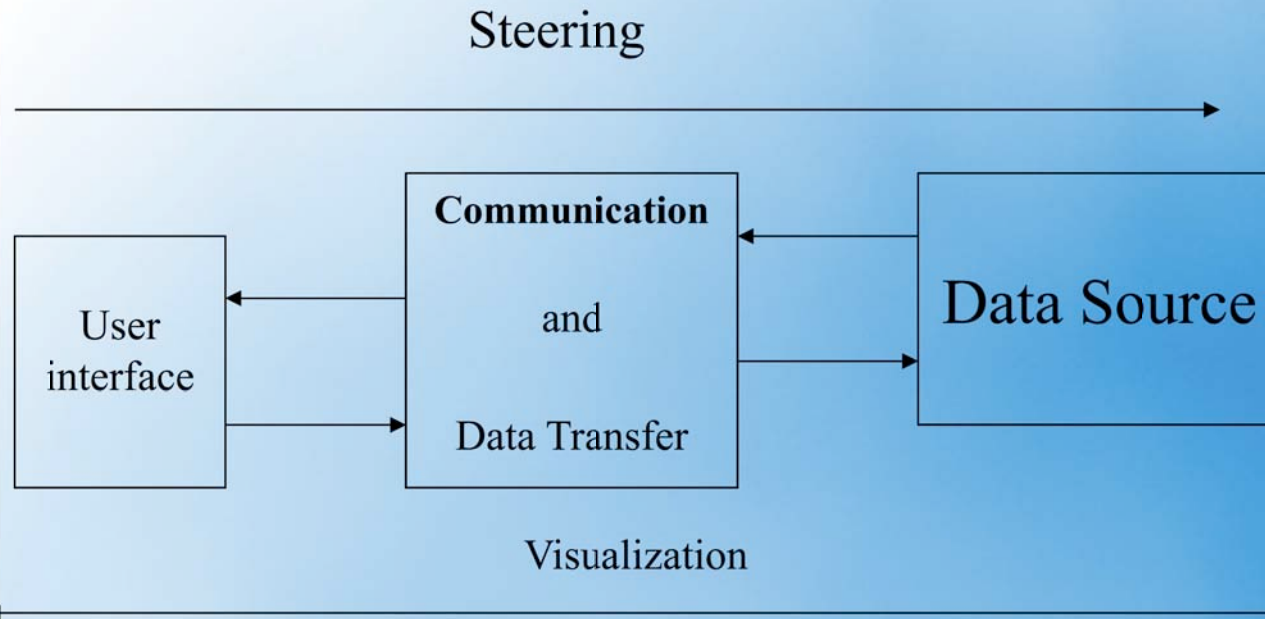
Motivations

Objectives

STEEL

Applications

Conclusions



Computational steering model



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Computational Steering Tasks

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



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Model exploration

The user is primary 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



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Algorithm experimentation

Informs the user about the  
application's program structure





# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Performance optimization

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



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Computational Steering Approaches

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



# Introduction



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

## Computational Steering Environments

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



# Motivations



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- The ongoing efforts in modernizing learning .
- The need to change the learning from the traditional "schoolhouse" model into the networked virtual classroom.



# Motivations (Cont.)



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- The need of using interactive techniques in electronic learning.



# Objectives



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- Designing collaborative and distributed computational steering environment for electronic learning.
- Developing a frame work for integration of existing application in the developed environment.
- Applying the developed environment in learning abstract scientific concepts.



# STEEL



Introduction

Motivations

Objectives

STEEL

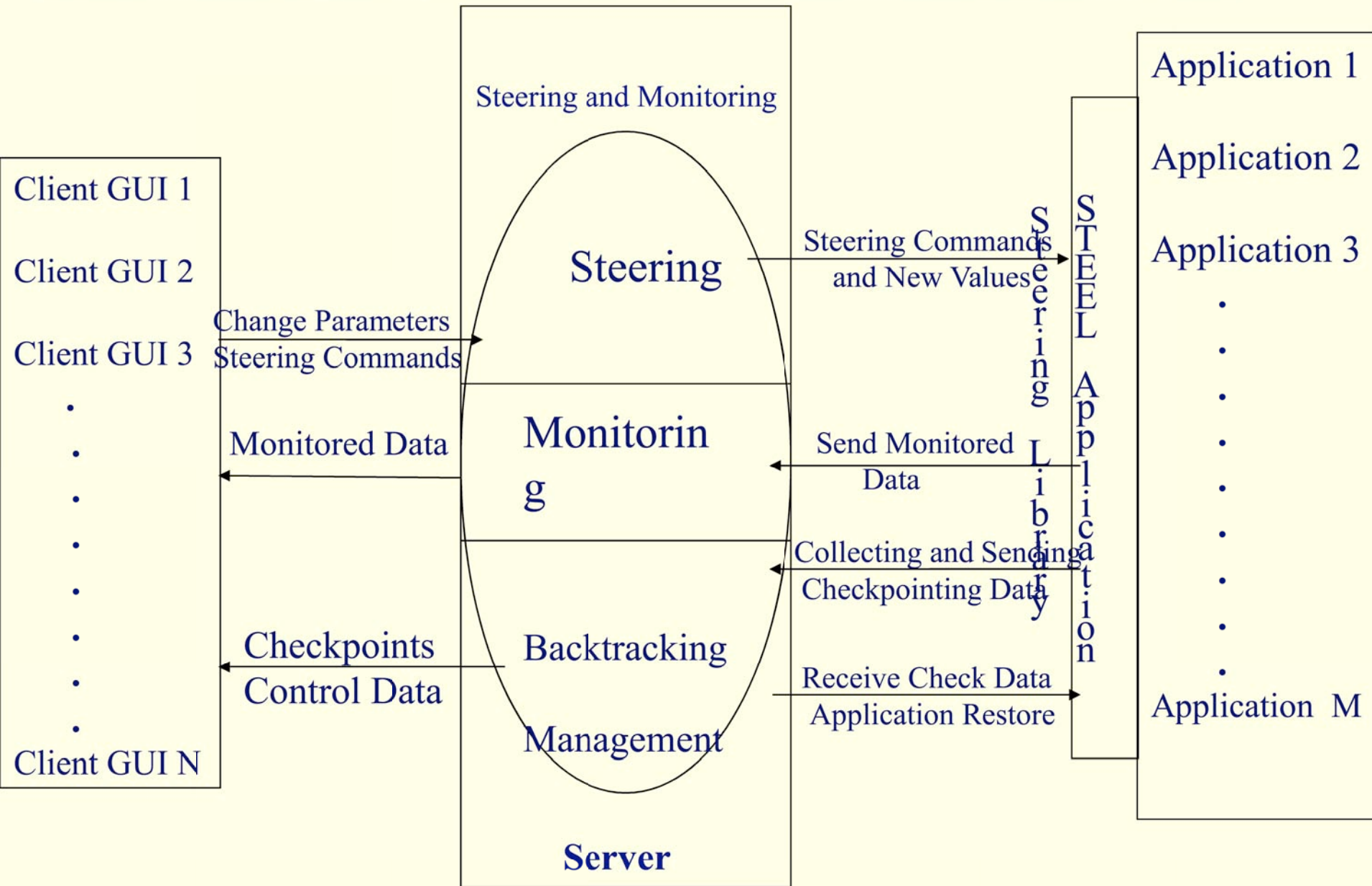
Applications

Conclusions

## STeering Environment for Electronic Learning.



# STEEL Architecture





# STEEL Architecture



Introduction

Motivations

Objectives

**STEEL**

Applications

Conclusions

- GUI Editor.
- Steering and Monitoring Server.
- Runtime Utility.
- Steering API Library.



# STEEL Architecture (Cont.)



Introduction

Motivations

Objectives

**STEEL**

Applications

Conclusions

SteeringGui - simann.std

File Edit View Objects Variables Run Help

Travelling sales Man Problem with simulated annealing technique

abl Length

Change

Current T

Successful Movies

SteeringGUI compilation output

Not Compiled

Ready NUM

**Check Pointing Keys Defination**

KeyData

Key\_label

Data Type

Variable Type

Single Variable

Array of Variables

Array Dimensions

Dims: 1 2 3 4

OK Update << >> Delete Add

**Properties**

property	value
Type	BUTTON
ID	16
Caption	&Stop
Action	Stop(App.)

