Visualization of Large and Unstructured Data Sets – Applications in Geospatial Planning, Modeling and Engineering

IRTG 1131 Workshop, March 19–21, 2010, Bodega Bay, U.S.

Edited by
Ariane Middel
Inga Scheler
Hans Hagen

OASIcs – Vol. 19 – VLUDS'10 www.dagstuhl.de/oasics

Editors

Ariane Middel
Decision Center for a Desert City
Arizona State University
ariane.middel@asu.edu

Inga Scheler
Regionales Hochschulrechenzentrum Kaiserslautern (RHRK)
University of Kaiserslautern
scheler@rhrk.uni-kl.de

Hans Hagen
Computer Graphics and Visualization Group
University of Kaiserslautern
hagen@informatik.uni-kl.de

ACM Classification 1998
I.3 Computer Graphics

ISBN 978-3-939897-29-3

Published online and open access by
Schloss Dagstuhl – Leibniz–Zentrum für Informatik gGmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at http://www.dagstuhl.de/dagpub/978-3-939897-29-3.

Publication date
April, 2011.

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Digital Object Identifier: 10.4230/OASIcs.VLUDS.2010.i

OASIcs – OpenAccess Series in Informatics

OASIcs aims at a suitable publication venue to publish peer-reviewed collections of papers emerging from a scientific event. OASIcs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

ISSN 2190-6807

www.dagstuhl.de/oasics

Contents

Finite Element Analysis for Linear Elastic Solids Based on Subdivision Schemes
Daniel Burkhart, Bernd Hamann, and Georg Umlauf 1

A Survey of Interface Tracking Methods in Multi-phase Fluid Visualization
Fang Chen and Hans Hagen 11

Survey on Benchmarks for a GPU Based Multi Camera Stereo Matching Algorithm
Klaus Denker and Georg Umlauf 20

A Survey of Interaction Techniques and Devices for Large High Resolution Displays
Taimur K. Khan 27

Detection and Identification Techniques for Markers Used in Computer Vision
Johannes Koehler, Alain Pagani, and Didier Stricker 36

Markerless Camera Pose Estimation - An Overview
Tobias Nöll, Alain Pagani, and Didier Stricker 45

On Moving Least Squares Based Flow Visualization
Harald Obermaier, Martin Hering-Bertram, Jörg Kuhnert, and Hans Hagen 55

Modeling and visualizing urban sprawl and carbon footprints in Phoenix metropolitan area
Sebastian Petsch, Subhrajit Guhathakurta, and Hans Hagen 64

Advanced Visualization and Interaction Techniques for Large High-Resolution Displays
Sebastian Thelen 73

Open Problems in Computational Steering of Massive Parallel Unstructured Grid Based CFD Simulations
Christian Wagner 82

Methods for Feature Detection in Point Clouds
Christopher Weber, Stefanie Hahmann, and Hans Hagen 90

Cartography of Mars in a Virtual Reality Environment
Rolf Westerteiger 100

Visualization in Human-Centered Virtual Factories
Xiang Yang, Eduard Deines, and Jan C. Aurich 111

IRTG 1131 Workshop 2010 (VLUDS'10).
Editors: A. Middel, I. Scheler, H. Hagen

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Preface

The International Research and Training Group (IRTG) Visualization of Large and Unstructured Datasets Applications in Geospatial Planning, Modeling and Engineering is a joint effort of the University of Kaiserslautern (Germany) and the U.S. partners University of California Davis, Arizona State University and University of Utah. It is funded by the German research foundation (DFG) under grant DFG GK 1131/2. In July 2009 the second 4.5 years phase of the IRTG started. The primary research goal of this graduate program is the enhancement of scientific and information visualization techniques applied to large and unstructured datasets. Every visualization task is based on application data. For providing these data, we integrate applications from the domain Geospatial Planning, Modeling and Engineering, which produce these huge amounts of unstructured data that are of interest for the visualization tasks at hand. This integration is necessary to allow a deeper understanding of the provided data due to the sharing of knowledge through the projects. Up to now, visualization of large and structured or small and unstructured datasets is the state of the art. Large and unstructured datasets are still not very well understood, especially with respect to visualization. In order to address these questions, we have defined a set of projects aiming at solving these problems. In detail, we are handling visualization problems, with respect to modeling, feature detection, and comparison tasks. For doing this, both the extension of existing techniques and the development of new ones are investigated. In the application areas there is an increasing need to handle huge amounts of unstructured data produced either by data from field measurements like environmental observation stations, from experiments, and from simulation. For example, environmental monitoring systems are capable of measuring data at a very high resolution and in a large number of frequency bands. On the other hand, scaled-down earthquake laboratory experiments within a centrifuge improved sensor technology permit the measurement of an increased number of participants at higher sampling rates. Finally, earthquake simulations produce more and more data because of more elaborate simulation techniques. All these improvements in measurement technology lead to large, high-dimensional data sets. Visualizing these data is very useful to get new insights into the problems involved. The visualizations themselves are based on improved or newly developed visualization techniques like volume modeling, feature detection and visualization, etc. In the issue of OASIcs - OpenAccess Series in Informatics we present the results of the annual workshop of this IRTG held in Bodega Marine Laboratory, Bodega Bay, California, U.S, March 19th to March 21st 2010. Aim of the workshop was to bring together all project partners, PHD students and advisors to report on the different research projects. After three days of presentations and discussions the graduates spent their time on writing papers that cover the outcome of the program and give surveys on related topics.

IRTG 1131 Workshop 2010 (VLUDS'10).
Editors: A. Middel, I. Scheler, H. Hagen

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

List of Authors

Jan C. Aurich
University of Kaiserslautern
Faculty Mechanical and Process Engineering
FBK - Institute for Manufacturing
Technology and Production Systems
Building 42, Room 468
aurich@cpk.uni-kl.de

Daniel Burkhart
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 234
67653 Kaiserslautern
burkhart@informatik.uni-kl.de

Fang Chen
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 220
67653 Kaiserslautern
chen@informatik.uni-kl.de

Eduard Deines
Institute for Data Analysis and Visualization
(IDAV)
University of California, Davis
One Shields Ave
Davis, CA 95616, USA
edeines@ucdavis.edu

Klaus Denker
Konstanz University of Applied Sciences
Faculty Informatics
Room E205
Brauneggerstrasse 55
78462 Konstanz
klaus.denker@ktwg-konstanz.de

Subhrajit Guhathakurta
Arizona State University
University Drive and Mill Avenue, Tempe
AZ, USA
School of Geographical Sciences and Urban
Planning GIOS 348
subhro.guha@asu.edu

Hans Hagen
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 226
67653 Kaiserslautern
hagen@informatik.uni-kl.de

Stefanie Hahmann
Laboratoire Jean Kuntzmann (LJK)
BP 53
38041 Grenoble cedex 9 (France)
Stefanie.Hahmann@imag.fr

Bernd Hamann
Department of Computer Science
University of California, Davis
1 Shields Avenue
Davis, CA 95616, USA
hamann@cs.ucdavis.edu

Martin Hering-Bertram
Rhein-Waal University of Applied Sciences
Faculty Informatics
Room 1.1.03
Nollenburger Weg 115
46446 Emmerich
martin.hering-bertram@hochschule-rhein-
waal.de

Taimur K. Khan
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 229
67653 Kaiserslautern
tkhan@informatik.uni-kl.de

Johannes Köhler
DFKI GmbH
Augmented Vision
Trippstadter Strasse 122
67663 Kaiserslautern
Johannes.Koehler@dfki.de

IRTG 1131 Workshop 2010 (VLUDS'10).
Editors: A. Middel, I. Scheler, H. Hagen
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Jörg Kuhnert
The Fraunhofer-Institut für Techno- und Wirtschaftsmathematik ITWM
Fraunhofer-Platz 1
67663 Kaiserslautern
joerg.kuhnert@itwm.fhg.de

Tobias Nöll
DFKI GmbH
Augmented Vision
Trippstadter Strasse 122
67663 Kaiserslautern
tobias.noell@dfki.de

Harald Obermaier
The Fraunhofer-Institut für Techno- und Wirtschaftsmathematik ITWM
Fraunhofer-Platz 1
67663 Kaiserslautern
harald.obermaier@itwm.fhg.de

Alain Pagani
DFKI GmbH
Augmented Vision
Trippstadter Strasse 122
67663 Kaiserslautern
alain.pagani@dfki.de

Sebastian Petsch
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 231
67653 Kaiserslautern
petsch@cs.uni-kl.de

Didier Stricker
DFKI GmbH
Augmented Vision
Trippstadter Strasse 122
67663 Kaiserslautern
didier.stricker@dfki.de

Sebastian Thelen
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 229
67653 Kaiserslautern
thelen@informatik.uni-kl.de

Georg Umlauf
University of Applied Science Constance
Department of Computer Science
Computer Graphics Lab
Room F 020
78462 Constance
umlauf@htwg-konstanz.de

Christian Wagner
German Aerospace Center
Simulation and Software Technology,
Software for Space Systems and Interactive Visualization, Braunschweig
and
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 233
67653 Kaiserslautern
wagner@cs.uni-kl.de

Christopher Weber
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 237
67653 Kaiserslautern
christopherweber80@googlemail.com

Rolf Westerteiger
German Aerospace Center
Simulation and Software Technology,
Software for Space Systems and Interactive Visualization, Braunschweig
and
University of Kaiserslautern
Computer Graphics and HCI Group
P.O. Box 3049
Building 36, Room 233
67653 Kaiserslautern
rolf.westerteiger@googlemail.com

Finite Element Analysis for Linear Elastic Solids Based on Subdivision Schemes*

Daniel Burkhart[1], Bernd Hamann[2], and Georg Umlauf[3]

1 **University of Kaiserslautern**
burkhart@cs.uni-kl.de
2 **University of California, Davis**
hamann@cs.ucdavis.edu
3 **HTWG Konstanz**
umlauf@htwg-konstanz.de

Abstract

Finite element methods are used in various areas ranging from mechanical engineering to computer graphics and bio-medical applications. In engineering, a critical point is the gap between CAD and CAE. This gap results from different representations used for geometric design and physical simulation.

We present two different approaches for using subdivision solids as the only representation for modeling, simulation and visualization. This has the advantage that no data must be converted between the CAD and CAE phases. The first approach is based on an adaptive and feature-preserving tetrahedral subdivision scheme. The second approach is based on Catmull-Clark subdivision solids.

Keywords and phrases Subdivision solids, Finite element method, Isogeometric analysis

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.1

1 Introduction

In engineering, one of the major problems is still the gap between computer-aided design (CAD) and computer-aided engineering (CAE). This gap results from different representations used for the design based on exact geometries, like boundary representations (B-Reps) or non-uniform rational B-splines (NURBS), and for the simulation based on approximative mesh representations.

As illustrated in Figure 1, design and analysis are typically done sequentially or iteratively in multiple design-simulation loops. In the initial CAD modeling and CAE pre-processing phases the boundary surface is modeled, the interior of the model is meshed and the boundary conditions, such as external forces, are defined. As the CAD and CAE model have different representations, in general a time consuming data conversion between the CAD and CAE system is required. This step also causes additional approximation errors. In the subsequent CAE processing phase the resulting system of equations is solved and in the CAE post-processing phase the solution is analyzed. If the simulation results are inadequate the geometric model can be adapted or the mesh can be refined to increase the accuracy of the

* This research was supported by the German Research Foundation (DFG), which has provided the funds for the International Research Training Group (IRTG) 1131, entitled 'Visualization of Large and Unstructured Data Sets'. We gratefully acknowledge DFG's support, and also thank our colleagues from the AG Computergrafik at the University of Kaiserslautern and the Institute for Data Analysis and Visualization (IDAV) at the University of California, Davis, for their many valuable comments.

simulation. This might also require a time consuming and approximating data conversion. These optional, iterative steps are marked as dashed arrows in Figure 1.

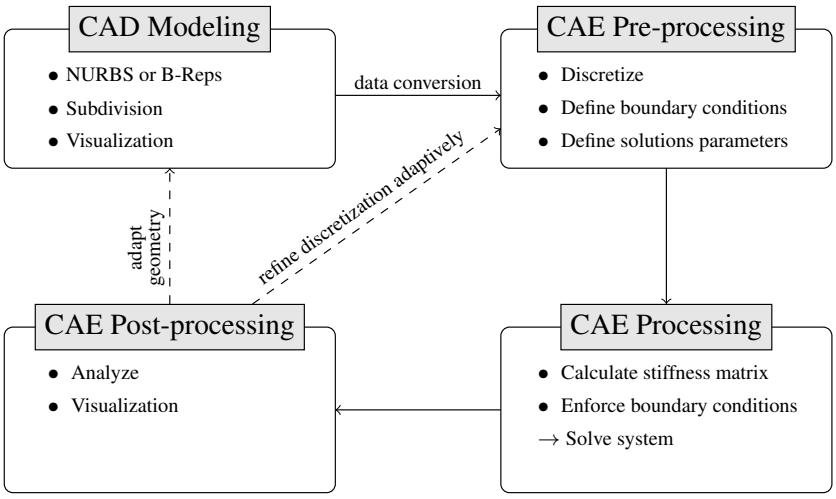


Figure 1 Different phases in the modeling and simulation process.

One solution to this time-consuming task is iso-geometric analysis (IGA), see [14]. The idea of this approach is to extend the finite element method such that it can also handle exact geometries. Thus, there is no need to transform the geometries to mesh representations which guarantees a seamless integration of CAD and CAE. Originally, IGA was based on NURBS, see [13]. Meanwhile, similar approaches for other geometric descriptions like B-Splines [17], T-Splines [1], or subdivision surfaces [10, 11] were presented. An important aspect of IGA is the fact that refinement or degree elevation of the exact geometric model can be used to increase the simulation accuracy without changing the geometry.

In this paper, we present two approaches based on subdivision solids. This extends the idea of IGA to unstructured, refinable volumetric meshes of arbitrary topology. The first approach is based on tetrahedral subdivision inspired by $\sqrt{3}$-subdivision for surfaces. This approach supports adaptive refinement and sharp features. The second approach is based on a hexahedral subdivision scheme, which generalizes Catmull-Clark subdivision surfaces to solids. This approach uses the same basis functions for the representation of the geometry and for the integration of elements during the simulation.

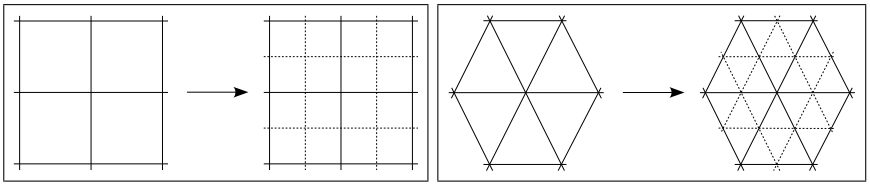
In Sections 2 and 3 we review subdivision surfaces and solids. Standard finite element techniques for linear elasticity problems are described in Section 4. In Sections 5 and 6 we describe two approaches for finite element analysis based on subdivision solids.

2 Subdivision surfaces

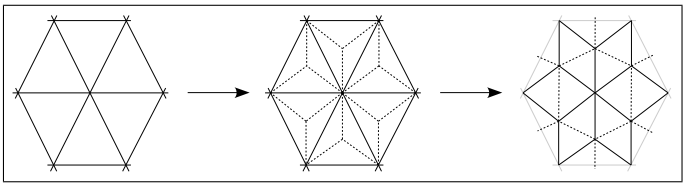
Subdivision surfaces are a powerful tool to model free-form surfaces of arbitrary topology. A subdivision surface is defined as the limit of an iterative refinement process starting with a polygonal base mesh M_0 of control points. Iterating the subdivision process generates a sequence of refined meshes $M_1, \ldots, M_n$, that converges to a smooth limit surface M_∞ for $n \to \infty$. Usually the subdivision operator can be factored into a topological refinement operation followed by a geometrical smoothing operation. While the topological refinement

inserts new vertices or flips edges, the geometrical smoothing changes vertex positions. To enforce and preserve sharp features such as corners and creases, special subdivision rules can be defined. Examples for such special rules, where tagged edges will yield creases on the subdivision surface, are presented in [15, 5, 23].

Subdivision surfaces either approximate or interpolate the base mesh. For approximating schemes the control points of M_i usually do not lie on M_{i+1}, $i \geq 0$. The Catmull-Clark algorithm [7] is an examples of such a scheme. Approximating schemes for arbitrary triangle meshes are the Loop algorithm and $\sqrt{3}$-subdivision [19, 18]. The corresponding topological refinement operators are illustrated in Figure 2. For interpolating schemes all control points of M_i are also in M_{i+1}, $i \geq 0$. Thus, the limit surface interpolates these points. The earliest interpolating subdivision scheme for surfaces is the butterfly scheme of [12]. For further details on subdivision surfaces refer to [21].



(a) Catmull-Clark subdivision. **(b)** Loop subdivision.



(c) $\sqrt{3}$ subdivision.

Figure 2 Topological refinement operators.

3 Subdivision solids

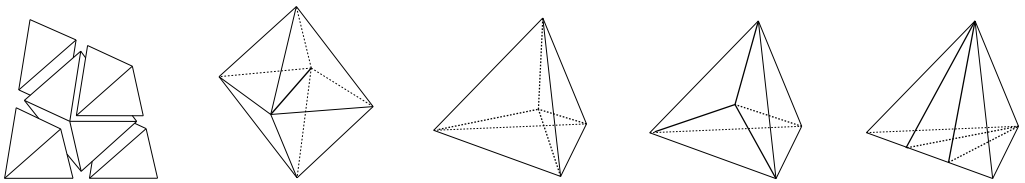
Like subdivision surfaces, subdivision solids are defined as the limit of an iterative refinement process, factored into topological and geometrical refinement operations. One of the first solid subdivision schemes is described in [16]. This is a generalization of Catmull-Clark subdivision to three-dimensional solids for smooth deformations based on unstructured hexahedral meshes. As the topological refinement operation of this algorithm made it hard to analyze the smoothness of the resulting limit solid a modified operation was proposed in [6]. The advantage of this scheme is its simplicity compared to the other subdivision solids, e.g. [8, 9, 20]. From a hexahedral base mesh, only hexahedral elements are generated, all inserted vertices are regular, i.e., they have valence six, and the limit solids are at least C^1 away from creases or corners. The subdivision rules for Catmull-Clark solids for hexahedral meshes are defined by five steps:

1. For each hexahedron with nodes $V_1, \ldots, V_8$ add a cell point $C = (V_1 + \cdots + V_8)/8$.
2. For each face add a face point $F = (C_0 + 2A + C_1)/4$, where C_0 and C_1 are the cell points of the two incident hexahedra and A is the face centroid.
3. For each edge add an edge point $E = (C_{\text{avg}} + 2A_{\text{avg}} + (n-3)M)/n$, where n is the number of incident faces, M is the edge midpoint, and C_{avg} and A_{avg} are the averages of

cell and face points of incident cells and faces, respectively.

4. For each hexahedron connect its cell point to all its face points and connect all face points to all incident edge points. This splits one hexahedron into eight hexahedra.
5. Move each original vertex V_{old} to $V_{\text{new}} = (C_{\text{avg}} + 3A_{\text{avg}} + 3M_{\text{avg}} + V_{\text{old}})/8$, where C_{avg}, A_{avg}, and M_{avg} are the averages of the cell, face and edge points of all adjacent cells, faces, and edges, respectively.

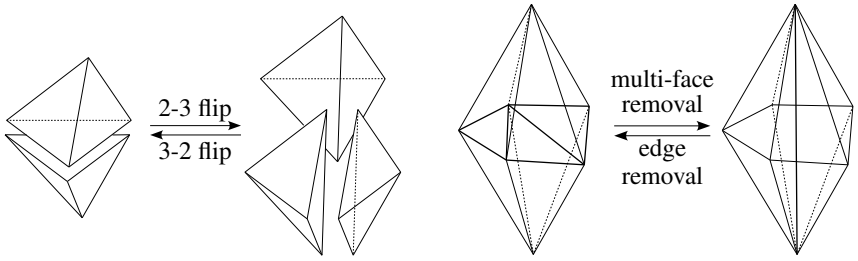
For faces, edges and vertices on the boundary of the solid corresponding rules for Catmull-Clark surfaces are applied.



(a) Edge bisection. **(b)** Diagonal in octahedron. **(c)** 1-4 split. **(d)** 1-3 split. **(e)** Edge trisection.

Figure 3 Split operations for tetrahedral subdivision.

A subdivision scheme for tetrahedral meshes based on trivariate box splines was proposed in [8, 9]. This scheme is approximating or interpolating depending on the geometrical smoothing operation. The topological refinement first splits every tetrahedron into four tetrahedra and one octahedron. This operations is illustrated in Figure 3a. Subsequently, every octahedron is split along one of its diagonals into four tetrahedra causing a potential directional bias as shown in Figure 3b.



(a) 2-3 and 3-2 flip operation. **(b)** Multi-face removal and edge removal.

Figure 4 Flip operations for tetrahedral subdivision.

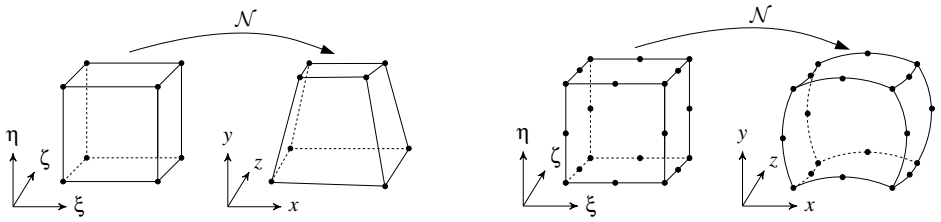
In [2] another tetrahedral subdivision scheme that generalizes the idea of $\sqrt{3}$ subdivision [18] for triangular meshes is described. While $\sqrt{3}$ subdivision is based on triangular 1-3 splits and edge flips, this tetrahedral subdivision scheme is a combination of 1-4 splits (Figure 3c) and 2-3 flips (Figure 4a) in the interior and the $\sqrt{3}$ scheme and edge removals (Figure 4b) on the boundary. For these boundary steps, tetrahedral 1-3 splits (Figure 3d) are required. For preservation of sharp features 1-3 edge splits (Figure 3e) are required. Additional optimization steps are used to guarantee high quality of the tetrahedra. In contrast to earlier solid subdivision schemes, this scheme allows for adaptive refinement by restricting the 2-3 flips and the boundary edge removals, control of the shape of the tetrahedra by adjusting the optimization steps, and preservation of sharp features by adjusting the two

smoothing operations. The latter can also be used to replace the original $\sqrt{3}$ smoothing by an interpolatory smoothing. These properties make this subdivision scheme suitable for FEM simulations. For details see [2].

4 Finite element analysis of linear elastic solids

Finite element analysis is a numerical method to solve partial differential equations by first discretizing these equations in their spatial dimensions. This discretization is done locally in small regions of simple shape (the finite elements) connected at discrete nodes. The solution of the variational equations is approximated with local shape functions defined for the finite elements.

For volumetric problems the most common element types are hexahedra and tetrahedra. Typically, these elements are defined in a local coordinate system. This simplifies the construction of shape functions also for higher-order elements with curved boundaries and the numerical quadrature arising during the assembly of the stiffness matrix. If the same shape functions are used to describe the variation of the unknowns, such as displacement or fluid potential, and the mapping between the global and local coordinates, the elements are called iso-parametric elements.



(a) Tri-linear hexahedral element **(b)** Tri-quadratic hexahedral element

Figure 5 Lagrangian hexahedral elements. Both elements are shown in local and global coordinates related by the corresponding shape functions $\mathcal{N}$.

A tri-linear and a tri-quadratic hexahedral element are illustrated in Figure 5, where (ξ, η, ζ) are local and (x, y, z) are global coordinates. The tri-linear element, for instance, has eight local shape functions $\mathcal{N} = [\mathcal{N}_1, ..., \mathcal{N}_8]$ defined over the cube $[-1, +1]^3$. For more details on elements of different order and their shape functions refer to [22].

The finite element approximation results in matrix equations relating the input (boundary conditions) at the discrete nodes to the output at these same nodes (the unknown variables). The contribution of each element is computed in terms of local stiffness matrices $\mathbf{K}_m$, which are assembled into a global stiffness matrix $\mathbf{K}$. This yields for static elasticity problems a linear system of equations $\mathbf{Ku} = \mathbf{f}$, where $\mathbf{u}$ is the vector of the unknown variables and $\mathbf{f}$ if the vector of external forces.

During the assembly of the stiffness matrix the shape functions and their derivatives with respect to global coordinates are involved. To convert these derivatives between the coordinate systems the Jacobian matrix given by

$$\mathbf{J} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi & \partial z/\partial \xi \\ \partial x/\partial \eta & \partial y/\partial \eta & \partial z/\partial \eta \\ \partial x/\partial \zeta & \partial y/\partial \zeta & \partial z/\partial \zeta \end{bmatrix}.$$

is used. For a linear elastic body Ω, the equations for the computation of $\mathbf{K}_m$ are typically

derived from the strain energy defined as

$$E_{\text{strain}} = \frac{1}{2}\int_{\Omega} \epsilon^T \sigma \, d\mathbf{x},$$

with the stress vector σ and the strain vector $\epsilon = [\epsilon_x \ \epsilon_y \ \epsilon_z \ \gamma_{xy} \ \gamma_{xz} \ \gamma_{yz}]^T$ defined as

$$\begin{aligned} &\epsilon_x = \partial u/\partial x, && \epsilon_y = \partial u/\partial y, && \epsilon_z = \partial u/\partial z, \\ &\gamma_{xy} = \partial u/\partial y + \partial v/\partial x, && \gamma_{xz} = \partial u/\partial z + \partial w/\partial x, && \gamma_{yz} = \partial v/\partial z + \partial w/\partial y. \end{aligned}$$

This can be rewritten as $\epsilon = \mathbf{B}u$, where $\mathbf{B}$ is the strain-displacement matrix.

$$\mathbf{B}^T = \begin{bmatrix} \partial/\partial x & 0 & 0 & \partial/\partial y & \partial/\partial z & 0 \\ 0 & \partial/\partial y & 0 & \partial/\partial x & 0 & \partial/\partial z \\ 0 & 0 & \partial/\partial z & 0 & \partial/\partial x & \partial/\partial y \end{bmatrix}.$$

Hooke's law $\sigma = \mathbf{C}\epsilon$ relates the stress vector σ to ϵ via the material matrix $\mathbf{C}$. For homogeneous, isotropic material $\mathbf{C}$ is defined by the Lamé constants λ and μ, and

$$\mathbf{C} = \begin{bmatrix} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix}.$$

Rewriting the strain energy and adding work applied by eternal forces $\mathbf{f}$ to the boundary Γ, yields the total energy function

$$E(\mathbf{u}) = \frac{1}{2}\int_{\Omega} \mathbf{u}^T\mathbf{B}^T\mathbf{C}\mathbf{B}\,\mathbf{u}\, d\mathbf{x} - \int_{\Gamma} \mathbf{f}^T\mathbf{u}\, d\mathbf{x}. \tag{1}$$

This energy function can be approximated with finite elements in terms of

$$\mathbf{K}_m = \iiint \mathbf{B}^T\mathbf{C}\mathbf{B}\, dx\, dy\, dz. \tag{2}$$

As the exact evaluation of (2) is in general not possible Gauss quadrature is used

$$\int_{-1}^{+1}\int_{-1}^{+1}\int_{-1}^{+1} \mathbf{f}(x,y,z)dxdydz \approx \sum_{i=1}^{n} W_i \mathbf{f}(x_i,y_i,z_i), \tag{3}$$

where x_i, y_i and z_i are the sampling points of the univariate quadrature rule and W_i is the product of the corresponding weights. As the elements are defined in local coordinates, combining (3) with (2) yields

$$\mathbf{K}_m \approx \sum_{i=1}^{n} W_i \det(\mathbf{J})\mathbf{B}^T\mathbf{C}\mathbf{B}, \tag{4}$$

where the Jacobian matrix $\mathbf{J}$ and $\mathbf{B}$ are evaluated at the sampling points. This requires evaluating the derivatives of the shape functions, see [24, 22] for details.

5 Adaptive tetrahedral subdivision for finite element analysis

In [3] we demonstrate the effectiveness of adaptive and feature-preserving tetrahedral subdivision for finite element simulations for the engineering part shown in Figure 6 (top left

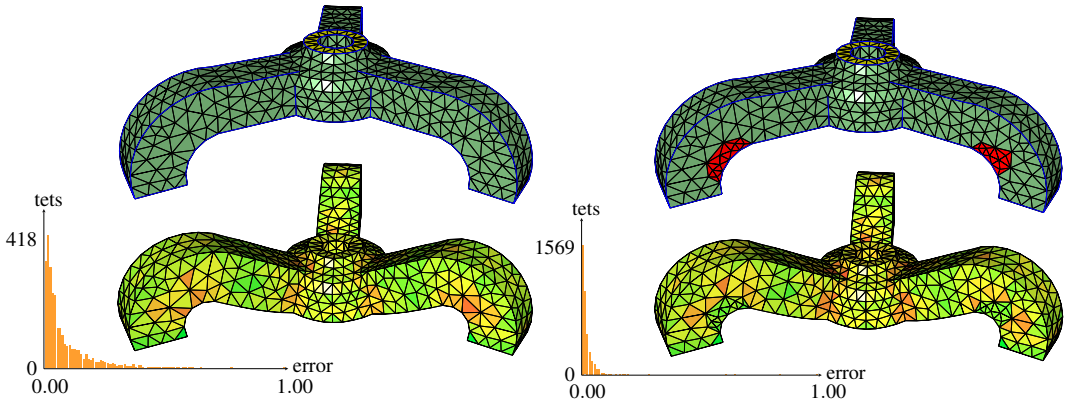


Figure 6 Adaptive subdivision and FE simulation. First column: tetrahedral base mesh (2,799 tetrahedra) and simulation with visualization of the approximation error (green=low – red=high) and the histogram of the error distribution; second column: adaptively refined mesh (4,540 tetrahedra) showing the refined regions in red, and simulation results for the once adaptively refined mesh.

model) consisting of 2,799 tetrahedra. To the top faces (yellow) of the tripod a vertical load is applied and the bottom of the legs of the tripod are fixed.

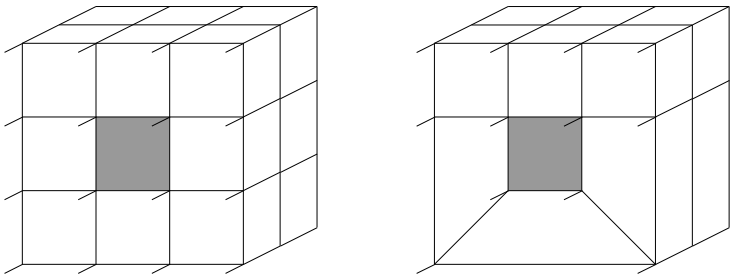
Figure 6 (bottom left model) visualizes the normalized approximation error of the deformed model, where the color hue is linearly interpolated from 0° (low error) to 120° (high error). The simulation took 491ms while the average normalized error is 0.08. The histogram shows the error distribution for the tetrahedra.

For the next step the mesh regions with the largest error are selected and refined. These refined regions are highlighted in red in Figure 6 (top right model). As some of these regions are isolated, one round of region growing is used to decrease the number of disconnected, refined regions. The adaptively refined mesh consists of 4,540 tetrahedra. Figure 6 (bottom right model) shows the deformation of this new tetrahedral mesh. The simulation took 596ms while the average normalized error is 0.03. Without adaptive refinement the mesh consists of 23,480 tetrahedra after one subdivision step. This yields a simulation time of 7,574ms with average normalized error 0.008 for the globally refined mesh. The decrease of error and the histograms getting narrower demonstrates that our method is effective. The efficiency of the proposed methods is demonstrated by reducing the computation times by a factor of twelve for the adaptively refined mesh compared to the globally refined meshes. For more details see [3].

6 Hexahedral finite element analysis based on Catmull-Clark solids

For the method presented in Section 5, tetrahedral subdivision was used to represent the geometry and to adaptively refine the mesh, but for the analysis, standard linear Lagrangian tetrahedral elements are used. In [4] we described a method that uses Catmull-Clark solids for the representation of the geometry and the approximation of the displacement field defined by Equation (1).

The major problem with this method is that the displacement field within an element does not only depend on the displacements of the nodes attached to the element but also on the displacements of the nodes of adjacent elements, because the support of the basis functions of Catmull-Clark solids overlaps a one-ring neighborhood of elements. This is



(a) Regular Catmull-Clark element

(b) Irregular Catmull-Clark element

Figure 7 To evaluate the highlighted hexahedron, all adjacent hexahedra are required.

illustrated in Figure 7, where the gray element is evaluated, but adjacent elements are also required to evaluate the derivatives in Equation (4).

For standard tri-linear and tri-quadratic elements these derivatives can be computed directly. For Catmull-Clark elements it is not obvious how to compute derivatives due to topologically arbitrary elements as shown in Figure 7b. However, evaluating the topological arbitrary elements can be reduced to evaluations of regular elements shown in Figure 7a. These regular elements can be evaluated directly with B-spline basis functions, since Catmull-Clark solids are generalizations of tri-variate cubic B-splines. For details on how to evaluate irregular elements see [4].

To demonstrate the effectiveness of our approach, we compare it to standard finite elements shown in Figure 5. As a test case we use the model shown in Figure 8a. This model is fixed at the left and right side, and a vertical load is applied to the top. We measure the maximum displacement in the direction of the load. Compared to standard tri-linear and tri-quadratic hexahedral elements Catmull-Clark elements converge faster to a reference solution. Furthermore, Catmull-Clark elements are numerically more stable than tri-quadratic finite elements. It seems that Catmull-Clark elements produce a more homogeneous stiffness matrix, which results in faster solution of the linear system of equations and in better conditioned stiffness matrices.

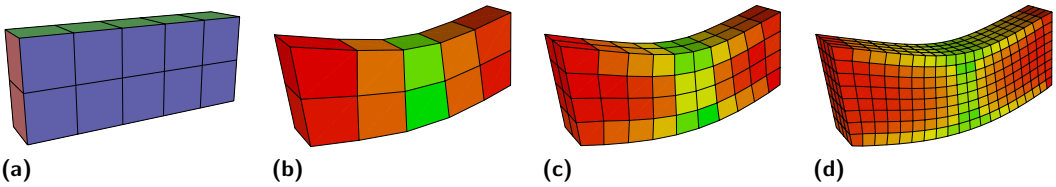


Figure 8 (a) Base mesh of model to be simulated. Red faces are fixed, to green faces a load is applied. (b) Simulation of base mesh, (c) simulation of once refined mesh, (d) simulation of twice refined mesh. For the visualization of the stress the same scale is used as in Figure 6.

Our approach is also applicable to unstructured meshes with irregular vertices and large, real-world examples. Figure 9 shows a simulation where a mesh with interior irregular vertices of valence six and ten is rotated. For the simulation, the red faces are fixed and for the vertices at the opposite side a fixed displacement is computed. Fore more examples and details concerning the convergence analysis see [4].

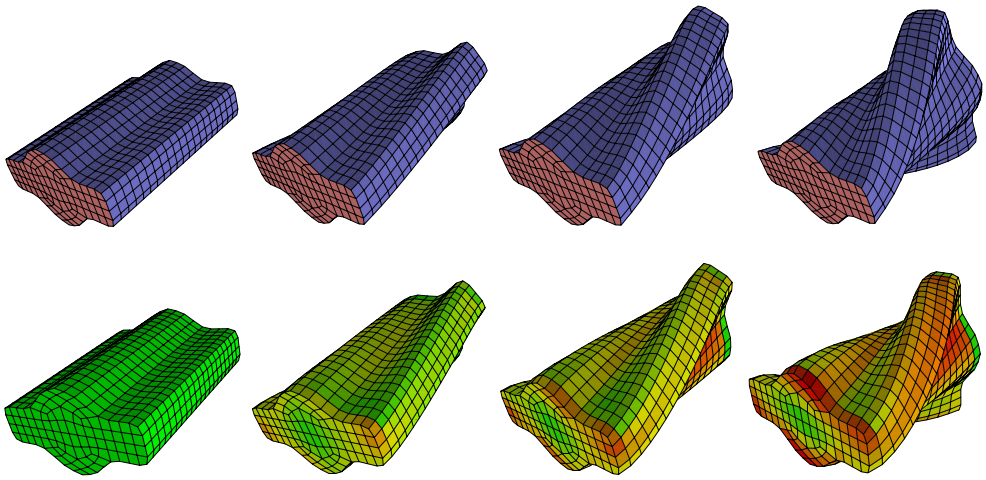


Figure 9 Simulated rotation of a model with interior vertices of valence six and ten. For the visualization of the stress the same scale is used as in Figure 6.

7 Conclusion and future work

In this paper we have presented two approaches for combining solid subdivision and FE analysis. The major advantage of these approaches is that only one representation is used for modeling, visualization and simulation of solid models, by means of an adaptive tetrahedral subdivision tailored for FE applications and an iso-geometric approach for finite element analysis based on Catmull-Clark solids.

For the future we plan to combine these subdivision schemes with more complex FE models, e.g. non-linear deformations and problems from fluid dynamics. For the second method only hexahedral meshes are supported but we are working on generalizations to arbitrary polyhedral meshes. Here, the evaluation technique of [4] can be generalized to the adaptive tetrahedral subdivision scheme presented in [2].

References

1 Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, and T.W. Sederberg. Isogeometric analysis using T-splines. *Comp. Meth. in Applied Mech. and Eng.*, 199(5-8):229 – 263, 2010.

2 D. Burkhart, B. Hamann, and G. Umlauf. Adaptive and Feature-Preserving Subdivision for High-Quality Tetrahedral Meshes. *Comput. Graph. Forum*, 29(1):117–127, 2010.

3 D. Burkhart, B. Hamann, and G. Umlauf. Adaptive tetrahedral subdivision for finite element analysis. In *Comput. Graph. International, Electronic Proceedings*, 2010.

4 D. Burkhart, B. Hamann, and G. Umlauf. Iso-geometric Finite Element Analysis Based on Catmull-Clark Subdivision Solids. *Proceedings of Symposium on Geometry Processing, Computer Graphics Forum*, 29(5):1575–1584, 2010.

5 Henning Biermann, Ioana M. Martin, Denis Zorin, and Fausto Bernardini. Sharp features on multiresolution subdivision surfaces. *Graph. Models*, 64(2):61–77, 2002.

6 C. Bajaj, S. Schaefer, J. Warren, and G. Xu. A subdivision scheme for hexahedral meshes. *The Visual Computer*, 18:343–356, 2002.

7 E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

8 Yu-Sung Chang, Kevin T. McDonnell, and Hong Qin. A new solid subdivision scheme based on box splines. In *Proc. 7th ACM Symp. on Solid Modeling and Appl.*, pages 226–233, 2002.

9 Yu-Sung Chang, Kevin T. McDonnell, and Hong Qin. An Interpolatory Subdivision for Volumetric Models over Simplicial Complexes. In *Proc. Shape Modeling International*, pages 143–152, 2003.

10 Fehmi Cirak, Michael Ortiz, and Peter Schröder. Subdivision Surfaces: A New Paradigm For Thin-Shell Finite-Element Analysis. *Int. J. Num. Meth. Eng.*, 47:2039–2072, 2000.

11 Fehmi Cirak, Michael J. Scott, Erik K. Antonsson, Michael Ortiz, and Peter Schröder. Integrated Modeling, Finite-Element Analysis, and Engineering Design for Thin-Shell Structures using Subdivision. *Computer-Aided Design*, 34:137–148, 2002.

12 Nira Dyn, David Levine, and John A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans. Graph.*, 9(2):160–169, 1990.

13 G. Farin. *Curves and Surfaces for CAGD: A Practical Guide*. Morgan Kaufmann, 5th edition, 2001.

14 T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comp. Meth. in Applied Mech. and Eng.*, 194(39-41):4135–4195, 2005.

15 Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. In *Proceedings of SIGGRAPH*, pages 295–302, 1994.

16 K. Joy and R. MacCracken. The refinement rules for Catmull-Clark solids. *Technical Report 96-1*, UC Davis, 1996.

17 Pavel Kagan, Anath Fischer, and Pinhas Z. Bar-Yoseph. Integrated mechanically based CAE system. In *ACM Symposium on Solid modeling and applications*, pages 23–30, 1999.

18 L. Kobbelt. $\sqrt{3}$ Subdivision. In *Proceedings of SIGGRAPH*, pages 103–112, 2000.

19 C. Loop. Smooth Subdivision Surfaces Based on Triangles. *Master's thesis, University of Utah*, 1987.

20 Valerio Pascucci. Slow growing volumetric subdivision. In *Proceedings of SIGGRAPH*, pages 251–251, 2002.

21 J. Peters and U. Reif. *Subdivision Surfaces*. Springer, 2008.

22 I. Smith and D. Griffiths. *Programming the Finite Element Method*. Wiley, 2004.

23 J. Warren and H. Weimer. *Subdivision Methods for Geometric Design*. Morgan Kaufmann Publishers, 2002.

24 O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method, Volume 1+2*. Butterworth, 5th edition, 2000.

A Survey of Interface Tracking Methods in Multi-phase Fluid Visualization

Fang Chen[1] **and Hans Hagen**[2]

1 **University of Kaiserslautern**
AG Computer Graphics and HCI Group, TU Kaiserslautern, Germany
chen@cs.uni-kl.de

2 **University of Kaiserslautern**
AG Computer Graphics and HCI Group, TU Kaiserslautern, Germany
hagen@cs.uni-kl.de

Abstract

A central feature that scientists are interested in is the dynamics of fluid interfaces or the so called material boundaries in multi-fluid simulation . Visualization techniques for capturing fluid interface are based on one of about three basic algorithms. In this paper, we give a survey of the existing interface tracking algorithms, including backgrounds, terms, procedures as well as pointers to details and further reading. We also provide a glance at the mathematical fundamentals of multi-fluid dynamics for scientists who are interested in understanding the underlying math and physics of multi-phase fluid simulation.

