Accurate Surface Embedding for Higher Order Finite Elements

Stefan Suwelack[1]* Dimitar Lukarski[2] Vincent Heuveline[1] Rüdiger Dillmann[1] Stefanie Speidel[1]
[1]Karlsruhe Institute of Technology [2]Uppsala University

Abstract

In this paper we present a novel approach to efficiently simulate the deformation of highly detailed meshes using higher order finite elements (FE). An efficient algorithm based on non-linear optimization is proposed in order to find the closest point in the curved computational FE mesh for each surface vertex. In order to extrapolate deformations to surface points outside the FE mesh, we introduce a mapping scheme that generates smooth surface deformations and preserves local shape even for low-resolution computational meshes. The mapping is constructed by representing each surface vertex in terms of points on the computational mesh and its distance to the FE mesh in normal direction. A numerical analysis shows that the mapping can be robustly constructed using the proposed non-linear optimization technique. Furthermore it is demonstrated that the numerical complexity of the mapping scheme is linear in the number of surface nodes and independent of the size of the coarse computational mesh.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling —Physically based modeling; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism —Animation;

Keywords: deformable models, higher order FE

1 Introduction

Co-rotated finite elements (FE) are used in many applications for physically and visually accurate interactive simulation of deformable models. The method does not only produce visually pleasant deformations in real-time, but also converges to a true physical solution. This is an import feature for medical simulations, where physical correctness is required. In the realm of mechanical engineering it is well known that quadratic tetrahedra usually perform much better than linear tetrahedral meshes [Cifuentes and Kalbag 1992]. This is in particular true for the simulation of incompressible objects. However, despite the widespread use of FE based simulations in computer graphics, to the best of our knowledge only three groups have investigated quadratic volumetric FE in the context of real-time deformable models [Kaufmann et al. 2009][Mezger et al. 2009][Weber et al. 2011]. They all concluded that higher order elements are superior to linear elements in terms of speed and accuracy.

The main obstacle that prevents a use of higher order FE methods is, despite their slightly more complex implementation, the difficult integration into the visualization pipeline. It is in particular challenging to accurately map high resolution surface meshes to the computational FE grid. Weber et al. showed that basis functions in Bernstein-Bézier form yield a high quality embedding [Weber et al. 2011]. However, the presented approach did not make use of curved, isoparametric elements. Mezger et al. showed that isoparametric FE clearly improve the geometric approximation of coarse computational meshes [Mezger et al. 2009]. They use a special meshing approach to make sure that all FE nodes lie on the surface mesh. If the geometry approximation of the computational grid is sufficiently close, this allows to extrapolate the displacements to surface points outside the FE mesh without visible artifacts.

In this context we present an efficient algorithm based on non-linear optimization in order to find the closest point in the curved computational FE mesh for each surface vertex. The algorithm makes use of the topology of both the surface and the volume mesh in order to achieve a numerical complexity that is linear in the number of surface nodes and independent of the size of the computational mesh. Furthermore, we propose a novel mapping scheme that allows to robustly extrapolate displacements from the computational to the visual mesh even for vertices that are further away from the FE grid. We represent each surface vertex in the undeformed configuration in terms of a point on the computational mesh and its distance to the FE mesh in normal direction. By computing a smooth normal field for every timestep, this representation yields a smooth mapping scheme that preserves local shape even under large rotations. The proposed optimization algorithm is extended in order to obtain this representation in a stable and efficient way for arbitrary FE meshes.

The main contributions of this paper are:

- We propose a surface representation that defines each surface vertex in terms of a point on the computational mesh and its distance to the FE mesh in normal direction. This representation yields a novel mapping scheme that allows to accurately embed high resolution surface meshes into higher order FE grids.
- We introduce an efficient algorithm based on non-linear optimization to find the closest point in the curved computational FE mesh for each surface vertex and to extract the aforementioned representation.
- A numerical analysis show that the algorithm's computational complexity is nearly independent of the resolution of the coarse FE mesh and linear in the number of surface vertices.