**Variables Information**

Var. Name	Type	Access
X	Float	W
PATH	Float	W
NT	Float	R
CT	Float	W
SUCC	Integer	W
I	Integer	CHKPoi.
J	Integer	CHKPoi.
K	Integer	CHKPoi.
T	Float	CHKPoi.
NN	Integer	CHKPoi.
DE	Float	CHKPoi.

STEEL GUI Editor

# STEEL Architecture (Cont.)



Introduction

Motivations

Objectives

**STEEL**

Applications

Conclusions

Server Status: Running

Number of Clients Connected: 2

Number of problems Running: 1

Server log: The Server started Successfully

Var. Name	Type	Data Type	Access	Var. Val.
AirO	STATIC...	Float	W	Dims. = ,
MINF	SIMPLE...	Float	R	0.000000
ALPHA	SIMPLE...	Float	R	0.000000
OMEGA	SIMPLE...	Float	R	0.000000
RRHO	STATIC...	Float	W	Dims. = ,
GRD	STATIC...	Float	R/W	Dims. = ,

STEEL Steering and Monitoring Server



# STEEL Architecture (Cont.)



Introduction

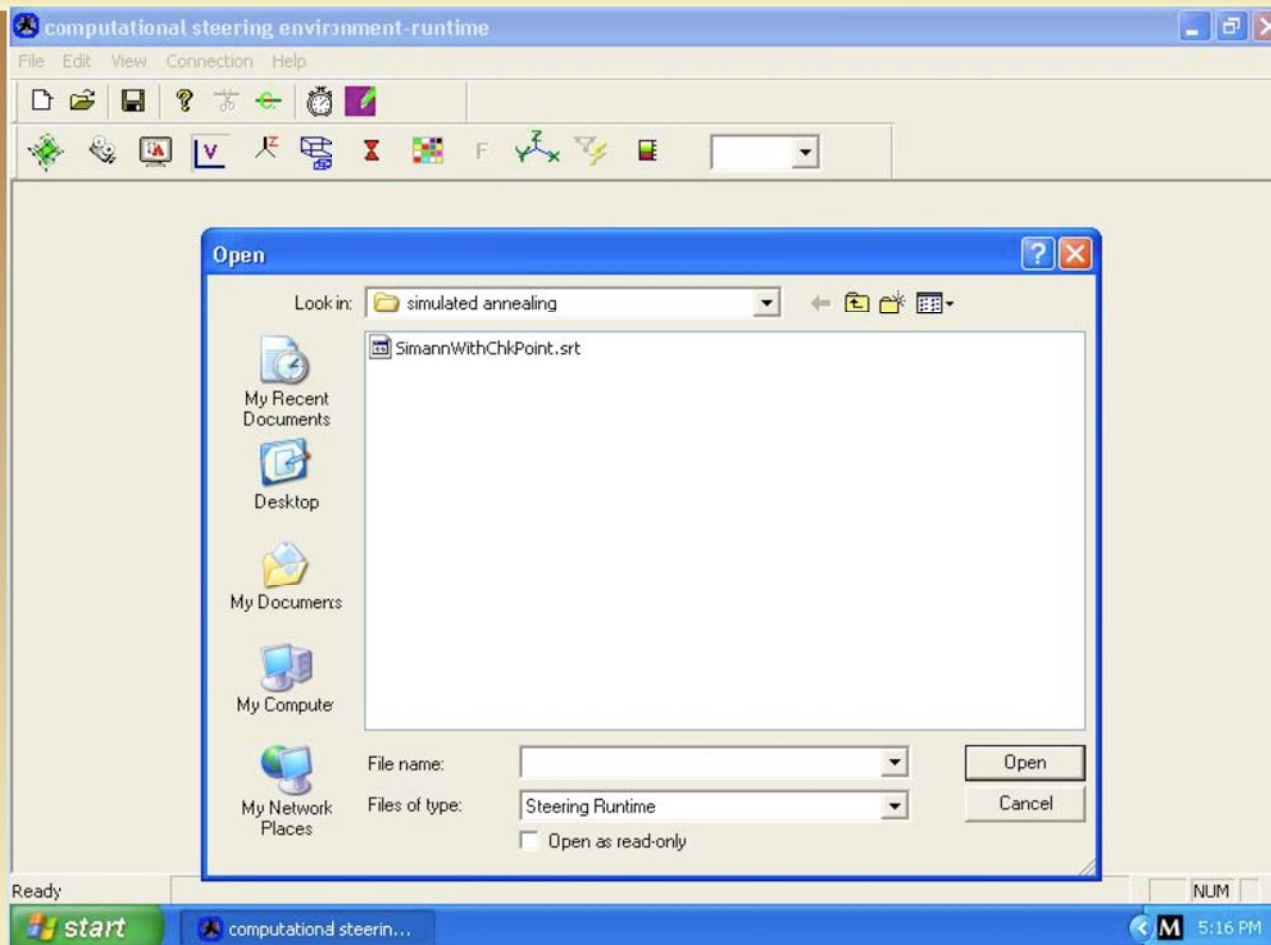
Motivations

Objectives

**STEEL**

Applications

Conclusions



STEEL Runtime Utility



# STEEL Architecture (Cont.)



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- 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 Steering API



# STEEL Users



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- 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



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- 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



