Visualisierungsinterface für Strömungssimulationsdaten

Atanas Atanasov

24.03.2008

Systementwicklungsprojekt

Wintersemester 2007-2008

Technische Universität München

Lehrstuhl für Informatik mit Schwerpunkt Wissenschaftliches Rechnen

Gliederung

1.Einführung

Numerische Simulation wird zu einem immer wichtigeren Werkzeug bei der erfolgreichen Entwicklung und Optimierung verschiedenster Anwendungen. Insbesondere der Simulation von Strömungen kommt dabei eine zentrale Rolle zu, z.B. beim Design von Turbinenschaufeln, Fluid-Struktur Wechselwirkungen etc. Da realistische Simulationsläufe stets eine genügend hohe räumliche Auflösung und damit große Datenmengen erfordern, ist neben einem Programm, dass diese Daten überhaupt in hinreichend geringer Zeit erzeugen kann, eine effiziente und aussagekräftige Visualisierung der Ergebnisse unerlässlich (R.W. Hamming: "The goal is insight, not numbers.").

Im Rahmen der Forschungsarbeit zu effizienten Strömungssimulationen wird am Lehrstuhl für Informatik V zur Zeit von mehreren Mitarbeitern und Studenten ein C++-Projekt namens Peano entwickelt. Zur graphischen Aufbereitung der Daten wird dabei bisher das Visualisierungswerkzeug ParaView verwendet, das - wohl auch aufgrund seiner Fokussierung auf "kleine", leistungsschwächere Architekturen, wie sie üblicherweise der Open Source Community nur zur Verfügung stehen - bereits jetzt an seine Grenzen stößt. Daher wurde vom Lehrstuhl das kommerzielle Visualisierungstool tecplot beschafft, das - in Verbindung mit einer performanten Opteron-64Bit-Quad-Maschine und einem hochauflösenden 30"-Apple-Cinema-Display - eine interessante Visualisierungsoption darstellt, die in Zukunft genutzt werden soll.

Im Rahmen dieses Systementwicklungsprojekts werden dazu zunächst die bereits vorhandenen ParaView-Output-Routinen des Projektes peano auf die tecplot-Syntax für das ASCII-Format umgestellt . Sowohl 2D als auch 3D Szenarien auf regulären und adaptiven kartesischen Gittern werden dazu verwendet. In einem zweiten Schritt wird dann das Ausgabeformat von ASCII auf Binärdaten umgestellt . Dazu sind eigene Binäroutputklassen entwickelt worden, die mit tecplot kompatibel sind. In dem dritten Schritt der Projektarbeit werden Mechanismen zum Plotten von Simulationsdaten in verschiedenen Detailstufen implementiert .

2.ASCII Ausgabeformat

Tecplot unterstützt das Einlesen von verschiedenen Ausgabeformaten und eines davon ist das natürliche Tecplot Format. Es werden zwei mögliche Darstellungen des Formats angeboten: ASCII und binäre Dateien. Das folgende Kapitel beschreibt kurz das ASCII Format und die entsprechenden Outputklassen in Peano. Ein einfaches Beispiel für eine Tecplot Datei in dem natürlichen ASCII Format sieht folgendermaßen aus:

```
TITLE = "test"
VARIABLES = "X" , "Y" , "Z",“pressure“
ZONE T="Zone1" N=4 E=1 DATAPACKING=BLOCK ZONETYPE=FEQUADRILATERAL
VARLOCATION=([1-3]=NODAL,[4]=CELLCENTERED)
SOLUTIONTIME=0
0.5 0.5 0.5 0.5
0.5 0.5 0.5 0.5
0.5 0.5 0.5 0.5
1

1 2 3 4
```