2 Related work

Co-rotated finite elements have become an established method for the real-time simulation of deformable bodies [Hauth and Strasser 2004][Muller and Gross 2004]. Several improvements such as stable extraction of element rotations and fast multigrid solvers have been proposed [Georgii and Westermann 2008]. The use of higher order isoparametric elements for real-time deformations has been investigated for the first time by Mezger et al. [Mezger et al. 2009]. Furthermore, Kaufmann et al. studied quadratic basis functions in the context of a Discontinuous Galerkin FEM framework [Kaufmann et al. 2009] and Weber et al. proposed a quadratic tetrahedron based on polynomials in Bernstein-Bézier form [Weber et al. 2011].

A straight forward approach to surface embedding is to completely

*suwelack@kit.edu

SCA 2013, July 19 – 21, 2013, Anaheim, California.

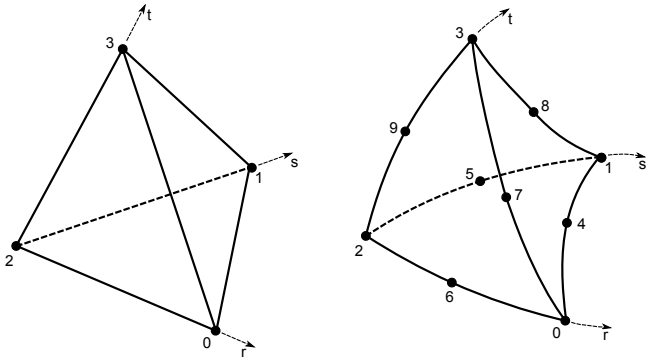

Figure 1: *Linear tetrahedron with 4 nodes (tet4) and quadratic tetrahedron with 10 nodes (tet10)*

submerge the high resolution surface mesh into the computational grid [Lee et al. 2009]. Unfortunately this approach is very inaccurate in comparison with boundary conforming FE meshes [Zhu et al. 2010].

Our novel embedding approach based on the surface normal is motivated by developments in the field of shape editing techniques. There, both the pyramid coordinate scheme developed by Sheffer et al. as well as the Green Coordinates introduced by Lipman et al. use a representation that includes the surface normal [Sheffer and Kraevoy 2004] [Lipman et al. 2008].

It has been shown that deformable models can be efficiently solved using multigrid schemes on structured hexahedral grids [Dick et al. 2011]. Although this approach allows to simulate large models in real-time, a lot more degrees of freedom are necessary in comparison with boundary conforming unstructured FE grids [Zhu et al. 2010]. However, volume locking of linear tetrahedral grids can severely degrade the performance of multigrid schemes. An accurate mapping scheme is an essential component in order to develop a multigrid scheme based on unstructured, isoparametric quadratic elements.

3 Co-rotated quadratic tetrahedra

The finite element method allows to find the exact solution of a variational formulation in the finite dimensional space V_h, that is spanned by the basis functions N_I. When using nodal basis functions, the position inside an n-node element

$$\mathbf{x}(t) = N_I \mathbf{x}_I(t) \tag{1}$$

is given by the linear combination of the n basis functions N_I weighted with the position of the element nodes $\mathbf{x}_I(t)$ (in the above equation and from here on out summation over repeated indices is implied).

Typically, the shape functions are defined in a local coordinate system (r, s, t) and all element based operations such as calculating deformations, extracting rotations or numerical integration are performed in this local coordinate system. Polynomial functions ('shape functions') are used to map the local coordinates to the global coordinate system. If the polynomial degree of the shape functions matches the order of the basis functions, the element is called isoparametric.

The isoparametric 4-node tetrahedron interpolates the position linearly between the nodes (Fig. 1). Consequently, the stress is constant over the element and the element shows significant volumetric locking for nearly incompressible materials [Belytschko et al. 2000]. In contrast, the 10-node tetrahedron interpolates positions with 2nd order polynomials and doesn't suffer from volume locking. Furthermore, the mapping from local to global coordinate is also quadratic, which allows curved boundaries and therefore better approximation of geometry (see Fig. 1). However, the quadratic mapping is not bijective and not analytically invertible. Instead, numerical optimization techniques have to be employed in order to find the local element coordinates for a given global position. For a detailed description of the tetrahedral shape function we refer to standard FE literature [Belytschko et al. 2000].

