3rd Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2006)
C. Mendoza, I. Navazo (Editors)

Real Time Simulation of Phaco-emulsification for Cataract Surgery Training

Marco Agus, Enrico Gobbetti, Giovanni Pintore, Gianluigi Zanetti, Antonio Zorcolo

CRS4 Visual Computing Group, POLARIS Edificio 1, 09010 Pula, Italy

Abstract

We present a real–time simulation of the phaco-emulsification task in a virtual reality training system for cataract surgery. Phaco–emulsification consists in breaking in small fragments and completely removing the eye crystalline lens by employing an ultra–sound tool called phaco–emulsificator. Our approach employs a mesh-less shape–based dynamic algorithm integrated with a simplex geometry representation in order to efficiently handle the rendering process and the continuous modifications involved by the surgical tool interaction, and with a smoothed particle hydrodynamics scheme with spatial ordering for handling fragments interactions. The complete training system also simulates other tasks involved in cataract surgery, like the corneal incision and the capsulorhexis. The simulator runs on a multiprocessing PC platform and provides realistic physically-based visual simulations of tools interactions. The current setup employs SensAble PHANToM for simulating the interaction devices, and a binocular display for presenting images to the user.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: Virtual Reality I.6 [Computing Methodologies]: Simulation and Modeling

Keywords: [Surgical simulation] [Real-time rendering] [Physically-based modeling]

1. Introduction

The word cataract is used to describe a clouding of the natural lens, that causes a gradual impairment of vision. The most efficient solution for restoring vision consists of extracting the cataract and substituting it with an **intra-ocular lens**. Modern advances in micro-surgical techniques allow cataracts to be removed safely and are very successful in restoring vision. Nowadays, this is probably the most frequently performed surgery in the world. Given the diffusion and the complexity of the specialty, training is considered very important. In this context, the usage of Virtual Reality based training systems would go along way towards improving the learning curves and the quality of apprenticeship. That's why a number of research groups are working to develop VR simulator systems realistically mimicking a patient-specific operating environment [WSM02a, SLA*02, LNS*01]. Phacoemulsification cataract surgery is a procedure in which an ultrasonic device is used to break up and then remove the cloudy crystalline lens, or cataract. It is difficult to simulate, since it involves the usage of a complex tool, a metal probe that vibrates back and forth at a high frequency in order to break the central nucleus into tiny pieces (emulsify) and to gently suck (aspirate) those pieces out of the eye.

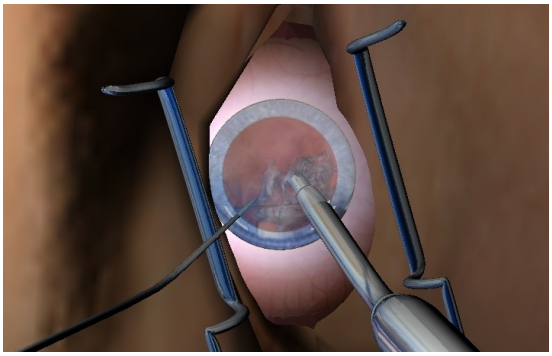


Figure 1: *A snapshot of our cataract simulator prototype.*

In this paper, we describe a simulation technique used to represent the phaco-emulsification task in the context of a real–time virtual reality training system for cataract surgery. The proposed approach involves: a mesh-less shape–based simulation scheme, containing improvements of the technique described in [MHTG05]; a simplex geometry representation aimed at efficiently handling rendering and the continuous modifications involved by the surgical tool interaction; a smoothed particle hydrodynamics based scheme with spatial ordering for handling fragments interactions [MCG03].