Keywords and phrases Multi-phase fluid, interface tracking, topology methods

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.11

1 Introduction

Implicit surface, dynamic and free boundaries are interesting topics in multi-phase fluid visualization. Aside from mass and momentum transportation of the flow, scientists are particularly interested in locating the sharp interfaces between the components of fluids. An interface or boundary is an implicit surface in the fluid mixture where two different materials are separated. To track the dynamic behavior of these interfaces, special algorithms are needed.

Interface tracking is a visualization technique that allows the scientist to identify and follow dynamics of fluid. It is of great interest in several application fields as diverse as chemical engineering, combustion and astrophysics.

There are three main approaches in fluids interface tracking, front tracking method, volume of fluid method, and level set method. In the next few sections, we will cover the central procedure and main idea of these algorithms, and provide also possible numerical solutions for solving the equations of computational fluid dynamics.

This paper is organized as follows: Section 2 gives a brief introduction to the mathematical simulation of multi-fluids. In section 3, we will discuss the main three types of algorithms in tracking fluids interface with two of them covered in more details. Section 4 covers the numerical schemes that can be applied to solving the corresponding hyperbolic partial differential equations when conducting those tracking algorithms. We especially expand our discussion in the direction of point-based/mesh-free methods, as they serve as a better and more efficient solution to transient multi-fluid problems.

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 11–19

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2 Simulation of multi-phase fluid

Interface tracking is a visualization technique that visualizes scientific data coming from multi-fluid simulations. To better understand the simulation data, it is also important to get an insight into the mathematical principles simulation process. In this section , we give a brief introduction to the mathematical formulation of multi-fluid simulation.

2.1 Navier-Stokes equations of multi-phase fluids

Numerical simulation of multiphase fluids are complex and challenging tasks. Current numerical algorithms mostly target a specific fluid model, and 3D simulations of multiphase fluids are still ongoing. What makes the problem so challenging is the large variety of physical phenomena in which the phases interact [9]. Here we provide a basic formulation of two-phase fluid problem, namely the conservation laws for mass and momentum.

Conservation laws for multiphase fluids problem are given follows: The first two equations describe the conservation of mass for each phase of flow, where α denotes the volume fraction of the dispersed phase. Here we consider the formulation proposed by Kuhnert and Tiwari [3, 17, 21]

$$\frac{d\alpha}{dt} = -\alpha(\nabla \cdot \mathbf{v_1}) \tag{1}$$

$$\frac{d(1-\alpha)}{dt} = -(1-\alpha)(\nabla \cdot \mathbf{v_2}) \tag{2}$$

Momentum conservation of the two phases are given as

$$\frac{d\mathbf{v_1}}{dt} = \frac{\nabla p}{\rho_1} + \frac{1}{\alpha_1 \rho_1}\nabla \cdot S_1 + \mathbf{g} + \frac{\alpha_1}{\rho_1}\mathbf{F_1} \tag{3}$$

$$\frac{d\mathbf{v_2}}{dt} = \frac{\nabla p}{\rho_2} + (1 - \frac{\rho_1}{\rho_2})\mathbf{g} - \frac{\mathbf{1}}{(\mathbf{1}-\alpha_1)\rho_2}\mathbf{F_1} \tag{4}$$

where the interfacial dragging force $\mathbf{F_1}$ is given by [24]

$$\mathbf{F_1} = -\frac{\mathbf{3}}{\mathbf{4}}(\mathbf{1}-\alpha)\rho_1 \frac{\mathbf{C}}{\mathbf{d_2}}|\mathbf{v_2} - \mathbf{v_1}|(\mathbf{v_2} - \mathbf{v_1})$$

and the interfacial stress tensor [3] is given as

$$S_1 = \alpha\mu_1[\nabla\mathbf{v_1} + (\mathbf{v_1})^{\mathbf{T}} - \frac{\mathbf{1}}{\mathbf{3}}(\nabla \cdot \mathbf{v_1})\mathbf{I}]$$

where μ is the dynamic viscosity of the liquid. In this formulation, the conservation law for different phase of fluid are coupled by volume fraction α of the dispersed phase. For instance $\alpha = 0.47$ means that the dispersed phase of liquid takes 47% of the total volume.

3 Main approaches in fluid interface tracking

This section gives a short overview of the current interface tracking methods. As two main streams of interface tracking algorithms, volume of fluid method and level sets methods are elaborated in the following subsections. A list of pros and cons are provided in the overview as an comparison of each algorithm. Then we introduce the general concepts of marker functions and one-phase formulation of the multi-phase fluid problem, followed by more detailed discussion of each algorithm. As a supplementary material, we will cover the mathematical fundamental concepts of multi-fluid simulation in Appendix.

3.1 Overview

Generally, there are three interface tracking methods : Front Tracking Method (FT) [23], Level Set Method [15], and Volume of Fluid method(VOF) [8].

Front Tracking Method [23, 22, 6] advects the marked interface from an initial configuration and keeps the topology of the interface during the simulation. Therefore, this method is limited to topological changes in multiphase-fluid, such as merging or breaking of droplets. Thus we will not include it in our discussion.

Level Set Method was first introduced by Osher [15] in 1988. The material boundary or interface is defined as the zero set of [13, 14, 18] isocontour or isosurface of the given scalar field. Sethian [20] and Lakehal [9] applied the idea level set method into fluid simulation. In 2002, Enright, Fedkiw et. al. [4] have combined Lagrangian marker particles with LSM to obtain a maintain a smooth geometrical description of the fluid interface. However, it has been pointed out by Müller [10] and Garimella et. al. [5] that material volume is not well preserved in level set method, which is a main draw back of this method.

Volume of Fluid Methods [8] is one of best established interface volume tracking method currently in use [11, 19]. As name indicated, it keeps tracking of the volume of each fluid phase with a sub-volume. This method is therefore based on subcells or sub-volumes, and one tracks the volume percentage that one type of fluid takes up a sub volume cell.

Aside from the above three interface tracking algorithms, there also exist research directions for reconstruction of material interfaces. Reconstruction methods are mainly working on rebuilding a continuously interfaces out of discrete pieces or piecewise functions, while interface tracking algorithms focuses on tracking the dynamic behavior of the interface. Material reconstruction researches include *simple line interface* (SLIC) [12] and piecewise linear interface construction (PLIC) [16]. Recent development of discrete reconstruction algorithms can be found in [2, 1], and it is beyond the scope of our discussion in the survey.

3.2 One-phase formulation of the multi-fluid problem

A significant idea in dealing with multi-phase fluids is to use a one-phase formulation for the coupled systems.

Rather than looking at the coupled equation systems, one can reduce the complexity of computation by introducing a marker/indicator function χ.

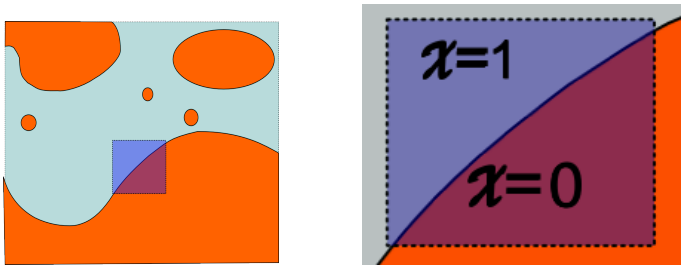
$$\chi = \begin{cases} 1 & \text{when it is in fluid 1} \\ 0 & \text{when it is in fluid 2} \end{cases} \tag{5}$$

As the name indicates, a marker function indicates the phase change by attaching a marker on one of the phases. Sometimes, markers can also be assigned to the exact interface between two materials, such as in level sets method. We will discuss that in the next subsection.

Therefore, instead of having two sets of variables for each phase of the fluid mixture (two phase formulation, see Appendix), a mark function marks the integration volume (i.e. when one look at a small volume element, 1b) by a characteristic function and conservation law for the two phase [9] results in only one :

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{6}$$

where $\rho = \rho_1 + (\rho_1 - \rho_2)\chi$.



(a) Two phase.

(b) One fluid formulation. Zoom in at an volume element

Figure 1 One and two phase formulation

3.3 Volume of Fluid

VOF method, originated by Hirt and Nichols [8] is a Eulerian method interface tracking method which fulfills the conservation of mass/volume. More importantly, topology change of the moving surfaces, such as merging and breakup of bubbles can be captured by VOF method.

The idea of VOF method is based on the one-fluid formulation. A fraction variable C is defined as the integral of the marker function χ in the controlled volume V:

$$C = \frac{1}{V}\int_V \chi(\vec{x},t)dV \tag{7}$$

Typically the controlled volume V is the computational cell volume.

A zero value of C indicates the cell is fully occupied by fluid one, and when $C = 1$, then the cell is fully filled with the other fluid. While $0 < C < 1$, the line/surface separating the two fluids goes across this cell.

Given a velocity field of V, the volume fraction function should fulfills the transportation theory

$$\frac{dC}{dt} + \vec{v}\cdot\nabla C = 0 \tag{8}$$

Volume tracking of VOF method do not only include a computation for volume frace at each time step, but also needs to approximate and reconstruct the interface. This method is accurate in the sense of volume preserving, but reconstruction of interface process makes it difficult to keep the topology of the interface.

3.4 Level Set Method

Osher and Fredkiw [14] have provided an overview of level set method and the most recent results. Enright et al. [4] use a fast first order accurate semi- Lagrangian method to improve the mass conservation. Similar to VOF method, LSM employs the idea of one-fluid simulation. The implicit material boundary/interface is given by the zero set of the scalar field ϕ:

$$\Gamma : \{(x,y,z)|\phi(x,y,z) = 0\}$$

$$\phi = \begin{cases} > 0 & \text{in fluid 1} \\ < 0 & \text{in fluid 2} \\ = 0 & \text{at boundary } \Gamma \end{cases} \tag{9}$$

Here ϕ is a similar marker function which is given in equation 5, the only difference is that $\phi = 0$ is exactly where the interface lies. The evolution of the zero sets satisfied the topology equation (8). Finding the initial zero sets normally involves extracting isocontour or isosurface at starting time. For a dataset with volume fraction information, one normally defines the zero set to be the isosurface of $\alpha = 0.5$.

Moreover, normal and curvature of the level set is given as [9, 13, 4]:

$$\vec{n} = \frac{\nabla\phi}{|\nabla\phi|} \tag{10}$$

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \tag{11}$$

Constraints on curvature should be applied when doing physically correct simulation. A general concern for curvature is the minimization of surface energy, which is also related to surface tension. However, this approach is more directed to free surface flows. In bubbly flow, e.g., mixture of two immiscible fluids, the topology change of interfaces contains more information.

Level set method is the most widely used method currently. It has an simple mathematical formulation and easy to solve. However, level set method has the draw back that volume is not always preserved while advecting the interface. Such disadvantages can be corrected by applying a volume correction after each numerical advection.

4 Numerical Methods for solving the governing equations

In computational fluid dynamics, there are two ways looking at a fluid property:

- Euler specification: In this way, a fixed time independent grid is used to compute the propagation of flow property. Metaphorically, the observer is sitting on the river bank, staring at a fix point in the river, and measuring the flow quantity at this fixed point as water flows by. This type of method is normally computationally expensive because to advect flow quantity, one has to update every grid point based on the previous quantity at this point and its four neighboring points.
- Lagrangian specification: A Lagrangian specification of flow dynamics is also known as the particle based specification. Compared to the previous one, now the observer rides on a single fluid particle, moves with the flow, and measures the change of flow quantities. Change of the flow properties are related to the so-called *material derivative*:

$$\frac{\mathrm{D}f}{\mathrm{D}t} = \frac{\partial f}{\partial t} + \vec{v} \cdot \nabla f$$

4.1 Grid-based method

One classical way of handling hyperbolic partial differential equations is to solve them numerically on a fixed grid points. These types of methods solves a partial differential equation in an Euler way. To solve the governing equations for both VOF and LS methods, one can apply the standard advection schemes, such as upwind scheme, central difference, Lax-Friedrich and Lax-Wendroff schemes. To avoid dissipation, a second order scheme with flux limiters can be considered. A large body of literature is available for Euler-type solvers, such as [14, 7, 8, 14, 13].

4.2 Particle-based method

In many simulation cases, volume fraction of second phase of fluid is given at a point cloud at each time frame. At each time slice, the point cloud was reinitialized, and there is correspondence between the points from one time slice to the next ones, see figure 2a. Therefore, conventional interpolation between two time slices would involve a resampling on a regular grid or project the points from one times step to the other. An intuitive way of interpolating the volume fraction in between two time steps is doing a linear combination between the two, such as:

$$f(t_0 + \Delta t) = \frac{t_0 - t_1 + \Delta t}{t_0 - t_1} f(t_0) + \frac{-\Delta t}{t_0 - t_1} f(t_1)$$

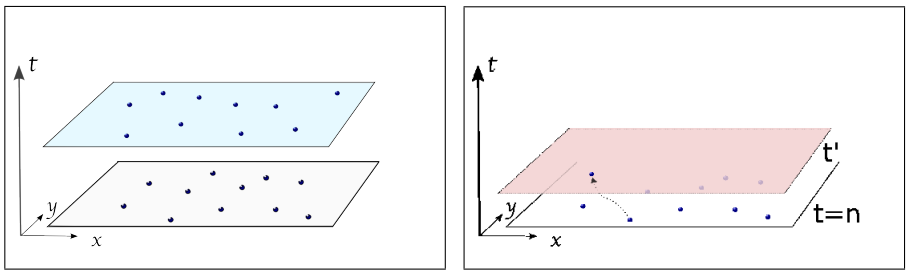
However, a linear blending gives us a physically incorrect interpolation resulting from the ignorance of flow direction. Displayed in figure 3a, the middle layer is a linear interpolation of the two slices. For a good interpolation, one would expect the high density region (marked as blue) moves a long the flow to the bottom time slice and then to the upper one. However, in the linear case, we can observe a non-smooth advection of these regions. What went wrong? We didn't consider flow directions. The solution to this problem is to use given velocity information and advect the flow property from one time step to another. To advect the flow property, we need to go back to the Navier-Stokes governing equation which described the motion of fluid:

$$\frac{\partial f}{\partial t} + \nabla \cdot (f\vec{v}) = 0 \tag{12}$$

where f is a fluid property such as mass or density. And Lagrangian formulation of equation (12) yields

$$\frac{\mathrm{D}f}{\mathrm{D}t} = -f\nabla \cdot \vec{v}. \tag{13}$$

where $-f\nabla \cdot \vec{v}$ is the advected term.



(a) Input point cloud **(b)** Lagrangian advection on a point

Figure 2 Original point cloud and Lagrangian methomaked

To solve the governing equation in a Lagrangian way, the following two steps are carried out:

- Advect a point along the velocity field (see Fig. 2b):

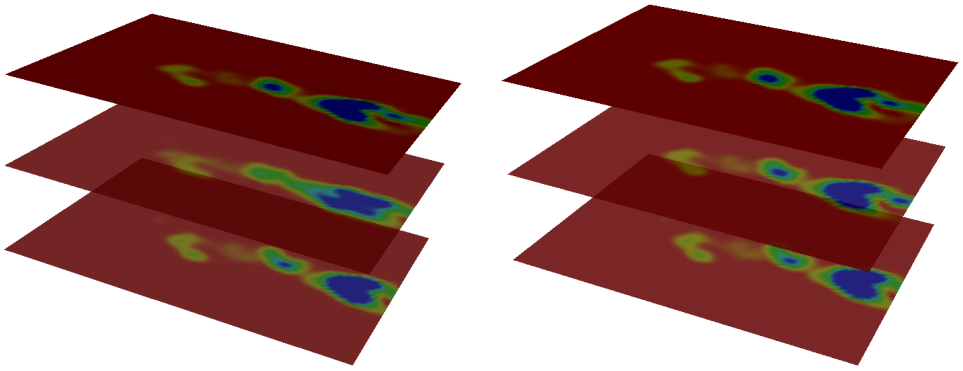
  $$\mathbf{x}^{n+1} = \mathbf{x}^n + \vec{v}\Delta t.$$

  For a higher order approximation or when then Δt is too large, one has to consider higher order scheme to advect the point, and adjust time step length when flow derivation is too large.

- Update the fluid property from equation (13)

$$f(\mathbf{x})^{t+1} = \frac{f(\mathbf{x})^t}{1 + \Delta t(\nabla \cdot \vec{v}^t)}$$

If convergence is desired, one can simply do a forward interpolation from t_n to t_{n+1}, and then perform a backward interpolation the other way around. A continuous result can be obtained by averaging the two. An example of Lagrangian interpolation is shown in figure 3b. Compare to Fig. 3a, one can observe that Lagrangian method gives us a smooth transition between time steps.



(a) Linear interpolation. Middle slice is a linear blending of the other two time slices.

(b) Linear interpolation. The side in the middle is interpolated using Lagrangian scheme.

4.3 Remarks

In both grid-based and particle based method, stability or convergence is an unneglegible issue. First order upwind scheme is unstable thus can cause unwanted numerical solution. In the case of particle based method, advection term of computational particles can be approximated by finite difference schemes. Moving particles around must be handle with care when velocity deviation is high. A common way of treating deviation problem is to apply an adaptive integration length.

5 Conlusion

In this paper, we presented a survey of the current interface tracking methods. The goal of interface tracking is to locate the interface separating fluid materials over time. Depending on the requirements of the interface tracking, such as continuity of volume preserving, some methods may be better suited than others. In this survey, we focus on a few current interface tracking methods that track the evolution of interface directly. Unlike the interface reconstruction methods, interface tracking methods track the interface propagation over time and require the knowledge of an initial stage.

We explore some key interface methods such as level set method which covers a wide range of interface-tracking problem. Depending on the mesh type of the given data, we also provide several corresponding numerical solvers to the underlying tracking equations. Thorough concepts and discussion can be found in further literatures such as [14, 15, 20].

Acknowledgements This work is supported by International Research and Training Group at University of Kaiserslautern (IRTG) and Deutsche Forschungsgemeinschaft (DFG , German research foundation).

References

1 John C. Anderson, Christoph Garth, Mark A. Duchaineau, and Kenneth I. Joy. Discrete Multi-Material Interface Reconstruction for Volume Fraction Data. *Comput. Graph. Forum*, 27(3):1015–1022, 2008.

2 John C. Anderson, Christoph Garth, Mark A. Duchaineau, and Ken Joy. Smooth, Volume-Accurate Material Interface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):802–812, 2010.

3 C. Drumm, S. Tiwari, J. Kuhnert, and H-J. Bart. Finite pointset method for simulation of the liquid-liquid flow field in an extractor. *Computers and Chemical Engineering*, 32(12):2946–2957, 2008.

4 D. Enright, R. P. Fedkiw, and Ferziger J. H. et al. A hybrid particle level set method for improved interface capturing. *J. Comput. Phys.*, 183:83–116, 2002.

5 R. V. Garimella and V. et. al. Dyadechko. *Interface Reconstruction in Multi-fluid, Multi-phase Flow Simulations.* pages 19–32, Springer, 2005.

6 O. Gloth, D. Hänel, L. Tran, and R. Vilsmeier. A front tracking method on unstructured grids. *Computers & Fluids*, 32(4):547–570, 2003.

7 Edwige Godlewski and Pierre-Arnaud Raviart. *Numerical approximation of hyperbolic systems of conservation laws.* Springer-Verlag New York, 1996.

8 C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–225, 1981.

9 D. Lakehal, M. Meier, and M. Fulgosi. Interface tracking towards the direct simulation of heat and mass transfer in multiphase flows. *International Journal of Heat and Fluid Flow*, 23:242–257, 2002.

10 Matthias Müller. Fast and robust tracking of fluid surfaces. In *SCA '09: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 237–245, New York, NY, USA, 2009. ACM.

11 M. Marek, W. Aniszewski, and A. Boguslawski. Simplified volume of fluid method for two-phase flows. *Task Quarterly*, 3:255–265, 2008.

12 W. Noh and P. Woodward. Simple line interface method. In *A. van de Vooren and P. Zandbergen, editors, Proceedings of the 5th international conference on fluid dynamics, Lecture notes in Physics*, volume 59, 1976.

13 S. Osher and R. P. Fedkiw. Level Set Methods: An Overview and Some Recent Results. *Journal of Computational Physics*, 169(2):463–502, May 2001.

14 S. J. Osher and R. P. Fedkiw. *Level Set Methods and Dynamic Implicit surfaces.* Springer, 2003.

15 S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

16 W. J. Rider and D. B. Kothe. Reconstructing volume tracking. *Journal of computational physics*, 141(2), 1998.

17 R. Rieger, C. Weiss, G. Wigley, H. J. Bart, and R. Marr. Investigating the process of liquid-liquid extraction by means of computational fluid dynamics. *Computers and Chemical Engineering*, 20(12):1467–1475, 1996.

18 J. A. Sethian. Curvature and the Evolution of Fronts. *Commun. Math. Phys.*, 101:487–499, 1985.

19 M. Sussman and E. G. Puckett. A coupled level set and volume-of-fluid method for computing 3D and Axisymmetric incompressible two-phase flows. *Journal of Computational Physics*, 162:301–337, 2000.

20 J. A. Sethian and P. Smereka. Level set methods for fluid interfaces. *Annu. Rev. Fluid Mech.*, 35:341–372, 2003.

21 S. Tiwari and J. Kuhnert. Modeling of two-phase flows with surface tension by finite pointset method (FPM). *J. Comput. Appl. Math.*, 203(2):376–386, 2007.

22 Hiroshi Terashima and Gretar Tryggvason. A front-tracking/ghost-fluid method for fluid interfaces in compressible flows. *Journal of Computational Physics*, 228(11):4012–4037, 2009.

23 S. O. Unverdi and G. Tryggvason. A front tracking method for viscous Incompressible multi-fluid flows. *Journal of Computational Physics*, 100(1):25–37, 1992.

24 F. Wang and Z. S. Mao. Numerical and experimental investigation of liquid-liquid two-phase flow in stirred tanks. *Industrial and Engineering chemistry research*, 44(15):5776–5787, 2005.

Survey on Benchmarks for a GPU Based Multi Camera Stereo Matching Algorithm

Klaus Denker[1] and Georg Umlauf[1]

1 **Computer Graphics Lab, Department of Computer Science**
University of Applied Sciences Konstanz
Brauneggerstr. 55, 78462 Konstanz, Germany
kdenker, umlauf@htwg-konstanz.de

Abstract

Stereo matching algorithms and multi camera reconstruction algorithms are usually compared using benchmarks. These benchmarks compare the quality of the resulting depth map or reconstructed surface mesh. We describe the differences between several known stereo and multi-view stereo benchmarks and their various datasets. Also the modifications that are necessary to use our own GPU based multi camera stereo matching algorithm with the data from these benchmarks are discussed.

Keywords and phrases Stereo matching, Multi camera, GPU

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.20

1 Introduction

Stereo matching is a technique to compute from two or more 2D camera images a depth map of the captured object. It is used in many applications ranging from remote sensing to robotics, archeology, cultural heritage, reverse engineering, and 3D face recognition [3, 6, 7, 15]. Its appeal comes from the fact that it is the only passive method to generate depth information. This means only natural light is used for the data acquisition and there is no interaction with the object that might harm the object.

Benchmarks in this field allow the comparison of the quality of stereo matching algorithms. As the applications of stereo matching algorithms differ, there are multiple benchmarks using diverse test datasets. They range from 2 to 363 images, taken from parallel, spherical, or arbitrary camera positions. They also use different result representations, depth maps or meshes, and different comparison methods [8, 11, 13].

Our GPU based multi camera matching [1] is tailored for fast face recognition. It does not fit well to any of the intended benchmark scenarios. This either causes bad results or adaptations of the algorithm to process the benchmark datasets and result data.

We describe the benchmarks in Section 2, our matching algorithm in Section 3, and the application to the benchmarks in Sections 4 and 5.

2 Benchmark survey

2.1 Middlebury stereo benchmark

One method to compare the quality of stereo matching algorithms is presented in [8]. Several pairs of photographs of carefully constructed test scenes are used. All stereo matching algorithms are compared to hand-labeled ground truth disparity maps.

The test photographs have a low resolution of 384×288 or 435×383 pixels. Each image pair differs only by a small horizontal displacement of the camera. The viewing directions

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 20–26

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of both images are exactly parallel. In the used test scenes most objects have flat surfaces parallel to the image planes of the cameras. This flat structure of the whole scene is necessary to create the ground truth disparity maps. They are build by hand-labeling the contours of all flat regions and computing an alignment for the whole region. In the disparity map the whole region is marked with the disparity of this alignment. Example images and the corresponding disparity map are shown in Figure 3.

Tested stereo matching algorithms have to provide a disparity map. This disparity map is compared with the ground truth disparity map based on the root mean square error. Additionally, the percentage of bad pixels with an error above a threshold is computed. Theses scores are computed for the whole image as well as for certain interesting regions. These regions are texture-less, occluded, and have depth discontinuities.

For algorithms using more than two images a set of extended datasets was generated using the method described in [9]. The extended datasets contain up to nine high resolution photographs (1800×1500 pixels) of constructed test scenes. As for the stereo pairs, the camera is moved horizontally and all viewing directions are parallel. The ground truth is recorded using a structured light approach. A projector projects multiple Gray-code stripe patterns on the scene. The photographs of these patterns are used to compute the exact disparity of each pixel in two reference images.

The official benchmark on the website [10] contains two sets from [8] and two reduced sets from [9]. The reduced sets consist of two images and two ground truth disparity images. Their resolution is reduced to 450×375 pixels.

2.2 Dense multi view stereo test images

The benchmark approach of [13] uses multiple photographs from quite arbitrary camera positions. Outdoor scenes with historic buildings are used as test datasets. Different image sets on the benchmark website contain between 6 and 30 images of the facades of different historic buildings. All pictures are taken using a digital camera without any special positioning device. A laser scanner generated the ground truth geometric data. Two example images and the corresponding depth map are shown in Figure 1 (left).

Reconstruction of the internal and external camera parameters is part of the benchmark. Thus, this benchmark can be used without camera parameters, with the internal camera parameters, or with all camera parameters. The ground truth camera parameters are reconstructed from feature markers in the scene using a maximum likelihood estimation.

Tested algorithms provide a depth map for each photograph. This depth map is compared to a reference depth map created from the laser scan using the ground truth camera parameters. An error value is computed by dividing each pixel's absolute difference by the local variance of the ground truth laser scan depth map. The benchmark results are presented as a histogram of the different error sizes on the website [12].

2.3 Middlebury multi view benchmark

The Middlebury multi view stereo benchmark described in [11] uses much more data. Pictures from 790 positions on a half-sphere are taken using the *Stanford spherical gantry* [4]. The spherical gantry uses two computer-controlled arms, one carries a camera, the other a light source. An object is placed in the center between the arms on a turn table. The spherical gantry has a precision of $0.01°$ and the captured images have a resolution of 640×480 pixels. A ground truth mesh G is created from multiple laser scans that are merged using a 3D grid.

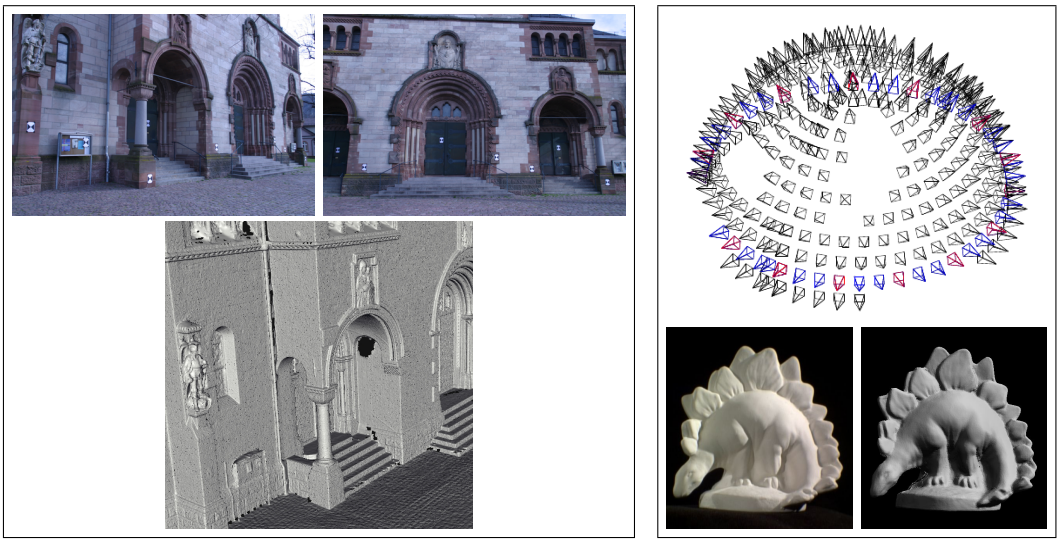


Figure 1 Dataset from *Dense multi view stereo testimages* (left) [13]. Hemisphere of camera positions, photograph, and result mesh from *Middlebury multi view benchmark* (right) [11].

Three different datasets of each object can be used for benchmarking. The *full* datasets contains a hemisphere of 317 to 363 images without shadows of the arms, see Figure 1 (right). Subsets of these datasets are the *ring* datasets that contain only one ring of 47 to 48 images around the object. The Further reduced *sparse ring* datasets contain 16 images on the same ring equally distributed around the object.

Each algorithm using this benchmark has to provide a mesh reconstruction R of the object. The benchmark measures the nearest distance between each point of R to G and of each point of G to R. The latter is only used to evaluate the completeness of R.

3 Our multi camera stereo matching

3.1 The camera system

For our system we use four USB Logitech® cameras QuickCam® Pro 9000. These cameras run at a resolution of 960×720 at five frames per second, because they are connected via two USB 2.0 Controllers. The USB-bandwidth limits the camera resolution, which could yield a much higher resolution.

To get the best results we use a Y-constellation of the four cameras as shown in Figure 2a. This arrangement has two advantages over e.g. linear configurations. First, the image of the central camera is used as reference image for matching and texturing. Second, we get a different angle between each camera pair. A camera constellation with a preferred direction could deteriorate the detection of features along this direction, e.g. horizontal stripes in the image cause problems in a horizontal camera constellation.

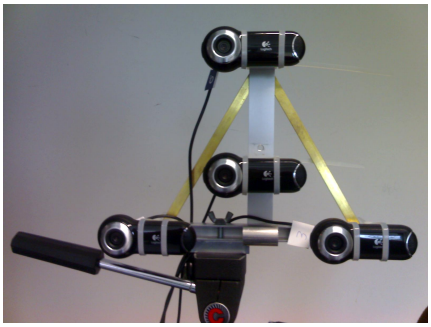
Independently of the used hardware system, our method can be adapted to arbitrary camera constellations. This adaption is easier for camera systems where all cameras are mounted on a plane with a viewing direction perpendicular to this plane. For details see [1].

3.2 Matching

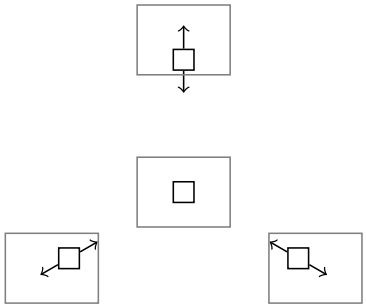
Stereo matching is an operation to find similar regions in two images. We call these square regions *matching windows*. To find the best matching position for these matching windows they will be moved over the images. Thus, the fundamental operation of stereo matching is the comparison of matching windows of the same size. To this end, each matching window position gets a score that evaluates the similarity of two matching windows. We use a *weighted normalized cross-correlation* similar to [5].

To get the depth information, the disparity between the images is estimated. For this the matching windows are moved simultaneously over all images to compute the total score of each position. So, the highest total score yields the best position of the matching windows.

To evaluate all possible camera positions is too expansive. Thus, the centers of the matching windows are moved only along epipolar lines with respect to the center point of the fixed matching window of the reference image, see Figure 2b.



(a) For our experiments we use a system of four USB cameras arranged in an upside down Y-constellation.



(b) Moving the matching windows (black squares) in all images (gray squares) along epipolar lines (arrows) simultaneously.

Figure 2 The camera configuration and simultaneous movement of the matching windows.

The step size for the movements is one pixel and per matching window in the reference image we test $3 \leq k \leq 9$ different positions. Due to the movement along the epipolar lines the color values for the score computations are bi-linearly interpolated. The matching window position with the highest score s_{best} then yields the best similarity of the matching windows. The number of movements gives the disparity d_{best} of the best match. From the disparity and the camera positions the depth is computed by reverse projection.

The process described so far gives the disparity information for one image at a fixed resolution. For large disparities this is computationally expensive, because many possible matching window positions must be evaluated. A multi-level approach can reduce this effort and allows for large disparities. Our approach is based on [16], where a matching pyramid with coupled levels is used. In our approach the different levels are completely independent and high level information is re-used to get a faster low level disparity computation. The camera image is stored in a mip-map on the GPU at seven different resolutions. Each resolution has half the horizontal and vertical resolution of the one before. The matching windows have a fixed size of e.g. 7×7 pixels. Starting with the image on the coarsest resolution level the disparities of all pixels in the coarsest reference image are computed. The matching windows are tested at $k = 9$ different positions. Then the image resolution is doubled and the same process starts again, while k is reduced linearly. The bi-linearly

interpolated disparities of the coarser level is used as starting position for the lower level matching windows. Thus, the low level matching windows move a fixed number of pixels around the best position of the previous level.

3.3 GPU implementation

Because our method mainly works on images and generates a depth map as an image, we use GLSL fragment shaders for our implementation. Fragment shaders are small programs, running highly parallel on the GPU, processing one or multiple texture images into one result image. For the complex shader operations required by our approach GPUs which support at least shader model 4.0 are necessary. Older GPUs are not able to perform enough computations in a single shader run.

The quite complex computations during stereo matching require the combination of multiple shader runs. Each step in our multi-level matching requires the run of three fragment shader programs. Two of them pre-calculate the average and auto-correlation data for the normalized cross-correlation. The third shader performs all matching operations, i.e. it moves the deformed matching windows, computes the total score, and finds the best score.

4 Results

4.1 Middlebury stereo benchmark

Our multi camera stereo matching algorithm is tailored for a four camera system. But it can be configured to any other planar camera configuration, where the viewing directions of all cameras are parallel.

The standard image set of the Middlebury benchmark *Tsukuba* consists of two images with small disparities. Our algorithm is optimized for much larger disparities. Therefore, the resulting depth map in Figure 3 is quite inaccurate, especially in the background regions.

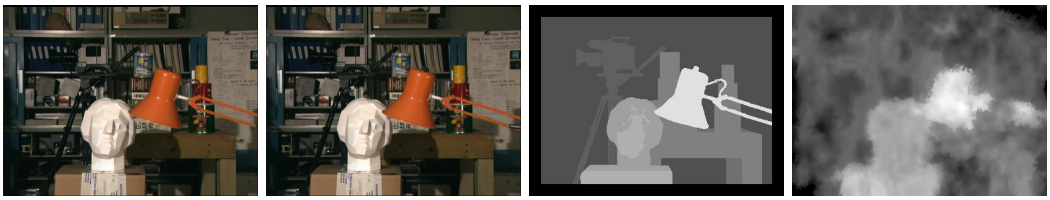


Figure 3 The *Tsukuba* dataset from the Middlebury stereo benchmark [10]. The two source images, reference disparity map, and our resulting depth map.

Much better results are possible using the extended dataset of the same scene. As described in Section 2.1 this dataset consists of five high resolution images. We scale these images to the resolution of our camera system of 960×720 pixels. Figure 4 shows a depth map generated from this image set. The quality is much better than in the previous example, only by using more pictures with better resolution for our simultaneous matching.

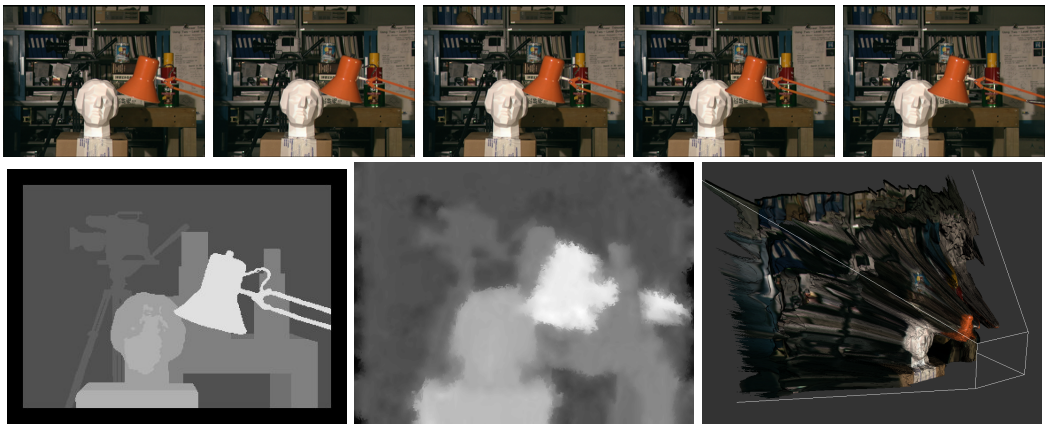


Figure 4 The extended *Tsukuba* dataset from the Middlebury stereo benchmark [10]. The five source images, reference depth map, our resulting depth map, and a 3D rendering of the result.

5 Necessary modifications

5.1 Dense multi view stereo test images

The camera positions of the pictures in the benchmark sets of [13] are quite arbitrary. They are not planar and the viewing directions are not parallel. Many of the simplifications of our matching shaders are not possible for arbitrary camera positions and viewing directions. Additional projection and re-projection calculations are necessary. This would affect the calculation speed and increase the complexity of the used GPU shaders.

Our multi camera stereo matching shaders support up to five images. The shaders can be modified to support larger image counts as long as the GPU has enough memory. At least the smaller image sets of this benchmark, that contain up to eleven pictures, can be used in a single matching step. Merging operations for multiple depth maps are not necessary.

5.2 Middlebury multi view benchmark

The Middlebury multi view benchmark uses a large number of camera images. Here, it is not possible to do simultaneous depth map calculation. So, a method to combine multiple depth maps and generate a high quality mesh of the object must be developed.

To use our algorithms subsets of the camera positions need to be chosen to generate multiple local depth maps. A cross-shaped set of five cameras with one in the center seems ideal. If these cameras are relatively close, this subset is almost planar and has almost parallel viewing directions. This yields a relatively small disparity between the cameras of a local subset, reducing the quality of the matching and complicating the merging of the depth maps. By choosing images that are far away from the center image, this can be improved. Thus, there is the trade-off between parallelism of the viewing directions and the magnitude of the disparities.

This approach generates a lot of depth maps from around the captured hemisphere of viewpoints. All of these depth maps have to be merged and converted to a single object mesh. Common methods for this problem use a volume or point cloud representation of the object and do a mesh reconstruction [2, 14]. All these methods have in common, that they are very slow compared to our fast multi camera stereo matching, and are not applicable to

our application setting.

Acknowledgements

This work was supported by DFG GK 1131 and AiF ZIM Project KF 2372101SS9.

References

1 K. Denker and G. Umlauf. Accurate real-time multi-camera stereo-matching on the gpu for 3d reconstruction. Accepted: Journal of WSCG, 2011.

2 Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *Proceedings of IEEE CVPR 2007*, pages 1–8, 2007.

3 R. Koch, M. Pollefeys, and L. Van Gool. Realistic 3-d scene modeling from uncalibrated image sequences. In *ICIP'99, Kobe: Japan*, pages 500–504, 1999.

4 M. Levoy. The stanford spherical gantry. `http://graphics.stanford.edu/projects/gantry/`, 2002.

5 J. P. Lewis. Fast template matching. In *Vision Interface*, pages 120–123, 1995.

6 D. Murray and J. Little. Using real-time stereo vision for mobile robot navigation. In *Autonomous Robots*, pages 161–171, 2000.

7 D.T. Pham and L.C. Hieu. Reverse engineering - hardware and software. In V. Raja and K.J. Fernandes, editors, *Reverse Engineering - An Industrial Perspective*, pages 33–30. Springer, 2008.

8 D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1-3):7–42, 2002.

9 D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Proceedings of IEEE CVPR 2003*, pages 195–202, 2003.

10 D. Scharstein and R. Szeliski. Middlebury stereo vision page. `http://vision.middlebury.edu/stereo/`, 2007.

11 S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of IEEE CVPR 2006*, pages 519–528, 2006.

12 C. Strecha. Multi-view stereo test images. `http://cvlab.epfl.ch/~strecha/multiview/`, 2008.

13 C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proceedings of IEEE CVPR 2008*, pages 1–8, 2008.

14 G. Vogiatzis, P. H. S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *Proceedings of IEEE CVPR 2005*, pages 391–398, 2005.

15 F. Voltolini, S. El-Hakim, F. Remondino, and L. Gonzo. Effective high resolution 3d geometric reconstruction of heritage and archaeological sites from images. In *Proceedings of the 35th CAA Conference*, pages 43–50, 2007.

16 R. Yang and M. Pollefeys. A versatile stereo implementation on commodity graphics hardware. *Real-Time Imaging*, 11(1):7–18, 2005.

A Survey of Interaction Techniques and Devices for Large High Resolution Displays

Taimur K. Khan[1]

1 **University of Kaiserslautern**
Kaiserslautern, Germany
tkhan@informatik.uni-kl.de

Abstract

Innovations in large high-resolution wall-sized displays have been yielding benefits to visualizations in industry and academia, leading to a rapidly growing increase of their implementations. In scenarios such as these, the displayed visual information tends to be larger than the users field of view, hence the necessity to move away from traditional interaction methods towards more suitable interaction devices and techniques. This paper aspires to explore the state-of-the-art with respect to such technologies for large high-resolution displays.