4 Closest point search in higher order meshes

In this section we present an efficient scheme to find the closest point $p(r, s, t)$ in the FE mesh $\mathcal{M}$ for each vertex v_i of the triangular surface mesh $\mathcal{S}$. First, we demonstrate how to find $p(r, s, t)$ for a single vertex v_i, before intruducing a recursive mapping scheme that computes $p(r, s, t)$ for all v_i in $\mathcal{S}$.

The distance $d(r, s, t)$ from a point $p(r, s, t)$ in the element $\mathbb{E}$ to an arbitrary surface vertex v_i is given by

$$d(r, s, t) = \|v_i - \mathbf{x}_I N_I(r, s, t)\| . \tag{2}$$

As s_i can be outside of $\mathcal{M}$, we have to constrict (r, s, t) such that $p(r, s, t)$ is indeed in $\mathbb{E}$. The corresponding constrained optimization problem reads

$$\begin{aligned} &\min d(p(r, s, t)) \text{ s.t.} \\ &r > 0, s > 0, t > 0 \text{ and } r + s + t \leq 1. \end{aligned} \tag{3}$$

If the shape functions N_I are linear, then the solution to this problem can be computed analytically using barycentric coordinates. However, in the quadratic case, non-linear programming has to be used. We solve problem (3) with an extended Levenberg-Marquardt algorithm and use finite differences to calculate the Jacobian [Nocedal and Wright 1999] [Kanzow et al. 2005]. It is noteworthy that a simple Newton-Raphson scheme is not guaranteed to converge even in the unconstrained case when v_i is inside $\mathbb{E}$.

The solution to problem (3) is the local optimum for $p(r, s, t) \in \mathbb{E}$. We use a recursive algorithm to efficiently select the element $\mathbb{E}_{\min}$ where $p(r, s, t)$ attains its global minimum distance $d_{\min}$ in $\mathcal{M}$ (see algorithm 1).

Starting with an initial guess $\mathbb{E}$, we first determine the closest point p to $\mathbf{v}$ in $\mathbb{E}$. The basic idea of the algorithm is to recursively call this procedure on the neighboring tetrahedra, if p is on a face of $\mathbb{E}$. The recursion is finished if a point p inside a tetrahedron is found ($d = 0$) or if the considered neighbor tetrahedron has already been visited (i.e. is in set $\mathcal{T}$). During each recursion step, the minimum distance $d_{\min}$ in $\mathbb{E}_{\min}$ is updated if a closer point $\hat{p}$ is found.

The algorithm efficiently computes the local minimum distance $d_{\min}$ for a given initial guess $\mathbb{E}$. However, this local minimum only corresponds to the global minimum, if $\mathbb{E}$ is close enough to the real solution.

4.1 Recursive mapping scheme

An outer recursion to algorithm 1 is introduced in order to map the whole surface mesh $\mathcal{S}$ and to efficiently find the global minimum $d_{\min}$ for all $\mathbb{E}_i$ in $\mathcal{M}$. We first establish an initial mapping by finding the closest surface vertex s_0 in $\mathcal{S}$ for an arbitrary point p_0 in

Algorithm 1 Recursively find $p(r,s,t) \in \mathcal{M}$

procedure CLOSESTPOINT($\mathbf{v}, \mathbb{E}, d_{\min}, \mathbb{E}_{\min}, \mathcal{T}$)
  Solve (3) to find closest point p with distance d to $\mathbf{v}$ in $\mathbb{E}$
  if $d < d_{\min}$ **then**
    $d_{\min} = d, \mathbb{E}_{\min} = \mathbb{E}$
  if p is on face of $\mathbb{E}$ **then**
    if p is on surface of $\mathcal{M}$ **then**
      if p is on surface edge **then**
        Find neighbour tetrahedron $\mathbb{E}_1$ that contains surface triangle which shares edge
    else
      Find tetrahedron $\mathbb{E}_1$ that shares face
    if $\mathbb{E}_1$ exists and $\mathbb{E}_1 \notin \mathcal{T}$ **then**
      Add $\mathbb{E}_1$ to $\mathcal{T}$
      $(\hat{p}, \hat{d}, \hat{\mathbb{E}})$ = CLOSESTPOINT($\mathbf{v}, \mathbb{E}_1, d_{\min}, \mathbb{E}_{\min}, \mathcal{T}$)
      if $\hat{d} < d_{\min}$ **then**
        $d = \hat{d}, \mathbb{E} = \hat{\mathbb{E}}, p = \hat{p}$
        $d_{\min} = d, \mathbb{E}_{\min} = \mathbb{E}$
  return $p, d, \mathbb{E}$