The phaco–emulsification simulation component is integrated in a cataract extraction training system containing also real–time physical simulation of the capsulorhexis task as well as a simple geometric simulation of the corneal incision task. The simulator runs on a multiprocessing PC platform and provides realistic visual feedback with real-time interaction. The current setup employs SensAble PHANToMs for simulating the interaction devices, and a binocular display for presenting images to the user. Figure 1 shows a snapshot of our prototype simulation system. The rest of the paper is organized as follows: section 2 reviews the related work on physical simulation of the task, section 3 describes the principal phases of the intervention. Furthermore, section 4 gives an overview of our simulation technique, section 5 presents the geometry and topology description, and section 6 describes the physical simulation system. Finally, section 7 provides a brief description of the architecture of the prototype simulator, with some details on the other principal tasks (corneal incision and capsulorhexis). We conclude with a discussion of the results obtained so far and a view of our future work.

2. Related work

Physically based deformable models have been widely embraced by the Computer Graphics community, and extensively applied to the simulation of organs and tissues for virtual reality based surgical training systems. For an extensive review of the most significant contributions of the past decade, we refer readers to [NMK*05, CCI*05] for deformable models in general, and to [VBB*04] for the application to medical virtual environments. In the following sections, we briefly discuss the approaches most closely related to ours. Broadly speaking, the techniques fall into two main categories: mesh–based and mesh–free methods.

Mesh–Based Methods. The most popular and powerful mesh–based approach for animating deformable and rigid objects is the Finite Element Method (FEM), normally used to solve PDE's on irregular grids. For simulating deformable objects, an *explicit* formulation of FEM is commonly employed, where masses and internal and external forces are lumped to the vertices [MDM*02, DDCB01]. Many studies successfully applied FEM to surgery simulation, as demonstrated in [CDA00, PDA01]. By the way, no matter how accurate, FEM–based approaches are not usable when dealing with many interacting objects, such as in the phaco–emulsification task. Another very popular class of mesh–based deformable models is composed by mass-spring systems, consisting of point masses connected together by a network of massless springs. The most popular dynamic simulation system is based on energetic considerations, in which deformation energies are derived from soft constraints and are to be maintained by the model during the evolution [BW98]. These approaches are well suited for off-line animation of surface–based models, but they cannot be applied to real–time simulation of volumetric objects, such as the crystalline lens, due to the high computation power required.

Mesh–Free Methods. Mesh–free methods treat objects regardless of their topology. In this category, Smoothed Particle Hydrodynamics (SPH) is a general technique originally proposed for astrophysical simulations, providing an interpolation method for particle systems, in which field quantities that are only defined at discrete particle locations can be evaluated anywhere in space. They are applied to the simulation of fluids [MSKG05, MCG03], such us blood [MST04], but they can be employed for simulating highly-deformable bodies [DG96] New trends in animation systems consider the combination of mesh–free physics with point-sampled surfaces in point-based animation. In this context, the most important contributions were provided by Muller et al [MKN*04], who introduced a mesh–free continuum-mechanics-based approach for the animation of elastic, plastic and melting objects. Their technique was modified by Pauly and others in order to cope with the fracture of a wide range of materials, from stiff elastic to highly plastic objects that exhibit brittle and/or ductile fracture [PKA*05]. These techniques are exclusively targeted for off-line precise and accurate animation systems. Finally, Muller et al proposed a real–time mesh-less method where nodes of a volumetric mesh are treated as point masses, and the original configuration of the points (rest configuration) is fitted to the deformed configuration, in the least squares sense, using shape matching techniques for point clouds [MHTG05]. Our approach for simulating the phaco–emulsification integrates an hybrid mesh–free method [MHTG05], with a topology description, in order to handle rendering and the continuous modifications involved by surgical instruments. It also employs an SPH scheme for handling fragment interactions. Our main contribution is a shape matching algorithm, originally derived for photogrammetry tasks, which provides a closed form solution to the absolute orientation problem [Hor87].