Keywords and phrases Interaction Techniques, High Resolution Displays

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.27

1 Introduction

In numerous industrial and academic endeavors either a projector or tiled large high-resolution display is preferred over a standard desktop in order to cater to a larger audience, for more efficient collaborative work, to further immerse the client in virtual reality applications, or to facilitate visualization of large and complex datasets by maintaining both overview and detail views simultaneously [21]. In these situations the major drawback of using traditional input mechanisms is the stationary installation of such devices - an individual would be required to sit at a console and follow instructions from others as how to interact with the environment. Other flaws associated with traditional devices include their lack of natural interactions and intuitiveness.

Through the course of this paper we shall survey both cutting-edge and established interaction technologies that deal with the above-mentioned issues. The common theme in these tools is the increased usability of interaction devices when integrating them with large high-resolution displays in a more natural manner than traditional setups.

2 Related Work

In order to interact with large high-resolution displays, researchers have mostly focused on: interaction techniques [2, 23, 20], a particular interaction device [52, 49, 55], or certain modalities [38, 48, 25].

There are not many papers that specifically compare interaction technologies for large high-resolution displays. In this regards, the presented survey is quite similar to the one conducted by Bierz in which he examined interaction metaphors, devices, and techniques for large and immersive displays [8]. The distinction lies in the approach of this survey to explore different modalities for relevant interaction technologies. Further, the scope has been extended to include recent publications.

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 27–35

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

3 Interactive System Components

Extensive studies have been conducted in the realm of high-resolution displays for desktops [29] as well as for large displays [19, 46, 5, 29]. In both contexts, a key benefit is the enhanced performance time for basic visualization tasks as documented by Ball et al. [4] - *high resolution displays help people find and compare targets faster, feel less frustration, and have more of a sense of confidence about their responses.*

However, large high-resolution displays stand out more so than desktops in their ability to not only immerse the user more deeply in a virtual environment, but also to efficiently explore and interact with complex datasets [56]. While implementing an interactive system, a number of factors have to be considered:

- Type of display technology: CAVE [17], Multi-Monitor Desktop [3], Tiled LCD Panels [43], Projector Arrays [50], or Stereoscopic Displays [18]
- Architecture and Data Distribution [21]
- User Interfaces and Interaction Devices

It is this latter criterion that shall be examined further in this paper, where distinct modality types are to be delved into. Relevant media and interaction techniques within each modality will be inspected by analyzing between two and three applicable implementations.

4 Modalities

In [30] modality is described as a means to convey information to, or receive information from a communication partner (human or machine). In the following sections, pertinent media and interaction techniques are further investigated.

4.1 Speech

The notion of implementing speech recognition and synthesis to issue digital commands is definitely not a novel concept. Yet supporters of multimodal interfaces will point out to its significance in modern-day environments, where the combination of speech and gestures is more natural and efficient [48].

An early implementation, Put-That-There multimodal system, allowed individuals to interact with large displays via deictic referenced speech commands [10]. More recent research asserts multimodal input as a shorter, simpler and more fluent modality than speech alone [41]. In this study, the authors considered speech and gestures in map-based user tasks and verified that the participants not only preferred this symbiosis but also that it was more efficient than speech alone.

Results from several research projects indicate the real benefit of speech technology is when it is merged with direct input or gestures to form a multimodal input device for selection and manipulation tasks [14, 15, 32, 16]. As the broad context of these studies suggests, a number of environments can benefit from a multimodal approach that utilizes speech including large high-resolution displays.

4.2 Tracking

Early implementations employed one of two main technologies that provided sufficient update rates, accuracy, and mobility: magnetic and optical tracking systems [8]. However, the latter was typically preferred due to its lower latency time and as magnetic trackers tend

to be susceptible to ferromagnetic substances [40]. This section focuses solely on tracking interaction devices, whereas gestures and user tracking are inspected further in Section 4.3.

In [52], the authors suggest a technique to simultaneously track and identify multiple laser pointers in a large screen collaborative environment. The crux of this method lies in attaching a microprocessor to each laser pointer to modulate its brightness. A camera-based optical system is then used to process images that detect, group, and identify the laser points. Resulting identification and location is fed back to the application in order to update each pointers position.

Robertson et al. preferred motion-sensing Wii Remote controllers to touch screen or camera-based trackers in their multiuser collaborative environment [44]. Their results indicate that these controllers can proficiently facilitate several users to cooperate in controlling the virtual camera and to explore graphical objects. The three-axis accelerometer and infrared fiducial tracking camera of the Wii was used to report position and orientation.

Mobile phones or so-called smart phones may be employed as pointing devices if they are equipped with either an accelerometer or a camera. While the former is generally found in some high-end phones, cameras are packaged in almost all mobile phones. Work similar to the one by Haro et al. utilizes tracking algorithms to analyze live input images. In their research, a feature-based tracking algorithm was used to estimate the motion of the device [24]. This sort of tracking algorithm can be used in navigation and gesture-based interaction in virtual and augmented realities [11].

A promising application area is the interaction of mobile devices with large screen displays in public domains, one which is being researched thoroughly [23, 13, 6]. Hardy et al. postulate a technique, *Touch & Interact*, that facilitates the direct manipulation of dynamic displays using touch-based mobile interactions. Their results indicate an easy, intuitive, and enjoyable mechanism that is significantly faster than the alternative - usage of the mobile phone to control a mouse pointer remotely. Vajk et al. employ mobile phones as controllers for games (or applications) running on large public displays. Their work aspires to provide a generic framework that utilizes on-board phone sensors such as cameras, accelerometers, and Radio Frequency IDentification (RFID)/Near Field Communications(NFC) to run games and simulations on large public displays via Bluetooth [49].

As inspected in the above-mentioned research, it is common practice to employ tracking technologies to navigate virtual and augmented environments efficiently. Depending on the nature of the environment, one may choose to track interaction devices, the user, or a combination of both. Devices may be tracked in collaborative environments to distinguish users. In other scenarios, trackers may be implemented to gauge head, hand, or eye motion; allowing the adjustment of the perspective projection and auditory inputs to provide further realism.

4.3 Gestures

Interfaces that utilize gestures offer a natural way of interaction, one which is more accurate than the use of tracking devices and which is extremely intuitive. Of the various media employed, mobile phones, hand gestures, and the Wii Remote are particularly of interest due to their extensive usage in everyday life.

4.3.1 Mobile Phones

Innovative interaction techniques are required to offset the limitations posed by input and output modes of mobile devices [22]. One such technique is the usage of gestures as an interaction mode, empowering people to explore and achieve a task in a more instinctive manner.

Bhandari et al. conducted participatory interviews to establish the acceptance of gestures as an interaction mode for mobile devices like mobile phones [7]. Participants were asked to perform tasks using gestures that included *touching of the screen with fingers, movement of the fingers across the screen, and moving the phone in several directions.* A key observation was that the participants clearly favored gestures over traditional key-clicking modes of interaction. Relevant metaphors were indicated, such as: *unfolding a blanket, scraping off the dirt from surface, opening up a picture to zoom in, and crossing out incorrect items or information.*

Similarly, Dachselt et al. advocate a set of gestures that utilizes accelerometer-enabled mobile phones both as a remote control and as a conduit to transfer documents to and from large display interfaces [20]. The former is explored through two distinct application scenarios: 1) *discrete directional tilt* gestures are mapped to stepwise panning, up and down, and zooming in or out of a User Interface for browsing large music collections, and 2) *continuous tilt* gestures that employ pan, zoom, and tilt modes that are applied on 3D maps. The latter employs *throw* and *fetch-back* gestures to transfer items between a mobile device and a large display.

It is concluded that such interactive gestures lead to a natural and seamless integration of mobile devices and large displays.

4.3.2 Hand Gestures

Both hand gestures and postures are intrinsic components of our everyday lives giving us conscious and unconscious cues for non-verbal communication. Recent research aims to apply these cues in their most natural form, without the use of props, as a medium for interaction.

Work presented in [27] clearly categorizes hand movement vision techniques into two aspects: *static and dynamic*, referring to hand postures and hand gestures respectively. A technique based on the Modified Census Transform to extract relevant features in images is explored for Hand Posture Recognition [28] while existing models such as Hidden Markov Models and Input-Output Hidden Markov Models are compared for Hand Gesture Recognition.

Researchers at the University of Coimbra, Portugal, were one of the early pioneers in using computer vision to decipher human computer interfaces based on hand gestures [42]. Benefits associated with vision techniques include the usage of only cameras to capture hand related movements rather than sensors or devices. The key motivation of this research was to develop a natural interface based on the recognition of a set of hand gestures without computational latency. Analysis of the temporal variation of the hand contour was the basic premise of this recognition system, an approach that has been since extended to newer interface paradigms [1].

The ability to interact in real-time has traditionally been the bottleneck in realizing optical marker-less tracking. This concern has been tackled in the above-mentioned techniques and further resolved by delegating time-consuming computation to the graphics processing unit (GPU). Research akin to Bierz et al. has focused on performing image processing such as skin detection, noise filtration, and outline extraction on the GPU [9].

4.3.3 Wii Remote

Several aspects of the Wii Remote have been successfully reverse-engineered [35], resulting in software libraries [53, 34] and documented technical information about its inner workings [54]. Leading researchers to adopt it for a variety of purposes, such as: motion capture (see Section 4.2), gesture based applications [45], and robot control [33].

4.3.4 Other Technologies

Facial expressions, gaze tracking, and body postures tend not be favored in fine-grain manipulation due to their coarse nature, however they can be efficient in rough initial interactions and in augmenting other interaction mechanisms [8]. Further details can be found in [38, 47, 51].

4.4 Haptics

A brief mentioning of haptic input devices is imperative due to their potential in virtual environment applications. These devices are quite common in their simplest form and can be found in the vibrating alert function of mobile phones as well as in gaming force feedback devices. More sophisticated versions interpolate mass or stiffness of the objects and collisions with those objects to force output [30]. The biggest setback with the latter is that most of them are static grounded devices lacking the portability and freedom of movement to be used in combination with large screens.

Haptic devices have been successfully used in many different application areas such as training of medical students [31, 36], modeling objects in virtual reality [12, 37], or operation of tele-operating systems [39].

5 Conclusion

Careful inspection of the "state-of-the-art" in interactions with large high-resolution displays indicates that researchers have indeed freed users from their desks; empowering them with multimodal mechanisms that may be a combination of gestures, speech, tracking, and hand-held devices. A common theme emerges as a result - an approach that moves towards multimodal environments that are both natural and "fun".

Natural modalities such as speech and touch that compose easier and more effective interactions with applications and services are being adopted [26]. Whereas hand-held devices in earlier interface scenarios are being replaced in academia and industry by devices analogous to game controllers and smart phones. Researchers such as Iftode et al. indicate that smart phones are destined for universal acceptance due to their Bluetooth capabilities, Internet connectivity, significant processing power and combination of modality modes.

It is for these reasons that smart phones have a high potential to be used as remote controls and dual connectivity devices in large display interactions. Further, it is envisioned that the recent development of tablet devices such as the Apple iPad will have a larger impact on such interactions - as they combine the features of smart phones with a larger multitouch finger-sensitive touchscreen interface.

6 Acknowledgement

I would like to thank members of the International Research Training Group (IRTG) for their cooperation, especially Peter-Scott Olech. The IRTG is supported by the German

Research Foundation (DFG) under contract DFG GK 1131.

References

1 T. Bader, R. Rapple, and J. Beyerer. Fast invariant contour-based classification of hand symbols for hci. In *CAIP09*, pages 689–696, 2009.

2 Brian Badillo, Doug A. Bowman, William McConnel, Tao Ni, and Mara G. Silva. Literature survey on interaction techniques for large displays. In *Technical Report TR-06-21*, Computer Science, Virginia Tech, 2006.

3 Robert Ball and Chris North. An analysis of user behavior on high-resolution tiled displays. In *In Interact 2005 Tenth IFIP TC13 International Conference on Human-Computer Interaction*, pages 350–363. Springer, 2005.

4 Robert Ball and Chris North. Effects of tiled high-resolution display on basic visualization and navigation tasks. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1196–1199, New York, NY, USA, 2005. ACM.

5 Robert Ball and Chris North. Visual analytics: Realizing embodied interaction for visual analytics through large displays. *Comput. Graph.*, 31(3):380–400, 2007.

6 Rafael Ballagas, Michael Rohs, and Jennifer G. Sheridan. Sweep and point and shoot: phonecam-based interactions for large public displays. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1200–1203, New York, NY, USA, 2005. ACM.

7 Shruti Bhandari and Youn-Kyung Lim. Exploring gestural mode of interaction with mobile phones. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 2979–2984, New York, NY, USA, 2008. ACM.

8 Torsten Bierz. Interaction technologies for large displays - an overview. *Visualization of Large and Unstructured Data Sets, GI-Edition Lecture Notes in Informatics (LNI)*, S-4, 2006.

9 Torsten Bierz, Achim Ebert, and Jörg Meyer. Gpu accelerated gesture detection for real time interaction. In *Visualization of Large and Unstructured Data Sets*, pages 64–75, 2007.

10 Richard A. Bolt. "put-that-there": Voice and gesture at the graphics interface. In *SIGGRAPH '80: Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, pages 262–270, New York, NY, USA, 1980. ACM.

11 Tolga Capin, Antonio Haro, Vidya Setlur, and Stephen Wilkinson. Camera-based virtual environment interaction on mobile devices. In *Lecture Notes in Computer Science 4263, 765*, page 773, 2006.

12 Hui Chen and Hanqiu Sun. Real-time haptic sculpting in virtual volume space. In *VRST '02: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 81–88, New York, NY, USA, 2002. ACM.

13 Keith Cheverst, Alan Dix, Daniel Fitton, Chris Kray, Mark Rouncefield, Corina Sas, George Saslis-Lagoudakis, and Jennifer G. Sheridan. Exploring bluetooth based mobile phone interaction with the hermes photo display. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 47–54, New York, NY, USA, 2005. ACM.

14 Philip Cohen, David McGee, Sharon Oviatt, Lizhong Wu, Joshua Clow, Robert King, Simon Julier, and Lawrence Rosenblum. Multimodal interaction for 2d and 3d environments. *IEEE Comput. Graph. Appl.*, 19(4):10–13, 1999.

15 Andrea Corradini and Philip Cohen. Multimodal speech-gesture interface for hands-free painting on virtual paper using partial recurrent neural networks for gesture recognition. In *Proc. of the Int'l Joint Conf. on Neural Networks (IJCNN)*, volume III, pages 2293–2298, New York, NY, USA, 2002. ACM.

16 Andrea Corradini and Christer Samuelsson. A generic spoken dialogue manager applied to an interactive 2d game. In *PIT '08: Proceedings of the 4th IEEE tutorial and research workshop on Perception and Interactive Technologies for Speech-Based Systems*, pages 2–13, Berlin, Heidelberg, 2008. Springer-Verlag.

17 Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: the design and implementation of the cave. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 135–142, New York, NY, USA, 1993. ACM.

18 Cyviz. `http://www.cyviz.com`; Online; Accessed 20-April-2010.

19 Mary Czerwinski, Greg Smith, Tim Regan, Brian Meyers, George Robertson, and Gary Starkweather. Toward characterizing the productivity benefits of very large displays. In *PROC. INTERACT*, pages 9–16. Press, 2003.

20 Raimund Dachselt and Robert Buchholz. Natural throw and tilt interaction between mobile phones and distant displays. In *CHI '09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 3253–3258, New York, NY, USA, 2009. ACM.

21 Matthias Deller, Sebastian Thelen, Daniel Steffen, Peter-Scott Olech, Achim Ebert, Jan Malburg, and Jörg Meyer. A Highly Scalable Rendering Framework for Arbitrary Display and Display-in-Display Configurations. In Hamid R. Arabnia and Leonidas Deligiannidis, editors, *CGVR*, pages 164–170. CSREA Press, 2009.

22 Assaf Feldman, Emmanuel Munguia Tapia, Sajid Sadi, Pattie Maes, and Chris Schmandt. Reachmedia: On-the-move interaction with everyday objects. *Wearable Computers, IEEE International Symposium*, 0:52–59, 2005.

23 Robert Hardy and Enrico Rukzio. Touch & interact: touch-based interaction of mobile phones with displays. In *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, pages 245–254, New York, NY, USA, 2008. ACM.

24 Antonio Haro, Koichi Mori, Tolga Capin, and Stephen Wilkinson. Mobile camera-based user interaction. In *In ICCV 2005 Workshop on HCI*, pages 79–89, 2005.

25 Seokhee Jeon, Jane Hwang, Gerard J. Kim, and Mark Billinghurst. Interaction techniques in large display environments using hand-held devices. In *VRST '06: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 100–103, New York, NY, USA, 2006. ACM.

26 Michael Johnston. Building multimodal applications with emma. In *ICMI-MLMI '09: Proceedings of the 2009 international conference on Multimodal interfaces*, pages 47–54, New York, NY, USA, 2009. ACM.

27 Agnès Just. *Two-Handed Gestures for Human-Computer Interaction.* PhD thesis, 2006. PhD Thesis #3683 at the École Polytechnique Fédérale de Lausanne.

28 Agnès Just, Yann Rodriguez, and Sébastien Marcel. Hand posture classification and recognition using the modified census transform. Idiap-RR Idiap-RR-02-2006, IDIAP, 2006. Published in Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition 2006.

29 Youn-ah Kang and John Stasko. Lightweight task/application performance using single versus multiple monitors: a comparative study. In *GI '08: Proceedings of graphics interface 2008*, pages 17–24, Toronto, Ont., Canada, Canada, 2008. Canadian Information Processing Society.

30 Andreas Kerren, Achim Ebert, and Jörg Meyer, editors. *Human-Centered Visualization Environments, GI-Dagstuhl Research Seminar, Dagstuhl Castle, Germany, March 5-8, 2006, Revised Lectures*, volume 4417 of *Lecture Notes in Computer Science*. Springer, 2007.

31 Olaf Körner and Reinhard Männer. Implementation of a haptic interface for a virtual reality simulator for flexible endoscopy. In *HAPTICS '03: Proceedings of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS'03)*, page 278, Washington, DC, USA, 2003. IEEE Computer Society.

32 David M. Krum, Olugbenga Omoteso, William Ribarsky, Thad Starner, and Larry F. Hodges. Speech and gesture multimodal control of a whole earth 3d visualization environment. In *VISSYM '02: Proceedings of the symposium on Data Visualisation 2002*, pages 195–200, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

33 Micah Lapping-Carr, Odest Chadwicke Jenkins, Daniel H Grollman, Jonas N Schwertfeger, and Theodora R Hinkle. Wiimote interfaces for lifelong robot learning. In *AAAI Spring Symposium*, Menlo Park, CA, USA, March 2008.

34 Johnny Chung Lee. `http://www.cs.cmu.edu/~johnny/projects/wii`; Online; Accessed 23-April-2010.

35 Johnny Chung Lee. Hacking the nintendo wii remote. *IEEE Pervasive Computing*, 7(3):39–45, 2008.

36 Alan Liu, Frank Tendick, Kevin Cleary, and Christoph Kaufmann. A survey of surgical simulation: applications, technology, and education. *Presence: Teleoper. Virtual Environ.*, 12(6):599–614, 2003.

37 Kevin T. McDonnell, Hong Qin, and Robert A. Wlodarczyk. Virtual clay: a real-time sculpting system with haptic toolkits. In *I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 179–190, New York, NY, USA, 2001. ACM.

38 Kai Nickel and Rainer Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 140–146, New York, NY, USA, 2003. ACM.

39 Norbert Nitzsche and Günther Schmidt. Force-reflecting telepresence in extensive remote environments. *Journal of Intelligent & Robotic Systems*, 50:3–18, 2007. 10.1007/s10846-007-9148-7.

40 Mark A. Nixon, Bruce C. McCallum, W. Richard Fright, and N. Brent Price. The effects of metals and interfering fields on electromagnetic trackers. *Presence: Teleoper. Virtual Environ.*, 7(2):204–218, 1998.

41 Sharon Oviatt. Multimodal interactive maps: designing for human performance. *Hum.-Comput. Interact.*, 12(1):93–129, 1997.

42 Paulo Peixoto and Joao Carreira. A natural hand gesture human computer interface using contour signatures. *Proceedings of the IASTED HCI Conference*, 476, November 2005.

43 Luc Renambot, Andrew Johnson, and Jason Leigh. Lambdavision: Building a 100 megapixel display. `http://www.evl.uic.edu/cavern/sage/pubs/LambdaVision-light.pdf`; Online; Accessed 20-April-2010.

44 Scott Robertson, Brian Jones, Tiffany O'Quinn, Peter Presti, Jeff Wilson, and Maribeth Gandy. Multiuser collaborative exploration of immersive photorealistic virtual environments in public spaces. In *VMR '09: Proceedings of the 3rd International Conference on Virtual and Mixed Reality*, pages 235–243, Berlin, Heidelberg, 2009. Springer-Verlag.

45 Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. Gesture recognition with a wii controller. In *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 11–14, New York, NY, USA, 2008. ACM.

46 Lauren Shupp, Robert Ball, Beth Yost, John Booker, and Chris North. Evaluation of viewport size and curvature of large, high-resolution displays. In *GI '06: Proceedings of Graphics Interface 2006*, pages 123–130, Toronto, Ont., Canada, Canada, 2006. Canadian Information Processing Society.

47 Veikko Surakka, Marko Illi, and Poika Isokoski. Gazing and frowning as a new human–computer interaction technique. *ACM Trans. Appl. Percept.*, 1(1):40–56, 2004.

48 Edward Tse, Chia Shen, Saul Greenberg, and Clifton Forlines. Enabling interaction with single user applications through speech and gestures on a multi-user tabletop. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 336–343, New York, NY, USA, 2006. ACM.

49 Tamas Vajk, Paul Coulton, Will Bamford, and Reuben Edwards. Using a mobile phone as a "wii-like" controller for playing games on a large public display. *Int. J. Comput. Games Technol.*, pages 1–6, 2008.

50 Visbox innovative display and interaction technologies. `http://visbox.com`; Online; Accessed 20-April-2010.

51 Daniel Vogel and Ravin Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 33–42, New York, NY, USA, 2005. ACM.

52 Florian Vogt, Justin Wong, Sidney Fels, and Duncan Cavens. Tracking multiple laser pointers for large screen interaction. In *Ext. Abstracts UIST*, pages 95–96, 2003.

53 WiiBrew. `http://wiibrew.org`; Online; Accessed 23-April-2010.

54 Chadwick A. Wingrave, Brian Williamson, Paul D. Varcholik, Jeremy Rose, Andrew Miller, Emiko Charbonneau, Jared Bott, and Joseph J. LaViola Jr. The wiimote and beyond: Spatially convenient devices for 3d user interfaces. *IEEE Comput. Graph. Appl.*, 30(2):71–85, 2010.

55 Chow Yang-Wai. 3d spatial interaction with the wii remote for head-mounted display virtual reality. *WASET: World Academy of Science, Engineering, and Technology*, 50:377–383, 2009.

56 Elena Zudilova-Seinstra, Tony Adriaansen, and Robert van Liere. *Trends in Interactive Visualization: State-of-the-Art Survey.* Springer Publishing Company, Incorporated, 2008.

Detection and Identification Techniques for Markers Used in Computer Vision

Johannes Köhler[1], Alain Pagani[1], and Didier Stricker[1]

1 **Department of Augmented Vision**
German Research Center for Artificial Intelligence GmbH
Trippstadter Str. 122, 67663 Kaiserslautern, Germany
{Johannes.Koehler, Alain.Pagani, Didier.Stricker}@DFKI.de

Abstract

This paper summarizes and compares techniques for detecting and identifying markers in the context of computer vision. Existing approaches use correlation, digital, or topological methods for marker identification. The comparison points out that all marker processing algorithms, which employ sophisticated digital codes are more robust and reliable. Existing bit representation schemes for these codes and marker designs are compared with each other. We illustrate in this context, why the marker processing algorithm introduced in [11] is the best performer regarding marker occlusion and minimal detectable pattern size.

Keywords and phrases Marker identification, computer vision

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.36

1 Introduction

A marker system consists of a set of patterns that can be detected by a computer equipped with a camera and an appropriate detection algorithm. Markers placed in the environment provide easily detectable visual cues for indoor tracking [13], robot navigation [20], augmented reality ([8], [9], [23], [4] among others) and, in general, all applications where the relative pose between a camera and an object is required. The maximum distance between marker and camera that still allows a successful detection should be as large as possible in the computer vision context. This differentiates and distinguishes "vision" markers (figure 1) from those that have the purpose of information transport only. Examples for the latter types are MaxiCode used by the U.S. postal service or QuickResponse [24], their high information density makes them detectable at close distance only. They are not suitable for computer vision and will not be covered in this paper. Algorithms that detect only single markers (i.e. they have the smallest possible marker library size) as well as techniques that demand special hardware other than a conventional camera are also not considered.

Figure 1 Markers that are used in computer vision ([13], [9] [5], [11]).

Marker trackers used in computer vision mostly use either square or circular tags. These geometric primitives are well-detectable in images and usually serve as an initial hint for the

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 36–44
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

presence of a marker. Topology based fiducial systems as reacTIVision ([1], [3] or [15]) are the only ones that diverge from this scheme. Their markers have an irregular shape and are in case of reacTIVision tailored for the use in multitouch-surface application. The precise camera pose cannot be derived from them. Both square- and circular markers have individual advantages: The contents of a square can be accessed uniformly using only homography and the camera pose computed from the vertices is unique in most cases [21]. In case of circular markers, the camera pose for accessing the marker content can be computed from the whole contour surrounding the marker, instead of only the 4 vertices. This makes the detection more robust for occlusion. It, however, demands the intrinsic data of the camera to be known and always yields a pose ambiguity when only a single marker is used ([11], [2]). According to [19], square tags carry a larger symbolic data payload than a circular tag of the same size, whereas circular tags offer better location and pose accuracy. The latter could be confirmed in [10].

Various marker identification techniques exist that are applied to the detected geometric primitives to extract the markers. Most approaches either measure the degree of correlation of a pattern found inside an image with known patterns or use digital methods, e.g. read a binary code from the marker. The code stores a non redundant ID for identification. A third identification approach extracts markers from the image topology. It can be said in general that correlation and topology based trackers perform least robust ([10], [4]). All approaches influence the marker design.

Several parameters for measuring the performance and robustness of a marker detection algorithm have been introduced. The camera pose dependent bin, or "bit bucket" size, is one criterion for detectability of digital code based markers. It was theoretically analyzed for square and circular markers in [19]. Another detectability measure is the detection rate as function of the tag size in an image, applicable to all types of markers. Robustness can be measured by the false positive, false negative and inter marker confusion rates [4].

Marker processing can in general be grouped into candidate detection and identification. Candidate detection finds e.g. geometric primitives that constitute a marker. The identification then separates true markers from false candidates. The remainder of this paper is thus grouped into 3 sections: Section 2 lists basic, initial detection approaches. Section 3 discusses existing identification approaches and their influence on the marker appearance. Available performance analyses are presented and compared in section 4. The paper concludes with a short summary and an outlook on future work.

2 Marker Candidate Detection

The detection of a potential marker is usually accomplished by detecting perspective distortions of the geometric primitives they are composed of. These are quadrilaterals in the case of squares and ellipses in case of circles. These primitives are extracted from contours that were found using either image binarization by thresholding or edge detection. There are algorithms that detect primitives only as a whole and algorithms capable of detecting defective shapes. The latter are important in case of occlusion. The interested reader is directed to [4] for quad detection and [10] for ellipse detection. In case of topological identification methods (section 3.4), the markers are independent of the geometry, thus no candidate detection is necessary. The marker is detected and identified at the same time by searching a subgraph in a topological region adjacency graph.

3 Marker Identification

After the detection of geometric primitives in the image it is still unclear, if they originated from a marker. Regardless of the application of the marker tracking system, there are most certainly more primitives that do not represent a marker and should not yield a positive detection. The marker therefore needs certain features to distinguish it from its surroundings and to serve as additional detection features. The identification can either be based on correlation, on a digital code or on characteristic topology. Various designs exist for the approaches that determine how information is encoded on the marker. Moreover a uniform access method for the marker contents is necessary for correlation and code based identification.

3.1 Uniform Content Access

Each marker type has a 2D reference layout M_{ref} that contains information on where to find the marker contents (e.g., code bins or detection features). In order to access a hypothetical marker M_{hyp} (i.e., a detected geometric primitive) that is perceived in an image, a mapping $f : M_{ref} \mapsto M_{hyp}$ is necessary, that correctly maps reference features to (potentially) observed features. For square markers, f is most commonly the homography computed from the vertices. For circular markers the camera pose relative to the marker can be computed from its contour ([2], [10]). Together with the camera intrinsic data matrix K, it can be used to access the marker content: $M_{hyp} = KPM_{ref}$. The marker orientation is unknown up to this point and needs to be derived from special features on the marker. This holds true for squares that naturally induces 4 possible orientations, and circles that have no natural orientation indicators. Possible solutions are:

- **Graphical indicators**: They serve as anchors and are used e.g. in CyberCode [17], TriCode [12] and Intersense [13] (figure 2).
- **Fixating correlation**: Mostly usable for square markers; the correlation towards a template is computed for all possible orientations, the orientation yielding the highest correlation is assumed to be correct and fixes the orientation. Used e.g. in ARToolkit.
- **Fixating codes**: Mostly usable for square markers; A code is placed on the marker that only decodes correctly at the correct orientation and fixates it. This technique was introduced in [4].
- **Mixtures** The mixture of graphical orientation features generates a set of possible orientations and enables the use of either correlation measurement or digital codes for estimating the orientation on circular markers [11].
- **Rotational invariance**: When the rotation of a tag is not of importance, rotational invariant detection methods can be used. The possible simplest example is the use of circular markers with a circular information encoding pattern like concentric circles [7] or fourier tags [20]. More advanced is the use of cyclic codes, that, with a special layout, can be read at arbitrary orientations [18]. In this way rotationally invariant markers can be extended with error correction.

Graphical orientation features work well, but are, in general, not robust towards partial occlusion of the tag. The same holds true for correlation based methods. When the orientation can be determined, the contents of the marker can be uniformly read.

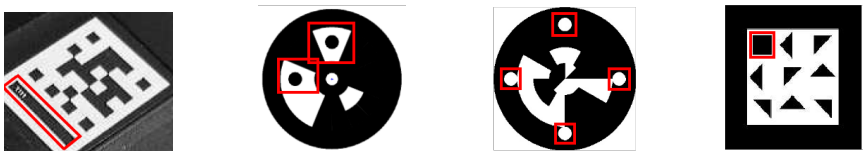


Figure 2 Various graphical orientation features used in existing marker systems

3.2 Identification Based on Correlation

In the case of correlation based marker identification, the marker is subdivided into grid cells that are correlated with reference grids. The correlation coefficient then indicates the "markerness" of an observation. Since this is a threshold decision, one can expect a high false positive detection rate of correlation based trackers [4]. The most popular and potentially most wide spread tracker which uses correlation based marker assignment is ARToolkit [8]. Since correlation does not require binary values, arbitrary images could be used on the marker. Thus, no consistent design can be provided. According to [16] the use of 2D direct cosine transform basis functions for marker pattern generation produces a higher reliability.

3.3 Identification Based on Digital Codes

In the case of digital code based marker identification, binary digits are read from the marker content that contain an ID. Error correction and detection as well as other digital methods can be applied to them. The marker appearance depends on the method of how the code is represented. The following methods exist (figure 3):

- **Code Bins:** The marker is subdivided into bins that store black/ white values which represent 1/0. This is the most commonly used method.
- **Fourier Encoding:** Code words are represented by evenly spaced, low-frequency signal bursts and visually represented in spatial domain using a fourier transform. This technique is used in [20].
- **TriCodes:** This encoding technique uses isosceles triangles which point in 1 of 8 directions. Each triangle thus encodes 3 bits. The technique is used in [12].

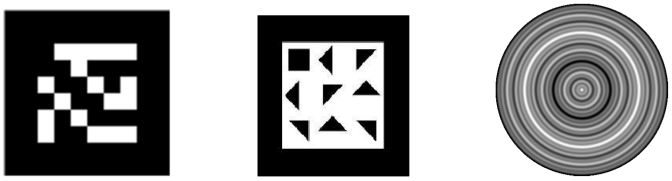


Figure 3 Bit representations; From left to right: Bins, TriCode, Fourier

Possible ways of constructing codes for digital markers are:

- **Raw codes:** The unprotected digital ID is placed on the marker ([13], [25]). Only the presence of orientation features ensures "markerness", thus a high false positive rate can be expected.
- **Error detecting codes:** A redundant part for error detection is computed from the marker ID and added to the marker. "Markerness" of a shape is checked by correct decoding. CRC codes are used in current marker systems ([4], [23], [11], [22]).

- **Error correcting codes:** A redundant part for error correction is computed from the marker ID and added to the marker. The codes used are usually not precisely stated ([22], [4], [17]), according to their description they could be categorized as BCH codes.

Reed Solomon codes were used for error correction and CRC codes for error detection in [10] and [11]. It was shown that the exploitation of additional error detecting capabilities of Reed Solomon codes can be used together with an appropriate minimal CRC generator polynomial to shorten the code used on markers.

3.4 Identification Based on Topology

Topology based identification relies on topological region adjacency. Images are thresholded and a region adjacency graph is computed, that reflects containment relationships. The graph is analyzed for patterns which resulted from a marker (figure 4 (left)). It is notable, that this type of identification is independent of the tag geometry up to a certain extend and thus does not need an a priori detection of geometric primitives. Distortions of the tags still yield a positive detection (figure 4 (right)), unless they change the topology. Topology based extraction algorithms are on the other hand not capable of handling occlusions and could yield a comparatively high false positive detection rate, if the topologic structure is not complex enough [15]. Systems employing these type of markers are [3], [1], [14] and [15].

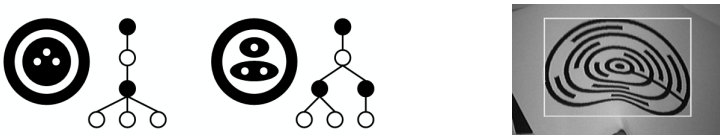


Figure 4 left: Topology graph for two exemplary markers, **right:** Deformed, detectable marker

4 Performance Analyses

There are currently no comparative studies of all existing marker processing systems, mostly due to the fact of their unavailability. To our knowledge ARToolkit, ARToolkitPlus and reacTIVision are currently freely available, the distribution of ARTag has been stopped. Available results and analyses for current marker processing algorithms are compared in the following section. A detailed list of metrics and evaluation criteria that can be applied to evaluate marker detection algorithms is found in [6]: False positive rate, inter-marker confusion rate, false negative rate, minimal marker size, vertex jitter characteristics, marker library size, immunity to lighting conditions, immunity to occlusion, perspective support, immunity photometric calibration and the speed performance. Table 1 shows known results for a set of existing marker systems. A subset of the previous characteristics were chosen according to the availability of data.

These results show that the library sizes vary greatly, ranging from 17 to up to 262144 markers. It is however questionable, if that many markers are needed in any field of application. Instead, it is more reasonable to decrease the library size. In the case of digital markers, all bits that are then unoccupied can be used to add redundancy for error correction or can be removed to improve the tag readability. No meaningful library size can be provided for correlation based trackers, since training of new patterns enables a continuous enlargement

	# Tags	Red	Occl	FP	MC	MS
ARToolkit [8]	NP	0%	0%	0.5-4.4%+	N/A*	25-65
ARTag [4]	2002	70.2%	5%	0%	0.0039%	20
ARToolkit+ [23]	4096	66.6%	N/A*	0%	N/A	N/A
Intersense [13]	2^{15}	0%	0%	N/A*	N/A*	16*
[11]	255	75%	25%	0%	0.153%	17.36
TriCode [12]	2^{18}	25%	0%	N/A*	N/A	N/A*
reacTIVision [1]	89	0%	N/A	$> 0\%$	N/A	N/A
Nishino [15]	17	0%	N/A	0%	N/A	40
Constanza [3]	N/A	0%	N/A*	0-16%	N/A*	N/A

Table 1 #Tags: Tag amount in the library; Red: Redundancy in the identification pattern; Occl: Amount of possible data part occlusion; FP: False positive detection rate; MC: Marker confusion rate; MS: Minimum detectable tag size in pixel.
"NP": not possible for the marker type; "N/A": no data available; "N/A*": no precise data available, but facts that allow a comparative estimation.
red: Correlation based-; blue: digital method based-; green: topology based identification.

of the library, which certainly will rise the false positive detection rate and marker confusion rate though.

Redundancy is only present in the case of digital markers. The corresponding "Red" value in table 1 reflects the amount of redundant bits and measures the information density. The occlusion value ("Occl") illustrates, to what extend the redundant data is used for error correction. It is only measured for the data part of the marker, not for its border and results from the amount of correctable bits in the case of a digital marker. Constanza's [3] topology based markers can also be detected under partial occlusion, which is achieved by adding a topology error tolerance, that does not permit a precise determination of the coverable area. It also raises the false positive detection and marker confusion rates (refer to [3] for more details). The use of error correcting codes in case of ARToolkitPlus is mentioned in [23], but a CRC code is reported for error correction. Since CRC codes however only detect errors, it is not clear to what extend ARToolkitPlus is able to correct errors.

The false positive detection rate is computed from available analyses that applied marker detection algorithms to images not containing a marker. It reflects the amount of frames where a marker was detected. Algorithms employing digital codes never reported a marker. The same holds for Nishino's topology based algorithm [15]. ARToolkit's false positive detection rate strongly depends on the correlation threshold (confidence factor, c.f.) used. This is reflected by the large interval (0.5-4.4%) which was computed from the results published in [4]. Only a single reference pattern was used for the experiment, [12] confirmed that the false positive detection rate dramatically increases when more patterns are used. The false positive rate for the Intersense algorithm [13] can be assumed to be larger than 0% because the marker orientation feature is the only factor for ensuring "markerness" (section 3.1). This feature might be found often by accident.

The theoretical marker confusion rate is only available for ARTag [11]. In both cases it represents the risk of marker confusion caused by error correction. The values have a pure combinatorial nature and do not necessarily represent rates applicable in practice, since the factors that cause errors (occlusion, errors in f (section 3.1)) can hardly be captured by theory. Refer to [10] and [4] for more details. The confusion rate in practice for the algorithms marked with N/A* can be expected to be high, since they do not use protection

mechanisms. In the case of ARToolkit, this depends on the amount of markers [12] and the type of patterns used [16]. In the case of topology based methods, the complexity of the tag structure influences the confusion rate [15]. An error tolerance, as used in Constanza's markers [3], that raises the false positive detection rate might also likely raise the confusion rate.

Detectable tag sizes vary from 17-65 pixels. The value 16, in the case of Intersense, is of theoretical nature and was not verified with experiments. [11] performs best with a minimal detectable tag size of 17 pixels, followed by ARTag. The second number represents the minimal tag size in case of defects (36 pixels). ARToolkit's minimal pattern size depends on the correlation threshold (c.f.). If a high tolerance is used, the tag size decreases but the false positive rate increases. With a strict tolerance, the tag size is comparatively large. The same holds for Nishino's algorithm [15].

4.1 Design of Digital Markers

A thorough comparative study on the design consequences of digital markers can be found in [19]. The authors of that paper use their machine vision framework, Cantag, for a theoretical evaluation of tag readability and a comparison of square and circular tag shapes. They furthermore conclude that square tags carry a larger symbolic data payload than a circular tag of the same size, whereas circular tags offer better location and pose accuracy. The pose accuracy aspect can be confirmed for the algorithm proposed in [10]. Using bilinear interpolation code bins down to a size of 3 pixels could be successfully read, which is close to the theoretical limit of 1 pixel. For the TriCode bit encoding scheme, one can expect the tag readability to be worse than this because the successful recognition of a triangle demands a larger shape.

5 Conclusion

In this paper an overview of existing marker systems was presented. The generalized detection steps were presented and current methods were arranged in this context. Among the known techniques, digital method based marker processing algorithms have been shown to be most reliable. Their performance and behavior in the presence of errors is predictable and can be quantified precisely. In numerous experiments they never gave rise to false positive detection or inter marker confusion and detectable tag sizes are small. The traditional bin encoding scheme is used for bit representation on most digital markers. Too few results were published for alternatives (TriCode, Fourier encoding) to compare the approaches in a fair way. It can however be said that the information density is currently lower for fourier tags due to the spatial pattern distribution. The same holds for TriCodes, since the triangles encode 3 bits, but the white space in between is wasted.

In contrast to this, the correlation based ARToolkit is possibly the most widespread marker tracker due to its availability. It, however, suffers from high false positive and marker confusion rates. The same holds for most topology based methods. Minimization of the marker code together with a pose computation from the whole marker contour make the circular marker processing algorithm proposed in [11] the top performer for occlusion and detectable tag size. The number of tags it offers is sufficient for a large field of applications. It is currently developed further to support erasure channel decoding for handling larger occlusions up to 50%. This will drastically increase the combinatorial marker confusion rate. Thus a more meaningful metric needs to be developed, that better adapts to the practical occurrence of error patterns.

References

1 Ross Bencina and Martin Kaltenbrunner. The design and evolution of fiducials for the reactivision system. In *Proc. 3rd International Conference on Generative Systems in the Electronic Arts*, 2005.

2 Qian Chen, Haiyuan Wu, and Toshikazu Wada. Camera calibration with two arbitrary coplanar circles. In *ECCV* (3), 521–532, 2004.

3 Enrico Costanza and J. Robinson. A region adjacency tree approach to the detection and design of fiducials. In *VVG*, pages 63–69, 2003.

4 Mark Fiala. Artag revision 1. a fiducial marker system using digital techniques. Technical report, National Research Council of Canada, 2004.