Introduction

Motivations

Objectives

STEEL

Application

Conclusions

- 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



Introduction

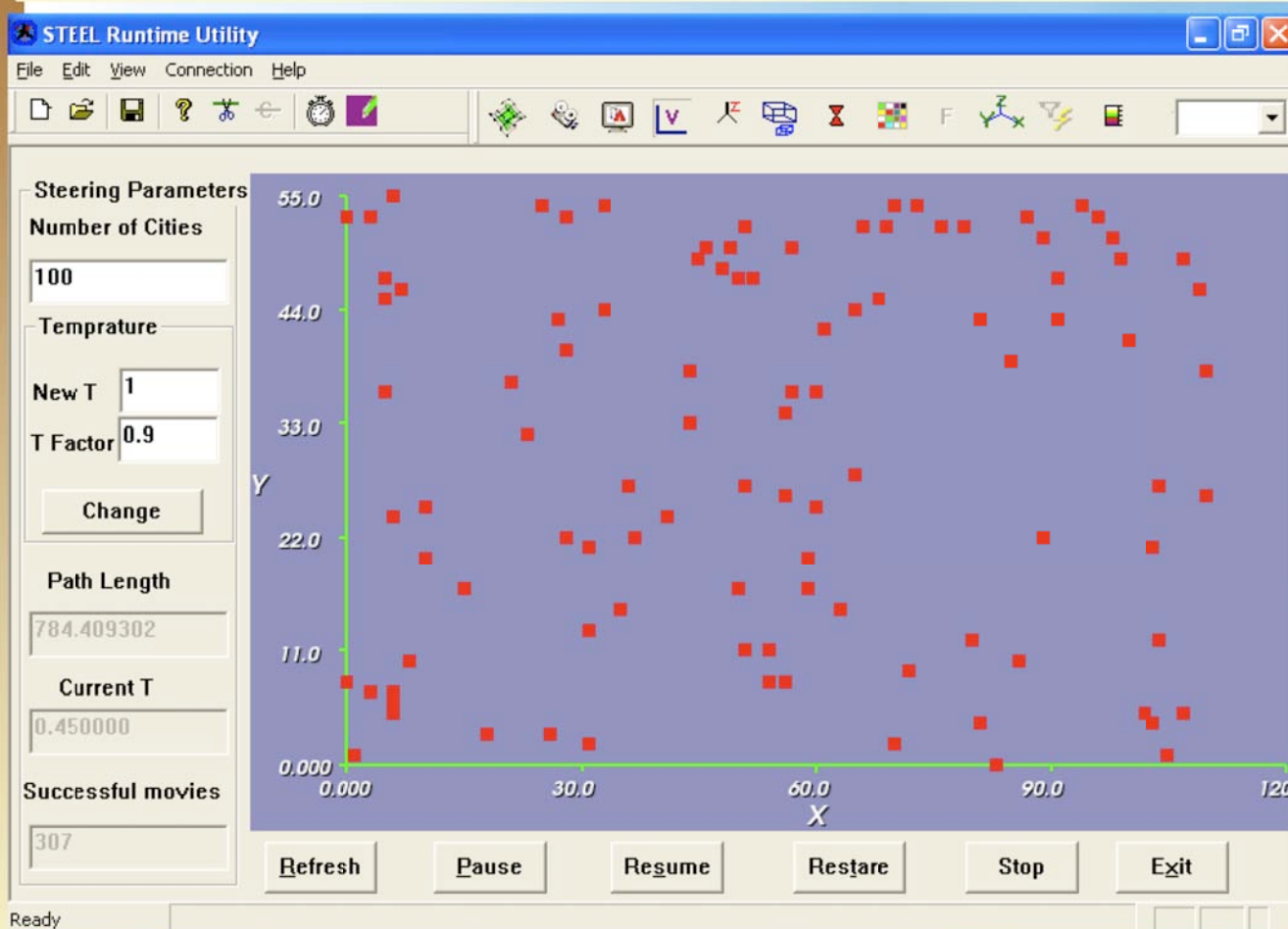
Motivations

Objectives

STEEL

Applications

Conclusions



Random Number of Points Representing 100 Cites

# The traveling salesman problem using simulated annealing Algorithm



Introduction

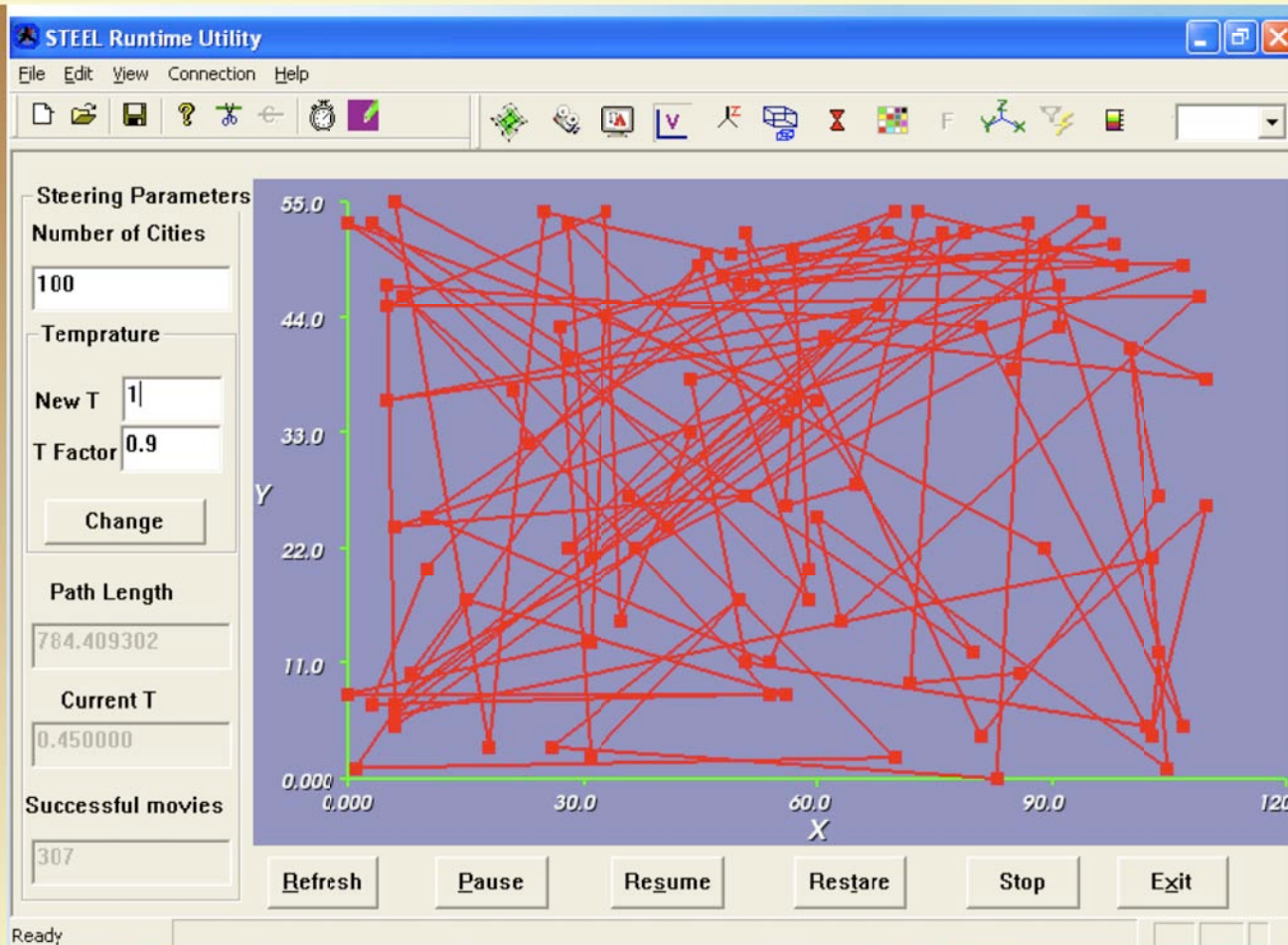
Motivations

Objectives

STEEL

Applications

Conclusions



Initial Path

# The traveling salesman problem using simulated annealing Algorithm



Introduction

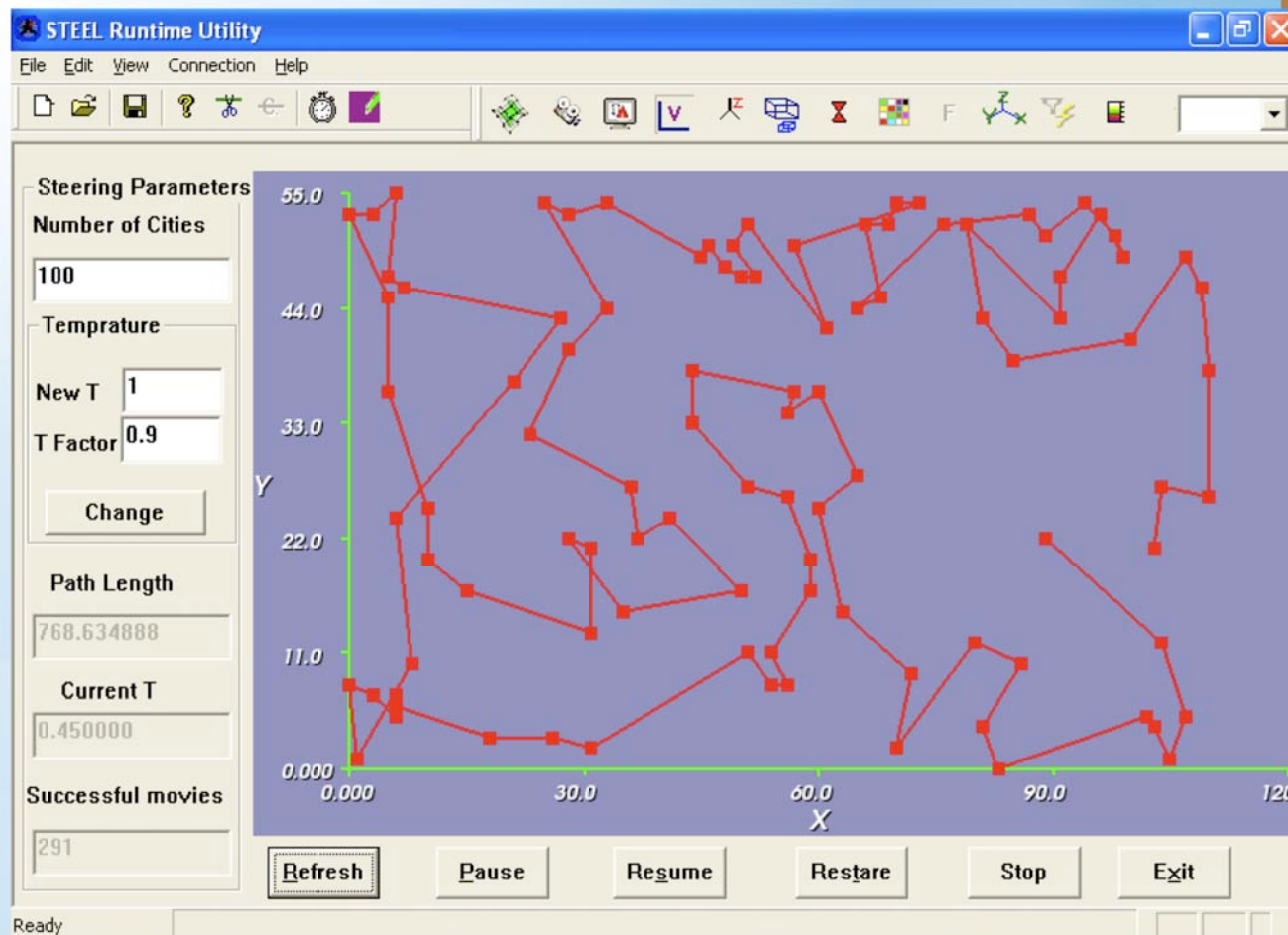