$\mathbb{E}_0$. Please note that this inverse mapping problem can be robustly solved as the resolution of $\mathcal{S}$ is much higher than the resolution of $\mathcal{M}$.

We arbitrarily select a triangle $\mathbb{T}_0$ that contains v_0 and start the recursive scheme detailed in algorithm (2). The basic idea is to map all points in a given triangle $\mathbb{T}$ and then use the current mapping result $(p, \mathbb{E})$ as the initial guess for mapping the neighbor triangles of $\mathbb{T}$.

Algorithm 2 Recursively map triangles in $\mathcal{S}$

procedure MAPTRIANGLE($\mathbb{T}, \mathbb{E}, d_t, \mathcal{P}$)
  triangleMapped = true
  for all Surface vertices $\mathbf{v}$ in $\mathbb{T}$ **do**
    if $\mathbf{v} \notin \mathcal{P}$ **then**
      Initialize empty $\mathcal{T}$, $d_{\min} = \inf$
      $(p, d, \hat{\mathbb{E}})$ = CLOSESTPOINT($\mathbf{v}, \mathbb{E}, d_{\min}, \mathbb{E}_{\min}, \mathcal{T}$)
      if $d < d_t$ **then**
        $\mathbb{E} = \hat{\mathbb{E}}$
        Map vertex $\mathbf{v}$ to point p in $\mathbb{E}$
        Add $\mathbf{v}$ to $\mathcal{P}$
      else
        triangleMapped = false
  if triangleMapped **then**
    for all Neigbour triangles $\mathbb{T}_n$ **do**
      MAPTRIANGLE($\mathbb{T}_i, \mathbb{E}, d_t, \mathcal{P}$)
  return

The initial guess that is used for algorithm 1 thus depends on the order in which the triangles are mapped. Due to this reason the initial guess is not guaranteed to lead to a global optimum for $d_{\min}$. That's why we control the mapping order by introducing the distance threshold d_t. The triangle $\mathbb{T}$ is only mapped (and the recursion only continues) if the distance for each surface vertex in $\mathbb{T}$ is below d_t. During the mapping, this distance threshold is incrementally raised in an outer iteration that calls algorithm (2) until all triangles are mapped.

We note that the recursive scheme has to be unrolled into a loop in the implementation in order to guarantee efficiency and stability for large meshes.

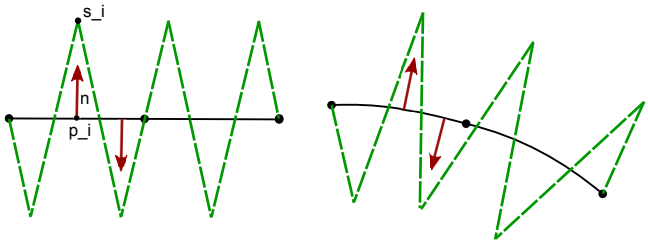

Figure 2: *Sawtooth surface attached to 1D quadratic edge. The deformation of the edge can be extrapolated to the surface while preserving local shape features.*

5 Accurate surface embedding

The goal of our approach is to map the surface mesh $\mathcal{S}$ onto the FE mesh $\mathcal{M}$ in such a way that the deformed surface $\mathcal{S}'$ can be extrapolated from the deformed FE mesh $\mathcal{M}'$ in a fast and correct way. In this section we present a mapping that preserves the smoothness of the deformation (and thus local shape features) even for points outside the computational mesh. This is achieved by representing each surface model vertex $\mathbf{v}_i$ in terms of a point $\mathbf{p}_i$ on the FE mesh and its distance d_i to the computational mesh in normal direction (see Fig. 2). The deformed vertex position

$$\mathbf{v}_i' = \mathbf{p}_i' + d_i \mathbf{n}_i' \tag{4}$$

can subsequently be constructed using the position of $\mathbf{p}_i$ in the deformed configuration ($\mathbf{p}_i'$) as well as the deformed normal $\mathbf{n}_i'$.