5 Mark Fiala. Artag, a fiducial marker system using digital techniques. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* 2:590–596. IEEE Computer Society, 2005.

6 Mark Fiala. Designing highly reliable fiducial markers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(RapidPosts), 2009.

7 Lance B. Gatrell, William A. Hoff, and Cheryl W. Sklair. Robust image features: concentric contrasting circles and their image extraction. In *Cooperative Intelligent Robotics in Space II*, pages 235–244, 1992.

8 Hirokazu Kato and Mark Bilinghurst. *ARToolkit User Manual*. Human Interface Technology Lab, University of Washington, 2000.

9 Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, San Francisco, USA, October 1999.

10 Johannes Koehler. Robust detection and identification of circular markers for estimation of the camera pose in real time. Master's thesis, Technische Universität Kaiserslautern, 2009.

11 Johannes Koehler, Alain Pagani, and Didier Stricker. Robust detection and identification of partially occluded circular markers. In *Proceedings of the 3rd International Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 387–392, 2010.

12 Jonathan Mooser, Suya You, and Ulrich Neumann. Tricodes: A barcode-like fiducial design for augmented reality media. In *ICME*, pages 1301–1304. IEEE, 2006.

13 Leonid Naimark and Eric Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *ISMAR '02: Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, page 27. IEEE Computer Society, 2002.

14 Hiroki Nishino. A split-marker tracking method based on topological region adjacency & geometrical information for interactive card games. In *SIGGRAPH ASIA '09: ACM SIGGRAPH ASIA 2009 Posters*, New York, NY, USA, 2009. ACM.

15 Hiroki Nishino. A 6dof fiducial tracking method based on topological region adjacency and angle information for tangible interaction. In *TEI '10: Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, pages 253–256, New York, NY, USA, 2010. ACM.

16 Charles B. Owen, Fan Xiao, and Paul Middlin. What is the best fiducial. In *The First IEEE International Augmented Reality Toolkit Workshop*, 2002.

17 Jun Rekimoto and Yuji Ayatsuka. Cybercode: designing augmented reality environments with visual tags. In *Designing Augmented Reality Environments*, 2000.

18 Andrew C. Rice, Christopher B. Cain, and John K. Fawcett. Dependable coding of fiducial tags. In Hitomi Murakami, Hideyuki Nakashima, Hideyuki Tokuda, and Michiaki Yasumura, editors, *UCS*, 3598:259–274 of *Lecture Notes in Computer Science*. Springer, 2004.

19 Andrew C. Rice, Robert K. Harle, and Alastair R. Beresford. Analysing fundamental properties of marker-based vision system designs. *Pervasive and Mobile Computing*, 2(4):453–471, November 2006.

20 Junaed Sattar, Eric Bourque, Philippe Giguere, and Gregory Dudek. Fourier tags: Smoothly degradable fiducial markers for use in human-robot interaction. *Computer and Robot Vision, Canadian Conference*, 0:165–174, 2007.

21 Gerald Schweighofer and Axel Pinz. Robust pose estimation from a planar target. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:2024–2030, 2006.

22 Daniel Wagner, Tobias Langlotz, and Dieter Schmalstieg. Robust and unobtrusive marker tracking on mobile phones. In *ISMAR '08: Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 121–124, Washington, DC, USA, 2008. IEEE Computer Society.

23 Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, 2007.

24 Denso Wave. Quick response specification. http://www.denso-wave.com/qrcode/index-e.html, 1994.

25 Xiang Zhang, Yakuo Genc, and Nassir Navab. Mobile computing and industrial augmented reality for real-time data access. Technical report, Siemens Corporate Research, Imaging and Visualization Department, 755 College Road East, Princeton, NJ 08540, USA, 2001.

Markerless Camera Pose Estimation - An Overview

Tobias Nöll[1], Alain Pagani[1], and Didier Stricker[1]

1 **Department of Augmented Vision**
German Research Center for Artificial Intelligence GmbH
Trippstadter Str. 122, 67663 Kaiserslautern, Germany
{Tobias.Noell, Alain.Pagani, Didier.Stricker}@DFKI.de

Abstract

Human perception shows that a correct interpretation of a 3D scene on the basis of a 2D image is possible without markers. Solely by identifying natural features of different objects, their locations and orientations on the image can be identified. This allows a three dimensional interpretation of a two dimensional pictured scene. The key aspect for this interpretation is the correct estimation of the camera pose, i.e. the knowledge of the orientation and location a picture was recorded. This paper is intended to provide an overview of the usual camera pose estimation pipeline as well as to present and discuss the several classes of pose estimation algorithms.

Keywords and phrases Pose Estimation

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.45

1 Introduction

A large number of different markerless pose estimation algorithms already exist [5] [8] [3] [17] [6] [12] . Usually the camera pose for a given image is estimated solely using a set of correspondences. A correspondence is a match of an object's natural feature (3D) detected on the image (2D). With a sufficient number of correspondences the camera pose can be determined uniquely. In this paper the different classes of pose estimation algorithms will be presented along with their advantages and disadvantages. The classes can be roughly categorized into the following:

- **Direct Linear Transformation:** These algorithms intend to directly estimate the camera pose ignoring certain restrictions regarding the solution space.
- **Perspective n-Point:** These algorithms intend to directly estimate values for the parameters parameterizing the solution space of all valid camera poses.
- **A priori information estimators:** Beside the point correspondences, additional information regarding the camera pose is often available before estimating. The algorithms belonging to this class use this information in order to provide more reliable or faster results.

Figure 1 gives a possible application of a camera pose estimation algorithm: Once the camera pose is known, the recorded image is extended with important 3D information. Not only can all objects whose three dimensional positions are known be identified on the image, the image can also be enriched with virtual information, allowing a fusion of reality and virtuality (virtual reality). This way images can be extended with additional virtual objects.

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 45–54

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

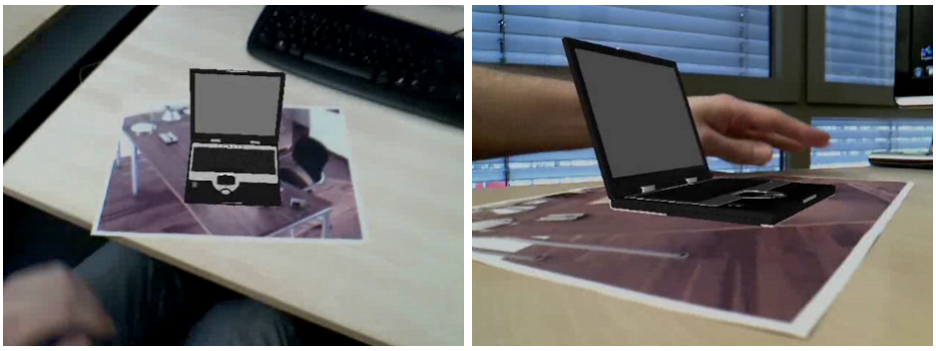


Figure 1 By estimating the camera pose, the live recorded image can be added with additional virtual objects (in this case a notebook). Even though the camera moves, the impression of the mixed reality and virtuality is still unbiased.

2 Background

The mathematical camera model which is used in the following is built on the principles of the pinhole camera. It models the central projection of 3D points through the center of projection onto the image plane. This mapping depends on the following parameters:

- The *internal parameters* of the camera: $K \in \mathbb{R}^{3\times3}$ which depends on certain properties of the manufactured camera (i.e. focal length, lens properties). These parameters can be estimated using camera calibration techniques such as [18] and are assumed to be known. K maps 3D points in camera coordinates to 2D image points.
- The *external parameters* of the camera: $\left[R \mid \mathbf{t}\right] \in \mathbb{R}^{3\times4}$. This mapping is commonly referred to as the *camera pose* which relates 3D points from their representation in a world coordinate system to their representation in the camera coordinate system by use of a rotation R and a translation $\mathbf{t}$.

Altogether a homogeneous representation $\mathbf{P} \in \mathbb{R}^4$ of a 3D point in world coordinates is mapped by the camera to the point $\mathbf{p} \in \mathbb{R}^3$. $\mathbf{p}$ itself is a homogeneous representation of the corresponding 2D point on the image. This mapping is defined by:

$$\mathbf{p} = K \left[R \mid \mathbf{t}\right] \mathbf{P} \tag{1}$$

Estimation of the camera pose from correspondences then refers to searching the camera pose $\left[R \mid \mathbf{t}\right]$ which best relates a set of given correspondences $C = \{\mathbf{P_i} \longleftrightarrow \mathbf{p_i}\}$ using Equation 1 under a given camera calibration K.

2.1 Correspondence generation

The set of correspondences C is usually automatically generated using feature detectors. Given a set of images, feature detectors try to identify and match several features of the captured objects on those images (see Figure 2). In context of camera pose estimation, one image is referred to as the *reference image*. Features on this image have – manually or automatically – been assigned to 3D coordinates in the world coordinate frame. This way, once a feature from the reference image with assigned 3D coordinate $\mathbf{P_i}$ is detected on position $\mathbf{p_i}$ in the captured image, a correspondence $\mathbf{P_i} \longleftrightarrow \mathbf{p_i}$ is generated. Popular feature detectors are SIFT [13] and Randomized Trees [11].

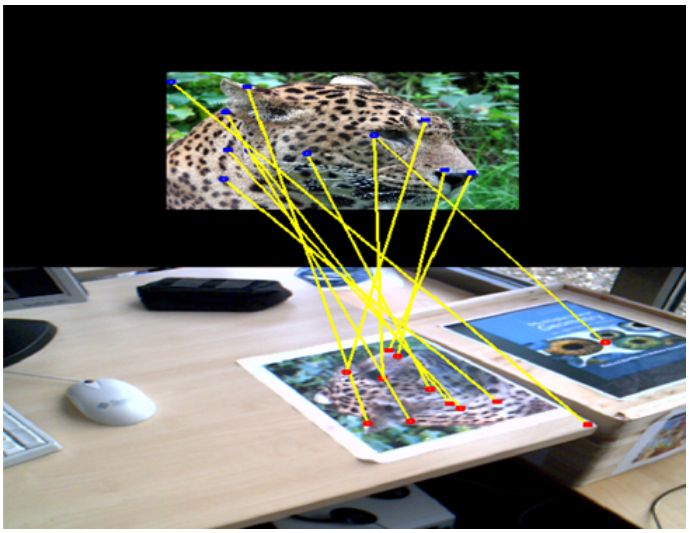


Figure 2 Two images (top: reference image, bottom: captured image) and their matched features. Some features have been misleadingly matched.

Since the correspondences are generated automatically, C will also contain a certain percentage of outliers, i.e. untrue correspondences $\mathbf{P_i} \longleftrightarrow \mathbf{p_i}$. These outliers are problematic because they potentially can have a huge negative impact on the quality of the solution. Therefore, methods such as RANSAC [5] need to be used to identify the inliers in order to perform the camera pose estimation only using this set.

2.2 Exclusion of outliers

RANSAC uses a stochastic approach in order to identify the inliers. Therefore this algorithm is of non-deterministic nature in the sense that it produces a reasonable result only with a certain probability. This probability increases as more iterations are allowed. The algorithm operates as follows:

1. A random subset C_S of the provided correspondences C is selected. It contains only the minimal number of correspondences required for camera pose estimation using an arbitrary pose estimation algorithm M.
2. A camera pose Q is estimated using M along with C_S.
3. All remaining correspondences $C \setminus C_S$ are then checked for integrity with Q. Therefore the points $\mathbf{P_i}$ are projected using the known camera matrix K and Q to the 2D points $\mathbf{p'_i}$. If $\|\mathbf{p'_i} - \mathbf{p_i}\| \leq \tau$, then the correspondence $\mathbf{P_i} \longleftrightarrow \mathbf{p_i}$ fits well to the estimated camera pose and will therefore also be considered as a hypothetical inlier.
4. Q is reasonably good if a sufficient number of correspondences has been classified as hypothetical inliers. Q is then reestimated from all hypothetical inliers using M, because it has only been estimated from the initial set of hypothetical inliers C_S.

This procedure is repeated a fixed number of times, each time producing either a camera pose which is rejected because too few points are classified as inliers or a refined pose together with a corresponding error measure. In the second case the refined pose is kept if its error is lower than the last saved pose.

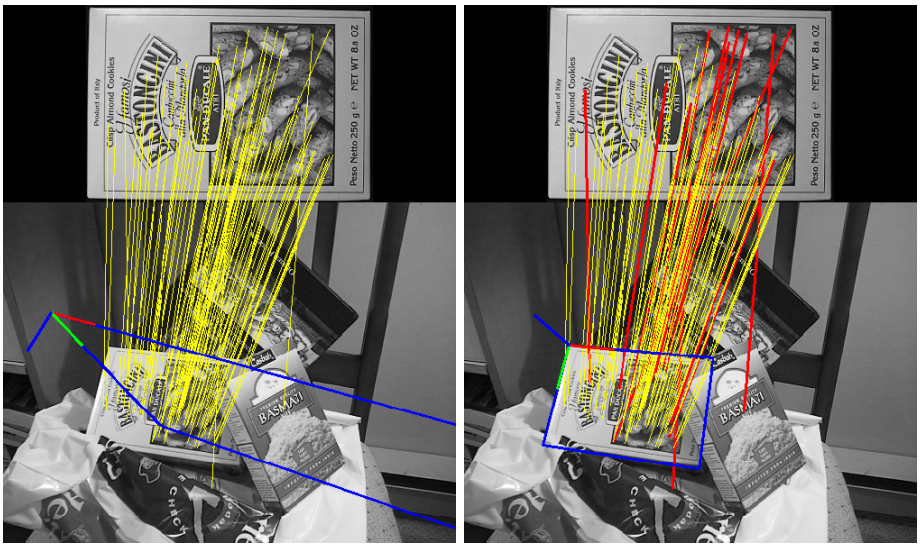


Figure 3 The yellow correspondences are computed using the SIFT feature detector [13] providing many good matches along with a small number of outliers. The borders of the reference image are reprojected in blue in the scene using the calculated camera pose. One can see that the quality of the solution is highly improved by identifying the outliers. Left: All yellow correspondences are used, including the outliers. Right: RANSAC is used in order to find the set of inliers (yellow correspondences). The red correspondences are treated as the outliers and not used in order to compute the final camera pose.

Figure 3 gives an example in the context of augmented reality, showing how important the identification of outlying point correspondences is in order to receive a reliable camera pose.

3 Direct Linear Transformation

For the Direct Linear Transformation (DLT) algorithm [9] it is assumed that the set of points $\mathbf{P_i}$ spans a real 3D space which means that those points are arbitrarily distributed in space an thus do not lie on a single point, line or plane. Let $\mathbf{P_i} = \begin{pmatrix} X_i & Y_i & Z_i & W_i \end{pmatrix}^T \longleftrightarrow \mathbf{p_i} = \begin{pmatrix} u_i & v_i & w_i \end{pmatrix}^T$ be n correspondences. A DLT $F \in \mathbb{R}^{3\times 4}$ now is a linear function $F : \mathbb{R}^4 \mapsto \mathbb{R}^3$ which maps the points $\mathbf{P_i}$ to the points $\mathbf{p_i}$. This can be expressed in the homogeneous context as

$$F\mathbf{P_i} \sim \mathbf{p_i} \Leftrightarrow \mathbf{p_i} \times F\mathbf{P_i} = \mathbf{0} \tag{2}$$

F has 12 unknowns:

$$F = \begin{bmatrix} {f_1}^1 & {f_2}^1 & {f_3}^1 & {f_4}^1 \\ {f_1}^2 & {f_2}^2 & {f_3}^2 & {f_4}^2 \\ {f_1}^3 & {f_2}^3 & {f_3}^3 & {f_4}^3 \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \mathbf{f^1}^T \\ \mathbf{f^2}^T \\ \mathbf{f^3}^T \end{bmatrix}$$

Rewriting Equation 2 using the fact that $\mathbf{f^i}^T\mathbf{P_i} = \mathbf{P_i}^T\mathbf{f^i}$ yields

$$\mathbf{p_i} \times F\mathbf{P_i} = \begin{pmatrix} v_i\mathbf{f^3}^T\mathbf{P_i} - w_i\mathbf{f^2}^T\mathbf{P_i} \\ w_i\mathbf{f^1}^T\mathbf{P_i} - u_i\mathbf{f^3}^T\mathbf{P_i} \\ u_i\mathbf{f^2}^T\mathbf{P_i} - v_i\mathbf{f^1}^T\mathbf{P_i} \end{pmatrix} = \underbrace{\begin{bmatrix} \mathbf{0}^T & -w_i\mathbf{P_i}^T & v_i\mathbf{P_i}^T \\ w_i\mathbf{P_i}^T & \mathbf{0}^T & -u_i\mathbf{P_i}^T \\ -v_i\mathbf{P_i}^T & u_i\mathbf{P_i}^T & \mathbf{0}^T \end{bmatrix}}_{\stackrel{\text{def}}{=} A'_i} \underbrace{\begin{pmatrix} \mathbf{f^1} \\ \mathbf{f^2} \\ \mathbf{f^3} \end{pmatrix}}_{\stackrel{\text{def}}{=} \mathbf{f}} \tag{3}$$

However each correspondence will only derive two *linearly independent* equations. The linearly independent system now reads

$$\underbrace{\begin{bmatrix} \mathbf{0}^T & -w_i\mathbf{P_i}^T & v_i\mathbf{P_i}^T \\ w_i\mathbf{P_i}^T & \mathbf{0}^T & -u_i\mathbf{P_i}^T \end{bmatrix}}_{\stackrel{\text{def}}{=} A_i} \mathbf{f} = \mathbf{0} \tag{4}$$

For $n \geq 6$ System 4 is overdetermined and hence F can be estimated by a *Singular Value Decomposition* (SVD). The camera pose can now be extracted from F:

$$F\mathbf{P_i} \sim \mathbf{p_i} \sim K\left[R \mid \mathbf{t}\right]\mathbf{P_i} \Rightarrow F \sim K\left[R \mid \mathbf{t}\right] \Rightarrow \left[R \mid \mathbf{t}\right] \sim K^{-1}F \tag{5}$$

Since by Equation 5 the pose is defined up to a scale, this scale factor should be chosen in a way that R mostly approximates a rotation matrix (i.e. $R^TR \approx RR^T \approx I, det(R) \approx 1$). Also an orthogonalization of R is advisable afterwards.

4 Perspective n-Point Problem

Formally, the Perspective n-Point (PnP) problem is defined as follows: Given a set of n 3D points $\mathbf{P_i}$ whose coordinates are known in some object coordinate frame O. Let $\mathbf{p_i}$ be a set of n 2D points which are the projections of the points $\mathbf{P_i}$ on the image plane I. Let $\vec{\mathbf{v_i}} = \overline{\mathbf{Cp_i}}$ be n directional vectors with $\mathbf{C}$ as the camera's center of perspective (Note: Since the camera is assumed to be calibrated, one can determine the vectors $\vec{\mathbf{v_i}}$ from the camera calibration matrix K). The PnP problem is defined as to determine the position of $\mathbf{C}$ and its orientation relative to O.

This problem was first solved directly in 1841 by the German mathematician Grunert [7]. In 1945 Church [2] first formulated an iterative solution algorithm which – if provided with a good starting value – constitutes an approximate solution. Since then, a great variety of different direct and iterative solution techniques were invented. However, for Computer Vision a good starting value of the camera pose is usually not known in advance and therefore the direct solutions of the PnP problem are of more interest. For those non-iterative algorithms like Dhome et al. [4], Fischler and Bolles [5], Gao et al. [6], Haralick et al. [8], Quan and Lan [17], it typically involves solving for the roots of an eight-degree polynomial with no odd terms, yielding up to four solutions, so that a fourth point is needed for disambiguation. The complexity of these algorithms varies according to [12] from $O(n^2)$ to $O(n^8)$. As for the iterative solution approaches the method POSIT by Dementhon et al. [3] has to be highlighted: Even though POSIT is an iterative algorithm with a precision increasing in each iteration step, no starting value is needed for initialization.

A non-iterative technique called EPnP (**E**fficient **P**erspective **n**-**P**oint Camera Pose Estimation) developed by Lepetit, Moreno-Noguer and Fua [12] allows the computation of an accurate and unique solution in $O(n)$ for $n \geq 4$. As in most of the existing PnP solution techniques, the idea in the implementation of EPnP is to retrieve the locations of $\mathbf{P_i}$ relative

to the camera coordinate frame. Then retrieving the camera orientation and translation which aligns both sets of coordinates is a simple task [1]. The key of the algorithm's efficient $O(n)$ implementation is, to represent the $\mathbf{P_i}$ as a weighted sum of $m \leq 4$ control points $\mathbf{C_1} \dots \mathbf{C_m}$ and perform all further computation only on these points. For large n this yields a much smaller number of unknowns compared to other algorithms and therefore accelerates further computations.

5 A priori information

Any information imaginable restricting the set of valid camera poses can be used whenever there exists a way of appropriately integrating it in a pose estimation algorithm. Two types of *a priori* information are considered:

- **Inclination information:** The inclination of the camera can be measured by inclination sensors. This reduces the three degrees of freedom for the rotation matrix R to only one degree of freedom.
- **Prior probability:** Often some subset of all possible camera poses is more probable or can be assumed. This information can be used in order to reduce the search space.

5.1 Inclination information

Inclination, in general, is the angle between a reference plane and another plane or axis of direction. The inclination relative to a surface can be measured using acceleration sensors which measure the inclination relative to each axis. Usually the sensing element consists of three acceleration sensitive masses. Daisuk Kotake et al. [10] propose a pose estimation algorithm similar to DLT being capable of integrating inclination information in the computation process. Both the sensor and the camera have their own coordinate frame. For the case where both coordinate frames are not identical, a calibration known as 'Hand-Eye Calibration' is necessary. For simplicity it is assumed that the rotation matrix R is parameterized using Euler parametrization and the sensor's world coordinate frame is identical to the camera coordinate frame. Thus the rotation R can be written as

$$R = R_x R_y R_z \stackrel{\text{def}}{=} R_{incl} R_z \tag{6}$$

This means that R can be represented as the product of three rotations R_x, R_y, R_z around each axis $\vec{\mathbf{x}} = \begin{pmatrix}1 & 0 & 0\end{pmatrix}^T, \vec{\mathbf{y}} = \begin{pmatrix}0 & 1 & 0\end{pmatrix}^T, \vec{\mathbf{z}} = \begin{pmatrix}0 & 0 & 1\end{pmatrix}^T$. The rotation around the X and Y axis R_{incl} can be measured using the inclination sensor, the remaining unknown part of the rotation is R_z. Since R_z is a rotation around the Z axis, one can write it explicitly

$$R_z = \begin{bmatrix} cos(\varphi) & -sin(\varphi) & 0 \\ sin(\varphi) & cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

At this stage one can already see that the overall rotation R, having three Euler angles as parameters, is reduced to a representation only having one degree of freedom $R_{incl} R_z(\varphi)$.

5.2 Prior probability

The a priori information can also be used to formulate probabilities regarding the final camera pose estimate (e.g. '*The camera will most likely have a small distances from the earths' surface and will not be positioned below it*', '*The camera will be flipped upside down*

Figure 4 Modeling the camera pose prior probability by mixtures of Gaussians. For simplicity, only the translation uncertainty is visualized (green ellipsoids). A real pose prior probability would normally consist of a 6D covariance. The mean values $\mathbf{x_Q}$ hereby specify the position, the covariances Σ_Q the shape of the ellipsoids.

only with a very low probability'). In context of probability theory, this prior probability is modeled using probability distributions. Depending on the pose parametrization, the pose $Q \stackrel{\text{def}}{=} [R \mid \mathbf{t}]$ can be written as a $m \geq 6$ dimensional parameter vector $\mathbf{x_Q}$ along with an invertible transformation function $T : \mathbb{R}^m \mapsto \mathbb{R}^{3\times4}$, which transforms the parametrization of the pose to the real camera pose. Mathematically the *pose prior probability* is modeled in the 6D pose parametrization space using *mixture of Gaussians* with a number of g *Gaussian components*. Each of the Gaussian components consists of a mean value $\mathbf{x_Q} \in \mathbb{R}^6$ along with a *covariance matrix* $\Sigma_Q \in \mathbb{R}^{6\times6}$ (see Figure 4).

Moreno-Noguer et al. [14] propose a method called BlindPnP where the prior probability is integrated in a pose estimation algorithm. Even though this algorithm was not intended to work with correspondences it can be easily modified for doing so, an optimization of BlindPnP is presented in [16]. In BlindPnP the prior probability is used primarily in order to increase the efficiency while identifying the outliers using an approach similar to RANSAC. The correct pose is found by minimizing the error functional

$$E(\mathbf{x_Q}) \stackrel{\text{def}}{=} \sum_{\mathbf{P_i} \longleftrightarrow \mathbf{p_i} \in M_Q} \|\mathbf{p_i} - Proj_{\mathbf{x_Q}}(\mathbf{P_i})\| + \theta |F_Q| \tag{8}$$

with $Proj_{\mathbf{x_Q}}(\mathbf{P_i})$ defined as the projection of $\mathbf{P_i}$ on the image using pose $T(\mathbf{x_Q})$ and $\theta \in \mathbb{R}$ as a penalty term that penalizes unmatched points. The minimization is computed by utilizing the prior probability: Roughly summarized, BlindPnP hypothesizes three 3D to 2D point correspondences consecutively which are compatible with the prior probability (see Figure 5). During this process, the camera pose $\mathbf{x_Q}$ evolves and the assigned covariance Σ_Q (i.e. uncertainty) reduces. In one cycle of BlindPnP all correspondences are projected on the image and checked for compatibility with the prior probability. All compatible correspondences are hypothesized in turn or marked as outliers. After three consecutive hypothesizing steps, the remaining correspondences can be checked for validity using this camera pose by projecting

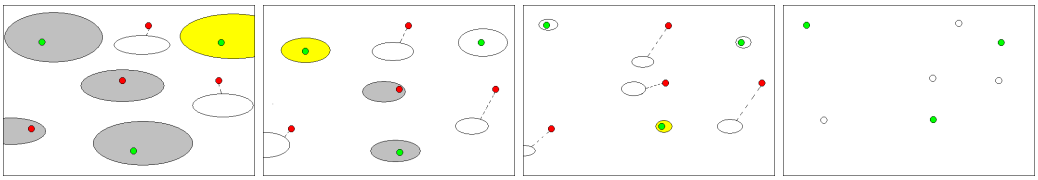


Figure 5 BlindPnP hypothesizing process: Red dots represent the 2D points which are assigned to wrong 3D correspondences (outliers), the green dots are correctly classified correspondences (inliers). The ellipses represent the region of uncertainty of the projected 3D points with the current camera pose estimate. If the ellipses are drawn in gray, the according correspondences are classified by BlindPnP as valid candidates for the hypothesizing process in the next step. The correspondence regarding the yellow marked ellipse is then selected for the current hypothesizing process. Thereby the camera pose changes (i.e. dots move) and the assigned uncertainty reduces (i.e. ellipses shrink).

the 3D points $\mathbf{P_i}$ on the image and checking their distance to their corresponding 2D point $\mathbf{p_i}$. This way the sets M_Q and F_Q can be constructed. Recursively iterating through all arguable subsets containing three correspondences valid for hypothesizing, the pose $T(\mathbf{x_Q})$ with the least error function value $E(\mathbf{x_Q})$ is then chosen as result.

6 Discussion

Nöll [15] showed that the weakness of the DLT algorithm lies in its unconstrained minimization. In the SVD solution method the vector $\mathbf{f}$ (i.e. the matrix F) is treated as unknown in all of its components with a total number of 11 degrees of freedom. However, a valid camera pose F has only 6 true degrees of freedom (three Euler angles for R and three coordinates for $\mathbf{t}$, K is known). It becomes clear that the minimization of $\mathbf{f}$ in Equation 4 does not provide a good solution F in all cases, especially if n is small or the correspondences are significantly affected by noise.

Algorithms belonging to the PnP class usually search the solution only in the valid solution space. As a consequence these algorithms are significantly more robust especially in these situations. If the correspondences contain a certain ratio of outliers, standard methods (i.e. DLT and PnP algorithms) usually utilize RANSAC in order to exclude those prior to computing the camera pose solely using the remaining consistent set of correspondences. In [15] it has been shown that the quality of RANSAC reduces as the ratio of outliers increases. At outlier ratios of 60%, in both synthetic and real data test settings, it was often not possible to estimate a robust camera pose within a reasonable time.

Two different methods have been presented that include a priori information in the camera estimation process, intending to thereby eliminate the weaknesses of the standard methods. In [10] the information is included directly in the equations which reduces the number of unknowns. Experiments in [15] showed that this results in a better runtime performance. Problematic in this approach is that in reality the inclination cannot be measured with infinite precision. This way errors are included in the estimation process. In order to receive robust estimations nevertheless, numerical optimizations such as Levenberg Marquardt have to be applied. Under those assumptions, experiments showed that this results in an even higher runtime and lower quality compared to standard methods. For correspondences containing low outlier ratios standard pose estimation algorithms such as RANSAC+EPnP are fast and robust enough even without using any a priori information. In [14] and [16] information in form of pose prior probability is used prior in order to exclude the outliers.

It was shown in [15] that for each scenario with appropriate a priori information available, there always exists an outlier ratio where these algorithms provide both more reliable and faster results than standard methods. In the evaluated scenarios this was the case for outlier ratios of 60% resp. 40% and above. A positive effect of the a priori information included in form of pose prior probability intending to identify the outliers could clearly be verified in both synthetic and real data test settings. However comparing to standard methods, BlindPnP and PPnP depend on a large number of variables which have to be assigned for each situation accordingly. Since these variables are mutually dependent, the assignment is not intuitive and usually a certain effort has to be put into testing different assignments before using the algorithms appropriately.

References

1 J. Challis. A procedure for determining rigid body transformation parameters. *Journal of Biomechanics*, 28(6):733–737, 1995.

2 E. Church. Revised geometry of the aerial photograph. volume 15, Syracuse, NY, 1945. Syracuse University Press.

3 D. Dementhon and L. Davis. Model-based object pose in 25 lines of code. *Int. J. Comput. Vision*, 15(1-2):123–141, 1995.

4 M. Dhome, M. Richetin, and J.-T. Lapreste. Determination of the attitude of 3d objects from a single perspective view. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(12):1265–1278, 1989.

5 Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

6 X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(8):930–943, 2003.

7 J. A. Grunert. Das pothenotische Problem in erweiterter Gestalt nebst über seine Anwendung in der Geodäsie. In *Grunerts Archiv für Mathematik und Physik*, 1:238–248, 1841.

8 R. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle. Review and analysis of solutions of the three point perspective pose estimation problem. *Int. J. Comput. Vision*, 13(3):331–356, 1994.

9 R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, New York, NY, USA, 2000.

10 D. Kotake, K. Satoh, S. Uchiyama, and H. Yamamoto. A hybrid and linear registration method utilizing inclination constraint. In *ISMAR '05: Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 140–149, Washington, DC, USA, 2005. IEEE Computer Society.

11 V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2:775–781, Washington, DC, USA, 2005. IEEE Computer Society.

12 V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: An accurate o(n) solution to the pnp problem. *Int. J. Comput. Vision*, 81(2):155–166, 2009.

13 D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

14 F. Moreno-Noguer, V. Lepetit, and P. Fua. Pose priors for simultaneously solving alignment and correspondence. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 405–418, Berlin, Heidelberg, 2008. Springer-Verlag.

15 T. Nöll. Robust camera position estimation by integration of a priori information. Master's thesis, University of Kaiserslautern, 2009.

16 T. Nöll, A. Pagani, and D. Stricker. Real-time camera pose estimation using correspondences with high outlier ratios. In *VISAPP 2010: International Conference on Computer Vision Theory and Applications*, pages 381–386, 2010.

17 L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):774–780, 1999.

18 Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, 2000.

On Moving Least Squares Based Flow Visualization

Harald Obermaier[1,2], Martin Hering-Bertram[3], Jörg Kuhnert[2], and Hans Hagen[1]

1 **Fraunhofer ITWM, Germany**
2 **University of Kaiserslautern, Germany**
3 **Rhine-Waal University of Applied Sciences, Germany**

Abstract

Modern simulation and measurement methods tend to produce meshfree data sets if modeling of processes or objects with free surfaces or boundaries is desired. In Computational Fluid Dynamics (CFD), such data sets are described by particle-based vector fields. This paper presents a summary of a selection of methods for the extraction of geometric features of such point-based vector fields while pointing out its challenges, limitations, and applications.

Keywords and phrases Moving Least Squares, Approximation, Flow, Scientific Visualization

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.55

1 Introduction

Computational Fluid Dynamics simulations allow the mathematically and scientifically well-founded modeling of fluid flow. Results of such simulations include the generation of pressure, velocity and temperature fields in domains with pre-defined boundary geometry for given liquids. While previous simulation techniques are mainly based on grids to construct computational meshes, modern particle-based methods, such as Smoothed Particle Hydrodynamics (SPH) and Finite Pointset Methods (FPM) [25], avoid the use of an explicit neighborhood relation in the simulation domain to cope with dynamic, rapidly changing boundary geometry and free surfaces. Due to these capabilities, grid-less methods are able to overcome challenges of industrial simulations such as stirring or mixing of multi-phase or heterogeneous liquids, without the need of frequent updates of neighborhood structures. This new generation method of flow field data poses interesting questions to the visualization community, as common visualization techniques often rely on structured data for feature extraction and rendering. In this paper, we give an overview of the main challenges arising from the lack of a pre-defined neighborhood and summarize existing work on moving-least-squares-based [16, 29] grid-less vector field segmentation. The main focus lies on the extraction of geometric flow features [24, 27, 2, 10] that can be used to segment point-based vector fields, covering an important topic of flow visualization, as (topological) segmentation [19, 30] is a key question in many of the affected application areas of meshfree CFD. *Moving Least Squares* (MLS) as the chosen reconstruction method is well accepted in the area of surface, object, and point-cloud reconstruction [1] and is also suitable for the problems posed in flow field approximation.

Section 2 gives an overview of challenges in grid-less vector field processing, proposes appropriate solutions, and discusses the concept of integral feature extraction in meshfree vector fields. In sections 3 and 4 we detail existing work on MLS-based stationary and time-varying vector field segmentation. Section 5 briefly recapitulates application areas and results of the presented papers. This paper is concluded in Section 6.

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 55–63

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2 Challenges

Challenges in the context of geometric feature extraction from grid-less fields rise from two different fields, namely i) reconstruction of a continuous field function and ii) extraction of feature geometry. On the one hand, the lack of an explicit neighborhood structure in the data set itself poses questions about the method of field approximation and domain decomposition, on the other hand, the defined geometric segmentation structures are desired to provide means for binary classification and should therefore hold information in the form of tessellations. We regard the following challenges to be the main questions to be answered in the context of point-based vector field feature extraction:

- Interpolation/Approximation schemes (Section 2.1)
- Domain decomposition (Section 2.2)
- Robust and correct boundary treatment (Section 2.3)
- Efficient visibility querying (Section 2.4)
- Definition of grid-less feature extraction methods (Section 2.5)

These challenges are detailed and discussed in the following.

2.1 Field Reconstruction

In most standard flow fields, the given grid-based simulation data implies the use of trilinear, barycentric, or higher order geometric interpolation based on the underlying cell types [26]. In contrast, the absence of a computational mesh in point-based fields requires the use of scattered data interpolation schemes [29], which can be adapted for the interpolation or approximation of grid-less flow fields [17]. A common property of these methods is the use of implicit local neighborhoods in the form of appropriate weighting functions. As such, *Radial Basis Functions* (RBF) and related techniques [6] implicitly define a spherical neighborhood around the point of evaluation. One polynomial reconstruction method is the MLS approximation scheme [16], where the quadratic distance of a polynomial f of a given degree to a set of n discrete function-values f_i at $p = (x\ y\ z)^T$ is minimized with respect to a pre-defined weighting function ω:

$$min\left\{\sum_{i}^{n}\omega(p,p_i)||f(p_i)-f_i||^2\right\} \tag{1}$$

The central advantage of grid-less approximation is the independence of a computational grid, i.e. that solutions are governed by the field's value rather than by the choice and characteristics of a (static) neighborhood structure. Moreover, concrete neighborhood structures for a field may not be defined in a unique way (cf.: decomposition of cubes into tetrahedra). However, faithful reconstruction of a data set is only possible, if the same interpolation method is used during creation, i.e. simulation of the phenomenon, and visualization of the data set. Reconstruction properties of MLS, for example, depend heavily on the chosen weighting function. In the case of ω being a two-dimensional Gaussian function, changes in the variance parameter, shape or radius of the smoothing function have great impact on the output with respect to scale space [28]. Especially in data sets with inhomogeneous particle densities, choice of the appropriate smoothing radius has an influence on the reconstructed function and is also a major factor contributing to increased computation times. The reconstructed functions are highly sensitive to changes in smoothing

length, as too small radii introduce noise or lead to singular systems, and too large radii blur important features of the field and reduce reconstruction accuracy.
One of the main reasons why grid-less approximation techniques are outperformed by grid-based methods is the computationally expensive gathering and weighting step of particle neighbors. As discussed in the next section, caching of particle data contributes to faster field approximation. In the case of MLS with a linear polynomial of the form $f(p) = c \cdot (1\ x\ y\ z)^T$, the following linear system of equations needs to be solved for c:

$$\left(\sum_i^n \omega(p, p_i) \begin{pmatrix} 1 & x_i & y_i & z_i \\ x_i & x_i^2 & x_i y_i & x_i z_i \\ y_i & x_i y_i & y_i^2 & y_i z_i \\ z_i & x_i z_i & y_i z_i & z_i^2 \end{pmatrix} \right) c = \sum_i^n \omega(p, p_i) \begin{pmatrix} 1 \\ x_i \\ y_i \\ z_i \end{pmatrix} f_i \tag{2}$$

We note, that the symmetric matrices resulting from a product of base-vectors on the left side of (2) is independent of the point of evaluation and is completely determined by the position of a data point p_i. Following this observation, matrix creation can be relocated to a pre-processing step, pre-computing one such matrix for each data point, thus removing the expensive matrix creation step from approximation. It is notable, that MLS matrices of higher order include those of lower order. Advantages of MLS over other scattered data approximation techniques are mathematical simplicity, and ease of control over polynomial degree and approximation error.

2.2 Domain Decomposition

Mesh-free data sets miss the data clustering property of a computational mesh that is available in grid-based fields. The use of local scattered data approximation techniques for function reconstruction suggests the utilization of suitable domain decomposition schemes to reduce computational complexity by restricting the processed point set during approximation [20, 9, 13]. The most important reason for using a domain decomposition scheme is locality during the neighbor gathering step of approximation with compactly supported kernels. An ideal domain decomposition scheme should have the following properties:

1. Fast identification of enclosing/neighboring cells
2. Good caching properties of it contents
3. A small memory footprint
4. Support of parallelization
5. Adaptivity with respect to particle densities

When used in time-varying vector fields, a fast reorganization of the decomposition scheme is desired as well. Properties 1.-4. can be guaranteed by a simple uniform grid layer. The memory footprint of the grid consists of storing the grid implicitly in the form of grid origin, a number of cells for each dimension and uniform values for cell width, height and length. Further memory is used by a list of indices of the points contained in every cell. Identification of relevant cells during field evaluation is as low as three division operations. Parallelization of particle advection or clustering of the domain is straight-forward in the case of the data set being shared between processes. Parallelization with distributed memory requires the inclusion of neighboring grid-cells in the form of ghost cells. Improved caching is obtained by reordering points' memory locations according to their cell membership.
This straightforward approach is however not able to fully incorporate varying particle

densities. While small variations in particle densities can be handled by reducing the width of the approximation kernel as long as the maximal radius does not exceed cell size, large variations lead to gathering of unnecessarily large particle sets in areas with high densities. Multiple layers of uniform grids, as in *Adaptive Mesh Refinement* [5], or even more adaptive data structures such as kd-trees [29] can be used in this context to allow adaptive cell sizes, if the increased time in cell identification is justified by shorter gathering times.

2.3 Boundary Treatment

There are different types of boundaries in point-based CFD [18]. Type one is the triangulated boundary geometry defined by the device or object which is subject to fluid simulation, such as parts of stirring machines, planes or vehicles. Triangulated boundaries are commonly used to represent more static behavior than free surfaces, especially their topology tends to be stationary. Type two are fluid boundaries, i.e. free surfaces, where the simulated fluid moves without being restricted by a solid geometry. This often appears in simulations of stirring devices, where liquid is moved by mixing blades and is bound to a device by gravity or viscous force only. The third type of boundary is a multi-phase or fluid interface between different liquids. Boundaries of the latter two cases are generally free surface boundaries/field discontinuities and require special treatment [18].
During feature extraction and field approximation, a position outside of the data set is detected either by a boundary collision or visibility test with a triangulated boundary as described in the next section, or by analysis of the neighboring data points. In the latter case, statistics [15] like a large empty angular segment in the neighborhood or a point (the domain of the filter kernel) indicates, that the current point of evaluation is outside of the data set, see Figure 1 (a). While this covers most convex boundary shapes, accuracy in boundary detection decreases near concave regions. Similarly, smoothness measures [11] can be used to approximate free boundaries.

2.4 Visibility Queries

In grid-based visualization, visibility queries in vector fields play a minor role, since the computational mesh defines data set boundaries and effectively separates the domain of the flow field from flow obstacles. However, in meshfree particle systems, visibility queries have to be executed prior to the particle gathering step during field approximation to avoid that particles located on opposing sides of obstacles influence each other [23] as shown in Figure 1 (b). Such visibility queries can either be performed during neighborhood-finding [4, 3] or rely partly on a pre-computed visibility data structure, subdividing the field volume into regions with homogeneous visibility properties, as provided by Binary Space Partitioning or related schemes [7]. The former method is based on run time ray-triangle intersections to determine visibility of a certain data point from a point of evaluation. As higher order approximation techniques require a large number of neighboring points, the amount of required visibility tests for one vector field approximation often has a large impact on performance. Thus, visibility tests have to be both reduced to a minimum number and optimized performance wise.