Motivations

Objectives

STEEL

Applications

Conclusions



Intermediate Path

# The traveling salesman problem using simulated annealing Algorithm



Introduction

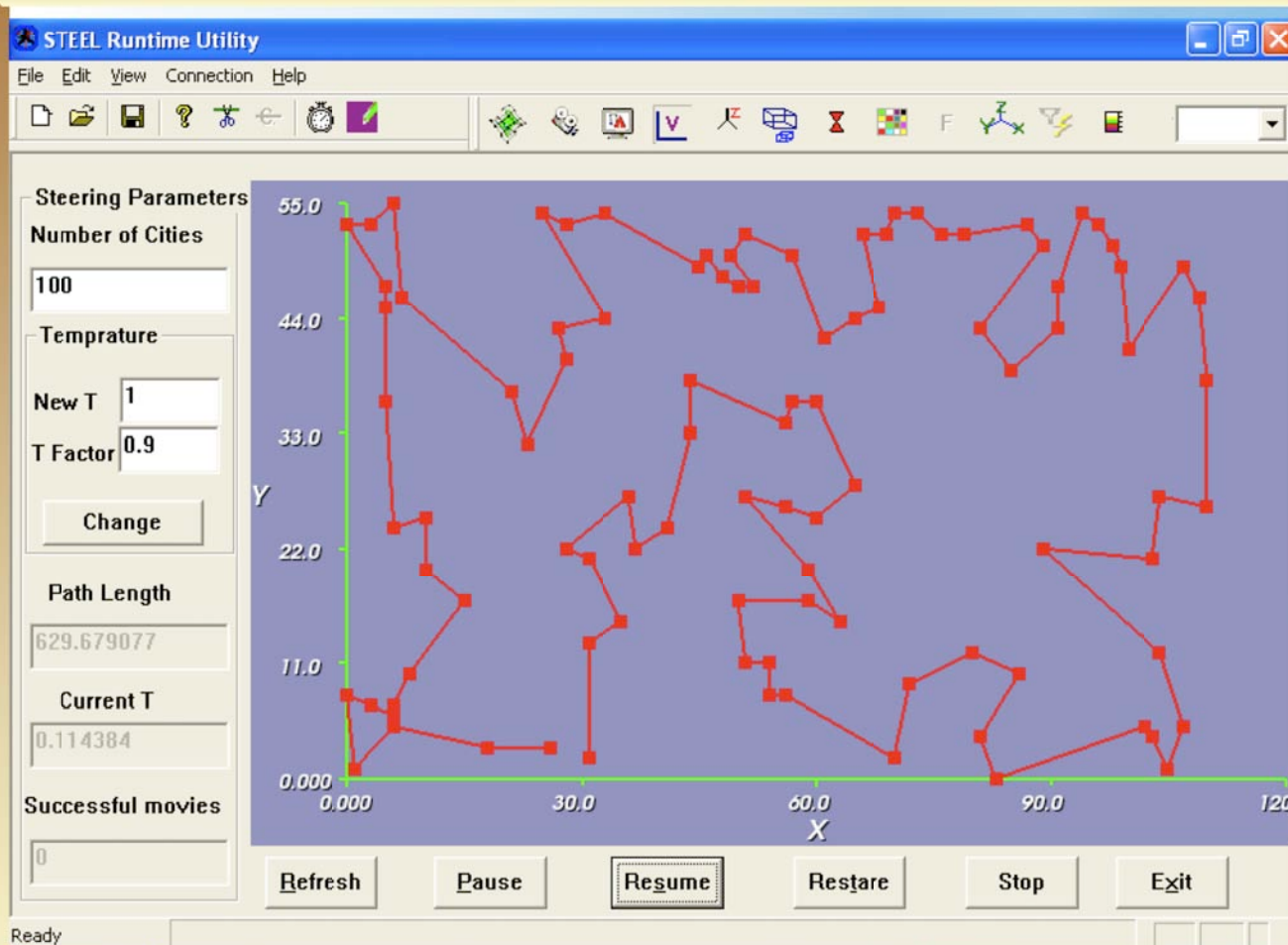
Motivations

Objectives

STEEL

Applications

Conclusions



Final Path



# The traveling salesman problem using simulated annealing Algorithm



Introduction

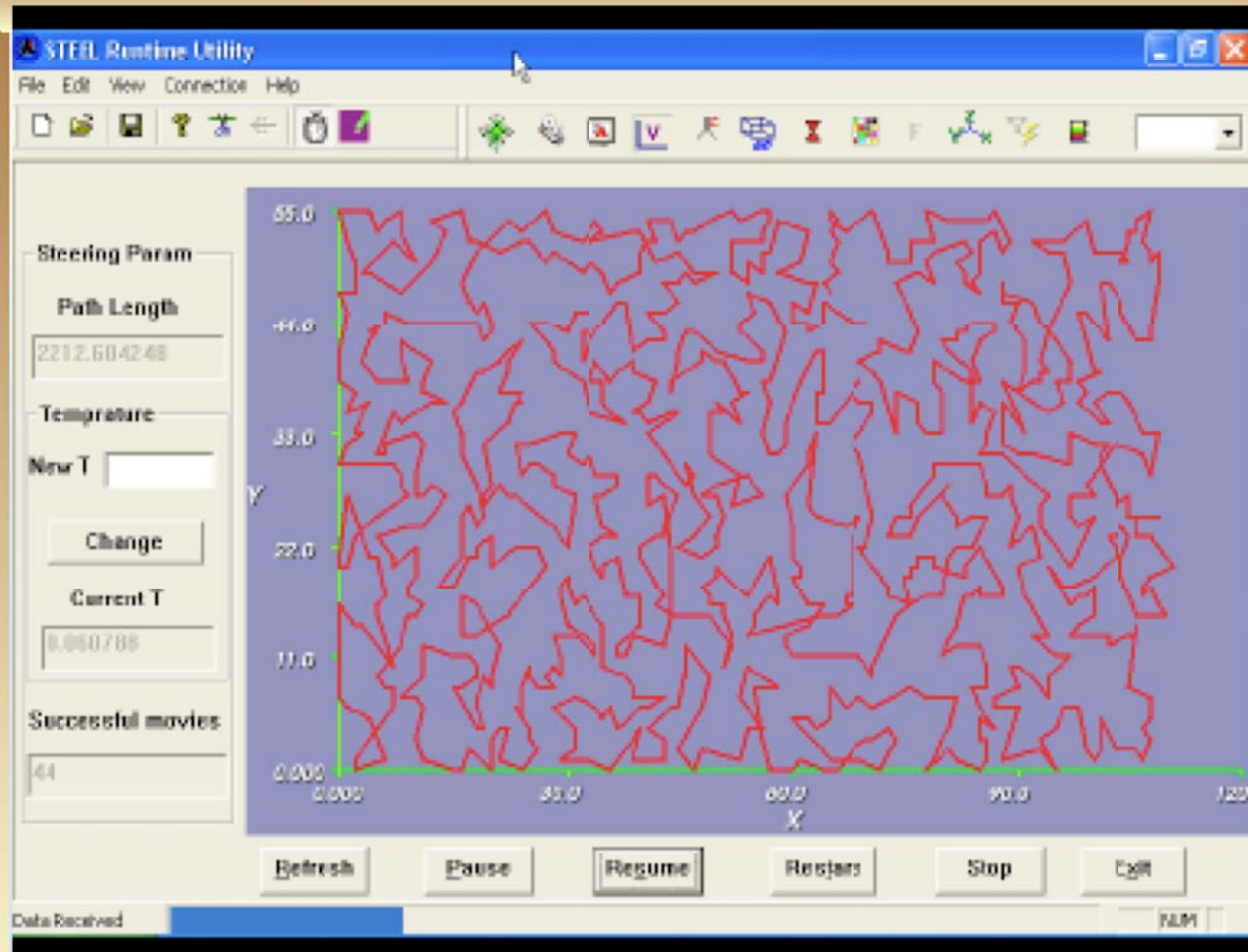
Motivations

Objectives

STEEL

Applications

Conclusions



Movie Demonstrating STEEL While Working



# The traveling salesman problem using simulated annealing Algorithm



Introduction

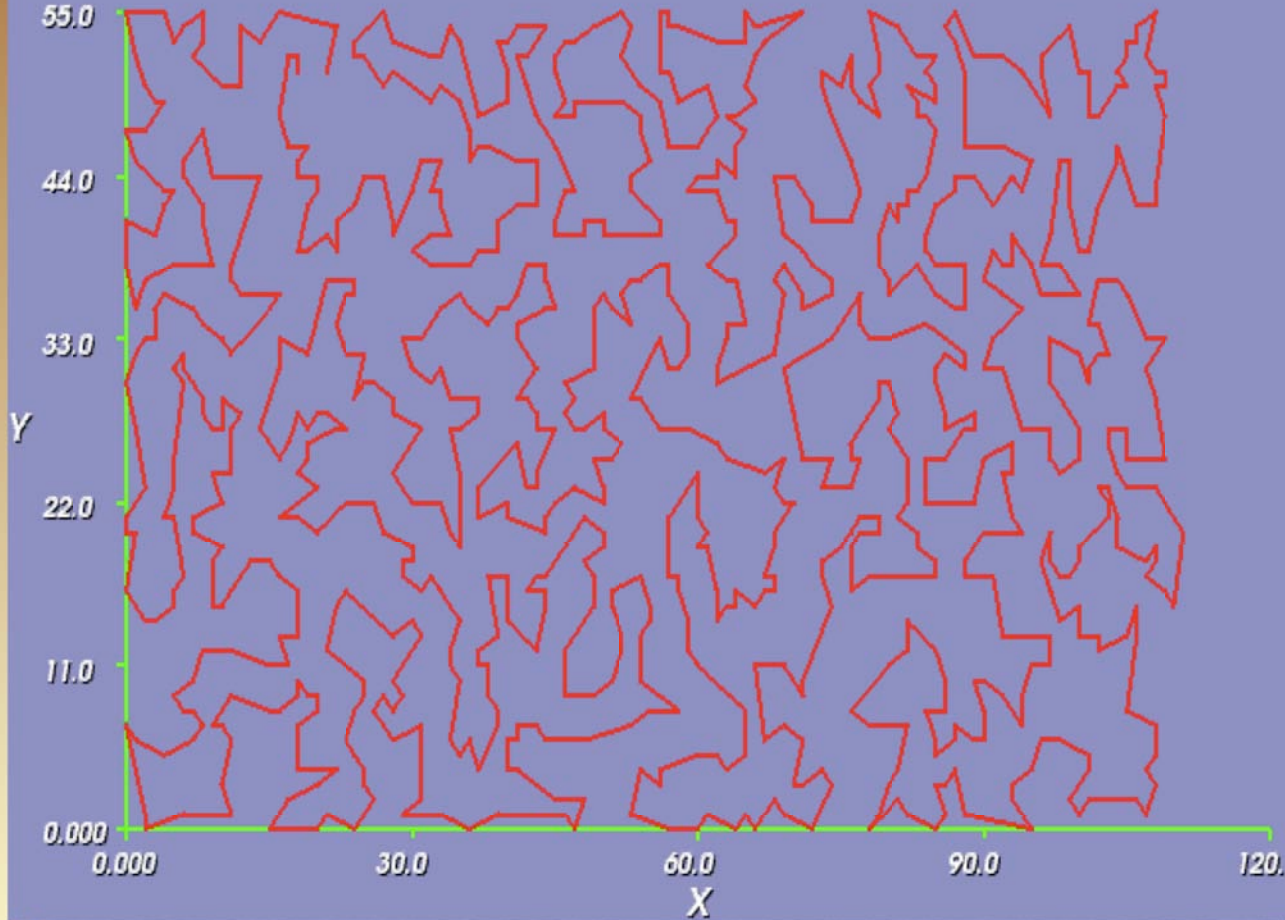