5.1 Smooth normal field

At any surface point p_i on the mesh given in local coords r_i, s_i with respect to the quadratic surface triangle, the spatial derivates

$$\frac{\partial \mathbf{x}(p_i)}{\partial r} = \mathbf{x}_I \frac{\partial N_I(r_i,s_i)}{\partial r}, \quad \frac{\partial \mathbf{x}(p_i)}{\partial s} = \mathbf{x}_I \frac{\partial N_I(r_i,s_i)}{\partial s} \tag{5}$$

can be calculated using the position of the nodes and the shape function derivatives at r_i, s_i of the 6 node quadratic triangle. The normal n_i at p_i is then given by

$$\mathbf{n}_i = \left(\frac{\partial \mathbf{x}(p_i)}{\partial r} \times \frac{\partial \mathbf{x}(p_i)}{\partial s} \right) / \left\| \frac{\partial \mathbf{x}(p_i)}{\partial r} \times \frac{\partial \mathbf{x}(p_i)}{\partial s} \right\|. \tag{6}$$

The smoothness of the proposed mapping scheme stems from a smooth normal field. However, the FE mesh is only C^0-continuous, which means that the partial derivatives (and thus the normals) are discontinuous at the element boundaries. In order to overcome this problem we average the normals at each node using the angle weighted average scheme. In this way we obtain a normal $\overline{n}_I$ for each surface vertex of the FE mesh and the normal

$$\mathbf{n}_i = (\overline{n}_I N_I(r_i,s_i)) / \left\| \overline{n}_I N_I(r_i,s_i) \right\| \tag{7}$$

can be interpolated using the standard shape functions of the quadratic triangle. The resulting normal field is not only smooth, but can also be efficiently computed. We note at this point that it is not necessary to re-compute the deformed vertex normals $\overline{n}_I'$ every timestep according to eq. (6). As detailed in section 5, we can instead define rotation matrices at each node of the FE mesh that can be used to rotate the normal to the deformed configuration.

5.2 Normal correction

The construction of the aforementioned representation (p_i, d_i) can be regarded as a global optimization problem. For complex, non-convex meshes the solution to this problem is not necessarily unique. Our strategy is to first find the closest point $p \in \mathcal{M}$ to s_i. Afterwards a correction step is performed, in order to make sure that the normal n_i at p_i does indeed point in the direction of v_i.

Due to the nature of the normal smoothing, the normal n_i at p_i does not necessarily point in the direction of v_i. We therefore introduce a final correction step to guarantee that the position of v_i can be expressed as $p_i + d \cdot n_i$. We start by defining the difference vector

$$\boldsymbol{\delta}(r,s,t) = v_i - (\mathbf{x}_I N_I(r,s,t) + d \cdot \overline{\mathbf{n}}_I N_I(r,s,t)), \quad (8)$$

where d denotes the distance as defined in eq. (2). The correction step can then be expressed in terms of the minimization problem

$$\begin{aligned} &\min \boldsymbol{\delta}(r,s,t) \text{ s.t.} \\ &r > 0, s > 0, t > 0 \text{ and } r + s + t \leq 1. \end{aligned} \quad (9)$$

We solve this optimization problem in the same way we solve problem (3): A recursive scheme along the lines of algorithm 1 is employed on top of a constrained Levenberg-Marquardt optimizer that uses finite differences to approximate the Jacobian.

6 Results

A co-rotated quadratic FE method and the novel mapping scheme were implemented in the SOFA framework [Faure et al. 2012]. The constrained Levenberg-Marquardt implementation from the Levmar library was used to solve the minimization problem (3) [Lourakis Jul. 2004]. All CPU computation are performed on a single core of an Intel i7-930.