2.5 Extraction and Segmentation Techniques

Geometry extraction in vector fields is often motivated by industrial mixing applications, where domain experts need to be able to analyze mixing quality of different devices. While

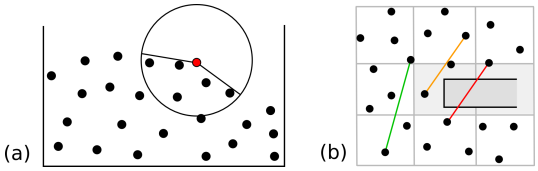


Figure 1 (a) Point of evaluation (red) outside of a free surface. (b) Visibility querying by ray testing. One query (green) is omitted, as affected cells do not contain obstacles. Two other rays are checked for intersection, one of which detects an obstacle (red).

certain other aspects of mixing, such as pressure and temperature are subject to optimization as well, velocity defines the central criterion for parameter optimization, as it directly influences the behavior of real and virtual material interfaces. Real interfaces between different liquids, as well as virtual interfaces defined by the user, device geometry or field topology and their evolution over time and space need to be analyzed in stationary fields and time-varying vector fields in order to be able to interpret the output of a complex mixing process. As demonstrated in the remainder of this paper, a selection of flow structures can be captured by the definition of integral flow features.
Integral flow features [12] are traces described by infinitesimal particles as they are advected through a vector field $f : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$. Positions p on these traces are defined by the solution of the basic governing differential equation $\frac{dp}{dt} = f(p(t), t)$, with $p(t_0) = p_0$. Work summarized in the following sections is concerned with feature extraction using integral line or surface geometry S at time t as defined by (3).

$$S(r, s, t) = c(s) + \int_{t-r}^{t} f(S(r - (t - x), s, x), x)dx \tag{3}$$

where $c : \mathbb{R} \rightarrow \mathbb{R}^n$ is a univariate point seeding curve parametrized by s. For integral lines, c is a point. Individual instances of lines or stream particles are identified by their age parameter $r \in [0, t]$. The following sections summarize a selection of existing work in the context of MLS based grid-less vector field processing that rely on the methods presented in the previous sections.

3 Extraction and Segmentation I: Stationary Flows

The boundary of instantaneous vector fields is static by definition. The work by Obermaier et al. [23] focuses on the topological segmentation of stationary mixing processes, see Figure 2. It is observed, that in stationary fields of mixing processes, flow obstacles such as mixing blades are the primary source of flow separation due to the lack of critical points or lines in practical applications. Thus, separation and attachment lines on inner flow obstacles are detected on the projected two-dimensional flow field by the criteria defined by Kenwright et al. [14] and used as source for the generation of integral separation structures. As such, separation and attachment lines define lines, where flow meets an obstacle or separates from it, stream surfaces integrated in forward or backward direction at these rakes represent three-dimensional separatrices. Modifications [23] to previous stream surface integration techniques [12, 10] to accommodate data set boundaries that violate no-slip conditions to allow line integration on triangulations as well as surface splitting caused by different flow obstacles yield a set of such separation surfaces. Identification and processing of different surface-surface intersections allows the geometry-based computation of surface segmentation. The resulting sets of stream

surface segments are reorganized and combined to form distinct, non-overlapping stream volumes that effectively segment the underlying domain into topologically homogeneous volumes.

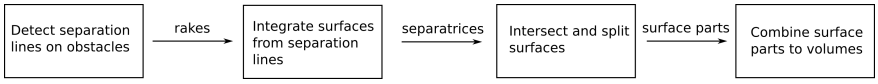


Figure 2 Steps of stationary stream volume extraction.

4 Extraction and Segmentation II: Time-Varying Flows

A more challenging and physically more realistic setting are time-varying data sets. In time-varying fields, changing data set boundaries and particle sets usually require the re-computation of domain decomposition or grid information for every time-step.

4.1 Two-Dimensional

In Obermaier et al. [22], a streak line based approach for the segmentation of two-dimensional, time-varying flow simulations into continuous areas, so called streak areas is introduced. Data set boundaries with non-vanishing velocities are incorporated in the streak line creation process, as they can be the source of streak line stretching, attaching and separation behavior. Streak lines are continuously seeded at user specified positions, indicating initial material boundaries. As the streak line is advected by the flow field, individual segments of the line are subdivided if either a maximal segment length is exceeded, or angles between neighboring segments fall below a given threshold, thus achieving curvature adaptivity and maintaining a pre-defined streak line resolution. Solid flow obstacles with non-zero boundary-velocity require special attention during the line generation process. In every time step, all line segments are tested for collisions with such obstacles. If a collision is detected at time step t_1, the affected segment is recursively split in half in $t_0 = t_1 - 1$ and the resulting two segments are advected and again tested for collision with a boundary in time t_1. This is repeated until no more segment-obstacle collisions are detected. Streak particles whose trajectory hits the boundary move along the projected field on the boundary, until they are released at separation points [14].

4.2 Three-Dimensional

In the three-dimensional case, streak surfaces provide means for segmenting a time-varying vector field when seeded at initial material interfaces or separation curves. A method to generate curvature adaptive streak surface geometry in grid-less flow is presented in [21], where MLS is used for vector field approximation as well as for surface curvature approximation. Particles seeded at a rake in a time-varying vector field naturally define a point-based streak surface. The true problem in accurate streak surface generation is to guarantee a certain desired surface mesh resolution. One solution to this adaptivity problem is to maintain a curved Delaunay-type surface mesh [8] by edge flipping as the surface is advected and deformed by the flow field. Triangle circumradii of this streak surface mesh serve as a measure of streak particle density and are used to insert new particles at regions with high surface curvature. As large Delaunay triangles reflect a low particle density, high MLS curvature at such regions indicates the need of additional particle insertion. In order to

reduce surface artifacts resulting from bad particle insertion positions and increase accuracy of adaptive surfaces when compared to high resolution streak surfaces, particle insertion positions in highly curved regions are traced back in time until a time step with a sufficiently low surface curvature is reached, before actual particle insertion is performed. Figure 5 shows four streak surfaces at different time-steps, parting a time-varying vector field into two volumes.

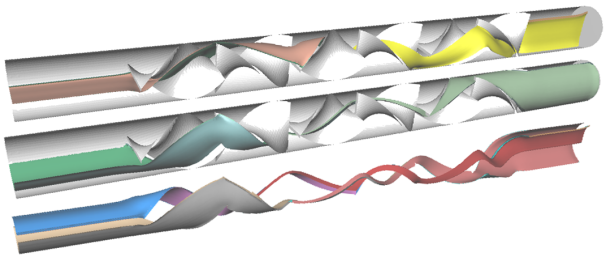


Figure 3 Visualization of two single stream volumes and a volume pair (bottom). The data set is segmented into a total of 64 distinct non-overlapping volumes.

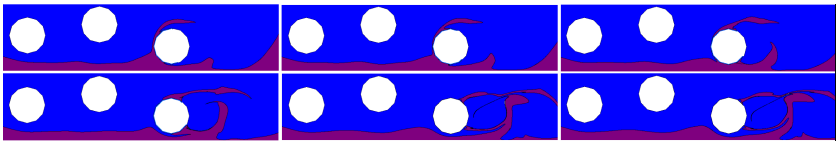


Figure 4 A sequence of six consecutive streak areas. The data set is segmented by a user specified material interface represented by a streak line. Intersections of the line with flow obstacles as well as cases where the streak line separates from the boundary geometry can be observed.

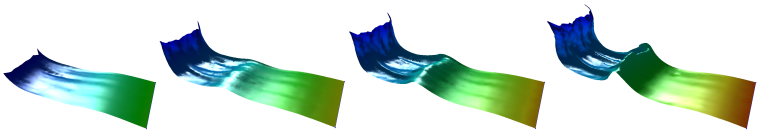


Figure 5 Four consecutive renderings of a segmenting streak surface colored according to seed time.

5 Results and Applications

The given examples demonstrate integral line and surface generation as tool to segment data sets for further analysis, processing and as general visualization method. Figure 3 depicts instantaneous flow in an industrial mixing process, however in reality time-dependent vector fields as shown in Figures 4 and 5 are more common, while sharing properties with stationary fields with respect to segmentation techniques. The shown results provide information about flow segmentation and allow domain experts to perform parameter optimization of mixing devices. The presented solutions to the main challenges of grid-less vector field segmentation are robust and applicable in stationary, as well as time-varying cases. Distinct segments of the data set that are bounded by integral feature geometry allow the computation of volumina, facilitate geometric intersection of different overlapping segments and form a basis

for binary classification of points in the data set and voxel based visualization techniques such as volume rendering.

6 Conclusions and Outlook

We have given an overview of advances in the field of MLS-based grid-less flow visualization and pointed out its challenges and capabilities. It was shown, that segmentation of such vector-fields is of extreme importance for certain industrial applications, providing insights about mixing and interface evolution. While the advantages of grid-less flow fields such as meshless modeling of boundaries and interfaces and grid independence carry over to the field of visualization by avoiding cell identification and reconstruction artifacts caused by grid characteristics, the drawback of expensive field approximation and visibility querying remains a hurdle on the way to real-time surface generation in point-based vector fields. The presented challenges of grid-less vector fields are subject to ongoing research and new solutions are expected to improve expressiveness and performance of approximation, extraction, and visualization techniques. Further work is required in the field of domain decomposition and distribution techniques, furthermore, a promising direction of future research is the study of topology changes caused by parameter variations in scattered data interpolation schemes.

References

1 M. Alexa, J. Behr, D. Cohen-Or, S. Fleishmann, D. Levin, and C. T. Silva. Point set surfaces. In *Vis. '01*, pages 21–28, 2001.

2 B. G. Becker, N. L. Max, and D. A. Lane. Unsteady flow volumes. In *Proc. of the 6th Conf. on Vis. '95*, page 329, 1995.

3 T. Belytschko, Y. Krongauz, M. Fleming, D. Organ, and W. K. S. Liu. Smoothing and accelerated computations in the element free galerkin method. *J. Comp. Appl. Math.*, 74(1-2):111–126, 1996.

4 T. Belytschko, Y. Y. Lu, and L. Gu. Element-free galerkin methods. *Internat. J. for Numerical Methods in Engineering*, 37:229–256, 1994.

5 M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comp. Physics*, 82(1):64–84, 1989.

6 T. Bobach and G. Umlauf. Natural neighbor concepts in scattered data interpolation and discrete function approximation. In *GI LNI, Visualization of Large and Unstructured Data Sets*, pages 23–35, 2007.

7 D. Cohen-Or, Y. L. Chrysanthou, C. T. Silva, and F. Durand. A survey of visibility for walkthrough applications. *IEEE TVCG*, 9(3):412–431, 2003.

8 R. Dyer, H. Zhang, and T. Möller. Delaunay mesh construction. In *Eurographics Sym. on Geometry Processing*, pages 273–282, 2007.

9 M. S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *J. Comp. Appl. Math*, 73:65–78, 1996.

10 C. Garth, H. Krishnan, X. Tricoche, T. Bobach, and K. I. Joy. Generation of accurate integral surfaces in time-dependent vector fields. *IEEE TVCG*, 14(6):1404–1411, 2008.

11 T. Gutzmer and A. Iske. Detection of discontinuities in scattered data approximation. *Numerical Algorithms*, 16:155–170, 1997.

12 J. P. M. Hultquist. Constructing stream surfaces in steady 3d vector fields. In *Proc. of Vis '92*, pages 173–175, 1992.

13 A. Iske and J. Levesley. Multilevel scattered data approximation by adaptive domain decomposition. *Numerical Algorithms*, 39:187–198, 2005.

14 D. N. Kenwright, C. Henze, and C. Levit. Feature extraction of separation and attachment lines. *IEEE TVCG*, 2(5):135–144, 1999.

15 S.-B. Kim. Eliminating extrapolation using point distribution criteria in scattered data interpolation. *Comp. Vis. Image Underst.*, 95(1):30–53, 2004.

16 D. Levin. The approximation power of moving least-squares. *Mathematics of Computation*, 67:1517–1531, 1998.

17 S. Li and W. K. Liu. *Meshfree Particle Methods.* Springer Berlin, 2007.

18 G. R. Liu and M. B. Liu. *Smoothed particle hydrodynamics: a meshfree particle method.* World Scientific, 2003.

19 K. Mahrous, J. Bennett, G. Scheuermann, B. Hamann, and K.I. Joy. Topological segmentation in three-dimensional vector fields. *IEEE TVCG*, 2(10):198–205, 2004.

20 L. B. Montefusco and C. Guerrini. A domain decomposition method for scattered data approximation on a distributed memory multiprocessor. In *EDMCC2: Proc. Dist. Memory Computing*, pages 274–282, 1991.

21 H. Obermaier, M. Hering-Bertram, J. Kuhnert, and H. Hagen. Generation of adaptive streak surfaces using moving least squares. In *Proc. of Dagstuhl Scientific Vis. Seminar (to appear)*, 2009.

22 H. Obermaier, M. Hering-Bertram, J. Kuhnert, and H. Hagen. Volume deformations in grid-less flow simulations. *Comp. Graph. Forum (Proc. of EuroVis)*, 28(3):879–886, 2009.

23 H. Obermaier, J. Kuhnert, M. Hering-Bertram, and H. Hagen. Stream volume segmentation of grid-less flow simulation. In *Topological Data Analysis and Vis.: Theory, Algorithms and Applications (Proc. of TopoInVis), (to appear)*, 2009.

24 T. Schafhitzel, E. Tejada, D. Weiskopf, and T.s Ertl. Point-based stream surfaces and path surfaces. In *Graph. Interface 2007*, pages 289–296, 2007.

25 S. Tiwari and J. Kuhnert. Finite pointset method based on the projection method for simulations of the incompressible navier-stokes equations. *Springer LNCSE: Meshfree Methods for Partial Differential Equations*, 26:373–388, 2002.

26 S.-K. Ueng, C. Sikorski, and K.-L. Ma. Efficient streamline, streamribbon, and streamlube constructions on unstructured grids. *IEEE TVCG*, 2(2):100–109, 1996.

27 W. von Funck, T. Weinkauf, H. Theisel, and H.-P. Seidel. Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments. *IEEE TVCG*, 14(6):1396–1403, 2008.

28 J. Van De Weijer and R. Van Den Boomgaard. Least squares and robust estimation of local image structure. *Internat. J. of Comp. Vision*, 64:143–155, 2005.

29 H. Wendland. *Scattered Data Approximation.* Cambridge University Press, 2004.

30 A. Wiebel, X. Tricoche, and G. Scheuermann. Extraction of separation manifolds using topological structures in flow cross sections. In *Topology-Based Methods in Vis. II*, pages 31–44, 2009.

Modeling and visualizing urban sprawl and carbon footprints in Phoenix metropolitan area*

Sebastian Petsch[1], Subhrajit Guhathakurta[2], and Hans Hagen[3]

1 **University of Kaiserslautern, Germany**
petsch@cs.uni-kl.de
2 **Arizona State University, USA**
subhro.guha@asu.edu
3 **University of Kaiserslautern, Germany**
hagen@cs.uni-kl.de

Abstract

Urban planners are dealing with problems of urban sprawl and CO_2 emissions. The multidimensional character of these phenomena requires new analysis and visualization tools that are unavailable in platforms like the Geographical Information Systems (GIS). This paper, first, presents an approach for measuring and monitoring urban sprawl and carbon footprints. Second, it offers a three-dimensional visualization method that takes into account the multi-dimensional nature of the data. The visualization of the data is based on an intuitive approach involving B-Splines and Bezier techniques to create three-dimensional surfaces. Finally the paper introduces an analysis tool for planners and decision makers to examine household carbon footprints in relation to their direct spatial neighborhood based on unstructured census data.

Keywords and phrases Urban Sprawl, GHG Emissions, Visualization, Case Studies

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.64

1 Introduction

In the era of globalization and climate change urban planners are dealing with problems related both to urban sprawl and increasing CO_2 emissions and even their interrelationships. Given the multidimensional and multidisciplinary aspects of these problems, the conceptual and visualization tools planners have typically employed are often inadequate [1]. This paper demonstrates how high bandwidth of visualization methods in computer science, especially the techniques developed in the field of information visualization, can be applied in planning problems related to urban sprawl and carbon emissions. The advances in computer technology provide a unique opportunity to use digital visualization techniques to represent planning issues especially in public communication and participation programs [2] [3].

While taking advantage of an existing method [4] to measure urban sprawl, the paper applies a new model for calculating carbon footprints at the level of individual households, which includes emissions from electricity use as well as from consumption behavior. By using a new form of three-dimensional visualization to present the results of both applications, the paper demonstrates how this form of analysis and visualization can support urban planning. The remainder of the paper is structured as follows. First, the paper introduces and describes the first application topic, urban sprawl, in section 2. In section 2.2, a new visualization

* This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG, grant number 1131) as part of the International Graduate School (International Research Training Group, IRTG) in Kaiserslautern, Germany.

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 64–72

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

tool using Coons Patches is presented in order to visualize the parameters of urban sprawl developed in section 2.1. The second application topic, carbon footprints at the level of individual households, follows in section 3. Here the paper presents a new model to measure carbon footprints at the level of individual households followed by an introduction of a visual analysis tool presented in section 4. The paper concludes by a short summary and a future outlook in section 5.

2 Urban Sprawl

2.1 Introduction urban sprawl

The term suburbanization is used to describe the process of movement of population from central areas of cities and towns to peripheral areas. This phenomenon can have a number of reasons associated with it such as high density of cities, pollution by industry or high levels of traffic. Suburbanization causes an increase urban sprawl. Urban sprawl is reported to be a significant contributor to traffic congestion, job-housing mismatches, racial and income segregation and environmental degradation [5]. The literature in urban sprawl is vast and there have been several efforts to define urban sprawl. An acceptable definition is found in Clawson [6]: "[the] rapid spread of suburbs across the previously rural landscape, tendency to discontinuity [...]". In other words urban sprawl can be defined as low density, leapfrog, commercial strip development and discontinuity [4] [7]. These common definitions bring us to one of the essential parts of this work; that is to develop quantitative indicators to measure urban sprawl. There are several studies that have derived indicators of urban sprawl [5] [8]. The research applies some of the measures introduced by Galster et al. [4], as a basis for indicator calculations in this study. Galster et al. defined urban sprawl as [...] a pattern of land use in an urban area that exhibits low levels of some combination of eight distinct dimensions: density, continuity, concentration, compactness, centrality, nuclearity, diversity, and proximity. The research reported in this paper uses three of these dimensions: density, continuity and diversity.

2.2 Visualizing indicators of urban sprawl

Based on Galster et al. [4] sprawl indicators were calculated and forecasted for Maricopa County, AZ, for 2000 to 2030. The study area is subdivided into one square mile grid cells. The analysis is based on the demographic projection data from a software-based simulation model called UrbanSim [9]. UrbanSim delivers those data on a household-per-grid-cell basis for the period of projection. Among others the data include the number of households by type, number of workers, and land use. The available simulation results over the next 30 years offer the possibility of visualizing trends in urban sprawl indicators over a medium-term future. As already mentioned in section 2.1, the most significant indicators for urban sprawl are density, continuity and diversity. The following definitions are adapted from Galster et al. [4]. Density is defined as "[...] the average number of households per square mile of developable land in the total area." That means in this case the number of residential units per grid cell. Density is the most widely used indicator of sprawl. Continuity is defined as "[...] the degree to which developable land has been developed in an unbroken fashion throughout the total area." In other words, leap-frog areas are considered to be more sprawl-like. Diversity is defined as "[...] the degree to which substantial numbers of two different land uses exist within the same area." Greater diversity values of land uses within a given area are considered as the opposite of sprawl. In other words, the interpretation of this

indicator is the average density of a particular land use (measured by number of households) in another land use's (measured by the number of employments) area. For each of those three indicators the higher indicator numbers indicate less sprawl. One main objective of this paper is to provide a compact visually pleasing three-dimensional visualization of urban sprawl. The research uses the location co-ordinates of the grid cell centre points as well as the calculated indicator values as a basis for our surface construction. First, a height-field comprised of the centres of the grid cells and the selected values as their heights is generated. Second, based on this height-field, a surface with C^0 continuity is constructed [10]. Figure 1 shows an example with a regular grid marked in green, its height field in red/orange, and the resulting surface in yellow.

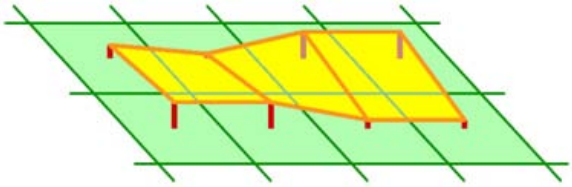


Figure 1 Regular grid (green), dedicated height-field (red/orange) [13].

The surface is built by tessellating the faces of the height-field using linear Coons Patches (a mathematical technique to describe and construct surfaces) [11]. With this approach the appearance of a smooth surface is maintained even though the surface is only C^0 continuous. It remains visible at the silhouettes, which is not the applications focus. Therefore, this method achieves a very good trade-off between the visualization's speed and quality. One advantage of this approach is its flexibility and adaptability. For example, the perspective and color of the surface can be changed (also see Figure 5 and 6). Furthermore the user is able to switch between a plain background and our background (Figure 2), created by using Google Earth [12].

This approach also offers the possibility to compare the results of different years by overlaying different surfaces by using different layers or transparency (see section 3.3: Figure 5 and 6).

3 Carbon footprints

3.1 Introduction carbon footprints

The urgency of reversing climate change is among the most pressing international concerns. One of the major anthropogenic contributors to the changing climate is carbon dioxide emission from burning fossil fuels. Given that global energy needs are overwhelmingly dependent on fossil fuels, reducing its use would require finding alternative energy sources together with increasing energy efficiency. Energy is needed for every aspect of human production and consumption - from extraction, to manufacturing, to transportation and finally for disposal of waste products. Therefore knowing what humans consume and the total energy required for making this consumption possible would provide a reasonable estimate of the carbon footprint of individuals. This paper offers a method to determine carbon footprints of cities and urban regions from consumption patterns of households and offers novel ways of visualizing this footprint. In this study the term "carbon footprint" describes the total amount of CO_2 emitted into the atmosphere by individuals and organizations

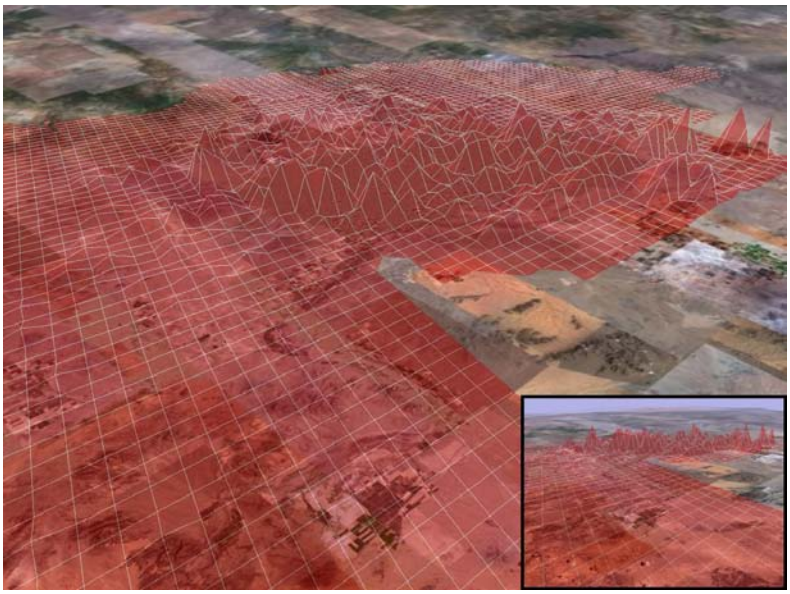


Figure 2 Density values for the year 2005 with "Maricopa County" background as well as different perspectives [10].

through the use of fossil fuels [13]. The paper describes an approach for estimating carbon footprints for different types of households and for different scenarios of housing development.

The focus of this approach is on the neighborhood scale. Thus the paper provides information about the distribution by type of households. The growing interest in monitoring and measuring carbon footprints has resulted in several studies across the globe. Among the path-breaking projects for monitoring CO_2 in the United States is The Vulcan Project [14], funded by NASA and led by a research team at Purdue University. It has achieved in quantifying United States fossil fuel CO_2 emissions at the scale of individual factories, power plants, roadways and neighborhoods. The results are represented on a common 10 km grid to facilitate atmospheric modeling. Weber [15] [16] defined 13 broad consumption categories of household level carbon footprints, for example education, home energy, and private transport. His study demonstrates, among other things, that the CO_2 emissions are proportional to the household expenditures. For 2004 household consumption totaled 8100 billion dollars and resulted in 5700 million-tons of CO_2, for an average CO_2 intensity of consumption of about 0.7 kg CO_2 per dollar. This suggests that each American household would have been responsible for 50 tons of CO_2 in 2004, if all of the production took place in the U.S. (without implicating import and export). Referring to studies focusing on national average results, he also concludes that global and distributional aspects would be important to consider in order obtaining an accurate picture of carbon emissions. Other studies such as Sovacool and Brown [17] and Wentz et al. [1] also offer alternative approaches for the calculation of carbon footprint.

3.2 Measuring carbon footprints at the level of local neighborhoods

When focusing on household carbon footprints this research distinguishes between different contributors of CO_2 emissions, namely energy (electricity) and consumption behavior. To complete a total household carbon footprint, future work will also include transportation

emissions. The basic information for the emission caused by the consumption behavior of different households is derived from the Consumer Expenditure Diary Survey (CES) for the year 2006 [18]. The CES provides detailed data about the consumption items per household as well as information about different types of household units. In other words we know how much is spent on a particular item by a certain category of household. These categories are generated for different family sizes, by different classes of income as well as by different races. The carbon intensity of each consumption item for each type of household was provided by the CES and the Berkeley institute of the environment [19] (Economic Input-Output Life Cycle Assessment EIO-LCA 10). Particular emission estimates were done for 42 different household types [13]. In this approach the consumption component is calculated based on the amount of money spent by a household for a particular item and the CO_2 emissions coefficient of for this item. With the variables KR, KF, KI in our matrix we can choose between the different values for income class, race and family size. C is a constant which does not depend on time.

$$\begin{bmatrix} E_R \\ E_F \\ E_I \end{bmatrix}(t) = \begin{bmatrix} K_R \times coeff \\ K_F \times coeff \\ K_I \times coeff \end{bmatrix} = C \times \begin{bmatrix} K_R \\ K_F \\ K_I \end{bmatrix}(t)$$

$$R = race, F = familysize, I = incomeclass, t = certainyear$$

Figure 3 shows an example of the expenditure distribution and the resulting CO_2 emissions for the household type 612. These resultant emissions give us baseline figures for the daily consumption decisions of specific household types. Comparisons can be made of carbon emissions of different consumables as well as the total emissions for different types of households.

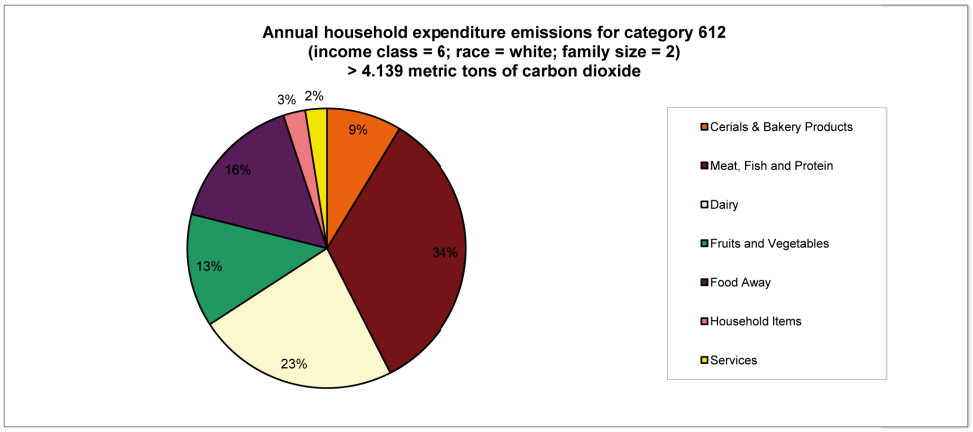


Figure 3 Household emissions for category 612 [13].

Furthermore detailed information on the total electricity consumption (in kWh) of every mentioned household category [20] is available. In conjunction with the energy coefficient for Arizona, precise amount of CO_2 emissions for each type of household as well as for all households in Maricopa County is calculated. The calculation of the household electricity component of emissions was based on the electricity consumption of each household category (KX) and the energy coefficient (C) for Arizona. Finally the emissions from household energy use and the emissions from the consumption items are combined for a total carbon footprint

at the household level. The distribution of the CO_2 emissions in metric tons by all of our 42 different household types is shown in Figure 4.

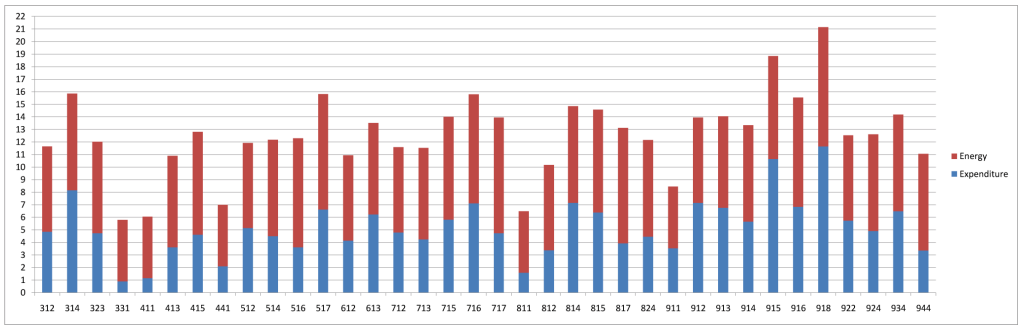


Figure 4 Distribution of CO_2 emissions in metric tons by household type [13].

3.3 Visualizing carbon footprints

Using the visualization tool, described earlier in section 2.2, results for household carbon footprints are shown three-dimensionally. To demonstrate the utility of this visualization method, calculations of carbon footprint in Maricopa County were made for two different development scenarios (emission values are in tons of CO_2 per grid-cell). The scenarios chosen for this exercise show the difference in development patterns between allowing state lands to be auctioned as per current rules ("BAU" scenario) and the alternative of freezing all state owned lands in Maricopa county to 2005 levels ("Stateland" scenario).

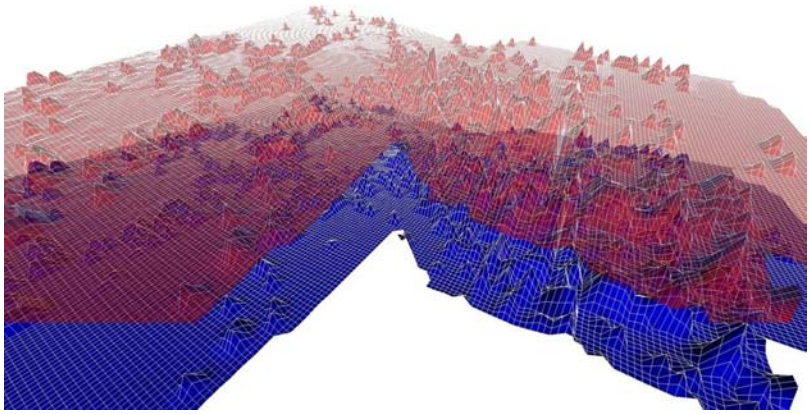


Figure 5 Transparent overlay of household emissions in tons of CO_2 per grid-cell for BAU 2020 and Stateland 2020 [21].

Using transparency and different surface layers (Figure 5), two datasets are compared in a way that shows specific areas of CO_2 pikes and how they are different in the other scenario. Figure 6 illustrates the differences between the total carbon footprints for BAU and Stateland in 2010. The grid cells are colored in blue if the BAU emissions are higher than the Stateland emissions and red in the opposite case. The height of each grid cell illustrates the absolute difference between the two scenarios. As expected, the BAU scenario provides

more sprawling and leapfrog development while the scenario Stateland (with no development on state lands) shows higher CO_2 emission values in the urban area of Maricopa County [13].

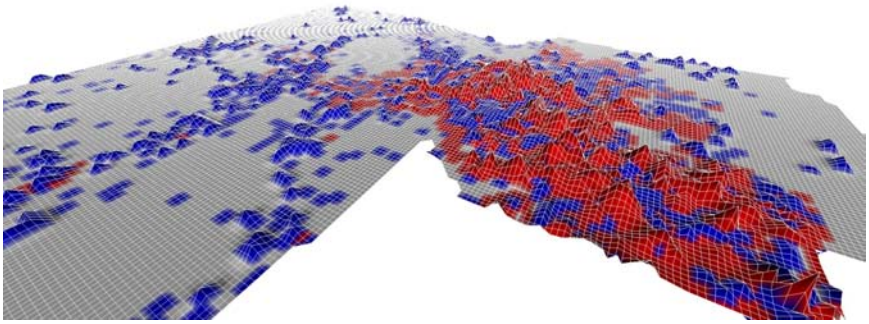


Figure 6 Differences between scenario emissions BAU and Stateland in 2010 [13].

4 Visual analysis tool

In this part of the paper the visualization toolbox is extended with a visual analysis tool which is able to demonstrate different quantities of CO_2 emissions for specific household categories in relation to the neighborhood. The visualization method presented earlier relies on an underlying grid-cell-based structure (one square mile grid cells). The motivation for this work is to create a visual analysis tool that is based on census tracts in order to cut down the workflow and to directly access census data. A novel type of diagram for a two-dimensional space subdivision in cells with weighted generator points is developed which builds on the original geometric construction of a Voronoi Diagram [22]. The weighting of each generator point is based on the calculated carbon footprints (section 3.2) for different household categories. The objective is to illustrate the differences in weights to the nearest neighbors. By doing so, differences depending on different household attributes or different numbers of households are shown. The analysis focuses on cell deformations depending on neighborhood values. The sizes of the new resulting cells represent the carbon footprint distribution for the census tracts in relation to the direct neighborhood cells. Figure 7 shows these resulting spatial subdivision diagrams based on average results of household carbon footprints for a particular part of Phoenix.

This distribution represents the typical household category in each census tract. To provide a better orientation, an underlying background map of this particular sector of Phoenix, adopted from Google Earth, is also included. By restricting this weighted approach to direct neighbors, the resulting cells always stay within the original borders of the study area which makes it easy to understand and reconstruct. Thus any other possible application within the same study area (within the same census tracts), such as visualizing urban sprawl indices, can be achieved by this method. Since this approach is based on U.S. census data, it can be adapted to any other metropolitan area in the United States.

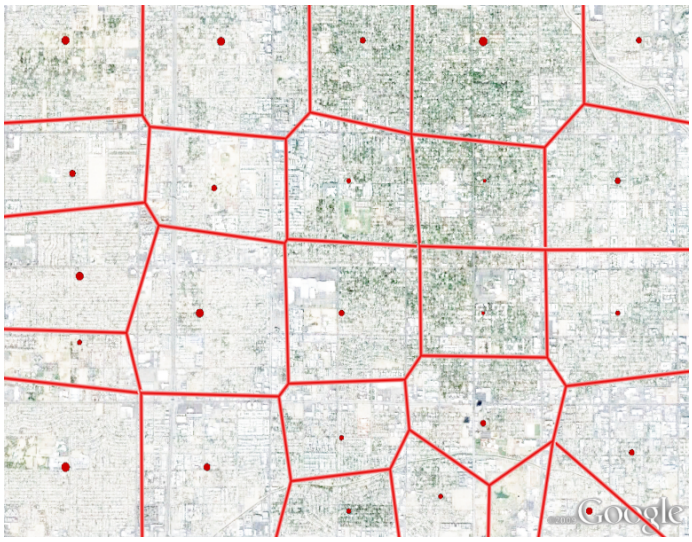


Figure 7 The resulting diagram for average household carbon footprint values per census tract [22].

5 Conclusion

The visualization approach presented in this paper provides an efficient way to visualize urban sprawl indicators as well as carbon footprint values three-dimensionally. Height fields were selected because in order to show multidimensional data (using different layers or multidimensional data within one layer). Possible alternatives, like the use of different colours, have to be considered carefully because of possible difficulties with coloured single spots that are hard to detect. Furthermore the paper presented a model to calculate carbon footprints at the level of urban neighborhoods. The high number of different household categories makes this model unique and gives the user a detailed tool for estimating the impacts of daily consumption decisions by household type as well as the resulting amount of CO_2 emissions. To complete a total carbon footprint at the level of households, future work will also include emission parameters from the transportation sector, for which we distinguish between different modes of transportation, the spatial locations of different household categories as well as different purposes of trips. By developing a novel approach for two-dimensional space subdivision in cells using weighted generator points, household carbon footprints in relation to their neighborhoods can be represented visually. This will enrich the toolbox for urban planners in analyzing and counteracting the causes of carbon footprints within a defined local area. As mentioned before, future work will contain other applications such as visualizing urban sprawl indices and combining both urban sprawl and carbon footprint results to see possible causes and correlations.

Acknowledgements This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG, grant number 1131) as part of the International Graduate School (International Research Training Group, IRTG) in Kaiserslautern, Germany. The author would like to thank the collaborating colleagues Daniel Engel and Luc Heischbourg as well as the supervisors Subhrajit Guhathakurta and Hans Hagen for their support.

References

1 E. A. Wentz, P. Gober, R. C. Balling, T. A. Day. Spatial Patterns and Determinants of Winter Atmospheric Carbon Dioxide Concentrations in an Urban Environment. *Annals of the Association of American Geographers*, 92(1):15–20 March 2002.

2 K. Al-Kodmany. Using visualization techniques for enhancing public participation in planning and design: process, implementation, and evaluation. *Landscape and Urban Planning*, 45(1):37–45, 15 September 1999.

3 K. Appleton, A. Lovett. GIS-based visualization of development proposals: reactions from planning and related professionals. *Computers, Environment and Urban Systems*, 29(3):321–333, Futurescapes, May 2005.

4 G. Galster, R. Hanson, M. R. Ratcliffe, H. Wolman, S. Coleman, J. Freihage. Wrestling sprawl to the ground: defining and measuring an elusive concept. *Housing Policy Debate*, 12(4):681–717, 2001.

5 R. Ewing. Is Los Angeles-style sprawl desirable?. *Journal of the American Planning Association*, 63(1): 107–126, 1997.

6 M. Clawson. Urban Sprawl and Speculation in Suburban Land. *Land Economics* 38(2):99-111, 1962.

7 Y.-H. Tsai. Al-Kodmany. Quantifiying Urban Form: Compactness versus Sprawl. *Urban Studies*, 42(1):141–161, 2005.

8 W. Fulton. Who sprawls most? *How growth patterns differ across the U.S.*. Washington, DC, USA, 2001.

9 P. Waddell. UrbanSim: Modeling Urban Development for Land Use, Transportation and Environmental Planning. *Journal of the American Planning Association*, 68(3):297–314, 2002.

10 S. Petsch, L. Heischbourg, K. Mueller, S. Guhathakurta, H. Hagen. Visualizing Urban Sprawl. Conference Compendium of IEEE Visualization Conference 2008, Columbus, OH, USA, 2008.

11 G. Farin. *Curves and Surfaces for CAGD, A practical Guide*. Wiesbaden, Germany, 1994.

12 Google. `http://earth.google.com`, Official website, accessed September 2008.

13 S. Petsch, L. Heischbourg, K. Mueller, S. Guhathakurta, H. Hagen. Modeling, Monitoring, and Visualizing Carbon Footprints at the Urban Neighborhood Scale. submitted to Journal of Urban Technology 2010.

14 Vulcan Project. `www.purdue.edu/eas/carbon/vulcan/research.html`, Official website, accessed March 2010.

15 C. L. Weber. *Trade, consumption, and climate change: an input-output study of the United States*. Carnegie Mellon University, Pittsburgh, PA, USA, 2008.

16 C. L. Weber, H. S. Matthews. Quantifying the global and distributional aspects of American household carbon footprint. *Ecological Economics*, 66(2-3):379–391, 2008b.

17 B. K. Sovacool, M. A. Brown. Twelve metropolitan carbon footprints: A preliminary comparative global assessment. *Energy Policy*, In Press, Corrected Proof, 2009.

18 U.S. Department of Labor, Bureau of Labor Statistics, Division of Consumer Expenditure Survey. *Consumer Expenditure Diary Survey 2006*. CD-Rom, 2008.

19 Berkeley institute of the environment. *Cool Climate Calculator*. `http://coolclimate.berkeley.edu/documentation`, accessed March 2010.

20 Energy Information Administration. *Official Energy Statistics from the U.S. Government*. `http://www.eia.doe.gov/pub/oiaf/1605/cdrom/pdf/e-supdoc.pdf`, accessed March 2010.

21 S. Petsch, L. Heischbourg, K. Mueller, S. Guhathakurta, H. Hagen. Monitoring und Visualisierung von Carbon Footprints im urbanen Raum. *Proceedings of REAL CORP*, 2009.

22 D. Engel, S. Petsch, S. Guhathakurta, H. Hagen. Visual Analysis of Carbon Footprint Indicators in Urban Planning. submitted to *Information Visualization Journal*, 2010.