Motivations

Objectives

STEEL

Applications

Conclusions



Movie produced using STEEL



# Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds



Introduction

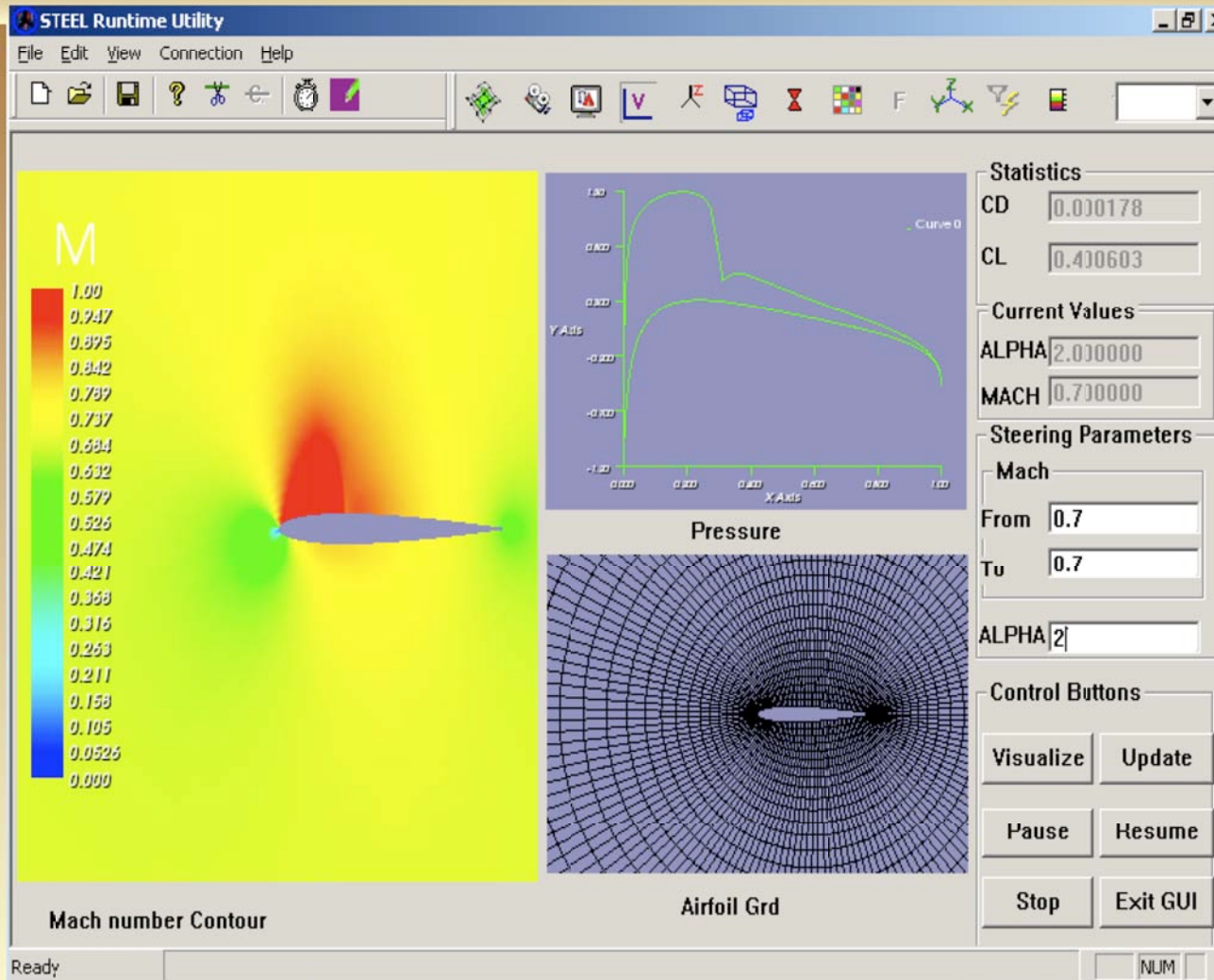
Motivations

Objectives

STEEL

Application

Conclusions





# Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds



Introduction

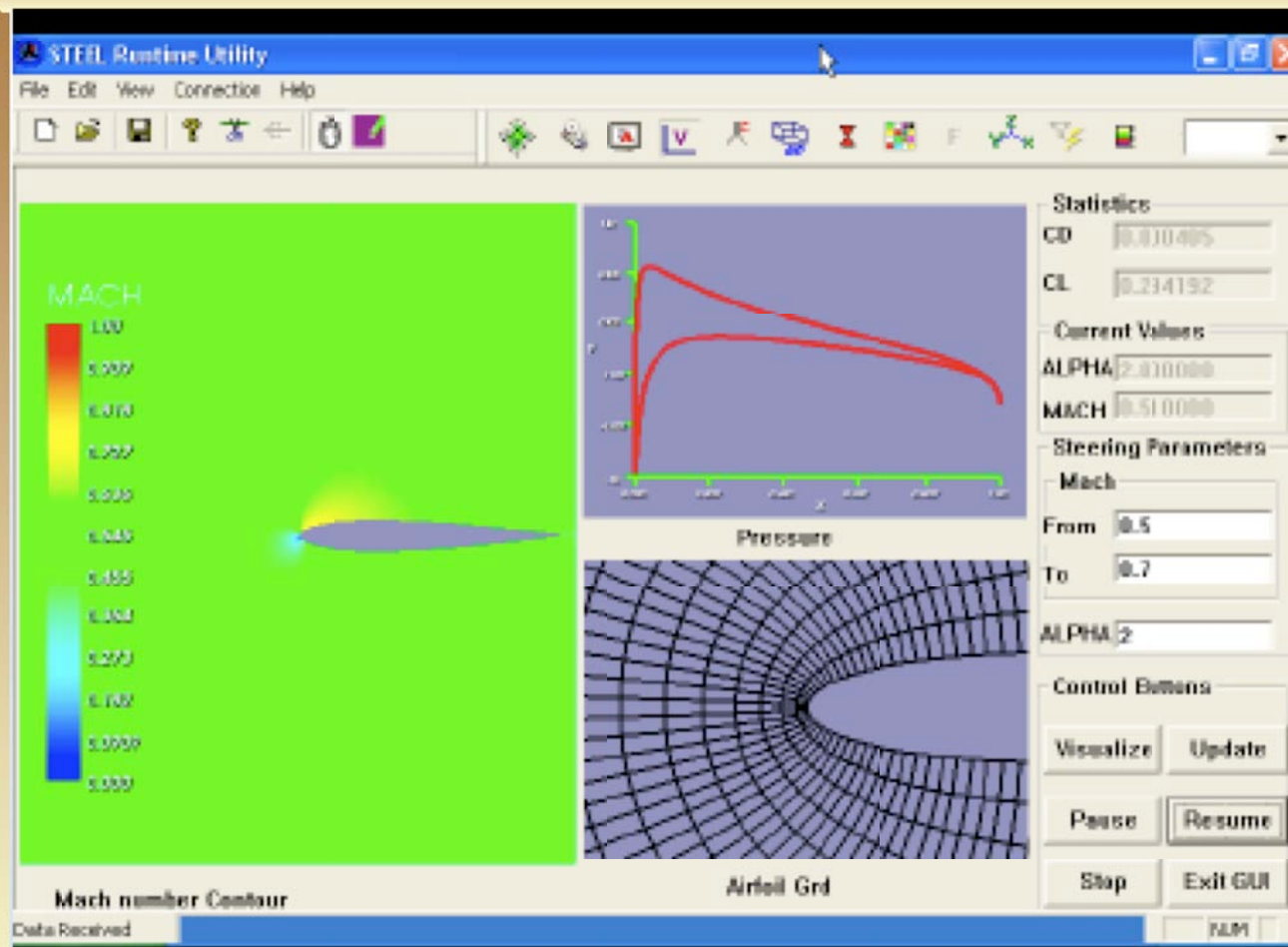
Motivations

Objectives

STEEL

Application

Conclusions



Movie Demonstrating STEEL While Working



# Numerical flow Simulation and Visualization past an Airfoil at Transonic Speeds



Introduction

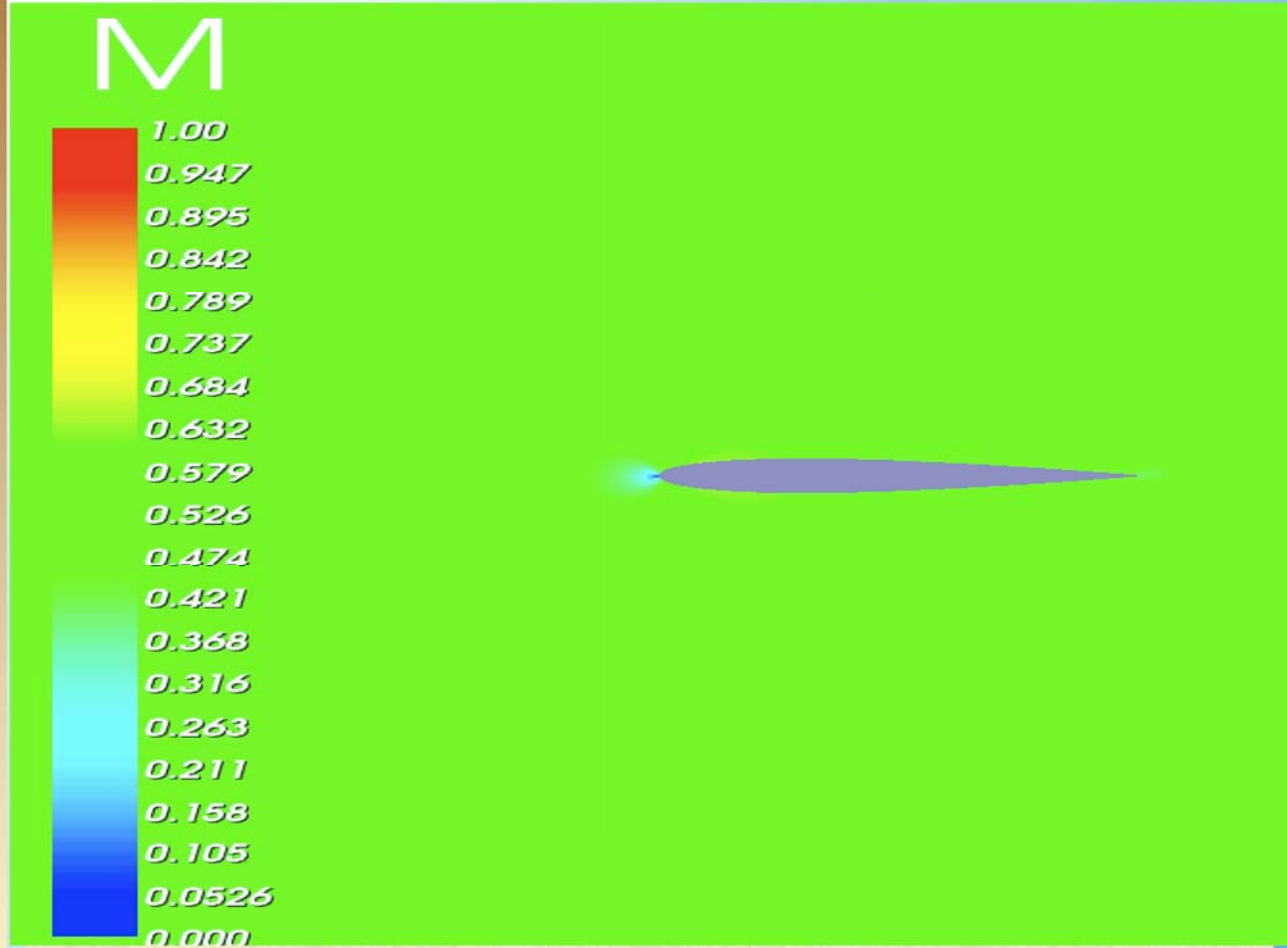
Motivations

Objectives

STEEL

Application

Conclusions



Movie produced using STEEL



# Image Analogies – Image Colorization



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

The screenshot displays the STEEL Runtime Utility window. The interface is divided into several sections:

- Source Example:** A grayscale image of two trees in a field.
- Target Unfiltered Image:** A grayscale image of a forest.
- Filtered Example:** The colorized version of the source image, showing green grass and blue sky.
- Target Filtered Image:** The colorized version of the target image, showing a green forest.