6.1 Accuracy

We consider the deformation of a seahorse model in response to a volumetric force. Starting from a high resolution surface mesh with 40k triangles three low resolution volumetric meshes for FE computations are constructed: A linear tetrahedral (tet4) mesh with 258 DOF and 181 elements, a linear tetrahedral mesh with 992 elements and 1488 DOF as well as a mesh with 171 quadratic elements (tet10) and 1260 DOF (see Fig. 3). The high resolution surface mesh is mapped to the linear tetrahedral FE meshes using barycentric mapping, while it is embedded into the higher order FE mesh using the proposed mapping scheme. All models are assumed to be nearly incompressible (Poisson's ratio $\nu = 0.49$).

If subjected to the volumetric force, the linear meshes show severe volume locking and consequently do not deform nearly as much as the quadratic FE mesh. In order to quantify the locking behavior we performed a comparable experiment on a simple beam geometry. This experiment showed that linear tetrahedral meshes need up to 40x more DOF than quadratic FE meshes to achieve the same accuracy. For the following visual comparisons, we subject the linear meshes to higher forces in order to overcome this artificial stiffness and achieve comparable deformation patterns for the three meshes (see Fig. 3, 4).

It is evident that the low resolution tet4 mesh does not only fail to capture the deformation, but the barycentric coupling with the surface mesh also produces large visible artifacts. In the presence of strong local rotations the surface features such as the dorsal fin are highly distorted (Fig. 4 left). In contrast, the proposed mapping scheme smoothly extrapolates the the deformation of the quadratic mesh is to the surface mesh and preserves local shape features (Fig. 3). The visible artifacts are substantially reduced if the higher resolution linear tetrahedral mesh is used for the FE computation. However, the volumetric locking can still be observed at the end of the Seahorse's tail; although the linear mesh has slightly more DOF (1488 vs. 1260) than the tet10 mesh, the movement of the tail is much better resolved by the quadratic FE.

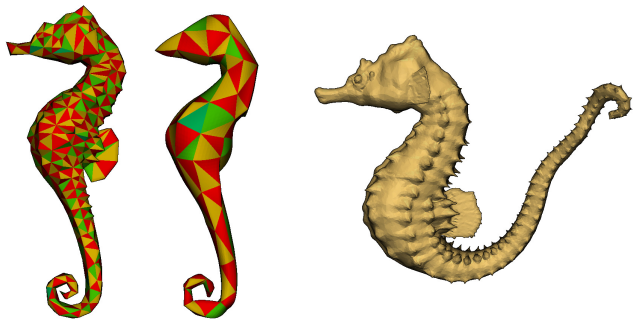

Figure 3: *Deformation of a seahorse model: Linear tetrahedral FE mesh with 992 elements, quadratic tetrahedral mesh with 171 elements (middle) and the deformed surface mesh mapped to the higher order FE mesh using the proposed mapping scheme (right)*

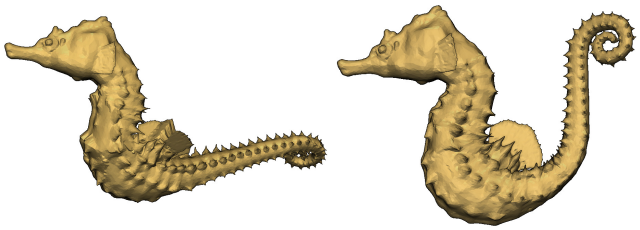

Figure 4: *Seahorse subjected to volumetric force: High resolution surface mesh mapped to a linear tetrahedral mesh with 182 elements (left) and 992 elements (right) using barycentric mapping*

Upon closer inspections, it can be seen that even for the higher resolution tet4 mesh, the thorns on the surface of the model are still visibly distorted (see Fig. 5 left). In the quadratic case, the proposed mapping scheme ensures that the shape of the thorns is preserved. Since the dorsal fin of the seahorse is not included in the tet10 mesh (Fig. 3), it has to be extrapolated from the body of the model. Fig. 5 shows that this might lead to undesired results (i.e. deformation of the fin) even if the rotation invariant mapping is employed.