Am Anfang jeder Datei muss eine Liste mit der Namen der Variablen existieren. In dem Beispiel gibt es vier Variablen (x,y,z und pressure). Danach folgt die Definition des Datensatzes. Die beginnt mit dem Kennwort zone. T bezeichnet hier den Titel der Zone, N die Anzahl der Datenpunkte und E die Anzahl der Zellen. Mit DATAPACKING=BLOCK wird festgelegt, dass alle Werte der ersten Variable in einem Block eingegeben sind und dann folgt einen Block für die zweite Variable und so weiter. Jede Variable in einer Zone kann sich auf den Knoten oder im Zentrum einer Zelle befinden. Das wird durch die Spezifikation der VARLOCATION festgelegt. Jede Zone kann auch einen Parameter zuständig für die Zeit haben (SOLUTIONTIME). Am Ende der Zone kommt die so genannte Connectivity List. Der Parameter ZONETYPE=FEQUADRILATERAL legt fest, dass eine Zelle aus vier Punkten besteht. In der Connectivity List steht die Information von welchen Punkten eine Zelle aufgebaut ist. In dem Beispiel hat die einzelne Zelle die Punkte 1 ,2 ,3 und 4.

3.Binary Ausgabeformat

Das zweite Format, das von TecWriter implementiert wird, ist das natürliche binäre Ausgabeformat von Tecplot. Das Format besteht aus zwei Teilen: **Header Section** und **Data Section**.

In der Header Section werden die Version und den Titel der Datei geschrieben. Sie enthält noch die Namen aller Variablen und die Zoneninformationen. In dem Data-Bereich kommen alle Daten, die zu den im Header beschrieben Zonen gehören. Ein Vorteil des Formats gegenüber der ASCII Dateien ist der kleinere Speicherverbrauch und die effizientere Interpretation innerhalb von Tecplot. Weitere Informationen über das Format können in der Dokumentation von Tecplot oder in dem Source von Preplot gefunden werden.

4.Implementierung

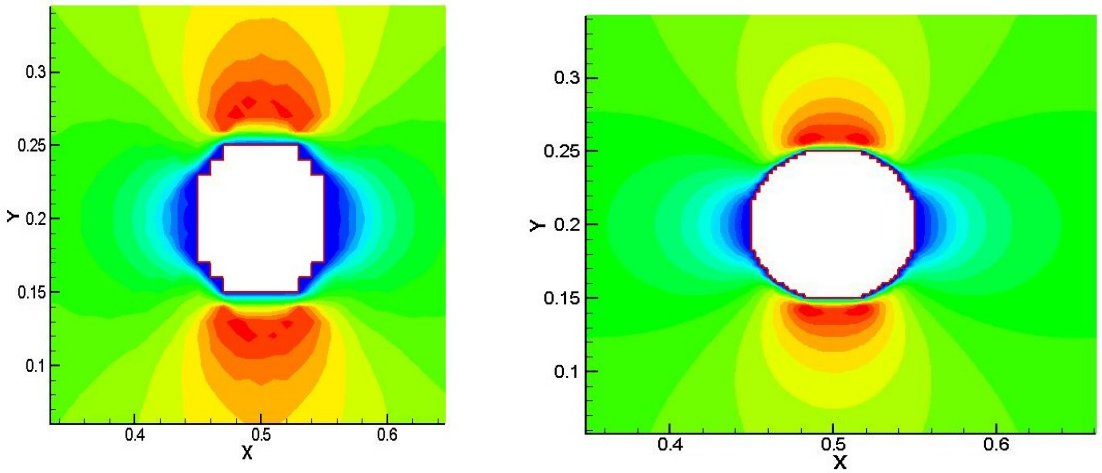
Um die Ergebnisse der Fluidsimualtion in einem Ausgabeformat lesbar für Tecplot darzustellen, wurden zwei neuen Adapter in der Fluid-Komponente eingefügt. Das Projekt unterstützt zwei Arten von Gittern: das reguläre Gitter(TrivialgridEventHandle2TECWriter4FluidAdapter) und das adaptive Peano-Gitter(GridEventHandle2TECWriter4FluidAdapter). Bei dem Durchgehen des Gitters werden spezielle Ereignisse ausgelöst (z.B Berührung eines Vertex oder Betreten einer Zelle). Bei diesen Ereignissen werden die Daten in temporäre Dateien durch verschiedene Writers gespeichert. Einzelne Writeres existieren für Koordinaten der Vertices(TecWriter::VertexWriter), Geschwindigkeiten(AbstractWriter::VectorVertexDataWriter),Vertexinformationen (AbstractWriter::ScalarVertexDataWriter) Zelleninformationen(AbstractWriter::CellDataWriter) und Konnektivität der Zellen(TecWriter::ElementWriter). Nach der Beendigung der Iteration wird die terminate-Methode des Adapters aufgerufen. Da werden alle Daten zu dem TecWriter mittels seiner plot-Methoden weitergeleitet.Bei dem Schließen des Writers werden diese Daten mittels der insert-Methoden in die Enddatei eingefügt. Diese Methoden sind für das ASCII-Ausgabeformat und für das Binary-Ausgabeformat implementiert. Ein Zeitschritt des Simulation wird in einer eigenen Datei gespeichert.