Advanced Visualization and Interaction Techniques for Large High-Resolution Displays

Sebastian Thelen[1]

1 **University of Kaiserslautern**
67653 Kaiserslautern, Germany
thelen@cs.uni-kl.de

Abstract

Large high-resolution displays combine the images of multiple smaller display devices to form one large display area. A total resolution that can easily comprise several hundred megapixels makes them suited for the visualization of data sets that could not be perceived entirely on desktop PCs or laptops due to their size. At the same time, user collaboration benefits from an extended screen area that facilitates interaction with screen contents as well as interaction among users. This paper discusses the challenges and opportunities of large high-resolution displays and examines ways to set up display clusters both in terms of hardware and underlying software technology. Furthermore, it investigates how to effectively harness the computational power and resources of rendering clusters to visualize giga-scale data sets. Last but not least, traditional interaction metaphors and their scalability to large displays as well as the effect of new techniques on the user experience are discussed.

Keywords and phrases Large High-Resolution Displays, HCI, Interaction, Visualization, Distributed Rendering

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.73

1 Introduction

Large high-resolution displays are getting common and have entered research laboratories, conference halls, and presentation rooms. Due to their size, they astonish the audience whenever used in tech demos or presentations. Exploiting their capabilities in order to be more than just giant displays showing scaled-up versions of existing applications is a challenging task. Programmers need reliable middleware to implement their applications without having to worry about low-level aspects of the system. Collaboration requires new interaction methods that scale with the size of the display and the size of the user group to offer intuitive ways to interact with each other and the application. Furthermore, since large displays are usually driven by a cluster of render nodes, sophisticated techniques are required to exploit the computational resources of the cluster and make use of the enormous resolution comprised in these systems for the visualization of giga-scale data sets.
The following sections give an overview of visualization and interaction techniques for large high-resolution displays. Each of the above mentioned aspects is addressed in more detail, and possible solutions to the challenges are presented.

2 System Aspects

This section discusses hardware- and software-specific aspects of large high-resolution displays. Typical display systems combine the resolution of multiple smaller devices (tiles) to form one large display area. Up to two tiles are driven by a single compute/render node, and nodes are

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 73–81

OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

(a) (b) (c)

Figure 1 Possible large high-resolution display configurations: (a) monitor-based tiled display wall, (b) projector-based tiled display wall, (c) Cave Automatic Virtual Environment (CAVE™).

connected via high-speed network to share data and communicate with each other. Hence, high-resolution displays can be described as network-based distributed render clusters.

2.1 Large High-Resolution Display Systems

There exist various ways to build large high-resolution displays, mainly differing in the number and type of display devices used in the setup [10]. Figure 1 illustrates three of the most common large display configurations.

Monitor-based tiled display walls (see figure 1a) combine the resolution of multiple LCDs and easily achieve a total resolution of several hundred megapixels. Since LCD screens have become cheap, monitor-based systems are an affordable way to set up large high-resolution displays, and compared to other approaches, their calibration is rather simple and mainly consists of mounting all tiles on a proper monitor rack. A major problem of monitor-based systems are screen bezels, which lead to discontinuities in visualizations and result in an ever-present French window effect.

Projector-based systems (see figure 1b) consist of multiple computer projectors that are arranged in a grid and project onto a large screen. Compared to monitor-based tiled walls, projector-based systems are capable of forming a truly seamless display area, which requires a careful projector calibration. However, calibration is challenging, since projectors differ in terms of luminance and color temperature. Also, geometric distortions have to be considered, which are due to lens effects and their location relative to the screen.

Cave Automatic Virtual Environments (CAVEs™) are non-planar projector-based installations that consist of up to six screens representing the sides of a cube/box (see figure 1c). CAVEs™ are immersive display systems in which users are literally surrounded by the virtual environment. Often, they use additional devices, such as tracking suits or stereo glasses, to increase the realism of the virtual environment.

2.2 AnyScreen - A Distributed Rendering Library

Once the hardware is set up, a distributed rendering library is required to drive the displays and take care of synchronization between nodes, data distribution, frustum culling, and I/O events. Rendering libraries can be characterized in various ways. One of them is in terms of the graphics pipeline stage, in which geometry is assigned to the tiles of a display cluster [9]. Figure 2 illustrates the stages of the graphics pipeline and the conversion of geometry data into pixel data.

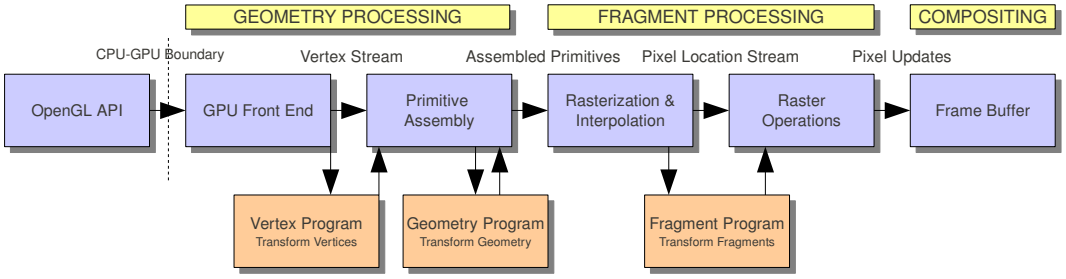


Figure 2 High-level representation of the graphics pipeline.

Sort-first libraries assign raw primitive data in the primitive assembly stage. This stage converts geometry data from three-dimensional object space into normalized screen space. After each node has been assigned its parts of data, the remaining pipeline stages are executed independently.

Sort-middle systems assign screen-space primitives between the primitive assembly stage and the rasterization stage.

Sort-last approaches assign actual pixel/fragment information in the rasterization stage. The rasterization stage is the last stage of the pipeline in which a series of fragment tests (e.g., scissor, alpha, and depth tests) are performed to determine whether and how a fragment is visualized as a pixel on the screen.

An alternative way to classify rendering libraries is in terms of the underlying execution mode [10]. Two different approaches have emerged: client-server and master-slave systems (see figure 3).

Client-server systems (see figure 3a) consist of a client node and a set of render servers. The client node is responsible for handling I/O events and is the only cluster node that runs an instance of the application. Render jobs are issued via network to the render servers, which means that the amount of network traffic depends on the complexity of a scene.

Master-slave systems (see figure 3b) execute an instance of the application on each node of the cluster. The master node in addition has to handle I/O events and forward these via network if necessary. Compared to client-server systems, the network traffic in master-slave setups is rather low, which makes this approach especially suited for the visualization of complex scenes.

AnyScreen [2] is a scalable and lightweight rendering library developed to drive arbitrary display configurations and support a variety of different stereo modes. The library consists of a set of C++ classes for managing synchronization and communication between executing threads and provides automatic viewport handling for each tile of a display cluster. AnyScreen implements a master-slave execution model, i.e., an instance of the application is executed on each compute node. Furthermore, a static sort-first approach is used to assign geometry to individual tiles in the primitive assembly stage. The library's visualization component is based on OpenGL and SDL (Simple Direct Media Layer), inter-node communication and synchronization have been implemented using MPI (Message Passing Interface). AnyScreen is flexible and has been used to drive a variety of display configurations, such as monitor-based tiled walls, stereoscopic Powerwalls, and different display-in-display configurations, such as combinations of 2D and 3D projector systems.

AnyScreen's advantage over other distributed rendering libraries is its high degree of versatility. CGLX [3], for instance, is a rendering framework with a similar architecture as AnyScreen. However, CGLX was specifically designed to drive monitor-based tiled systems and thus

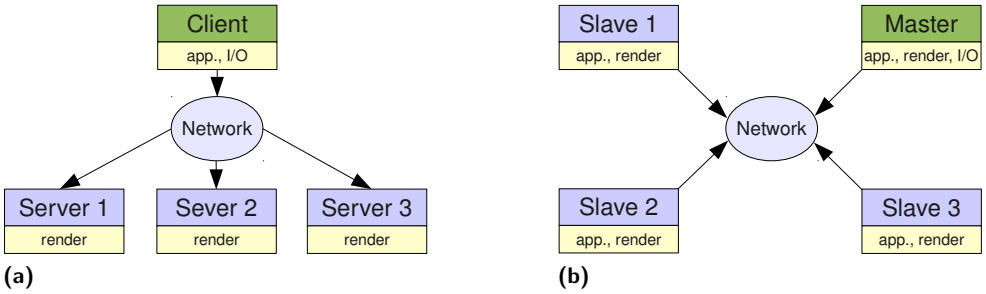


Figure 3 Execution modes: (a) client-server mode, (b) master-slave mode.

hardly adapts to more exotic display configurations. Chromium [5] is a client-server system supporting both sort-first and sort-last rendering. However, with increasing data size, network bandwidth may pose a limit to this technique, as clients constantly need to issue render requests to the servers.

2.3 Tiled++ - An Enhanced Tiled Hi-Res Display Wall

Monitor bezels are an inherent problem of monitor-based tiled display walls, causing effects similar to a French window and distracting users. Bezels either lead to discontinuities when they are completely ignored, i.e., lines are slightly offset when they cross the boarder between two adjacent screens, or they hide information when treated as overlays. Both approaches affect user perception and potentially lead to misinterpretations of a scene.
Tiled++ [4] is an approach that explicitly addresses the bezel problem. The idea is to enhance the bezel area of monitor-based display walls with the image of one or more computer projectors that are placed at arbitrary positions in front of the display wall. The projectors provide information that would normally be lost or disrupted on the bezel area, which for this reason has been covered with diffuse reflecting card board. In order to avoid optical interference, projectors only project onto the covered bezel area and spare the display area of the LC panels. They have been carefully calibrated using a camera-assisted homography-based calibration tool to compensate linear distortion effects that are due to a non-perpendicular placing of projectors in front of the wall (by placing projectors beneath the ceiling or at the side of a large display, one can prevent users from entering the projector beam and causing shadows).
Figure 4 shows an example that compares the results achieved with Tiled++ to those achieved when treating monitor bezels as an overlay to the scene. With the overlay method, bezels hide the company logo so that it is difficult to determine the type of the car or the manufacturer. Tiled++ provides the missing information on the bezel grid and the car can be easily identified.

3 Interaction Aspects

A major question when working with large high-resolution displays is how to interact with applications that run on a 200 megapixel tiled display wall or in a CAVE™? One can employ traditional input devices and metaphors designed for desktop PCs, such as keyboards, joysticks, computer mice, or tablets. Depending on the application or type of display that is used, however, one might consider using more sophisticated input devices that better fit

Figure 4 Tiled++ provides important image information that would otherwise be hidden or disrupted due to monitor bezels.

the needs of the user and allow to navigate data sets faster and more intuitively. Therefore, devices that are commonly used in virtual reality applications are full body tracking suits, data gloves, or head mounted displays.

When it comes to collaboration in front of large high-resolution displays, many input techniques suffer a couple of common issues. From an interaction point of view, the most severe one is that very often one person takes the lead and controls the application, while the rest of the group members depend on that one person navigating the data set. Furthermore, many approaches do not *scale* with the *size of the display* or the *size of user groups* working with the application. The mouse cursor metaphor, for example, hardly scales with the size of a display. In order to cope with the increasing distances that a cursor has to travel, one has to increase the mouse sensitivity, thereby making it harder for users to select small items on the display wall. Magnetic tracking on the other hand works fine for single users or very small groups, but equipping dozens of users with full body tracking devices is not feasible for obvious technical and financial reasons. Further questions that arise in the context of collaborative work with large displays are:

- How is it possible to support *user groups* and present information that is *customized* based on the status of a user (e.g., researcher and student)?
- How is it possible to deal with confidential information and implement a set of *security mechanisms*, so that not every bit of information can be accessed by every user?
- How can *independent data exploration* be realized for every user without running into scalability issues?

3.1 Large Display Interaction using Visual Tags

The interaction approach presented in this section explicitly addresses the problems and challenges discussed in previously. The technique is based on camera-enabled cell phones and two-dimensional barcodes (visual tags) acting as links between users and the application on a large display [12, 11]. The main idea is to label a set of data items using two-dimensional barcodes and display them in the application. Users employ their cell phone camera to scan and decode these barcodes with a special tag reader software installed on their mobile device. The information encoded in each tag is used to trigger a series of web services that, in their simplest form, display information about the data item on the cell phone screen. However,

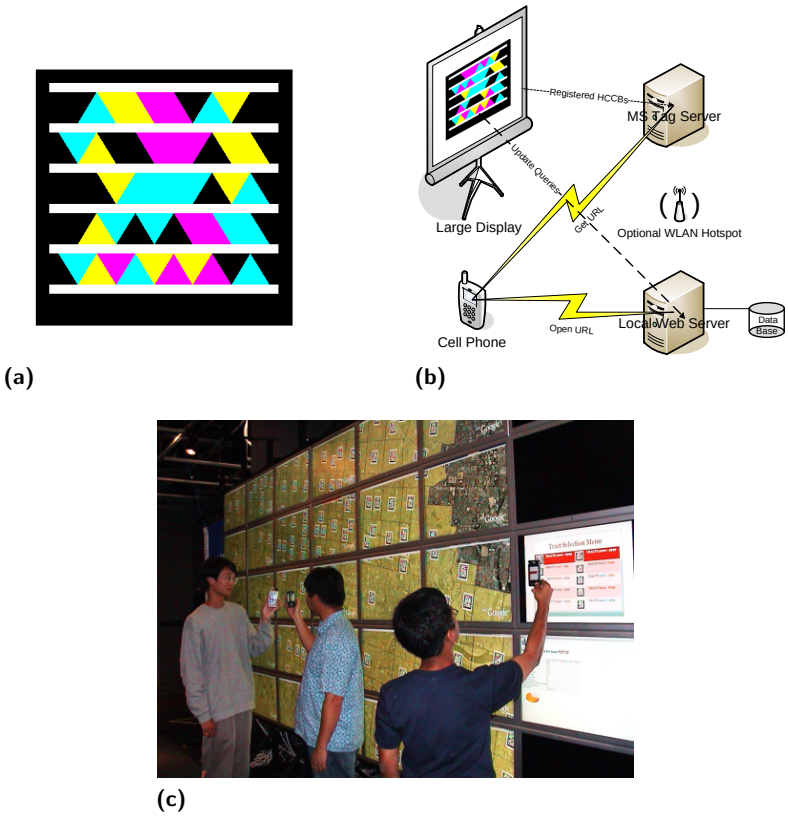
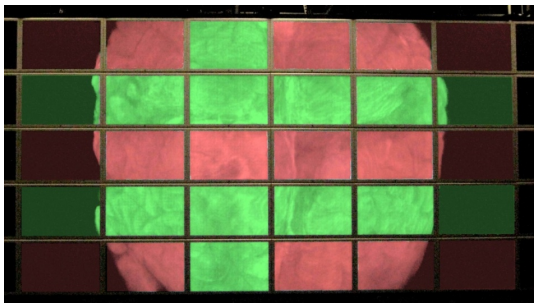


Figure 5 Tag-based interaction with large high-resolution displays: (a) two-dimensional HCCB, (b) general system setup, (c) collaboration in front of a 200 megapixel display.

more complicated interaction forms can be implemented that, for example, modify data sets or influence the visualization on the display.

The application presented here makes use of a special type of visual tag called *High Capacity Color Barcode* (HCCB) (see figure 5a). HCCBs have been invented by Microsoft (MS) and are supposed to be an alternative to already existing barcode systems. The general setup for an application using HCCBs as an interaction method is depicted in figure 5b. Data has been labeled with a set of barcodes and is visualized on a large display. The information encoded by each tag links to an entry on a MS tag server. The tag server stores the information that has been registered with the tag when it was created. At the current stage, this information is a unique URL to a web server in the local network. The cell phone retrieves the URL from the MS server in order to open it in a web browser. Querying the URL triggers a series of Python CGI scripts on the web server that generate the HTTP output to be displayed in web browser.

The approach is used in a Google Earth-based application on a fifty tile display cluster for the investigation of census data (see figure 5c). Census data consists of several hundred census tracts, each tract associated with a high-dimensional feature vector. The center of each census tract is labeled with a two-dimensional barcode that can be scanned in order to retrieve information about that tract. The system supports further interaction concepts, such as *information layers*, *public screens*, *description balloons*, and *level-of-detail information querying* that are also based on the HCCB approach (see Thelen et al. [12] for details).



(a)

(b)

(c)

Figure 6 Multi-resolution volume rendering on a 200 megapixel display wall: (a) tiles display at different levels of detail, (b) mandible data set displayed at wavelet level 1 (corresponds to $1/8 = 12.5\%$ level of detail), (c) mandible data set displayed at wavelet level 0 (corresponds to 100% level of detail).

Other smart phone-based interaction techniques featuring two-dimensional tags have been presented previously [1, 6, 7]. However, a majority of these methods seek to offer large display alternatives for navigation in typical window environments, i.e., selection, scrolling, and zooming. Instead, the focus of the HCCB approach is on exploiting two-dimensional codes to overcome common issues in collaborative large display environments.

4 Visualization Aspects

In order to handle the enormous resolution comprised in large display systems, tiled walls are usually driven by a cluster of render nodes that communicate via network. Render nodes are often built from off-the-shelf commodity components and drive up to two tiles of a display wall. They basically resemble modern desktop PCs, i.e., core 2 duo processors, 2-4 GB RAM, 512-1024 MB VRAM.

A major reason for using large display systems is their ability to display giga-scale data sets at a much higher level of detail than single monitor systems due to their pixel count. The Visual Human Project, for instance, provides 40 GB of volumetric data for the female cadaver. However, special techniques are required to visualize this enormous amount of data using the resources of the render cluster.

4.1 Handling Large Volumetric Data Sets

The volume rendering system depicted in figure 6a combines octree-based space subdivision with a (Haar) wavelet-based multi-resolution representation [8] to display large volumetric data sets at fairly interactive rates on a 200 megapixel tiled display wall. The algorithm to visualize data sets is divided into three stages:

- Per tile octree-based frustum clipping.
- Wavelet-based level of detail (LOD) determination.
- Direct volume rendering using 3D texture mapping.

The first stage of the algorithm determines for each tile of the display wall, which subvolume of the data has to be rendered on each tile and discards all other parts in order to make optimal use of system resources. The second stage of the algorithm determines the maximum level of detail, based on the Haar wavelet representation, that lets subvolumes still fit into the video memory of the respective render node. Once the subvolume has been transferred to the GPU's texture memory, 3D texture mapping is used to render the data.
As illustrated in figure 6a, the final rendering consists of multiple tiles, each displaying its part of the data set with highest possible quality. In figure 6a, all green tiles display data at wavelet level 0, corresponding to a level of detail of 100%. All red tiles display data at wavelet level 1, which corresponds to $1/8 = 12.5\%$ of the detail comprised in the original data. The difference in terms of visual quality is depicted in figure 6b and figure 6c for the mandible data set. The results shown in figure 6b correspond to what can be achieved on a single monitor PC without frustum clipping and multi-resolution representation by downsampling the entire data set to make it fit into the texture memory of the graphics card. Figure 6c shows the results achieved with the octree-wavelet data structure. Especially in the upper half, figure 6b is blurry and does not reveal as many details as figure 6c.

5 Conclusion and Future Work

Large high-resolution displays provide exciting research opportunities in various disciplines of computer science. In order to make users benefit from the applications executed on large displays, numerous challenges have to be addressed. The paper illustrated these benefits cannot be achieved without rethinking concepts of human computer interaction and developing new visualization techniques. At best, this is supported by reliable middleware abstracting from underlying system details and providing transparent programming interfaces to developers. First solutions have been proposed, yet the fact that most laboratories/institutes reimplement from scratch shows more research is required to provide standardized methods and concepts for setting up large high-resolution systems.

Acknowledgements This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) as part of the International Graduate School (International Research Training Group, IRTG 1131) in Kaiserslautern, Germany.

References

1 Rafael Ballagas, Michael Rohs, and Jennifer G. Sheridan. Sweep and point and shoot: phonecam-based interactions for large public displays. In *CHI '05: CHI '05 extended abstracts on Human Factors in Computing Systems*, pages 1200–1203, New York, NY, USA, 2005. ACM.

2 Matthias Deller, Sebastian Thelen, Daniel Steffen, Peter-Scott Olech, Achim Ebert, Jan Malburg, and Joerg Meyer. A Highly Scalable Rendering Framework for Arbitrary Display and Display-in-Display Configurations. In *CGVR*, pages 164–170, 2009.

3 Kai-Uwe Doerr and Falko Kuester. CGLX Project (accessed September 16, 2010). `http://vis.ucsd.edu/~cglx/`.

4 Achim Ebert, Sebastian Thelen, Peter-Scott Olech, Joerg Meyer, and Hans Hagen. Tiled++: An enhanced tiled hi-res display wall. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):120–132, 2010.

5 Greg Humphreys, Mike Houston, Ren Ng, Randall Frank, Sean Ahern, Peter D. Kirchner, and James T. Klosowski. Chromium: a stream-processing framework for interactive rendering on clusters.

6 Seokhee Jeon, Jane Hwang, Gerard J. Kim, and Mark Billinghurst. Interaction with large ubiquitous displays using camera-equipped mobile phones. *Personal Ubiquitous Comput.*, 14(2):83–94, 2010.

7 A. Madhavapeddy, D. Scott, D Sharp, and E. Upton. Using camera-phones to enhance human-computer interaction. In *Sixth International Conference on Ubiquitous Computing (Adjunct Proceedings: Demos)*, 2004.

8 J. Meyer, R. Borg, B. Hamann, K.I. Joy, and A.J. Olsen. Network-Based Rendering Techniques for Large-Scale Volume Data Sets. In *Farin, G., Hamann, B. and Hagen, H., eds., Hierarchical and Geometrical Methods in Scientific Visualization*, pages 283–296, Heidelberg, Germany, 2002. Springer-Verlag.

9 Steven Molnar, Michael Cox, David Ellsworth, and Henry Fuchs. A sorting classification of parallel rendering. *IEEE Computer Graphics and Applications*, 14(4):23–32, 1994.

10 Tao Ni, Greg S. Schmidt, Oliver G. Staadt, Robert Ball, and Richard May. A Survey of Large High-Resolution Display Technologies, Techniques, and Applications. In *VR '06: Proceedings of the IEEE conference on Virtual Reality*, page 31, Washington, DC, USA, 2006. IEEE Computer Society.

11 Sebastian Thelen, Joerg Meyer, Achim Ebert, and Hans Hagen. A 3D Human Brain Atlas. In *Modelling the Physiological Human. Proceedings of the second 3D Physiological Human Workshop*, 2009.

12 Sebastian Thelen, Joerg Meyer, Ariane Middel, Peter Scott Olech, Achim Ebert, and Hans Hagen. Tag-based interaction with large high-resolution displays. In *Proceedings of the 4th IASTED International Conference on Human-Computer Interation (IASTED-HCI)*, pages 356–363, 2009.

Open Problems in Computational Steering of Massive Parallel Unstructured Grid Based CFD Simulations

Christian Wagner[1]

1 **German Aerospace Center**
Brunswig, Germany
christian.wagner@dlr.de
and
University of Kaiserslautern
Kaiserslautern, Germany
wagner@cs.uni-kl.de

Abstract

Traditionally, analysis of flow fields resulting from computational fluid dynamics (CFD) calculations is a sequential process. The flow area defined by surrounding geometry is tessellated, a mesh is generated and divided into subregions, transferred to a cluster or supercomputer and the result is transferred back. Then, a variety of post-processing tasks should give insights to the physical problem. At that point, parameters chosen wrong can be identified and the simulation has to be done again with tweaked parameters. This is an iterative process that can be time consuming, especially if one iteration lasts more than a few days. In general, aiming at reducing the simulation times by shortening the time used to identify wrong parameters results in high productivity enhancements.

In this paper, the need for on-line monitoring and computational steering approaches for massive parallel unstructured flow simulators are presented with aircraft design as one of many possible application domains. This involves software integration aspects, data streaming and explorative visualization. Many challenges still have to be solved and this paper summarizes most important ones.

Keywords and phrases Computational Steering, CFD simulation, Interactive Visualization, Explorative Visualization, Virtual Reality

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.82

1 Introduction

Aircraft design has many challenges in the near future forcing the design to heavily rely on numerical simulations. The number of expected passengers in 2020 is as triple as high as nowadays number and the price per passenger has to be reduced by half. Additionally, aviation has to deal with governmental restrictions like massive reductions in emissions and noise levels.

These requests need technological leaps and cannot be achieved through continuous improvements of traditional techniques like the continuous improvements of wing shapes. New aircraft configurations have to be discovered and it becomes necessary to account for the whole flow field around an aircraft. The aim in numerical CFD research is to enable aircraft simulations over its entire flight.

To support engineers in the development of advanced designs, CFD as the main future design tool also needs to provide high usability. The nowadays very sequential CFD work-flow

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 82–89

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

is therefore the topic of this paper. Traditionally, CAD design, mesh generation, simulation and post-processing are separate tasks, most of them have effort of more than a few days. This sequential pattern has the effect that scientific results as well as errors are recognizable at very late stages and decreases the efficiency of the whole work-flow.

The following paper is structured as follows. In the next section we will discuss how CFD work-flows can be enhanced with computational steering methods. Then, we will cite work related to the stated CFD work-flow. After that, we will discuss the open problems of software integration, data streaming and explorative visualization needed to be solved followed by a conclusion section.

2 CFD Optimizations

Computational Fluid Dynamics became the most important tool to give researchers insight into complex flow structures. Different strategies can be used to support results more quickly. On the one hand, the CFD simulation itself can be improved by using faster algorithms and hardware acceleration. On the other hand, the overall work-flow can be restructured to enable higher responsiveness for simulation systems. For the latter approach, this chapter will discuss the benefits of computational steering capabilities.

2.1 Computational Steering to optimize CFD work-flows

The traditional work-flow used in computational fluid dynamics is very sequential. The tasks of setting up the flow conditions, the simulation and the analysis of the results are strongly separated. If some parameters were set inappropriately they have to be tuned and the work-flow starts again from its beginning. Since the simulation itself can last for a few days or longer, the process is full of very long waiting periods.

Especially in production processes long waiting periods have to be avoided. Different approaches to overcome these problems are still in development. In-situ visualization approaches try to move most of the analytic calculations into the simulation that have to be done in the post-processing otherwise. Besides the challenge to bring visualization algorithms to the same scaling as the simulation code, sophisticated knowledge about the expected results are needed.

Another approach is the usage of computational steering techniques. Computational Steering normally comes in combination with on-line monitoring and visualization of the ongoing simulation, giving the ability to evaluate the actual solver runs. Interactive steering then allows to tweak simulation parameters and guide the running simulation. The resulting fast feedback gives a lot of potential to the researchers. Having the possibility to change simulation parameters on the fly shortcuts the traditional sequential work-flow, because inappropriate parameters can be identified much earlier before the post-processing task and can be tuned immediately instead of starting at the setup task again. Also, being capable of fixing issues in the underlying simulation mesh like multi-block decompositions or mesh refinement levels can guide the simulation to a faster convergence. Last but not least, through getting an immediate visual feedback the researcher gains additional insights in the simulated effects.

Even in the setting of attaching a computational steering system to an ongoing simulation, a fast and effective visualization is needed for the analysis of not well-known physical phenomena. Explorative visualization has proven to support researchers many times in the past. But the high amount of data in physically correct computational fluid dynamics still challenges data management as well as visualization algorithms.

Some of the open problems needed to solve to make such a steerable system available are presented in section 4.

3 Related work

To support the computational steering approach described in the last section we are introducing a very general solution that future research will aim on, as described in section 4.1. This section gives an overview of work with close relation to at least one aspect to our whole system view.

The idea to use visualization guided computational steering systems is not new. [5] describes a framework to develop steered algorithms, but focuses more on the combination of algorithms. To allow for a better visual responsiveness [6] introduces a parallel pipeline-driven front-end. A more CFD-specific system can be found in [8]. An overview of the different approach of in-situ visualization can be found in [12].

The benefits of virtual environments for the analysis of flow phenomena are discussed in [11]. [18] introduces a parallel back-end to support post-processing in virtual environments. These two articles also discuss the advantages of interactive and explorative post-processing.

In the field of progressive streaming a lot of work has already been done. [17] shows how to use space-filling curves to reorganize data on regular grids and give them a multi-resolution meaning that can easily be streamed. For unstructured meshes the additional problem with indexing schemes from cells to nodes arises. [7] therefore introduces an interleaved streaming file format that is used in [13] to enable for streaming iso-surfaces. These streaming formats are used for online monitoring in [16] for a simulation on regular grids and in [3] for progressive volume rendering of unstructured grids.

The addressed problems of computational fluid dynamics systems in this paper are guided by the TAU flow solver used and developed by the German Aerospace Center [20]. [19] shows how this system was wrapped into a python-scripting interface.

4 Open Problems

As described in section 2.1, design processes relying on CFD simulations can benefit a lot from computational steering capabilities. In this section, we will summarize the work-flow with computational steering capabilities as an aimed solution in section 4.1. After that, we will discuss arising problems. We will describe software architecture problems in section 4.2, communication problems in section 4.3, and finally visualization issues in section 4.4.

4.1 Aimed solution

The aimed solution proposed for CFD work-flows can be found in figure 1. Instead of iterating through the time-consuming traditional work-flow again and again until an appropriate solution is found setup and output should be necessary only once.

After the computation of the simulation is started it can be observed via online monitoring. For this purpose, data from the simulation is transferred to the visualization system and the running simulation can be evaluated immediately. To help the viewer in unknown situations, explorative visualization techniques will be at hand to find the interesting features in the simulation. In the field of computational fluid dynamics this can be various flow specific visualization techniques like the interactive seeding of stream lines and stream surfaces as well as interactive iso-value selection techniques for the scalar fields in the simulation. If inappropriate parameters or meshing is found the researcher can change them on the fly

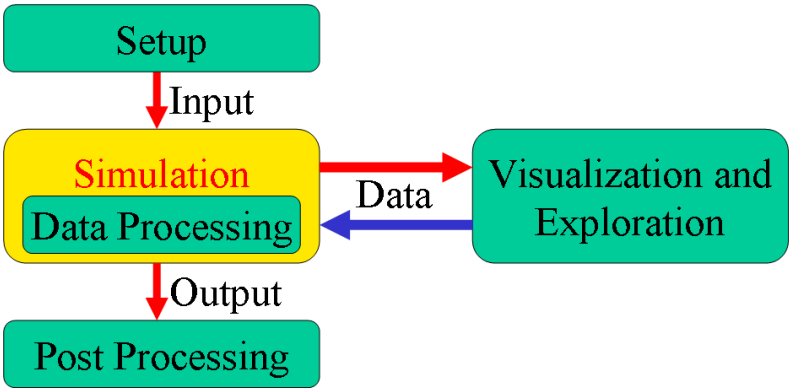


Figure 1 Aimed solution. A running simulation can be observed by an On-line Monitoring visualization. Explorative visualization guides the viewer through unknown behavior. For the determination of visualization features causing high CPU load a Data Processing module assists the visualization module. Since the simulation can be guided to appropriate solutions, setup and has to be done only once. Finally, the one-time written output has to go through additional post processing steps only once.

and the simulation will continue with the new values. Since the analysis in the traditional post-processing task involves normally also higher order visualization techniques, e.g. vortex region and vortex core line extraction, these should also be possible. For that reason, a module supporting these tasks should be combined in the system.

For the realization of such a system a long way is still to go. As shown in the last section, a lot of the problems are already solved or at least got research attention. But there are still many open questions. Some of them are described in the following sections.

4.2 Software Architecture

To enable CFD simulations for computational steering a system architecture with supporting functionalities has to be developed.

Traditional simulation systems were built as a collection of executables concerning different steps of the simulation. Therefore, the simulation was done by calling these programs in a way suitable for the simulation purpose, each of them reading input data from the file system and writing output data for the following one. This results in a huge file I/O overhead, that is nowadays canceled out by wrapping the functionalities in a scripting wrapper, the Python language proved to be effective, and keeping the data in memory. As an additional advantage for the researcher, the growing amount of functionalities can be combined in a wide and fine-granular variety enabling new and optimized simulation settings.

A computational steering system has to take account of the manifold functionalities of these simulation scripts while not constraining the researcher in the freedom of writing simulation scripts. Therefore, well-designed and flexible interfaces are required. The question how to build these interfaces between simulation, computational steering system and visualization is still unsolved. Some interfaces to bring data out of the simulation are used in existing on-line monitoring systems. Mainly copying raw geometry and field data, they are not very flexible and the simulation has to deal with the on-line monitoring data format. The way back into the simulation is even harder and a computational steering system has to know all the parameters that can be changed.

It is not very likely that a common interface standard can be established, however, application domain-specific computational steering standards could eventually evolve like the CGNS file format did for saving fluid dynamic data.

In the existing computational steering approaches concurrency between the simulation and the visualization system was addressed seldom. Normally, the data is copied between every iteration step to the visualization as well as changed parameters.

Especially, when the visualization becomes more distant to the simulation system, as described in section 4.3, this becomes an unwanted scenario. If the time to transmit data becomes much longer than the time to calculate one iteration step concurrent transmission and visualization of the data is very reasonable for not slowing down the simulation while monitoring a running simulation. However, from computational steering point of view the changes into the simulation have to be serialized or the simulation might have to get back to the time step the viewer is actually seeing.

4.3 Communication

The system architecture needed for large CFD simulations is running distributed on large supercomputers or cluster systems and communication is therefore inherent in computational steering. For on-line monitoring data has to be transferred to the visualization host. In the first computational steering systems the visualization host was located nearby the simulation host and the bandwidth compared to the amount of data was high.

This changes in the usual setting of computational fluid dynamic simulations. In figure 2 the computational environment often found in research institutes is depicted. On the one hand, you can find a simulation cluster or supercomputer with a very high bandwidth and low latency intercommunication network. On the other hand, for visualization purposes a different system is used, sometimes as an additional small visualization cluster system with a virtual reality system. Normally, these two systems are connected by only a local area network with a rather low bandwidth and high latency. Additionally, the computation systems mostly have only one or a few login nodes with high firewall protections that can be used to transfer data. The separation becomes even much stronger if distant visualization is used in a collaborative system supporting a few visualizations at very distant locations.

The usage of such a system causes auxiliary trouble for computational steering systems. The concurrency issues where already discussed in section 4.2. Through domain decomposition techniques additional blocking of the data is introduced that should be handled without slowing down the simulation by using a traditional transfer through the simulations master node.

To deal with low bandwidth, progressive streaming approaches were already introduced years ago. This streaming technique is to reorganizing data to show results from the very beginning of data transmissions. First, a lower quality preview is transmitted providing quick overviews. Further data is then gradually increasing details. For data on structured grids this is a solved problem for most applications and for data on unstructured grids some cases out of the variety are already addressed.

For the usual kind of computational fluid dynamics systems shown in section 4.3 new challenges come into play. View-dependent progressive streaming algorithms have to take into account the additional domain decomposition that should not be handled by loading the simulations master node.

In order to additionally minimize the necessary data to transfer, domain specific algorithms should be introduced handling the conditions of computational fluid dynamics situations. For unsteady simulations grid adaptations are often required to support complex flow features

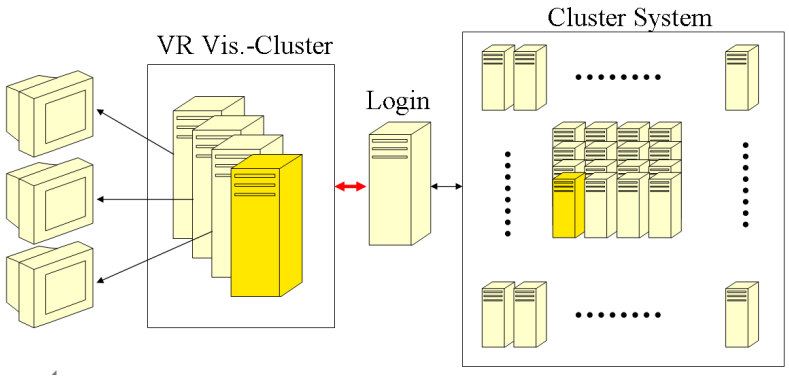


Figure 2 Usual computational system in CFD environments. Because of their scaling performance, a cluster system is often used on the computational side. The visualization is located at a distant place, here a virtual reality system with its own visualization cluster is depicted. These two systems are mostly connected through a local area network with a low bandwidth resulting in an additional bottleneck.

changing only portions of the underlying simulation grid. Transferring incrementally only the updated portions results in less data to handle in a progressive streaming. The same is true for resulting data on top of the grid that has to be updated after each simulation iteration step.

4.4 Visualization

Finally, a complete computational steering environment needs to support adequate visualization capabilities. For the analysis of unknown flow structures explorative visualization techniques have shown to support the researcher in a very effective way. The existence of real time feature extraction algorithms is required. For flow fields arising from CFD solutions stream lines are suitable to guide the researcher through the flow field and algorithms to extract them in real time are already at hand. Especially in complex flow situations stream surfaces provide better visual clues, but porting to graphics processing units to provide interactive extraction techniques has just started.

A system to determine complex features causing high CPU load should be introduced to assist the explorative visualization on-demand. But the best system architecture is unclear. Two solutions are possible. On the one hand, an additional visualization cluster can be attached to a running simulation. Data then has to be copied from the simulation to the visualization, resulting visualization features are copied to the visualization front-end. On the other hand, feature extraction can be done on the simulation nodes, this approach is called in-situ visualization. This approach promises more accurate results, since the full amount of data can be accessed. But the extraction algorithms fight for computation time with the running simulation and have to scale as well as the simulation does.

Another unsolved problem is how to provide the graphics processing units with all the data needed for the computation. Since the resulting flow fields are way to large for the GPUs memory, adapted data has to be streamed over the connecting bus system.

5 Conclusion

In this paper we had a look at the traditional work-flow for computational fluid dynamics systems. Even if the evolution of CFD system by adding capabilities of scripting and coupling with many other solvers transforms them to a more general toolbox, it is hard to use their full potential. This is caused by long waiting periods and manually iterated tasks as a result of lacking compatible steering frameworks.

Therefore we stated an idealized work-flow consisting of a one-time data load at startup that has the potential to also write the results only once. This can be achieved by tuning all parameters and guiding the simulation to an appropriate result without the bottlenecks of the traditional CFD work-flow.

To realize the idealized work-flow we determined three main research fields. First, flexible software architectures need to be developed that are capable of handling the variety of application scenarios and with interfaces flexible enough to support future applications and hardware architectures. Second, communication between subsystems will get more important in the future. With increasing supercomputer sizes their availability becomes more centralized and therefore distant analyzes and visualization becomes more and more common. Progressive streaming techniques are promising to handle bandwidth bottlenecks and provide better responsiveness. And thirdly, enhanced visualization techniques are required to support engineers with analyzes techniques suitable to their application domain. For efficient feature extraction, these algorithms need to scale as well as the hardware they are running on.

Nevertheless, although research is going on now for over forty years, we have to keep in mind that also computational fluid dynamic is far away from reaching its end. If the performance of nowadays simulation systems can be increased by a factor of at least 1000, scientists might start to think of modeling and simulating most aircraft design situations.

That is why computational steering systems still need to be flexible and application dependent in order to support efficient work-flows.

References

1 Allen Bierbaum, Christopher Just, Patrick Hartling, Kevin Meinert, Albert Baker, and Carolina Cruz-Neira. Vr juggler: A virtual platform for virtual reality application development. In *Virtual Reality Conference, IEEE*, 1:89–96, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

2 Steve Bryson. Virtual reality in scientific visualization. *Commun. ACM*, 39(5):62–71, 1996.

3 Steven P. Callahan, Louis Bavoil, Valerio Pascucci, and Claudio T. Silva. Progressive volume rendering of large unstructured grids. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1307–1314, 2006.

4 Andries van Dam, Andrew S. Forsberg, David H. Laidlaw, Joseph J. LaViola, and Rosemary M. Simpson. Immersive vr for scientific visualization: A progress report. *IEEE Comput. Graph. Appl.*, 20(6):26–52, 2000.

5 J. Davison de St. Germain, John McCorquodale, Steven G. Parker, and Christopher R. Johnson. Uintah: A massively parallel problem solving environment. *High-Performance Distributed Computing, International Symposium on*, 0:33, 2000.

6 Aurelien Esnard, Nicolas Richart, and Olivier Coulaud. A steering environment for online parallel visualization of legacy parallel simulations. In *DS-RT '06: Proceedings of the 10th IEEE international symposium on Distributed Simulation and Real-Time Applications*, pages 7–14, Washington, DC, USA, 2006. IEEE Computer Society.

7 Martin Isenburg, and Peter Lindstrom. Streaming meshes. In *Proceedings of Visualization'05*, pages 231–238, October 2005.

8 O. Kreylos, A. M. Tesdall, B. Hamann, J. K. Hunter, and K. I. Joy. Interactive visualization and steering of cfd simulations. In S. Müller and W. Stürzlinger, editors, *VISSYM '02: Proceedings of the symposium on Data Visualisation 2002*, pages 25–34, Aire-la-Ville, Switzerland, 2002. Eurographics Association.

9 Oliver Kreylos, Tony Bernardin, Magali I. Billen, Eric S. Cowgill, Ryan D. Gold, Bernd Hamann, Margarete Jadamec, Louise Kellogg, Oliver G. Staadt, and Dawn Y. Sumner. Enabling scientic worksflows in virtual reality. In *Proceedings of the ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications VRCIA 2006*, pages 155 – 162. ACM SIGGRAPH, ACM SIGGRAPH, 2006.

10 Oliver Kreylos, E. Wes Bethel, Terry J. Ligocki, and Bernd Hamann. Virtual-reality based interactive exploration of multiresolution data. *Hierarchical and Geometrical Methods in Scientific Visualization*, 2003.

11 T. Kuhlen, A. Gerndt, I. Assenmacher, B. Hentschel, M. Schirski, M. Wolter, and C. Bischof. Analysis of flow phenomena in virtual environments - benefits, challenges, and solutions. In *Proceedings of the 11th International Conference on Human Computer Interaction*, 2005.