6.2 Speed

In order to evaluate the performance and robustness of the mapping scheme, we construct surface and volume meshes in different resolutions from the Asian dragon model that is included in the Stanford 3D scanning repository. We then map surface meshes of different resolution to a volume mesh of 3000 elements and 1000 elements, respectively. In all cases the optimization algorithm robustly finds the correct surface representation in terms of each associated point on the computational mesh and the corresponding distance to the FE mesh in normal direction. It is clear from the design of the recursive mapping scheme that its numerical complexity should be

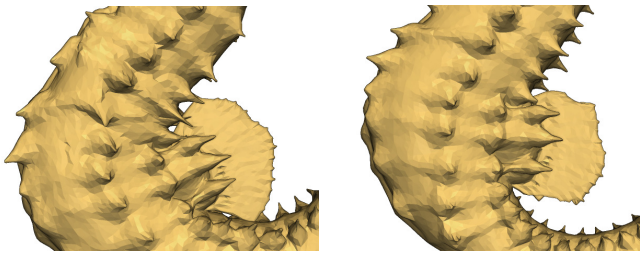

Figure 5: *While the barycentric coupling leads to visible distortions (left), the proposed mapping preserves local shape features (right).*

Figure 6: *Asian dragon model with 100k surface triangles (right) is mapped to a computational mesh with 1000 quadratic tetrahedra (left).*

linear in the number of surface nodes and independent of the size of the coarse computational mesh. The results of the convergence analysis (Fig. 7) exactly reproduce this expectation.

Due to these excellent convergence properties, even large surface meshes with several 100k elements can be mapped in less than a minute, although the constrained Levenberg-Marquardt optimization for each element is numerically complex. As the construction of the proposed surface representation is done in a pre-process, the obtained coordinates can also be saved and simply loaded upon startup of the simulation similar to texture coordinates.

The application time of the mapping during the simulation is negligible even for larger surface meshes. The only additional computation in comparison with standard barycentric mapping is the calculation of the node normals for the deformed computational mesh. This can be done very efficiently by simply applying nodal rotation matrices to the normals in the undeformed configuration.

7 Conclusion

In this paper we proposed a novel mapping scheme for embedding highly detailed triangular surface meshes to a coarse computational grid. This mapping preserves local shape features and causes only negligible overhead during the simulation. It is constructed by representing each surface vertex in terms of points on the computational mesh and its distance to the FE mesh in normal direction. We proposed a recursive mapping scheme based on non-linear optimization to robustly construct this representation for curved isoparametric higher order FE meshes. The numerical complexity of the mapping scheme is linear in the number of surface nodes and independent of the size of the coarse computational mesh.

The proposed method permits the use of a small higher order FE meshes to animate complex surface models. It is thus possible to leverage the computational efficiency of higher order isoparametric

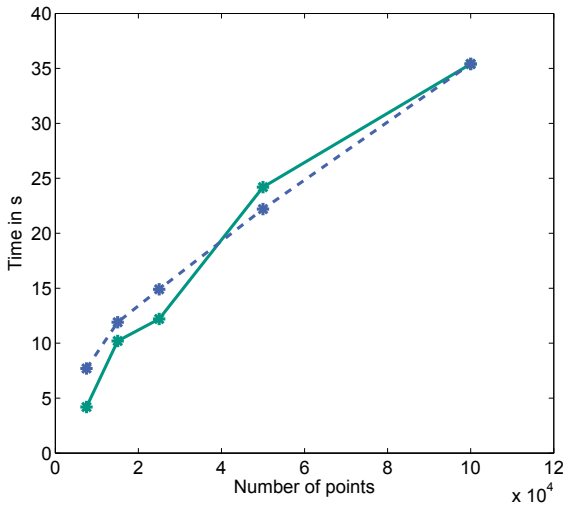

Figure 7: *Different surface meshes are mapped to a 3000 element mesh (blue) and a 1000 element (mesh). The mapping time is linear in the number of points of the surface mesh and independent of the computational mesh size.*

elements for real-time computations.