4.1.Experimentelle Ergebnisse für das reguläre Cartesian Grid.

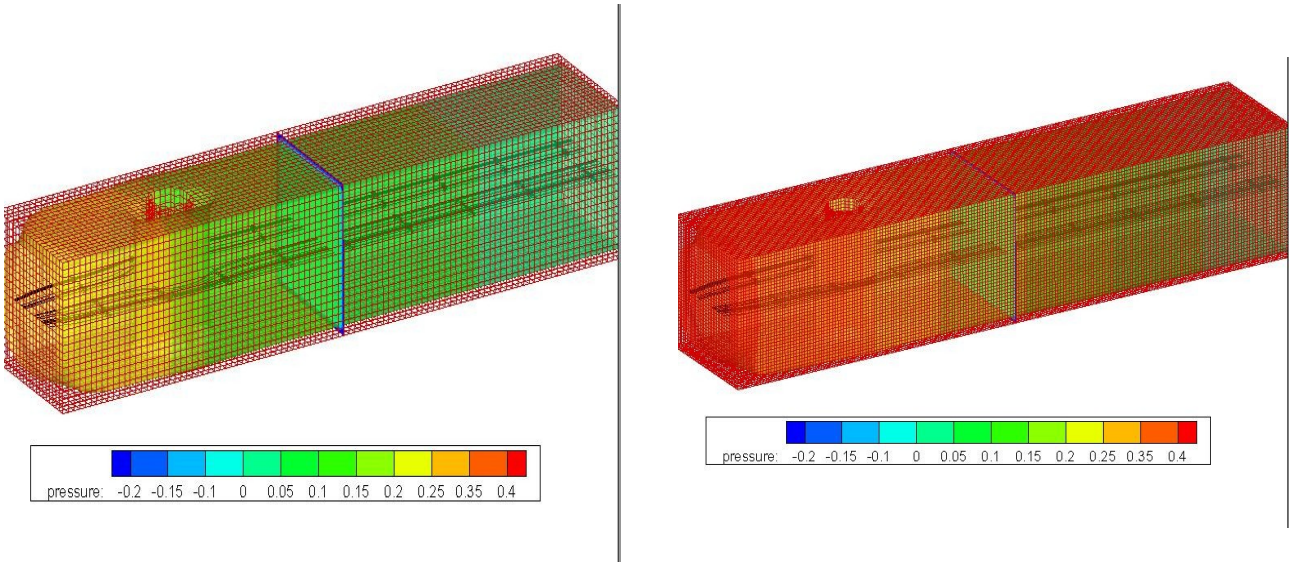
- Die folgende Tabelle und Bilder zeigen die Testergebnisse für 2D DFG Zylinder-Szenario. Bei größeren Auflösungen wird der Vorteil des Binary-Formats immer deutlicher.

Results

Identifier	scenario	Zellen	file size ASCII	file size Binary	time
1.	DFG-Kanal 2D	10162	525.4KB	740.8 KB	11.44 sec
2.	DFG-Kanal 2D	40656	3 MB	2 MB	45.77s
3.	DFG-Kanal 2D	162676	12.7MB	8.1 MB	234.44s



- Die nachfolgenden Bilder zeigen die Ausgaben für einen Cylinder-Test im regulären Cartesian Grid -3D mit Auflösungen:`100x20x20 und 400x80x80.`Der Druck wird durch Farben zwischen blau und rot dargestellt.