12 Kwan-Liu Ma. In situ visualization at extreme scale: Challenges and opportunities. *IEEE Computer Graphics and Applications*, 29:14–19, 2009.

13 Ajith Mascarenhas, Martin Isenburg, Valerio Pascucci, and Jack Snoeyink. Encoding volumetric grids for streaming isosurface extraction. In *3DPVT '04: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium*, pages 665–672, Washington, DC, USA, 2004. IEEE Computer Society.

14 Tom Meyer and Al Globus. Direct manipulation of isosurfaces and cutting planes in virtual environments. Technical report, NAS Systems Division, NASA Ames Research, 1993.

15 Jurriaan D. Mulder, Jarke van Wijk, and Robert Van Liere. A survey of computational steering environments. *Future Generation Computer Systems*, 13, 1998.

16 V. Pascucci, L. Linsen, B. Hamann, and F. Gygi. Real-time monitoring of large scientific simulations. In *In ACM Symposium on Applied Computing'03, ACM*, pages 194–198. Press, 2003.

17 Valerio Pascucci and Randall J. Frank. Global static indexing for real-time exploration of very large regular grids. In *Supercomputing '01: Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)*, pages 2–2, New York, NY, USA, 2001. ACM.

18 M. Schirski, A. Gerndt, T. van Reimersdahl, T. Kuhlen, P. Adomeit, O. Lang, S. Pischinger, and C. Bischof. Vista flowlib - framework for interactive visualization and exploration of unsteady flows in virtual environments. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003*, pages 77–85, New York, NY, USA, 2003. ACM.

19 Andreas Schütte, Gunnar Einarsson, Axel Raichle, Britta Schöning, Matthias Orlt, Jens Neumann, Jürgen Arnold, Wulf Mönnich, and Thomas Forkert. Numerical simulation of maneuvering aircraft by aerodynamic, flight mechanics and structural mechanics coupling. In *AIAA-Journal of Aircraft*, 2007.

20 Dieter Schwamborn, Thomas Gerhold, and Ralf Heinrich. The dlr tau-code: Recent applications in research and industry. In *ECCOMAS CFD 2006*, The Netherlands, 2006. TU Delft.

Methods for Feature Detection in Point Clouds

Christopher Weber[1], Stefanie Hahmann[2], and Hans Hagen[1]

1 **Technische Universität Kaiserslautern, Germany**
2 **Laboratoire Jean Kuntzmann, Université de Grenoble, France**

Abstract

This paper gives an overview over several techniques for detection of features, and in particular sharp features, on point-sampled geometry. In addition, a new technique using the Gauss map is shown. Given an unstructured point cloud, this method computes a Gauss map clustering on local neighborhoods in order to discard all points that are unlikely to belong to a sharp feature. A single parameter is used in this stage to control the sensitivity of the feature detection.

1998 ACM Subject Classification Edge and feature detection

Keywords and phrases point cloud, sharp features, reconstruction, Gaussmap, clustering

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.90

1 Introduction

The use of 3D scanning technologies in art, design, manufacturing and research has become more and more common over the last years. Possible applications range from research and medical applications to prototyping and design. For example during the development of a new product, 3D scanning devices can be used to digitize and validate a manually optimized prototype, thus speeding up the development process. It is of huge importance that no details of a scanned object are lost during the scanning process. But not only the scanning process itself needs to be optimized. Also the reconstruction of the raw point set data delivered by the scanning device needs to be performed properly. During the reconstruction of the surface one does not want to loose specific features of the original object, e.g. sharp edges or corners. Sharp corners and edges are often used as design element, for example in the car manufacturing industry to visually break up huge planes and underline the dynamic and unique design of a car. In such cases it is useful to know the exact positions of sharp features in the raw point cloud data. Other examples for the reasonable use of feature detection might be quality measurement, monitoring of a manufacturing process or medical applications. In non-photorealistic rendering feature lines are used to enhance the visual perception. Additional, in the case of mesh generation, mesh simplification and segmentation knowledge about the position of features can be of great help. Depending on discipline and application, "feature" can have different meanings. In computer graphics and CAD for 3D shapes this term is usually used for *free form features* including all kinds of visually prominent characteristics of a shape, from salient edges, ridge and valley lines to sharp features as line-type or corner features.

This paper aims first to give an overview over prominent existing feature detection methods. These methods are classified into different groups, polygonal based methods in Section 2 and point based methods in Section 3. We then present in Section 4 a new method especially dedicated to the extraction of sharp features on point-sampled surfaces. In our case, a point-sampled surface is a simple unstructured point cloud, where the points belong to a 2-manifold, without any further information about (mesh) connectivity, topology,

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 90–99

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

parameterization or differential properties. Our algorithm is based on the observation that *sharp features* have the property to separate clearly two or more local surface parts with a tangent discontinuity. We thus introduce Gauss map clustering as a sharp feature detection operator. The presented method does not need any prior surface reconstruction. It can be applied directly on the point cloud and only a local neighbor graph need to be computed. The method is therefore fast and works for a wide range of sampling resolutions. It provides a set of sharp feature points as result, even in the presence of noise.

2 Polygonal Methods

There exist multiple techniques for feature extraction relying on polygonal meshes [6, 12, 9, 7, 13]. The following methods represent a group of different approaches. The list of methods is not intended to be exhaustive.

Hubeli and Gross [6] use a normal based multi-resolution framework and generate a set of edges with a normal-based classification operator. In a classification phase they assign a weight to every edge in the input mesh, proportional to the probability of belonging to a feature. The authors provide different types of operators for different mesh types like a "second order difference" operator for very coarse data sets, an "extended second order difference" operator for finer meshes, or a computational more expensive "best for polynomial" operator which performs well on noisy points sets. After this, in a detection phase they reconstruct the features from the information gained in the classification phase. According to the weights, they produce piecewise linear curves from the collections of edges that are assumed to belong to a feature. For thresholding they use a hysteresis thresholding. An edge is added as feature if it's weight is larger than an upper bound. If the weight is smaller but a neighboring edge is already selected as feature, the hysteresis thresholding provides another lower bound and accepts the edge if its weight is above this second threshold. All other edges are discarded as features. A thinning process then refines the edges to generate clear feature lines. For the thinning all patch-boundary edges are first inserted to a linked list. A first condition removes edges that are perpendicular to the mesh feature being extracted. A second condition makes sure that an edge is only removed if the patch will not become disconnected. If an edge is removed, new edges are inserted into the list and have to be analyzed since they became boundary edges. The process continues until the list is empty. They also present a multi resolution approach for their feature extraction to improve the quality. The process is not fully automatic since a user has to choose the classification operator and some parameters for the detection phase.

Hildebrand et al. [7] use anisotropic filtering on third order derivatives of the surface mesh. The derivatives are approximated by discrete differential geometric approximations. This way, the authors compute discrete extremalities, which are then smoothed and used to trace feature lines in regular triangles. Singular triangles need a special treatment. In this case the adjacent triangles are used to determine the feature line intersections with the singular triangle. After the first feature line extraction, a threshold filter is used to improve the stability of the feature extraction and to remove small ridges. The last step is an optional smoothing of the feature lines. Both methods ([6], [7]) use extrema triangles to build a set of sharp feature edges.

Watanabe and Belyaev [12] use the so called focal surfaces to detect curvature extrema on dense triangle meshes. If k_{max} and k_{min} are the largest and smallest principle curvature then the principle centers of curvature are points situated at the surface normal with a distance of $1/k_{max}$ and $1/k_{min}$ from the surface. These principle centers form the focal surface. It

consists of two sheets, one for the minimal, and one for the maximal principal curvature. Watanabe now uses the property, that the singularities of the focal surfaces, called focal ribs, correspond to lines on the original surface where the principal curvature has extrema. They present a method for the estimation of the principle curvature on a dense triangle mesh and then show how the associated focal ribs can be used to identify the features. Also here a thinning process of the first results is necessary to construct a final feature line. The resulting lines show the regions of maximal curvature. The method is not specialized for the detection of sharp features.

All mesh-based techniques use the connectivity information and normals associated with the underlying mesh. But often surface scanning devices do not deliver a mesh as raw data, but an unsorted set of point data representing the original surface. In this case, a mesh-based method has to rely on the proper reconstruction of the features during the mesh generation.

3 Point Based Methods

Very few feature detection methods are dedicated to point-sampled geometry only. The major problem of these point based methods is the lack of knowledge concerning normal and connectivity information. This makes feature detection a more challenging task than in mesh based methods.

Gumhold et al. [5] present a method that uses the Riemannian graph to construct the connectivity information for the point cloud. The Riemannian is the graph, that contains the edges to the k nearest neighbors for every data point. The algorithm first analyzes the neighborhood of each point via a principal component analysis (PCA). The eigenvalues of the correlation matrix are then used to determine a probability of a point belonging to a feature. The analysis of the ellipsoid formed by the three eigenvectors and their eigenvalues allows further conclusions about the underlying feature type. This way the algorithm can differentiate between line-type features, border and corner points. The result is a quite dense set of points covering all kinds of features independent if the feature is sharp or not. This set of points is then reduced by computing a minimal spanning tree followed by a branch cutting. This is an elegant way to obtain a sparse set of points representing the feature line.

Pauly et al. [11] extended the PCA approach with a multi scale analysis of the neighborhoods. Based on the eigenvalue analysis of the covariance matrix, they compute a value for the surface variation in the local area around a sample point. To obtain more information, they use a multi scaling approach that varies the size of these neighborhoods. That means, they apply their feature detection operator to multiple neighborhood sizes, which allows to measure the persistence of the feature. A jump in the graph of the surface variation during the multi scaling shows the existence of new surface parts. Especially in noisy datasets, the multi scaling approach enhances the result of the usual PCA analysis. Since the method analyzes up to 200 neighborhood sizes for each point in the dataset, it is computationally more expensive. To handle relative huge neighborhood sizes of over 200 neighbors, they also show a way to solve the problem of neighbors not belonging to the same connected region. To estimate when a neighborhood becomes too large, they use a heuristic that looks for strong deviations in the normal direction. The algorithm recognizes all kinds of visual eminent features, but for the identification of only the sharp features inside the dataset, this method has to be modified. One way to adapt this approach to sharp features may be achieved with an adjustment of the thresholds for the feature recognition. With well chosen lower and upper thresholds the method may be able to identify only sharp features.

Demarsin et al. [2] also searched for sharp features in point cloud data. Their goal is to

produce closed sharp feature lines. They choose a region growing method that segments the point cloud into clusters and identify the regions of sharp features. Based on the analysis of the normals of the points, they segment the point cloud in clusters with equal normal behavior. From these clusters they build up a graph that connects the neighboring clusters. The edges in this graph are then used as indication for the existence of a sharp feature in the related area. Similar to Gumhold [5] and Pauly [11], they use a graph approach and construct a minimum spanning tree of these candidates. This gives them an initial reconstruction for the feature lines. A fixed parameter for the maximum branch length is then used to cut of the short branches of the tree. In the next step, they close the feature lines. For each open endpoint in their graph they compute the n nearest neighbors among the other endpoints. The distance and the length of the paths of the neighboring endpoints is used to determine a good connection. After this they cut of the branches of the possibly remaining endpoints and smooth the graph to get their final closed feature lines.

The mentioned techniques for the detection of features in point clouds are mostly used as a preprocessing step for another processing step, e.g. a surface reconstruction with sharp features, but there exist also several reconstruction methods that preserve sharp features during a surface reconstruction of a point cloud without preprocessing. For example the methods shown by Amenta et al. [1], Guy and Medioni [4], Fleischmann et al. [3] and Öztireli et al. [10]

4 Feature extraction via Gaussian Map Clustering

This section presents our feature extraction method. It is a point-based method like the methods mentioned in Section 3. The method can be divided into three steps: In the first step, the data structure used for the analysis of the point set is built. Subsequently this data structure is used to generate local neighborhoods inside the point set. In the last step these neighborhoods are analyzed, and points belonging to sharp features are identified.
We define a *point cloud* as a simple set of 3D point coordinates $P = \{p_1, p_2, ..., p_N\}$, $p_i \in \mathbb{R}^3$ without any normal or connectivity information. The data points are unstructured, but supposed to belong to a 2-manifold surface. Let $N = |P|$ be the number of points.
The type of sharp feature we want to detect in the point cloud can vary from edges or lines between two surfaces to corners where three or more surfaces meet.

4.1 Analysis of neighborhoods

To detect sharp features in the point cloud, we have to analyze the neighborhood of every point in the dataset, similar to [11],[5], and decide if the point is a sharp feature point or not. As neighborhood, we use the *k-nearest*, i.e. the k points with the shortest distance to the sample point. The k-nearest neighbor search is a well studied problem and many algorithms exist, since it is just a variant of the nearest neighbor search problem. For performance reasons, we use a kd-tree implementation as underlying data structure. Building the kd-tree is performed in a pre-processing step for our algorithm. After the construction of the local neighborhood N_p for a sample point $p \in P$, the next step is to analyze it and decide if p belongs to a sharp feature or not. During tests a neighborhood size of 16 turned out to deliver good results. For the following analysis we apply a Gauss map clustering.

4.2 Discrete Gauss map

Let N_p be the neighborhood of p containing the k nearest neighbors and $I_p = \{1, ..., k\}$.

We now take a set T of all possible $k \cdot (k-1)$ triangles with p and two neighborhood points as vertices

$$T = \{ \quad \Delta_{ij} = \Delta(p, p_i, p_j) \quad | \quad i \neq j, \quad i, j \in I_p \}.$$

An example for three possible triangles is shown in Figure 2. The normal vector of one of these triangles Δ_{ij} is given by

$$n_{ij} = \overline{pp_i} \times \overline{pp_j}. \tag{1}$$

Note that $n_{ij} = -n_{ji}$. The discrete *Gauss map* of the neighborhood of p can now be defined as the mapping of T onto the unit sphere S^2 centered at p as follows

$$\begin{array}{ll} G_p & : T \to S^2 \\ \Delta_{ij} & \mapsto x_{ij} := p + \frac{n_{ij}}{\|n_{ij}\|}. \end{array} \tag{2}$$

Figure 1 shows the projection onto the gaussian sphere.

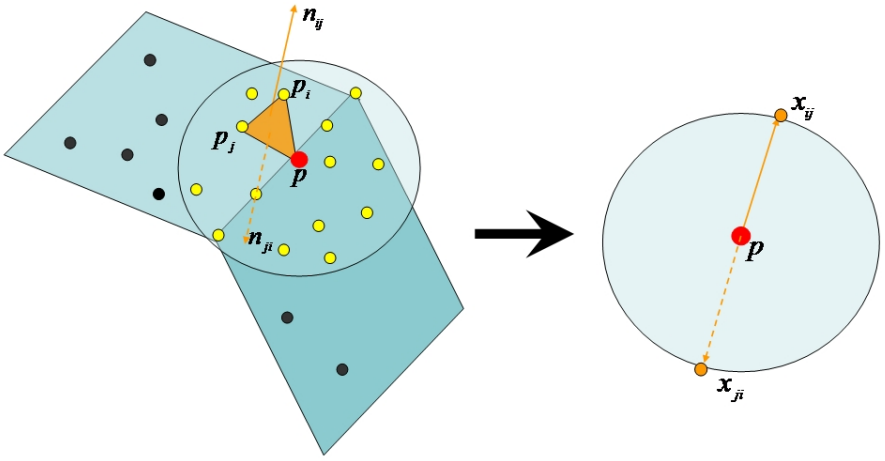


Figure 1 The projection of the normal of one of the possible triple (p, p_i, p_j) onto the gaussian sphere, with the two possible results x_{ij} and x_{ji}

4.2.1 Gauss map clustering

Feature detection is now performed by analyzing the clustering behavior of these normals on the Gauss map G_p of the set $T = \{\Delta(p, p_i, p_j)\}$.

The motivation for this idea is the fact that in the case of a smooth piecewise C^0 surface, the patterns of resulting cluster on the Gauss map is different whether the point is flat, curved (elliptic, hyperbolic or parabolic) or tangent plane discontinuous. In the nearly flat case, the Gauss map of neighbor points will present one cluster of points on the sphere, see Figure 3 (left). In the case of a curved point (parabolic, hyperbolic, or elliptic) the points will not form clusters, but spread on the sphere over a larger region, see Figure 3 (middle). And a tangent plane discontinuity will lead to a pattern, where the points of the sphere form two distinct clusters, see Figure 3 (right).

Working with a simple point cloud, the difficulty is that we have only very limited information about the underlying surface. No local triangulation or normal vectors are available. For this reason, we defined the Gauss map as projection of all triangles in T (see Sect. 4.2). As a consequence of using all triangles $\Delta(p, p_i, p_j)$, the resulting Gaussian map will contain some additional noisy points that correspond to the triangles with p and one data point from each of the planes. These noisy points are distributed over the whole gaussian sphere (see Figure 2 right), while the correct points (i.e. points belonging to triangles of p and two points of one of the planes) are positioned in clearly recognizable clusters. The dominance of the clusters makes it easy to ignore the noisy points during the computation of the clusters. Figure 2(right) illustrates these assumptions for the common case of p lying on the sharp edge of two intersecting planes, with half of the $k = 16$ neighborhood points lying on each plane. The result of the Gauss map computation shows two pairs of (opposite) clusters of $O(k^2/4)$ identical points. These pairs of clusters correspond to each of the intersecting planes. Note that the reason for the pairwise clusters is the lack of knowledge about the original surface, especially the normal directions. The points in the neighborhood are unsorted and so we can not control which triangle ($\Delta(p, p_i, p_j)$ or $\Delta(p, p_j, p_i)$) is used for the gaussian clustering. Based on this fact, both normals, n_{ij} and n_{ji} are normals of possible triangles, also see Figure 1. The result are clusters on opposing sides of the sphere, although they belong to the same plane, but where arbitrary projected into opposing directions. The red points on the right picture in Figure 2 are an example for these opposing clusters. All other points on the sphere (noisy points) correspond to situations with one point of each plane and the sample point forming a triangle. This noise is sparsely distributed over the sphere. The example in Figure 2(right) was implemented in Matlab.

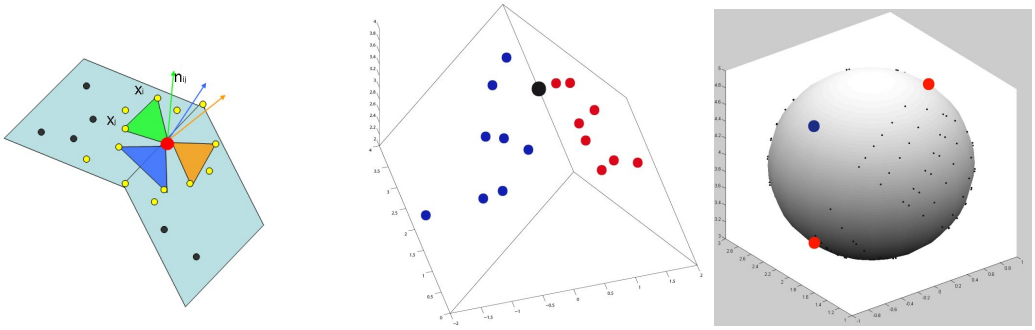


Figure 2 Left: Computation of the normal vectors used for feature identification. Middle: An example for a sharp feature situation. Right: The Gauss map G_p for the feature identification of the example in the middle, showing the clusters (big red and blue points) and the noisy points generated from wrong triangles.

Using this knowledge about the clustering behavior on the Gauss map, called Gauss map clustering, we can now determine whether the sample point p belongs to a sharp feature or not. Regarding cluster analysis, we have to consider the effect of the opposing clusters mentioned above. Since we don't have any information whether the normals points outside or inside of the surface and thus $n_{ij} = -n_{ji}$, each point x_{ij} in one of the clusters has its counterpart x_{ji} in the cluster on the other hemisphere of the Gaussian map.

For this reason and to avoid possible problems and wrong detections, we have to guarantee that opposing clusters are treated as one cluster. So instead of the original vectors, we use the lines defined by these vectors for the clustering. After clustering, the results are interpreted as follows. The case of one resulting cluster (i.e. two opposing clusters) corresponds to a flat

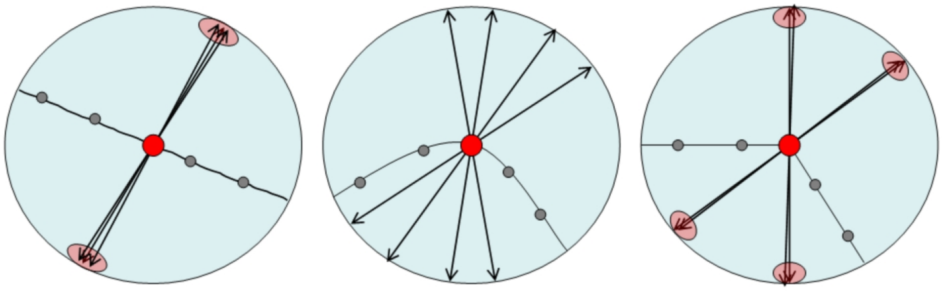


Figure 3 2D examples for clustering during feature detection

or nearly flat areas (Figure 3 left). Two or more clearly distinguishable clusters show the existence of a sharp feature nearby and the sample point of this neighborhood belongs to this feature (Figure 3 right). The case of no clustering and a sparsely distribution of points on the Gaussian map shows, that this region is curved or has at most a smooth feature, but not a sharp one (Figure 3 middle).
In many clustering algorithms one needs to specify the number of clusters to produce. However, we have to go the other way around. We want to distinguish our real clusters from sparsely distributed points in the Gauss map, and afterwards count the number of clusters. Therefore we use a hierarchical agglomerative ("bottom-up") clustering method [8]. Starting with one separate cluster for each point, we merge them step by step into larger clusters. Here the distance, which is used as the criterion for the merging process, and its definition is of importance for the success of the clustering.
We use the mean distance D_c between the elements as criterion, and define the distance as the angle between the lines which are defined by the opposing clusters as follows:

$$D_c(S_1, S_2) = \frac{1}{|S_1| \cdot |S_2|} \sum_{x \in S_1} \sum_{y \in S_2} d(x, y), \tag{3}$$

where S_1, S_2 are two clusters to be compared, $|S|$ is the number of elements in a cluster and d is the distance measure on our Gauss map. Each agglomeration increases the distance between the clusters. As a consequence, we can stop the clustering algorithm when the distance between the existing clusters exceeds a certain threshold $\sigma \in [0, \frac{\pi}{2}]$.
After the clustering we have to discard all clusters with only a few points. We do so to eliminate the effects of the noisy points mentioned earlier. If only one cluster remains, we know that the sample point does not belong to a sharp feature since the underlying area is a flat plane. If two to four clusters remain, we say that the point belongs to a feature. If the result consists of more than four clusters or no cluster at all, we decide that the point is not a feature since the result of many or no clusters is a strong signal for a curvy area without a sharp feature. For datasets with sharp features where more than four edges join, the parameter can be adjusted to match the occurring situation.

4.3 Sensitivity to parameter choice

The threshold σ is the parameter we can use for the sensitivity of the feature detection. It corresponds to the angle of the sharp feature we want to detect. The choice of the value of σ depends on the dataset. A dataset with obtuse angles will need a lower value for σ than a

dataset with acute angles. The effect of σ is shown in Figure 4. On the left side a too small σ was chosen, the features were not detected properly. In the middle σ was well chosen, the features are well detected. On the right side σ was chosen too big, resulting in the detection of too many feature candidates. The user adapts the σ value with a trial and error method.

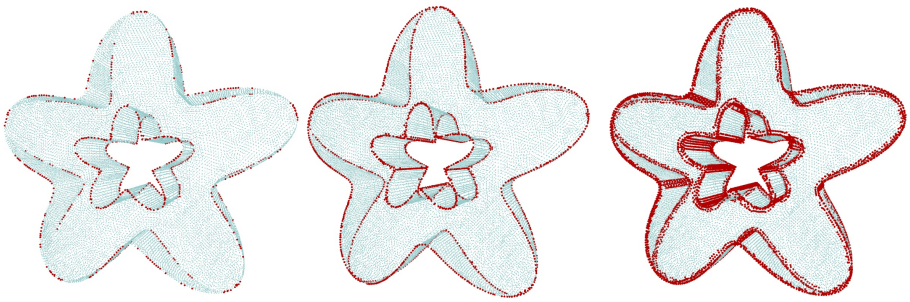


Figure 4 Different values of σ tested with the trim-star model, σ growing from left to right.

4.4 Sensitivity to noise

We also tested the methods behavior regarding noise. To do so, we generated noise by moving the points of the original dataset in a random direction with a maximum length of 5%, 10% and 15% of the dataset's bounding box. The results are shown in Figure 5. One can see that the method is relatively stable concerning noise and only strong noise results in the presence of outliers and false positive features (i.e. points which are detected as features but which are not features). To improve the result it would also be possible to use some smoothing techniques to reduce the outliers in the presence of strong noise. Indeed, just removing isolated candidates which are not surrounded by other features would reduce the resulting number of outliers significantly.

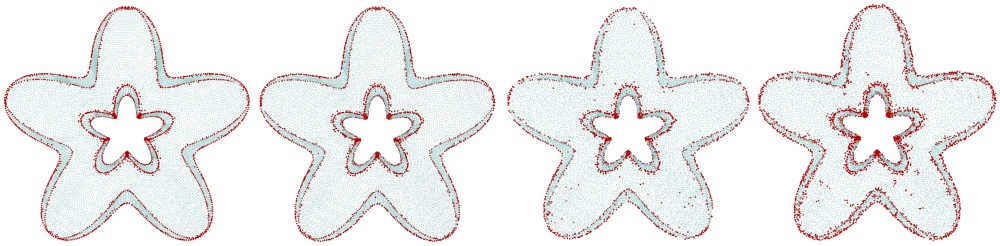


Figure 5 Noise test on the trim-star model. From left to right: no noise, 5%, 10%, 15% noise

5 Results and conclusion

The proposed method for sharp feature detection was tested on various constructed and real world examples and delivered good results, see Figure 6. It does not rely on local surface reconstructions, and no normal information is required. The resulting point cloud with the marked sharp features can be used for several applications like surface reconstruction, non-photorealistic rendering, mesh generation or surface modeling. In [5] a whole section is devoted to possible applications. Currently, the threshold for the sensitivity is user controlled and specific for a dataset. For most datasets, an automatic computation of the parameter

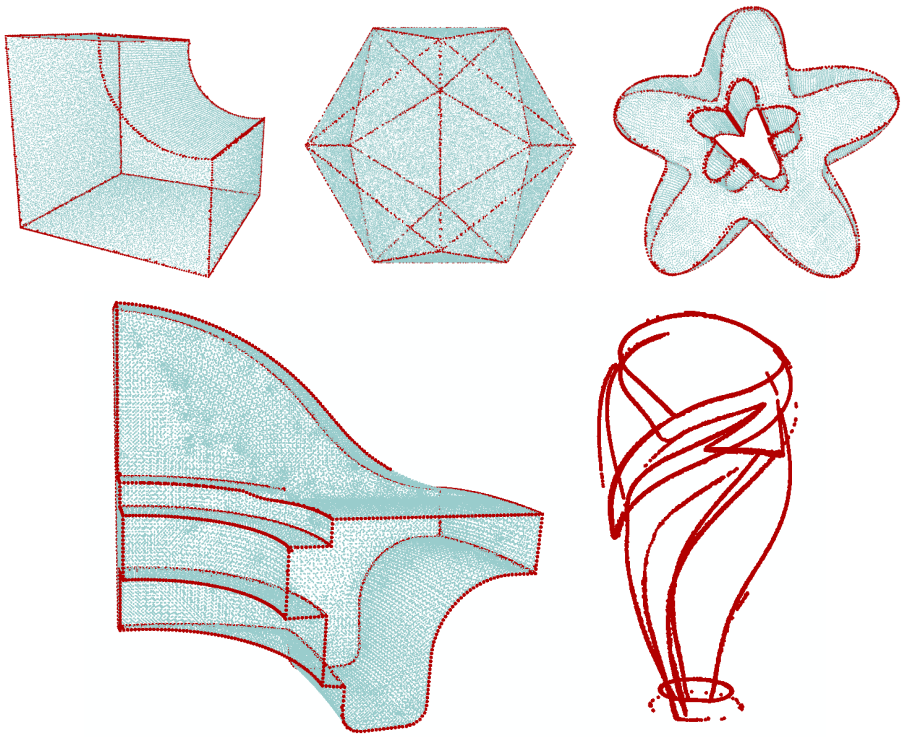


Figure 6 Sharp feature detection on a cube, an icosahedron, the trim-star model model, the fandisk and a vase model.

can reduce user dependency. The method detects line-type and corner-typed sharp features. Cone peak features are currently not recognized by it, but it should be possible to extract a particular clustering behavior for these cases and adapt the method to recognize these.

Acknowledgements

The vase model is provided courtesy of INRIA, the fandisk and trim-star are provided courtesy of MPII, all by the AIM@SHAPE Shape Repository. This work was supported by the IRTG 1131 of the DFG (German Research Foundation). It has been partially supported by the Deutsche Forschungsgemeinschaft INST 248/72-1.

References

1 Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *SMA '01: Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266, New York, NY, USA, 2001. ACM.

2 Kris Demarsin, Denis Vanderstraeten, Tim Volodine, and Dirk Roose. Detection of closed sharp edges in point clouds using normal estimation and graph theory. *Comput. Aided Des.*, 39(4):276–283, 2007.

3 S. Fleischmann, Daniel Cohen-Or, and C.T. Silva. Robust Moving Least-squares fitting with sharp features. *ACM Trans. Graph.*, pages 37–49, 2005.

4 G. Guy and G. Medioni. Inference of surfaces, 3D curves, and junctions from sparse, noisy, 3D data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1265–1277, 1997.

5 Stefan Gumhold, Xinlong Wang, and Rob McLeod. Feature extraction from point clouds. *Proceedings of 10th International Meshing Roundtable*, 2001.

6 Andreas Hubeli and Markus Gross. Multiresolution feature extraction for unstructured meshes. *Proccedings of IEEE Visualization*, pages 287–294, 2001.

7 K. Hildebrand, K. Polthier, and M. Wardetzky. Smooth feature lines on surface meshes. *Proceedings of Symposium on Geometric Processing*, 2005.

8 Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, Boston, MA, USA, 2th edition, 2009.

9 Leif Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 57–66, New York, NY, USA, 2001. ACM.

10 Cengiz Öztireli, Gael Guennebaud, and Markus Gross. Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression. *Computer Graphics Forum*, 28(2), 2009.

11 Marc Pauly, Richard Keiser, and Markus Gross. Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 2003.

12 Kouki Watanabe and Alexander G. Belyaev. Detection of salient curvature features on polygonal surfaces. *Computer Graphics Forum*, pages 385–392, 2001.

13 Tino Weinkauf and D. Günther. Separatrix Persistence: Extraction of Salient Edges on Surfaces Using Topological Methods. *Computer Graphics Forum (Proc. SGP '09)*, 28(5):1519–1528, July 2009.

Cartography of Mars in a Virtual Reality Environment

Rolf Westerteiger[1,2]

1 **DLR Deutsches Zentrum für Luft- und Raumfahrt (German Aerospace)**
Lilienthalplatz 7, 38108 Braunschweig, Germany
rolf.westerteiger@dlr.de
2 **Computer Graphics and HCI Group, University of Kaiserslautern**
67653 Kaiserslautern, Germany

Abstract

This work aims to investigate the suitability of applying Virtual Reality techniques in the exploration of Mars terrain features in order to support the creation of topographic maps of the planet. Traditionally, these tasks are performed using Geo-Information-Systems (GIS) on desktop workstations, with a two-dimensional projection of the collected map data as the basis on which an operator performs manual feature extraction. After identifying features and characterizing them quantitatively using measurement operators, they are ultimately represented visually by so called geo-objects which are then entered into a GIS database. Within the scope of this project, a system will be developed which enables this workflow to occur entirely within a VR-environment, using appropriate navigation and interaction metaphors. The main goal of the project is to examine whether the more natural immersion in the VR-environment can help to improve the identification and spatial analysis of surface features.

Keywords and phrases Virtual Reality Environment, interaction metaphors, spatial analysis

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.100

1 Introduction

Nowadays the creation of topographic maps is supported by 2D Geographic Information Systems (GIS) running on desktop computers. For the experts in the field, working within a 2D environment is a natural choice because, depending on the chosen projection type, certain geometric properties (such as angles) can be preserved. Historically, this has been critical for performing spatial analysis on drawn or printed map material. However, using software systems the visual representation of the data can be decoupled from the measurement process. This gives rise to the question whether a virtual reality system, consisting of an immersive 3D display along with intuitive input and navigation methods, can support the user in identifying surface features by immersing him within a natural representation of the dataset and therefore closing the cognitive gap between the user and his data. Furthermore, since a perspective projection naturally implements a focus and context scheme, it has the potential to enable more data to be displayed at the same time while keeping visual clutter at an acceptable level. In order to test these hypotheses, a VR-GIS system is being developed in coorporation with cartography experts at the German Aerospace (DLR) whose ultimate goal is the creation of a topographic map of Mars.

2 Related work

The following section gives an overview of the state of the art in 3D-GIS systems, focusing on aspects of navigation, spatial analysis and manipulation of geo-objects which can be

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 100–110

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

considered the core set of functions a GIS-system needs to implement. Furthermore, the terrain rendering algorithms and data structures being employed by these systems will be mentioned.

2.1 LandExplorer

LandExplorer[5] is a desktop 3D-GIS system, which allows the users to create 3D-maps by drawing both simple geometric primitives like landmarks or polygons as well as instantiate more complex pre-fabbed geo-objects such as buildings or bridges on top of a given DTM. In order to keep rendering work manageable, the approximation-tree LoD-Scheme[2] is employed, which uses a hybrid representation mixing triangulated irregular networks (TINs) and regular grids. Geo-referenced 2D-vector overlays are rendered on-the-fly to an off-screen texture using techniques described in [9] before being mapped onto the terrain geometry. This strategy avoids some of the problems such as clipping artifacts associated with the more traditional approach of displaying 2D-geometry as offset or extruded lines on top of the terrain. The LoD-adaptivity of this approach, however, only considers viewer distance and cannot take into account the slope of the terrain, which can warp the projected area of individual texels significantly. A more elaborate approach such as vector texture maps [13] could solve this problem and provide pixel-exact rasterization of projected 2D-geometry. To facilitate exploration of the dataset at hand, information visualization metaphors such as filter lenses are being provided in order to superimpose additional data within a user-defined region, while keeping the neighbouring region uncluttered to provide orientation. One drawback of this system, however, is that it does not support out-of-core rendering of large datasets (streaming) and furthermore requires long preprocessing times. Therefore, it is better suited for localized tasks like urban planning rather than explorative planetary research.

2.2 Google Earth

Google Earth is a popular desktop application which enables the user to browse a high-resolution digital terrain model (DTM) of Earth (and nowadays, Mars) at interactive rates by using Level-of-Detail (LoD) techniques and on-demand streaming of DTM data. In spite of a well-performing implementation, LoD-transitions are readily apparent, owing to the bandwidth and latency limitations of the internet. By eliminating some degrees of freedom in camera movement, the software successfully unifies the virtual trackball metaphor with traditional pan & zoom interaction, depending on whether the view axis is tangential or perpendicular to the planet's surface, enabling a natural transition between the two modes. Even though Google Earth is not geared towards the expert user, it nevertheless provides some tools resembling simplified versions of tools provided by GIS systems, like a simple ruler tool to measure distances. Furthermore, 3D objects (geo-objects) can be constructed and geo-referenced within the system, mainly to provide the user with the means to augment Google's database by adding a visual representation of buildings in urban areas, which are not representable within the DTM model due to undersampling.

2.3 RIMS

The real-time interactive mapping system[3] is a 3D-GIS research prototype which uses a quadtree data-structure to support a level-of-detail rendering scheme. While screen-space projected size of tiles is being considered in rendering, the primary criterion for LoD selection is a user-defined region of interest (ROI). Geometric detail is highest at the center of the ROI

while gradually decreasing with distance. The motivation for this is that spatial analysis tasks are commonly performed locally at a pre-selected site.

To support this spatial analysis, the sytem provides a so-called virtual geological compass to measure the orientation of surface features, which is visualized in the form of a plane which is tilted and translated manually in order to fit to surface features. The orientation of the plane is then displayed on-screen and can be used to annotate features quantitatively, which can be represented by polylines drawn directly on top of the DTM.

Furthermore, curviplanar surfaces can be created based on the polyline control points in order to visualize hypotheses about the original configuration of geological features before erosion. User studies were performed, showing that the 3D interaction with the data indeed can improve the identification phase of feature extraction.

3 Research goals

3.1 Cartography workflow

3.1.1 Identification and classification of features

The primary motivation for moving the cartography workflow into a VR environment is to investigate whether the immersive effect of VR in combination with adequate navigation interfaces can improve the perception of surface features that are of interest to the expert users. As opposed to the top-down projection used in traditional 2D-GIS, where the height attribute can only be incorporated indirectly, a perspective projection with stereoscopic rendering has the potential to close the gap between the user and the data by allowing him or her to perceive the actual topography of the terrain in a natural way, leveraging the potential of the human brain to recognize structures.

3.1.2 Instantiation of geo-objects

The process of creating a topographic map involves the creation of so-called geo-objects, which are symbolic representations of the surface features that have previously been identified. The most trivial geo-object considered here is a point object, marking a location on the planet's surface. These can be considered waypoints designating areas of interest that the user might wish to return to later for presentations or further analysis. Extended surface features are traditionally labelled using polygon primitives tracing the outline (footprint) of the feature.

Accurate placement of these geometric primtives on a workstation can be considered a simple task when performed using a mouse or a similar pointing device. However, input devices within a VR environment have different characteristics. The natural equivalent of a mouse cursor in VR is a picking ray, which is controlled by the user using a so-called wand, a input device resembling a pointer. The selected point on the surface is given by the intersection of the picking ray with the terrain geometry. Whether this method of input can improve upon traditional techniques will be investigated within the scope of this work.

Even though the final goal is implementation of the full GIS workflow within the proposed system, it is still necessary to be able to interface with common GIS databases. This will eventually enable a user to quickly switch from his workstation to a VR environment to perform a detailed analysis of a questionable terrain configuration, for example. On the other hand, geo-objects created during a VR-session could be taken to a desktop GIS in order to further characterize and organize them.

3.1.3 Spatial analysis

To help fully understand surface processes, virtual measurement tools will be provided to quantify geometric properties of features previously identified. For example, given a crater, a central quantity of interest is the volume of dislocated material. This can be computed by using a surface integral over the height field data, constrained to the inside of a user-specified polygon which represents the crater boundary. This capability can be extended to include the computation of statistic properties of other data channels, such as hydration, within a polygonal footprint.

3.2 Use as collaboration platform

Since most VR setups are multi-user capable by design, the question arises how this can be leveraged for a VR cartography system. A realistic use-case would be a geologist wanting to present a hypothesis to his or her peers. After visualizing it using geo-objects and other glyphs, a camera trajectory could be recorded which clearly documents the area in question and can serve as an anchor for an (interactive) presentation.

3.3 Public relations

In addition to its purpose in supporting geoscientific research, a secondary role of the proposed system will be to present the data and results in a way that is both accessible and appealing to a general audience. To support this usage scenario, additional rendering effects such as atmosphere, shadowing and advanced shading will be provided. Synthetic ground detail will be overlaid on top of interpolated source data in order to hide low sample-density in views close to the surface (see 9.1).

4 Reference dataset

The reference dataset which will be used in the development and initial evaluation of the system is a data product obtained by the Mars Express (MEX) mission, the first european mission to Mars. The probe, which was launcehd in June 2003, consists of an orbiter for remote sensing as well as a lander (Beagle 2) for geological surveys on-site. One of the instruments aboard the orbiter is the High Resolution Stereo Camera (HRSC), which was developed by the Institute for Planetary Research (IFP) of the German Aerospace (DLR) in Berlin. It consists of an array of line-sensors, capturing different wavelengths simultaneously while being swept across the surface (known as pushbroom-configuration). Due to this mode of operation, the resulting dataset fragments each represent long strips of the planet's surface. The sensor array consists of 2 spectrometers, 4 color sensors (red, green, blue, infrared), as well as 5 panchromatic (sensitive within the whole visible spectrum) black-and-white sensors. The latter are primarily used to acquire image data under different viewing angles in order to reconstruct height information by correlation of the 5 channels. This process is known as photogrammetry and results in a digital terrain model (DTM). The multi-spectral channels will not be available directly at first. Instead, derivative products will be used for visualization, which, for example, includes a scalar field encoding surface water density.

Eventually, the DTM as well as the individual color channels are de-warped, inverting the perspective projection. This process yields so called ortho-images, which are characterised by a constant ground sample distance (GSD), which is the distance between adjacent pixels of the image (10 to 40 meters in this case). For details about the camera operation and data processing pipeline, see [7] and [14].

Since the MEX mission is still ongoing, as of this writing only two thirds of the planet's surface have been captured. Therefore, the NASA MOLA (Mars Orbiter Laser Altimeter) dataset, which provides much less resolution but full coverage of the surface, will be used to fill in the gaps in order to provide a complete view of the planet.

5 System requirements

5.1 Data streaming

One of the requirements for the system is that the whole dataset has to be explorable as a continuum, without noticeable loading times. Considering the large size of the dataset (around 500 GiB), this calls for a streaming approach, where required data are requested from the storage backend as required. Because of the inevitable latencies involved in this data access, it is ncessary to predict user movements in order to anticipate which data tiles will most probably be required for rendering future views and issues requests to load that data in advance (prefetching).