We would like to point out that the mapping scheme is neither restricted to quadratic elements nor to tetrahedral elements. The approach can be used to map highly detailed surface mesh to shell or volume meshes of arbitrary order and element type.

It is well known that multigrid based techniques constitute very efficient solving scheme for elastic problems [Zhu et al. 2010]. In the context of incompressible materials the performance of multigrid schemes based on unstructured linear tetrahedral meshes can be severely degraded by the volume locking effect. In this scenario, quadratic elements provide a promising alternative. The proposed mapping is an essential building block in such a scheme as it can be used to transfer displacements between the grids at different resolution.

Acknowledgements

This work was supported by the German Research Foundation in the framework of the SFB/Transregio 125 Cognition-Guided Surgery.

References

BELYTSCHKO, T., LIU, W., AND MORAN, B. 2000. *Nonlinear finite elements for continua and structures*, vol. 36. Wiley.

CIFUENTES, A., AND KALBAG, A. 1992. A performance study of tetrahedral and hexahedral elements in 3-d finite element structural analysis. *Finite Elements in Analysis and Design 12*, 3, 313–318.

DICK, C., GEORGII, J., AND WESTERMANN, R. 2011. A real-time multigrid finite hexahedra method for elasticity simulation using cuda. *Simulation Modelling Practice and Theory 19*, 2, 801–816.

FAURE, F., DURIEZ, C., DELINGETTE, H., ALLARD, J., GILLES, B., MARCHESSEAU, S., TALBOT, H., COURTECUISSE, H.,

BOUSQUET, G., PETERLIK, I., ET AL. 2012. Sofa: A multi-model framework for interactive physical simulation. *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, 283–321.

GEORGII, J., AND WESTERMANN, R. 2008. Corotated finite elements made fast and stable. In *Proceedings of the 5th Workshop On Virtual Reality Interaction and Physical Simulation*, 11–19.

HAUTH, M., AND STRASSER, W. 2004. Corotational simulation of deformable solids. In *J. Winter School of Computer Graphics*, 137–145.

KANZOW, C., YAMASHITA, N., AND FUKUSHIMA, M. 2005. Levenberg–marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *Journal of Computational and Applied Mathematics 173*, 2, 321–343.

KAUFMANN, P., MARTIN, S., BOTSCH, M., AND GROSS, M. 2009. Flexible simulation of deformable models using discontinuous Galerkin FEM. *Graphical Models*.

LEE, S., SIFAKIS, E., AND TERZOPOULOS, D. 2009. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Transactions on Graphics (TOG) 28*, 4, 99.

LIPMAN, Y., LEVIN, D., AND COHEN-OR, D. 2008. Green coordinates. *ACM Trans. Graph. 27*, 3 (Aug.), 78:1–78:10.

LOURAKIS, M., Jul. 2004. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. [web page] `http://www.ics.forth.gr/~lourakis/levmar/`. [Accessed on 31 Jan. 2005.].

MEZGER, J., THOMASZEWSKI, B., PABST, S., AND STRASSER, W. 2009. Interactive physically-based shape editing. *Computer Aided Geometric Design 26*, 6, 680–694.

MULLER, M., AND GROSS, M. 2004. Interactive virtual materials. In *Proceedings- Graphics Interface*, CANADIAN INFORMATION PROCESSING SOCIETY, 1 Yonge St, 2401, Toronto, ON M 5 E 1 E 5, Canada.

NOCEDAL, J., AND WRIGHT, S. J. 1999. *Numerical optimization*. Springer verlag.

SHEFFER, A., AND KRAEVOY, V. 2004. Pyramid coordinates for morphing and deformation. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, IEEE, 68–75.

WEBER, D., KALBE, T., STORK, A., FELLNER, D., AND GOESELE, M. 2011. Interactive deformable models with quadratic bases in bernstein–bézier-form. *The Visual Computer 27*, 6, 473–483.

ZHU, Y., SIFAKIS, E., TERAN, J., AND BRANDT, A. 2010. An efficient multigrid method for the simulation of high-resolution elastic solids. *ACM Transactions on Graphics (TOG) 29*, 2, 16.