In der Tabelle werden drei Testfälle für reguläre Gitter in 3D abgedeckt.

Results

Identifier	scenario	Zellen	file size ASCII	file size Binary	time
1.	DFG-Kanal 3D	4880	456.2KB	315.5 KB	15 sec
2.	DFG-Kanal 3D	39360	3.9 MB	2.4 MB	96 sec
3.	DFG-Kanal 3D	316480	33.5MB	18.5 MB	794 sec

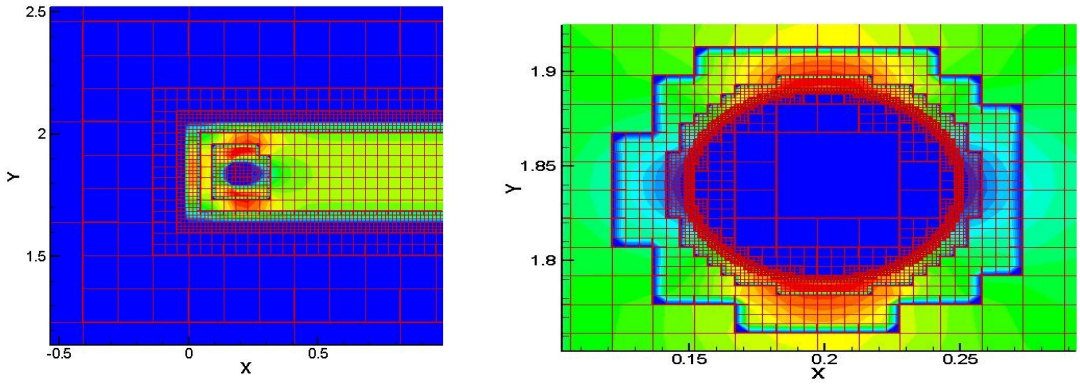
4.2.Experimentelle Ergebnisse für Adaptiven-Peano-Grid

Die folgende Tabelle zeigt die Ergebnisse aus dem 2D DFG Zylinder-Szenario in dem adaptiven Peano-Gitter. Der Druck auf den Bildern wird durch die Farbkodierung dargestellt. Dabei werden vier Auflösungen getestet.

Die Tabelle beinhaltet Information über den Zeit- und Speicherverbrauch.