5.2 Interactivity

In order to achieve a full immersion of the user in the virtual environment and to prevent so-called simulator sickness it is imperative to guarantee a certain bound on framerate and latency (delay between user input and effect of that input). Since the rendering of the terrain data can reasonably be expected to be the most time-consuming operation in updating the display, this translates into certain requirements on the chosen algorithm. Most importantly, given a target framerate, the algorithm has to restrict the choice of LoD-levels as to not violate this bound. For the same reason it has to be possible to suspend inter-frame coarsening / refinement operations, which can become problematic if the user moves across the surface at low altitude and high velocity, which calls for frequent reloading of data from the backend. Until requested data is made available to the frontend, the next lower available LoD-level for each region will be displayed instead.

5.3 Accuracy of rendering

In order to faithfully retain the content of the source data, it is necessary to visualize it with high accuracy. This demand is in conflict with the requirement of maintaining interactive framerates, however, since LoD rendering techniques decrease the resolution of terrain geometry with increasing distance to the viewer. This is acceptable as long as this decrease in resolution stays below the visual threshold. This can be guaranteed by algorithms which employ a screen space error threshold as a metric for controlling the LoD selection. If this threshold is set to a value smaller than a single pixel, the output of the algorithm will not be distinguishable from a full-resolution rendering. However, if the required data is not available at sufficient detail to satisfy these demands because of disk- or network-related delays, the LoD criterion might be violated, resulting in visible artifacts. Since these situations cannot, in general, be avoided, a compromise solution is to indicate within the environment whether the view represents the converged result.

An additonal challenge in faithfully representing the data lies in the fact that remote sensing data frequently contains redundancy due to overlap between adjacent datasets. Since, in the general case, the structured grids of the individual datasets are not aligned, handling these situations presents a challenge. A straightforward approach would be to resample the data on a common grid spanning the whole surface, however this would imply an undesirable

low-pass filtering of the data. A possible solution to be explored is to merge the two grids involved, giving a new grid which is not regular anymore but still has a simple topology. However, the resulting data cannot be directly rendered by the LoD terrain rendering algorithms considered in this scope. On the other hand, as long as the overlap regions are not too large, a brute-force approach can be applied by rasterizing the undecimated mesh directly.

6 System architecture

The initial development and evaluation platform for the software will be a Powerwall at the DLR (German Aerospace) in Braunschweig. This is an active stereoscopic display (using shutter glasses) equipped with infrared cameras for tracking the head or input devices. The rendering frontend consists of three nodes driving one projector each as well as a master node, using an InfiniBand interconnect. The storage backend will be realized by a server offering the data to the frontend by means of a network filesystem.

6.1 Distributed execution

The system will be a distributed application executing on the frontend nodes. A SPMD (single-program, multiple-data) programming model will be employed, which is facilitated by the usage of MPI as a network stack. In order to distribute the rendering and handle frame-synchronization, OpenGL Equalizer[6] will be used, which is a lightweight minimally invasive library supporting all common VR display configurations as well as providing built-in drivers for head-tracking devices. Any input devices will be connected to the master node, which collects input events and broadcasts them to the rest of the nodes. Since the program state at every node is only a function of the processed events, this strategy, in combination with synchronization barriers at each frame swap, will ensure that all the nodes share a common, global state at any time.

6.2 Scripting environment

In order to make the system easy to extend, non-performance critical modules of the system are implemented in the Python language while rendering and other costly operations are performed within a library written in C++. Since Python is easily learned and already widely popular, this design choice will allow even non-professional programmers to easily adapt the system to different usage scenarios without sacrifying much performance.

7 Navigation

To enable the user to move within the virtual world, intuitive navigation metaphors have to be provided. As proposed in [11], independent of the navigation mode, a 2D overview map showing the current location and view direction will always be available for reference. The user will be able to jump to any location on the map by using a pointer device or by selecting a landmark from the GIS database. In order to prevent disorientation, the transition from the previous location to the target location must not be abrupt but should follow a smooth interpolating trajectory, as demonstrated by Google Earth. To improve immersion, the system will support head-tracking, adapting the view frustum to the user's position relative to the screen. Optionally, the camera trajectory will be recorded during

navigation for later playback or export to a third party system for offline rendering of high quality movie sequences.

7.1 Virtual walk

Keeping the camera height fixed at eye-leve above the planet's surface by sampling the DTM below the viewpoint position, the hypothetical scenario of an astronaut walking the surface of Mars will be simulated. To prevent the user from being distracted by having to learn an unknown input system, a gamepad will be used as the primary input device in this mode. This mode of input is accessible to a wide audience while at the same time providing expert users with a large set of controls that can be mapped to functions within the VR environment. Keeping the up-vector of the camera perpendicular to the surface, the two remaining degrees of freedom defining the viewing direction can be directly mapped to an analog stick, while the two additional degrees of translatory freedom will be controlled by the direction pad.

7.2 Flyover

To get an initial impression of the topography of a larger area, the viewer can fly through the atmosphere at medium altitudes, looking down onto the terrain. This will be especially useful for surveying a larger area before beginning detailed analysis or for exploring elongated features like ridges or canyons. The parallax effect experienced due to the movement across the terrain can be expected to improve the perception of depth even further in addition to the stereoscopic rendering. By relaxing the requirement that the up-vector be perpendicular to the planet's surface, even tilting the camera would be possible, however the necessity to handle three degrees of freedom simultaneously would surely overburden most casual users, unless it was coupled to a flight dynamics simulation.

7.3 Top-down view

A top-down view of the planet, resembling the view from a satellite's position, will be provided for navigation at large scales and public presentations (flights towards the planet). A virtual trackball metaphor enables the user to rotate the planet by picking a surface point using a wand and dragging it to a new location. Alternatively, the analog stick of the gamepad can be used to control rotation along the two axes perpendicular to the view direction. Upon approaching the planet, a seamless transition to the flyover mode will occur (as in Google Earth). This means that the view direction, which initially is perpendicular to the surface, will gradually be tilted to lie in a tangential plane.

8 Terrain rendering

In the following some terrain rendering algorithms will be presented which are considered suitable for this project, based on the criteria previously given in section 5. The algorithms presented here are optimized to run efficiently on modern hardware, which due to the relative increase of the performance of GPUs as opposed to CPUs favors simple data-structures which can be represented as textures and do not perform fine-grained LoD-adaption as in previous approaches. Storage as textures implies that the input data have to be sampled on a regular grid, which is given for both the MEX and MOLA datasets considered in the reference application. Regular grids also facilitate the implementation of spatial analysis functionality, since most queries on the dataset can be trivially implemented, given that the

finest LoD-level is available in memory. Applying these rendering algorithms to unstructured (e.g. LIDAR) data directly is not possible and necessitates a resampling onto a regular grid, which for irregular data implies a tradeoff between undersampling, which removes data content by low-pass filtering, or oversampling which incurs a large memory overhead. For a more general survey of terrain rendering algorithms, see [12].

Most publications concerning terrain rendering due not address the use-case of a high-altitude viewpoint, which reveals the curvature of the planet, with the extreme case requiring a spherical rendering of the data. It can be expected that this issue can be solved in a straightforward way by performing projection to a spherical surface within a vertex shader. However, care must be taken to make sure that the LoD-metrics used by the chosen algorithm are adapted to be aware of this projection.

8.1 Geometry Clipmaps

The Geometry Clipmaps[10] scheme by Losasso and Hoppe renders concentric, rectangular rings of geometry centered at the viewer's position, with a factor two decrease in linear resolution between adjacent rings as distance increases. The on-disk representation of the dataset consists of a simple mipmap pyramid which is constructed from an original full-resolution image by successive filtering and downsampling. The simplicity of this data-structure allows the application of image compression algorithms and delta-encoding of finer LoD-levels against coarser ones, since the latter is required to be already in memory when the finer level is being loaded. A compression factor of 100 for the DTM of the United States being used as reference in the publication is reported.

At runtime, the algorithm maintains a LoD-pyramid in RAM, with each level storing a square subregion (window) of the corresponding level on disk. When the viewpoint moves, these windows are shifted and the data are updated correspondingly. By making use of toroidal memory addressing, copying operations are eliminated and only the L-shaped region of newly acquired data needs to be written to RAM. If the required data cannot be acquired in time for the next frame, during rendering the next lower LoD-level is used in its place. Therefore, data structure updates can be budgeted in order to guarantee a given minimal framerate.

8.1.1 GPU implementation

The original Geometry Clipmaps algorithm as described above uses vertex buffers to present heightfield data. These are updated each time the viewer positions shifts and have to be re-transferred to the GPU in full. Soon after the original publication, a newer generation of hardware enabled vertex shaders to access texture memory directly. This has been exploited by Asirvatham and Hoppe in their follow-up work, which unifies the handling of DTM and color data by storing both as textures. For rasterization, a proxy geometry is used which does not contain z-values but only texture coordinates. Thus, for each level of resolution only one of these proxy geometries is needed which is scaled and translated as required. For each vertex (x, y) within the geometry a vertex shader procedure then reads the corresponding z-value (height) from the currently bound texture. In order to update the L-shaped region within a texture, this revised algorithm employs render-to-texture functionality, reducing bus-traffic even further.

8.2 RASTeR

This algorithm[4] makes use of essentially the same disk and memory structure as the previous algorithm, however it uses a more complex proxy-geometry constructed using a subdivison scheme based on triangular patches. In order to control the level of subdivision, a screen-space error threshold is guaranteed by evaluating the size of bounding volumes of subtrees when projected to the screen. The detail levels of the texture pyramid correspond with the level of geometry subdivision and are selected accordingly, with the triangle vertices being centered on the texels, avoiding low-pass filtering due to bilinear interpolation. The advantage to the Clipmaps approach is that a more fine-grained LoD adaptation occurs.

9 Future work

9.1 Rendering

Once basic rendering of the geometry including texture-mapping and normals for lighting works and has been optimized to fulfill interactivity and accuracy demands, further work will including improving the perceived visual quality in order to enhance immersion.

In order to make the navigation metaphor of a virtual walk visually pleasing, once must consider that the ground resolution (ground sample distance) of the available dataset is 10m at best. Because the camera is very close to the ground, this would result in large areas of flat shading. As proposed in [10], additional geometric detail (that is not present in the original data) could be synthesized by generating detail maps to modulate the normal vector.

Further ideas include rendering the sun (possibly using HDR rendering to floating point buffers followed by tonemapping) and shadow mapping. Going even further, a global illumination solution could be precomputed for a given, fixed sun position and rendered by modulating the terrain brightness using lightmaps. The motivation for this is that global illumination has been shown to improve perception of geometric configurations and might therefore simplify the identification of terrain features.

Another idea to improve depth perception in the distance, where stereoscopic separation becomes close to imperceptible, is to visualize the planet's atmosphere. Enabling OpenGL fog (fading towards a constant color based on z-distance) is a straightforward solution that has the added benefit of hiding the geometry cutoff at the z-far clipping plane, which can be used to reduce the rendering workload by culling the distant parts of terrain. On the opposite end of complexity, a weather model could be employed, which gives density distributions of gases and particles within the atmosphere. Their effect on the light arriving at the viewer could then be simulated by tracing a ray from the eye-point to the geometry for each pixel, integrating atmospheric effects such as attenuation and scattering along the path.

9.2 Extension to Earth

After evaluation of the proposed system for topographic mapping of the Mars surface using the reference dataset, it is planned to extend the functionality to different usage scenarios on earth. Considering that earth harbours a more differentiated set of surface features than Mars as well as vegetation and man-made artifacts, the corresponding GIS representations can be expected to be much more complex, which will certainly require extension of the database interfaces. Also the available remote sensing data of Earth covers a wider range of the EM spectrum, including multi-spectral images, which will at the very least complicate the user interface for selecting the mapping of data to its visual representation.

10 Conclusion

A visualization framework was presented which enables geological studies and mapping of the Mars surface within a VR environment. Interaction methods were defined on the basis of a simplified cartography workflow, with a focus on different navigation modes and corresponding input devices. From a description of two different usage scenarios, requirements for the terrain visualization component were derived which, together with assumptions about the nature of the source data, motivated the selection of a LoD terrain rendering algorithm. Furthermore, a hardware architecture was presented, consisting of a PowerWall driven by a PC cluster as a rendering frontend as well as a backend server providing data access.

References

1 Ingo Assenmacher. *Low latency technology for interactive virtual environments*. PhD thesis, RWTH Aachen, 2009.

2 Konstantin Baumann, Juergen Doellner, Klaus Hinrichs, and Oliver Kersting. A hybrid, hierarchical data structure for real-time terrain visualization. *Computer Graphics International Conference*, 0:85, 1999.

3 T. Bernardin, E. Cowgill, R. Gold, B. Hamann, O. Kreylos, and A. Schmitt. Interactive mapping on 3-d terrain models. *Geochemistry, Geophysics, Geosystems (G-cubed)*, 7, 2006.

4 Jonas Boesch, Prashant Goswami, and Renato Pajarola. Raster : Simple and efficient terrain rendering on the gpu. In *Proceedings EUROGRAPHICS Areas Papers*, pages 35–42, 2009.

5 Jürgen Döllner, Konstantin Baumann, and Oliver Kersting. Landexplorer - ein system für interaktive 3d-karten. *Kartographische Schriften*, 7:67–76, 2003.

6 Stefan Eilemann, Maxim Makhinya, and Renato Pajarola. Equalizer: A scalable parallel rendering framework. *IEEE Transactions on Visualization and Computer Graphics*, 15:436–452, 2009.

7 K. Gwinner, F. Scholten, M. Spiegel, R. Schmidt, B. Giese, J. Oberst, C. Heipke, R. Jaumann, and G. Neukum. Derivation and validation of high-resolution digital terrain models from mars express HRSC data. *Photogrammetric Engineering & Remote Sensing*, 75(9), 2009.

8 Lok M. Hwa, Mark A. Duchaineau, and Kenneth I. Joy. Adaptive 4-8 texture hierarchies. In *VIS '04: Proceedings of the conference on Visualization '04*, pages 219–226, Washington, DC, USA, 2004. IEEE Computer Society.

9 Oliver Kersting and Jürgen Döllner. Interactive 3d visualization of vector data in gis. In *GIS '02: Proceedings of the 10th ACM international symposium on Advances in geographic information systems*, pages 107–112, New York, NY, USA, 2002. ACM.

10 Frank Losasso and Hugues Hoppe. Geometry clipmaps: terrain rendering using nested regular grids. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 769–776, New York, NY, USA, 2004. ACM.

11 Verbree E.and Maren G.V.and Germs R.and Jansen F.and Kraak M-J. Interaction in virtual world views-linking 3D GIS with VR. *International Journal of Geographical Information Science*, 13(4), 1999.

12 Renato Pajarola and Enrico Gobbetti. Survey on semi-regular multiresolution models for interactive terrain rendering.

13 Nicolas Ray, Thibaut Neiger, Xavier Cavin, and Bruno Lévy. Vector texture maps. Technical report, INRIA - ALICE, 2005.

14 F. Scholten, K. Gwinner, T. Roatsch, K.-D. Matz, M. Wählisch, B. Giese, J. Oberst, R. Jaumann, and G. Neukum. Mars express HRSC data processing - methods and operational aspects. *Photogrammetric Engineering & Remote Sensing*, 71(10), 2005.

Visualization in Human-Centered Virtual Factories

Xiang Yang[1], Eduard Deines[2], and Jan C. Aurich[3]

1 **Institute for Manufacturing Technology and Production Systems (FBK), University of Kaiserslautern**
P.O. Box 3049, Kaiserslautern, 67657, Germany
yang@cpk.uni-kl.de

2 **Institute for Data Analysis and Visualization, University of California, Davis**
One Shields Ave, Davis, CA 95616, USA
edeines@ucdavis.edu

3 **Institute for Manufacturing Technology and Production Systems (FBK), University of Kaiserslautern**
P.O. Box 3049, Kaiserslautern, 67657, Germany
aurich@cpk.uni-kl.de

Abstract

In a manufacturing system (MS), a wide range of human activities are applied in production processes. The human factor plays a core role and should be incorporated into the design, planning and decision making processes. In this work we describe different definitions, developments and existing concepts of a Virtual Factory (VF) and discuss VFs from the human oriented point of view. Furthermore, we analyze the potential need and use of visualization methods in VF study and propose a human-centered VF concept. Following this concept we introduce an example implementation and describe how our model facilitates the decision making and design process in MS. In addition, we show an example of a noise analysis of working environment, which is based on our virtual factory model.

Keywords and phrases Visualization, Virtual Reality, Virtual Factory, Sound Simulation

Digital Object Identifier 10.4230/OASIcs.VLUDS.2010.111

1 Introduction

Rapid development of theories and concepts of the Virtual Factory (VF) in the last two decades significantly improved the design and analysis of the manufacturing system (MS) in industry. Human factors play a key role in MSs in particular when facing challenges such as faster product changes and shorter planning time. However, in existing VF designs not enough attention is paid to human factors. People have to perform increasingly complex operation processes to ensure the functioning of the MS. Even if modern machining centers together with other support systems are capable of performing, for example, very precise and fast automatic cutting processes, failures are often caused by incorrect configuration and/or operation by the workers. Therefore, new methods for training employees need to be developed to enable machine operators to fulfill these tasks.

In manufacturing, employees are stressed by different factors in their working environment, such as, noise, air pollution, heat, or low/high levels of humidity. To protect health and safety of workers in manufacturing industry, manufacturers have to follow strict laws and guidelines, e.g. in Germany, the Federal Ministry of Labor and Social Affairs (BMAS) limits noise and vibration levels within Germany's Occupational Safety Law (Arbeitssicherheitsgesetz - ASiG), German ordinance (LärmVibrationsArbSchV) and other additional legal guidelines. Thus, the working environment needs to be analyzed during the design and

Visualization of Large and Unstructured Data Sets– IRTG Workshop, 2010.
Editors: Ariane Middel, Inga Scheler, Hans Hagen; pp. 111–119

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

planning process with regard to occupational health and safety aspects. Dedicated methods and tools are needed for measurement, simulation, analysis, evaluation, visualization, and optimization of activities related to MSs.

Large companies are already adopting more and more new technologies into their MSs. Due to high costs and long start-up time, for small and medium enterprises (SMEs) it is still difficult to obtain enough know-how to perform various instruments to enhance the human factor in MS [4, 18]. There is a need for a cost efficient and easy to customize VF framework to enable manufacturing SMEs to view, analyze, inspect and optimize facility layout during the planning process and to train their workers [17].

Currently available visualization techniques aid in understanding and analysis of complex simulations or measurements and facilitate presentation and communication of them[25, 26]. The significance of the visualization of manufacturing systems has been discussed in [16, 12].

In our work we review existing human-centered VF concepts and discuss the potential use of visualization methods in this context. The remainder of our paper is organized as follows. In the next section we define the term VF, discuss existing concepts and introduce the human-centered view in VF. In section 3 we outline our human-centered VF concept and visualization framework. In section 4 we show noise analysis example in a factory model, before we conclude our work in section 5.

2 Related work

In literature, a large range of VF definitions and classifications of VF concepts have been proposed. For example, Jain et al. [10] classified the various VF definitions into four categories: i) representation of all major aspects of a factory, ii) a virtual organization, iii) the Virtual Reality (VR) representation of a factory, and iv) an emulation facility for production activity in a factory. Before we discuss the human-centered VF concept and our example implementation in section 3, we provide in the following a brief overview of various VF definition with special regard to the human-centered perspective.

2.1 The Virtual Factory concept

In the narrow sense, VF is a rebuilt model of a real factory. It is basically a 3D geometric model of facilities and machines, consisting of primitive shapes, colors, and textures etc., usually structured in parent-child hierarchies. Furthermore, a VF represents all activities and aspects in the real factory. Every single production activity is modeled and simulated to support or test operational, tactical and strategic decisions. In this case, a VF provides a basic virtual environment for simulation and visualization. Kelsick and Vance [11] define and develop a VF as "a visual, three-dimensional space in which to explore the effect of various product mixes, inspection schedules, and worker experience on productivity". Bodner and Reveliotis [3] define a VF as "an environment for use as a high fidelity test bed for manufacturing system design and control". Souza et al. [13] build a VF as a VR environment for modeling and simulation of a factory where both the layout design and the production process are taken into account.

In its broad sense, VF is a comprehensive concept which includes various instruments and can be studied from two views. One point of view is based on the technical aspect, in terms of computer aided modeling, simulation and visualization [23]. In this case VF can be understood as a general digital support system, which is a powerful tool in research and development areas as well as in production or controlling areas of manufacturing, especially in the planning phase. VF systems contain application software for possible activities in

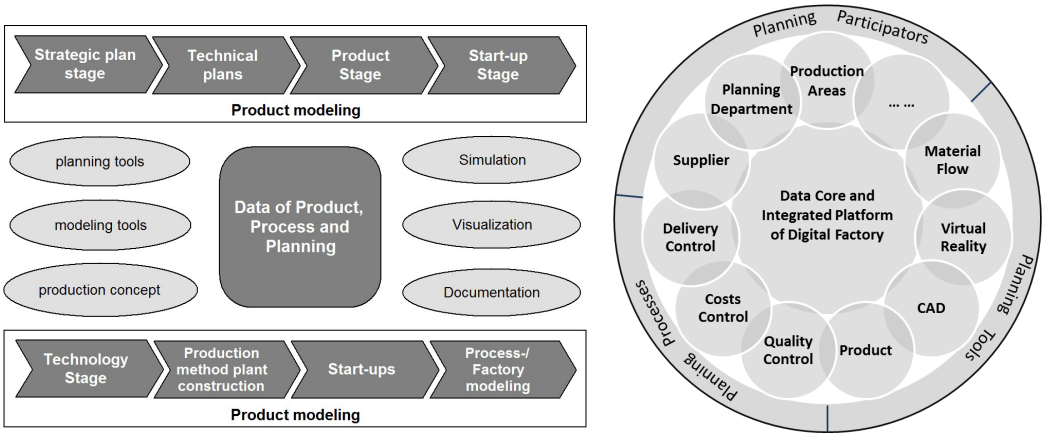
MSs. Banerjee and Zetu [1] describe virtual factories as "a distributed, integrated, computer-based composite model or a total manufacturing environment, incorporating all the tasks and resources necessary to accomplish the operation of designing, producing, and delivering a product". In [25], Zhong and Shirinzadeh built a VF framework using an integrated methodology. Their framework enables the direct integration of discrete event simulations with 3D animation.

Another point of view is the perspective of the management and organization. VFs are understood as "a co-operation of legal and economic independent enterprises (or units of them) with the aim to set up together the necessary preconditions to be able to identify opportunities in the markets and to grasp them quickly and efficiently together with partner" [24]. Schuh et al. [9] use a similar definition of VFs and discuss an application and branch customized VF concept. Upton and McAfee [20] emphasize VFs as collaborative and internetworked environments enabling several partners to share information and tools about a product, process or project.

In our work we define a VF as follows: a VF is a virtual environment, which contains a geometric model of the factory, machines and humans. In addition, measurement, simulation, visualization, documentation, VR, evaluation etc. are included as well. Furthermore, the feedback or modified information from a VF is coupled with real activities in MS. For example, material flow simulation in production processes, finite element simulation (FEM) of machining processes, collision detection simulation for layout planning or dynamic simulation for product design provide supporting information to build new or to modify existing factory models.

2.2 Concept of a Digital Factory

The established concept of the Digital Factory (DF) is closely related to the VF concept and is often used synonymously to VF in the literature [6]. The Association of German Engineers (VDI), defines in guideline 4499, DF as the generic term for a comprehensive network of digital models, methods and tools - including the simulation and 3D/VR-visualization for analysis of different processes in a factory.



(a) DF methods and tools, adapted from [12]

(b) DF modules, adapted from [22]

Figure 1 DF methods, tools and modules

Kühn [12] uses the VDI definition and describes several DF methods and tools (see Figure 1a) such as planning, modeling, production, simulation, and visualization, to support product, process, and factory modeling. These methods and tools share the data in the DF through a consistent data management system. The goal is the holistic planning, evaluation and ongoing improvement of all essential processes and resources of the factory. Zülch [26] defines a DF as a set of comprehensive technical and organizational tools for design, visualization and running of future production system in a digital model. Westkämper and Zahn [22] describe DFs as a development method and a platform that uses centered data and integrates many different tools and concepts. The authors discuss different modules of DF (see Figure 1b), e.g.: production, material flow, planning, supplier, product design, etc. as well as their tasks in a DF. Furthermore, they emphasize the significance of DF for factory and production planning.

Nylund et al. [14] summarize some general characteristics of VF and DF as follows: i) "an emerging and integrated approach to improve product and production engineering processes and technology", ii) "computer-aided tools, such as modelling and simulation, for planning and analyzing real manufacturing processes" and iii) "a framework for new technologies, including the collection of systems and methods." There is no need to distinguish the VF and DF in this field regarding MSs. In this paper we do not differentiate between DF and VF; meaning that DF is equal to VF.

2.3 Human-centered perspective

Most MSs are technology, not human, driven. The typical technology driven perspective is as follows: science finds, industry applies, man confirms. In contrast, the human-centered perspective can be described as: people propose, science studies, technology confirms. As shown in Figure 2, in the human-centered view, the human factor is considered as one of the most important pillars among technology and system. More information about human-centered systems can be found in [19].

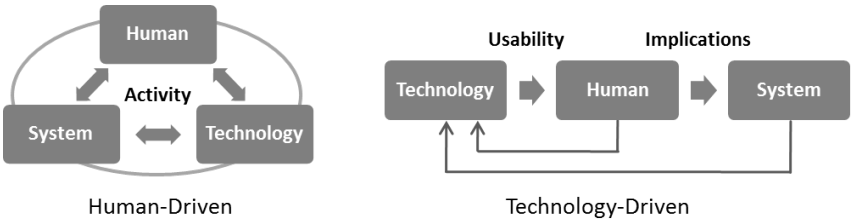


Figure 2 Human-Driven vs. Technology-Driven, adapted from [7]

Product design, factory layout design and production processes are usually accounted for VF concepts, for example in [23, 5, 25], while human factors are considered less important. Even if in real factories, many efforts are done to consider human factor. But its integration into VF is not sufficient. In VDI 3633 several criteria are studied from ergonomics point of view such as the staff work load regarding capacity demand, the degree of overstrain and fatigue, and the influence of the qualification of employees on their human error probability. This provides a starting point for a broad study field.

In the research project DiFac (Digital Factory for Human Oriented Production System) [18], Sacco et al. describe ergonomics, presence, and collaboration in an industrial virtual environment. Presence contains function components such as training, product development, and factory design. Zülch classifies ergonomics study into micro- and macro-ergonomics

[26]. The micro-ergonomic study analyzes human posture and body motion during tasks execution, whereas in macro-ergonomics, the stress and strain of the tasks to be accomplished by the worker are investigated. Furthermore, human-centered criteria are used to investigate and evaluate the working conditions. In [21], Viganò et al. introduce the GIOVE VF toolkit, which enables human oriented planning. They conclude that the major advantages of human oriented VF are increased efficiency of collaboration, reduced complexity of communication, reduced factory design time, and better working conditions.

In our human-centered VF concept we focus on training in the presence study and on macro-ergonomics in the ergonomics analysis. In the next section we describe the VF concept based on the above discussed human-centered point of view. In addition, we present an implementation example of our visualization-based framework.

3 Human-centered VF concept

Current work aims to use a low-cost solution to enable SMEs to visualize and analyze the information obtained from product lines or even the whole manufacturing system. Typical needs of SMEs in this context include enhancing the understanding of processes, reducing the potential risks of working environment in earlier planning level, and supporting the decision making. In many cases, these needs depend on what kind of products are produced, what kind of manufacturing methods are used, or what key issues have to be worried about. For example, thermo forming companies need to visualize air temperature, whereas for other machining enterprises, noise analysis may be more important. Hence, there is a need for a VF concept that can adapt to different fields and so its implementation has the following requirements: open architecture, flexibility, expandability, high compatibility and transparent development models.

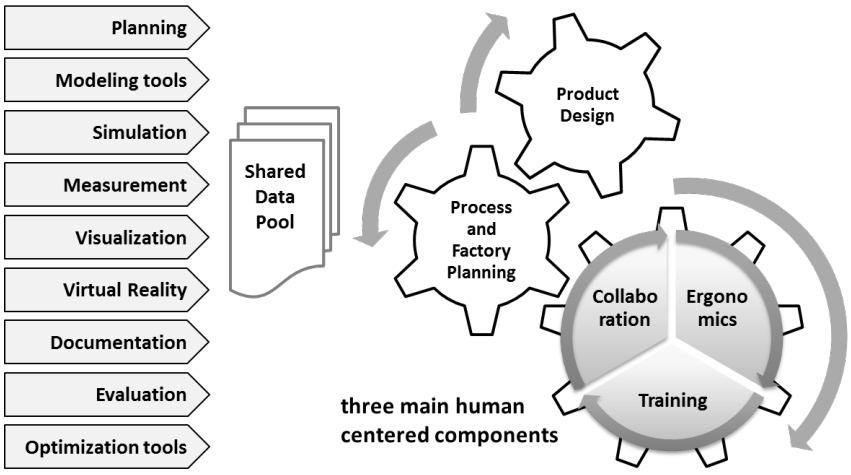


Figure 3 Human-centered study in VF

Figure 3 shows the structure of our human-centered VF concept. In our integrated VFs there are three main driving gears: i) human factor, ii) process and factory planning, and iii) product design. The three gears are interconnected. Thus, training, ergonomics and collaboration aspects are taken into account as are the other two major processes. A shared data pool enables the exchange of information between an extensive range of tools such as planning, modeling, simulation, production concept, visualization and others. As a result of

the complexity of the processes in MSs, the information obtained from the above mentioned tools is also complex, not intuitive and difficult to understand. Visualization serves as a key method to help analysts to verify models, understand simulation results and to communicate them to non-technical audience [15, 16, 12].

The outline of our scene-graph-based visualization framework is shown in Figure 4. The workflow begins with modeling or simulation software. In addition to the geometric model, measured and/or simulated data is collected, selected and integrated into the scene. OpenSceneGraph (OSG)[1] is used for graphical rendering and interactive scene building. We made this choice because OSG is a cross-platform graphics toolkit, which is widely used in different application areas running on a desktop computer as well as on VR system. Individual VF modules are developed on top of this flexible scene-graph-based model which helps us to fulfill individual user requirements and need.

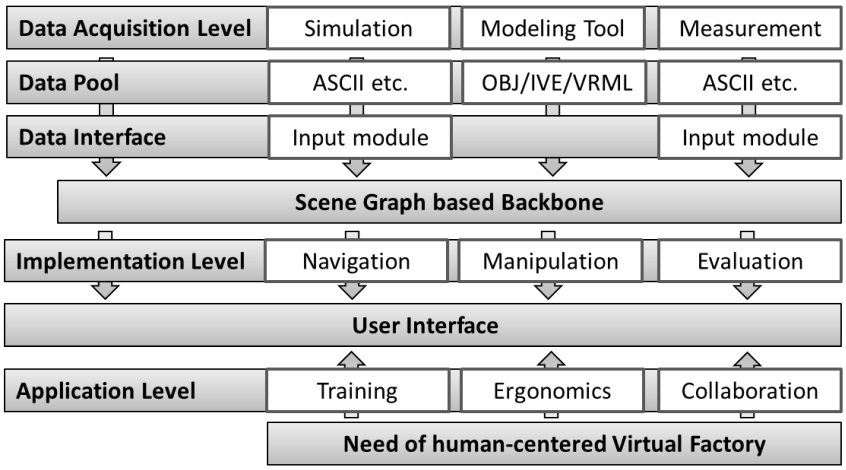


Figure 4 Framework structure

For example a scene graph can be modeled in 3DSMax[2] and exported as OSG native format (IVE), Alias Wavefront OBJ, or VRML2 (Virtual Reality Modeling Language) standard, in which all objects are organized in a tree like structure. In OSG this structure and node related information such as settings of the position of the objects, animations of objects, or definitions of logical relationships between objects are represented. This gives planners a clear understanding of the structure of the MS: the facility is the root, machines are usually nodes and the parts of the machines are leafs with additional information, such as dynamics and material characters.

In built scene graph several basic functions could be performed, such as view, navigation and manipulation of scene. Furthermore, by means of simulation, e.g. via the finite element (FEM) software DEFORM3D[3], analysis of machining processes in VFs becomes possible.

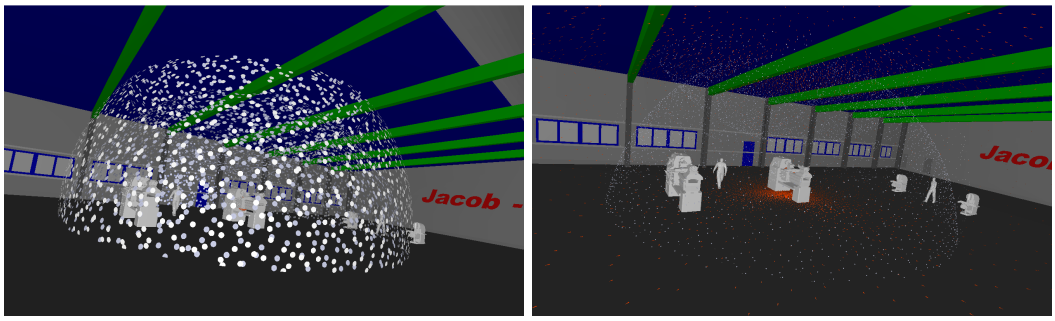
Based on our framework, SMEs will be able to build a software system to optimize facility layout and train their workers in a virtual and safe environment with low acquisition, administration and maintenance costs while reducing the time required for planning and training.

[1] OpenSceneGraph website: www.openscenegraph.org/projects/osg (2010)
[2] 3DS Max website: usa.autodesk.com (2010)
[3] DEFORM3D website: www.deform.com (2010)

4 Noise study example in a factory model

Noise from machining processes is one of the most frequent occupational hazards, it influences workers health and can even cause diseases. In Germany, about five millions employees are exposed to noise, which may cause hearing damage[4]. Therefore, it is important to analyze the noise in the VF. In this section we show an example implementation for sound analysis in a virtual model of a factory using our VF framework. A lot of research has been done in developing algorithms for indoor sound simulation. It is beyond the scope of this work to discuss these methods, the interested reader is referred to [8] for a detailed description of existing approaches.

For simulation of the sound in a VF based on given geometry, sound sources, and sound absorption at the surfaces, we integrated the phonon tracing algorithm described in [2] into our framework. In the "Data Acquisition Level" two passes are performed. In the first pass (phonon tracing), the method computes the energy or pressure decomposition for each sound particle (phonon) sent out from a sound source. In a second pass (phonon collection), the information is used to construct the impulse response (IR) for different listeners. IR is saved as ASCII file and imported to a scene graph. And from the IR, different metrics such as sound intensity (indicating the noise level) can be derived. Phonon tracing is coupled with an interactive visualization showing the sound propagation from a source, sound waves reflected from the scene, as well as received sound energy at a listener position. For more details about phonon tracing as well as sound visualization methods please refer to [2, 8].



(a) Sound particle propagation **(b)** Sound particle reflected from the floor

Figure 5 Sound visualization example

In the "Application Level", the sound simulation results is visualized in scene graph. Figure 5 depicts an example of sound visualization in a VF. The left figure shows sound particles building a wave front emitted from a machine, while the right figure shows particles reflected from the floor. The visualization of the sound propagation helps to identify the propagation of noise. The effect of different materials on the spectral energy/pressure distribution can be observed. The first few reflections already show whether certain frequency bands are rapidly absorbed and how much sound energy is reflected back into the scene causing a higher noise level. Scene materials of high reflectance can be identified and replaced in the virtual model to reduce the noise level in the VF.

[4] Federal Institute for Occupational Safety and Health (BAUA) website: www.baua.de (2010)

5 Conclusion and future work

In our paper we have discussed the human-centered VF concept and shown an example implementation to study the noise in a VF. Our scene-graph-based visualization framework is shown as a good choice for SMEs to implement various applications and fulfill their need of human factor study. The open structure of the visualization framework enables initiative applications to adapt enterprises depending questions in MSs. On the other hand, to perform such applications more available data sources and interfaces are needed. For example, to validate the sound simulation results, measurement data is requested. In the future work, these issues will be taken into account. Furthermore, more aspects of the human factor will be included into our VF model, and the use of VR systems for human-centered VF will be investigated.

6 Acknowledgement

This work was supported by the German Science Foundation (DFG) within the International Research Training Group (IRTG) 1131 Visualization of Large and Unstructured Data Sets Applications in Geospatial Planning, Modeling, and Engineering. Jacob Composite GmbH, Wilhelmsdorf Germany, provided a realistic facility model for the implementation of the described concept.

References

1 Prashant Banerjee and Dan Zetu. *Virtual Manufacturing*, page 10. John Wiley&Sons. Inc, New York, 2001.

2 Martin Bertram, Eduard Deines, Jan Mohring, Jevgenij Jegorovs, and Hans Hagen. Phonon tracing for auralization and visualization of sound. In *Proceedings of IEEE Visualization*, pages 151–158, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

3 Douglas A. Bodner and Spiridon A. Reveliotis. Virtual factories: an object-oriented, simulation-based framework for real-time fms control. In *Proceedings of the 1997 IEEE International Conference on Emerging Technologies and Factory Automation*, pages 208–213, 1997.

4 Uwe Bracht and Thomas Masurat. Modernes planungsvorgehen mit werkzeugen der digitalen fabrik - gerade auch für kmu bestens geeignet. In *Tagungsband "Digitale Fabrik-Potenziale für kleine und mittlere Unternehmen"*, Erfurt, 2002.

5 Joe Cecil and Atipol Kanchanapiboon. Virtual engineering approaches in product and process design. *The International Journal of Advanced Manufacturing Technology*, 31:846–856, January 2007.

6 Danfang Chen and Torsten Kjellberg. The digital factory and digital manufacturing - a review and discussion. In *Proceedings of 42nd CIRP Conference on Manufacturing Systems*, Grenoble, 2009.

7 Byoung K. Choi and Byung H. Kim. A human-centered vms architecture for next generation manufacturing. In *Proceedings of 2000 International CIRP Design Seminar*, pages 169–174, Israel, 2000.

8 Eduard Deines. *Acoustic Simulation and Visualization Algorithms*. PhD thesis, University of Kaiserslautern, 2008.

9 Asa Göransson Günther Schuh, Kai Millarg. *Die Virtuelle Fabrik*, page 13. Carl Hanser Verlag, 1998.

10 Sanjay Jain, Ngai Fong Choong, Khin Maung Aye, and Ming Luo. Virtual factory: an integrated approach to manufacturing systems modeling. *International Journal of Operations & Production Management*, 21:594 – 608, 2001.

11 Jason J. Kelsick and Judy M. Vance. The vr factory: discrete event simulation implemented in a virtual environment. In *Proceedings of ASME Design for Manufacturing Conference*, Atlanta, GA, 1998.

12 Wolfgang Kühn. *Virtual factory: an integrated approach to manufacturing systems modelling*, pages 2–4. Carl Hanser Verlag, 2006.

13 Arthur José Vieira Porto Mariella Consoni Florenzano Souza, Marco Sacco. Virtual manufacturing as a way for the factory of the future. *Journal of Intelligent Manufacturing*, 17(6):725–735, December 2006.

14 Hasse Nylund, Kai Salminen, and Paul Andersson. Digital virtual holons – an approach to digital manufacturing systems. In *Proceedings of the 41st CIRP Conference on Manufacturing Systems*, pages 103–106, Tokyo, Japan, 2008.

15 Jochen M. Quick, Chao Zhu, Haibin Wang, Meehae Song, and Wolfgang Müller-Wittig. Building a virtual factory. In *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 199–203, New York, NY, USA, 2004. ACM.

16 Maffhew W. Rohrer. Seeing is believing: the importance of visualization in manufacturing simulation. In *Proceedings of Winter Simulation Conference (WSC'00)*, volume 2, pages 1211–1216, Atlanta, GA, 2000.

17 Marco Sacco, Stefano Mottura, Luca Greci, Giampaolo Viganò, and Claudio Roberto Boër. Virtual factory as the way to support the design of a modular plant: design a mass-customised shoe production factory. In *Proceedings of IFAC-MIM2004 conference*, Athens, Greece, 2004.

18 Marco Sacco, Claudia Redaelli, Carmen Constantinescu, Glyn Lawson, Mirabelle D'Cruz, and Menelaos Pappas. Difac: digital factory for human oriented production system. In *HCI'07: Proceedings of the 12th international conference on Human-computer interaction*, pages 1140–1149, Berlin, Heidelberg, 2007. Springer-Verlag.

19 Nancy Talbert. Toward human-centered systems. *IEEE Computer Graphics and Applications*, 17:21–28, 1997.

20 David M. Upton and Andrew McAfee. *The real virtual factory*, pages 69–89. Harvard Business School Press, Boston, MA, USA, 1996.

21 G.P. Viganò, L. Greci, and M. Sacco. Giove virtual factory: the digital factory for human oriented production systems. In *Proceedings of CARV 3rd CIRP International Conference on Changeable*, Munich, Germany, 2009.

22 Engelbert Westkämper and Erich Zahn, editors. *Wandlungsfähige Produktionsunternehmen: Das Stuttgarter Unternehmensmodell (German Edition)*, pages 203–204. Springer, 1 edition, 2009.

23 Wenbin Zhai, Juanqi Yan, Dengzhe Ma, Jin Ye, and Fan Xiumin. Production engineering-oriented virtual factory, a planning cell-based approach to manufacturing systems design. *Int J Adv Manuf Technol*, 28:957–965, 2006.

24 Li Zheng and Frank Possel-Dölken. *Strategic Production Networks*, page 422. Springer, Heidelberg, 2002.

25 Yongmin Zhong and Bijan Shirinzadeh. Virtual factory for manufacturing process visualization. *Complexity International*, 12, 2005.

26 Gert Zülch. *Integrating Human Aspects into the Digital Factory: New Tools for the Human-oriented Design of Production Systems*, volume 160, pages 85–99. Springer Boston, 2005.