On the right side, there are control panels for:

- Output Statistics:** Algorithm Total Time: 51.0000S
- Algorithm Options:** Search Method (1-8): 6, ANN epsilon: 1.00000, Passes: 1, Filter Proc: 0. Checkboxes for Use Bias, Use PCA, Use Gain, and Use Random (checked).
- Neighborhoods:** Coherence eps: 1.00000, Source Weight: 200.000, Width: 5, NUM Level: 2, Spline Weight (checked).
- Control Panels:** Buttons for Refrsh, Update Var, Pause, Resume, Exit GUI, and Exit App.

The status bar at the bottom shows "Ready" and "NUM".

<http://mrl.nyu.edu/projects/image-analogies/>

# Image Analogies- Texture by Numbers



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

The screenshot displays the STEEL Runtime Utility application window. The interface is divided into several sections:

- Target Unfiltered Image (Mask):** A top-left panel showing a binary mask with red, blue, and black regions.
- Target Filtered Image (New Path):** A top-right panel showing a grayscale aerial image with a blue path overlaid on a green landscape.
- Source Example (Path):** A bottom-left panel showing a binary mask similar to the target unfiltered image.
- Filtered Example (New Mask):** A bottom-right panel showing a grayscale aerial image with a blue path overlaid, similar to the target filtered image.
- Output Statistics:** A panel on the right showing "Algorithm Total Time" as 297.000 S.
- Algorithm Options:** A panel on the right with various settings:
  - Search Method (1-8): 6
  - ANN epsilon: 1.00000
  - Passes: 1
  - Filter Proc: 0
  - Use Bias:
  - Use Random:
  - Use PCA:
  - Use Gain:
- Neighborhoods:** A panel on the right with settings:
  - Coherence eps: 1.00000
  - Source Weight: 200.000
  - Width: 5
  - NUM Level: 4
  - Spline Weight:
- Control Panels:** A panel on the right with buttons for "Refresh", "Update Var", "Pause", "Resume", "Exit GUI", and "Exit App".

The status bar at the bottom shows "Ready" on the left and "NUM" on the right.



# Conclusions and Future Work



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- 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



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- 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.



# Publications



Introduction

Motivations

Objectives

STEEL

Applications

Conclusions

- M. Herajy, B. Eldesouky, and E. Atta , collaborative and distributed electronic learning through interactive simulation and steering, proceeding of the 5th international internet education conference , september 2006, cairo, egypt.
- M. Herajy, B. Eldesouky, and E. Atta, A distributed computational steering environment for electronic learning applications, proceeding 3rd international conference on intelligent computing and information systems, 2007, cairo, egypt.