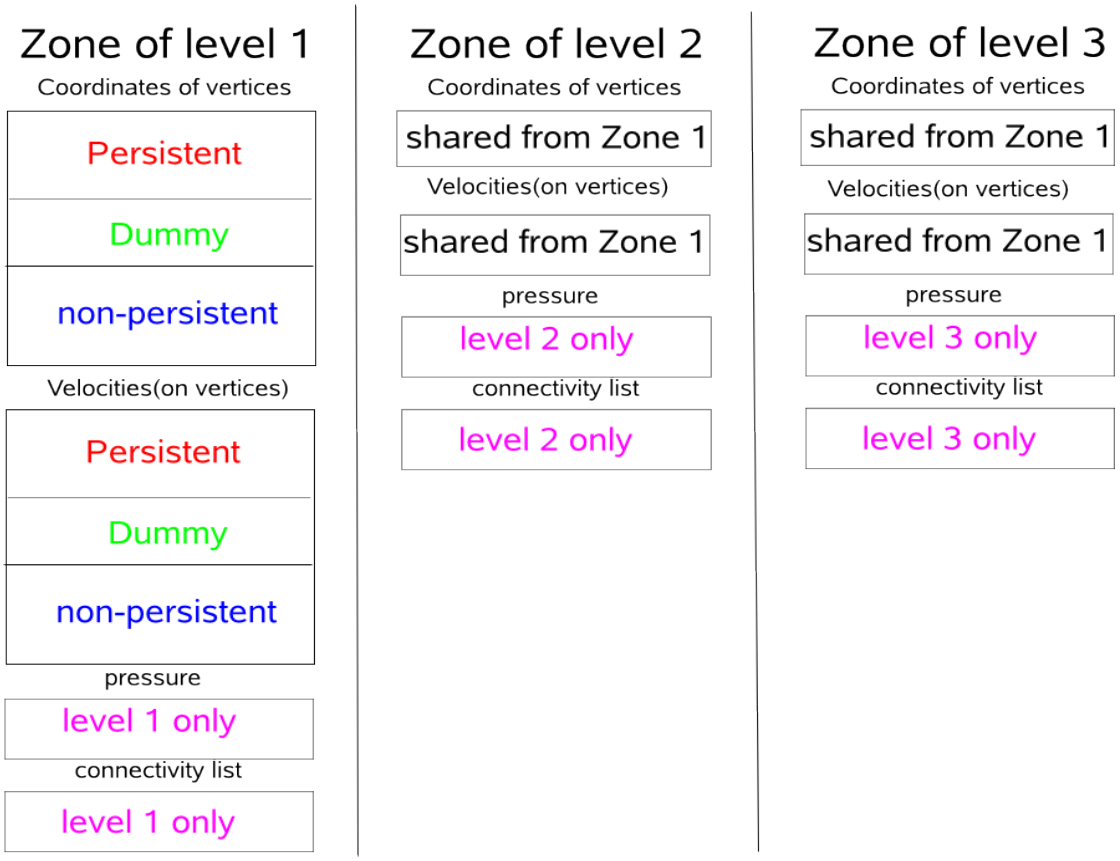
Results

Identifier	scenario	cells	file size ASCII	file size BIN	time
1.	DFG-Kanal 2D	3377	223.8 KB	190.7 KB	74.69 sec
2.	DFG-Kanal 2D	3825	256.2 KB	216.3 KB	87.01 sec
3.	DFG-Kanal 2D	5233	360.2 KB	296.8 KB	125.03 sec
4.	DFG-Kanal 2D	9009	634.3 KB	512.5 KB	234.02 sec

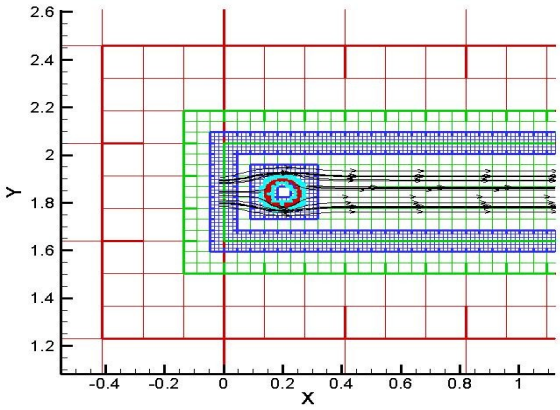


5.Multilevel Visualisierung.

Um multilevel Daten visualisieren zu können,wurde einen neuer Adapter (GridEventHandle2TECWriter4FluidLevelAdapter) in die Fluid-Komponente des Projekts eingefügt. Die Idee besteht darin, bei dem Durchgang über das Gitter die Daten für die verschiedenen Detailstufen unabhängig von einander zu plotten. Jede Detailstuffe wird in einer eigenen Zone eingefügt. Die erste Zone in der Datei enthält alle Vertexdaten und die nachfolgenden Zonen benutzen die brauchbaren Informationen durch der Shared List-Eigenschaft des Tecplot-Datenformats. Die Konnektivität der Zellen wird durch die globalen Vertices-Nummer bestimmt, d.h für jede Zone nach der ersten in der Datei sollen nur die Konnektivitätinformation und die Eigenschaften der Zellen (pressure) gespeichert werden. Die Konfiguration unterstützt die Eingabe des minimalen und des maximalen Levels durch die Attributen level-min und level-max. Alle Daten die außerhalb des bestimmten Intervalls liegen,werden in einer Default-Zone (Zone 1) gespeichert.Die nachfolgende Skizze zeigt ein Beispiel für die Aufbau einer Multilevel-tecplot Datei.



Eine Beispielausgabe sieht folgendermaßen aus :



Die einzelnen Levels werden in verschiedenen Farben dargestellt. Die Richtungen der Geschwindigkeitsvektoren werden mittels Streamtraces von Tecplot angezeigt.