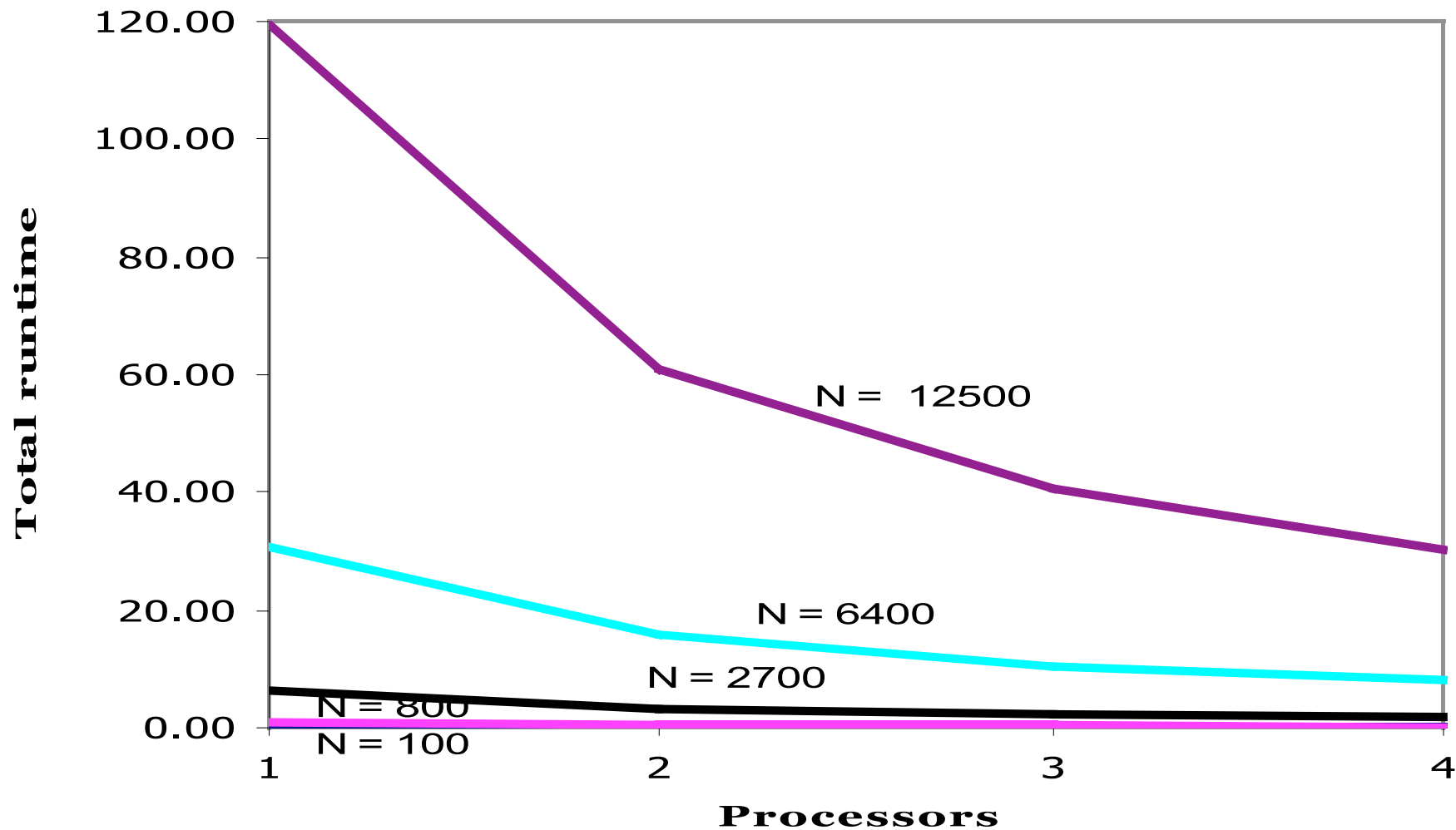
Thank You



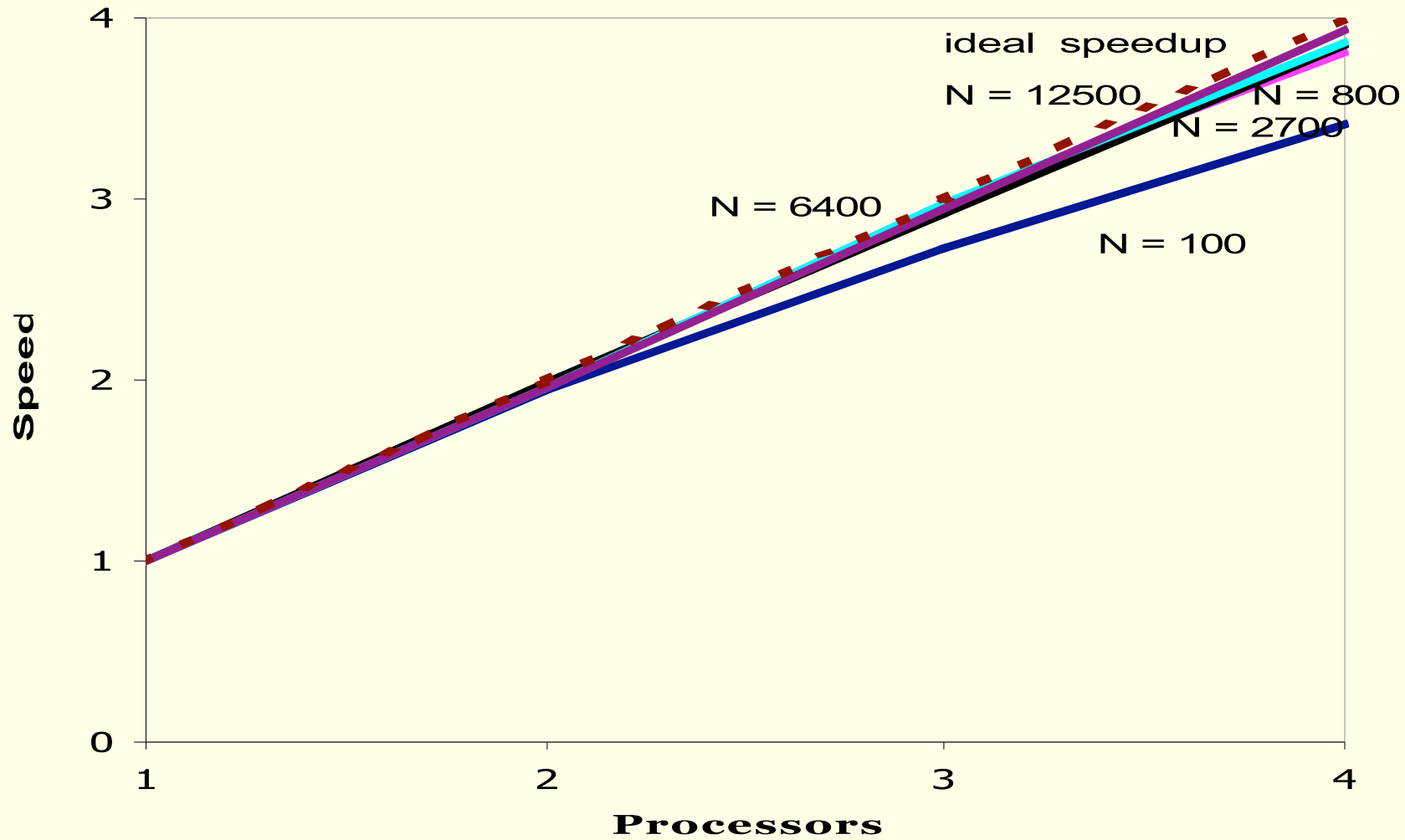
# Comparison with other systems

System	Distributed	Allow Parallelism	Scope			User Interface	Fault tolerance and Backtracking	Application
			Model Exploration	Algorithm experimentation	Performance optimization			
USE	YES	NO	YES	NO	NO	Visualization and steering through PGO	NO	
CUMULVS	YES	YES	YES	NO	YES	Visualization through AVS, textual steering	Fault tolerance	
SCIRun	YES	YES	YES	YES	NO	Steering through tcl/tk , visualization module	NO	
POSSE	YES	YES	YES	NO	NO	Use existing package for visualization	NO	
DISCOVER	YES	NO	YES	YES	NO	Web based visualization and steering	NO	
VASE	YES	NO	YES	YES	NO	Visualization through existing ,packages	NO	
RealityGrid	YES	YES	YES	NO	NO	Steering through textual inputs	Backtracking	
Progress & Magellan	YES	NO	YES	NO	YES	Visualization through existing packages, steering through command line and GUI	NO	
STEEL	YES	YES	YES	YES	NO	Steering and visualization through dedicated client	Backtracking	E-learning

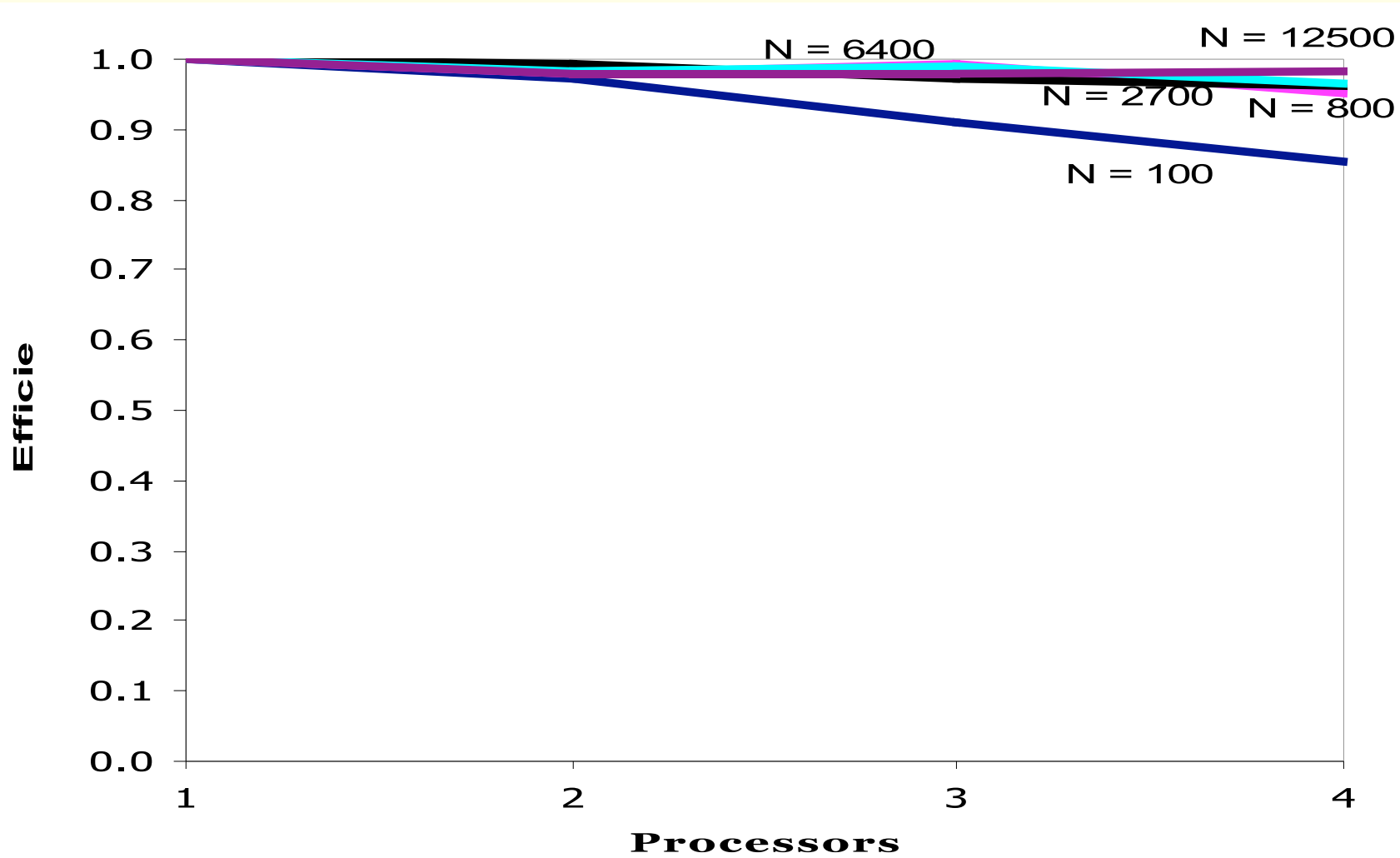
# Total runtime as a function of the number of processors



# Speedup as a function of the number of processors



# Efficiency as a function of the number of processors



# The performance analysis with fixed problem size