3. Application area

Phacoemulsification consists in breaking the hardened crystalline lens into very minute fragments by employing local-

ized high frequency waves (figure 2 shows a complete eye diagram indicating the lens position and shape). The lens fragments are then easily removed out with a small aspirator. Both the ultrasound generator and the aspirator are combined into a single thin instrument: the phacoemulsificator. The most common way of fracturing the lens nucleus consists in sculpting a ditch in a 6 o'clock direction and repeating the same operation after a 90 degree rotation, in order to obtain four arms of a shallow deep cross.

In order to open the way to the phacoemulsificator, first of all a small incision (the right port or tunnel) of about 3 mm, has to be performed on the cornea at about 5 o' clock with a lance knife. The corneal tunnel section should be Z-shaped in order to limit the outgoing liquid flow and maintain the internal pressure. Subsequently, a capsulorhexis is performed in order to uncover the upper surface of the crystalline. The capsulorhexis procedure consists in removing the anterior capsule, by first creating a small flap (a L-shaped incision) in the central area of the capsule with a hook, then engaging the flap by forceps and pulling it in a circular manner to create a regular round opening on the membrane. A second tunnel (the left port) is often created on the cornea at about 8 o' clock. It is used to perform auxiliary tasks, such as controlling eye globe movements or holding and moving the crystalline lens during the phacoemulsification stage. It is also used to introduce an irrigator attached to a balanced salt solution in a bottle placed 90-95 cm above eye level, necessary to stabilize the anterior chamber pressure during all intervention phases.

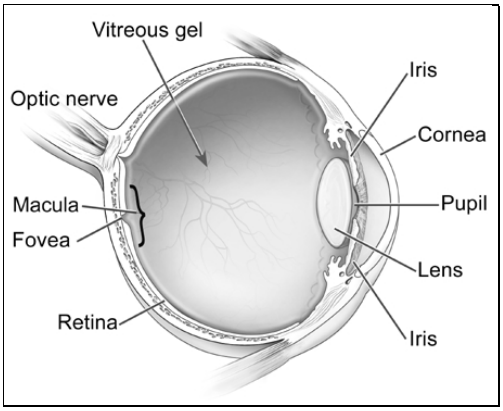


Figure 2: *Complete eye diagram with labels. Courtesy of U.S. National Eye Institute.*

4. Overview of the Phacoemulsification simulation

For the phacoemulsification task, we model the eye lens as a collection of simplices, built from a tetrahedron mesh, with mass particles placed in tetrahedrons barycenters. Links connecting particles are maintained in order to provide geometrical information for deriving the external triangular mesh for rendering, and for recognizing crystalline lens independent fragments when phacoemulsification is performed.

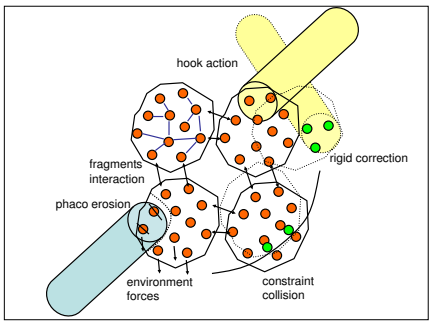


Figure 3: *A drawing indicating principal components of the phaco–emulsification simulation system.*

The simulation is performed by applying the following evolution steps to the particles composing the crystalline lens, as shown in figure 3.

- applying environment forces to particles (gravity and viscosity);
- eroding crystalline lens, by removing particles in the zone of influence of phaco, updating the topology, and recognizing independent fragments;
- detecting collisions between particles and boundaries (lens capsula and cornea);
- detecting collisions between particles and tools (hook and phaco–emulsificator);
- detecting collisions between independent fragments;
- restoring the original shapes of the fragments by employing a quasi–rigid correction.

Phacoemulsification simulation relies on a number of parameters, that are adequately tuned in order to get realistic effects. In the following sections we detail the geometric description, as well as the physical simulation approach.

5. Geometric description

The crystalline lens nucleus has approximately the shape of a disk, and it is wrapped by the anterior camera. It can be modeled as two half-ellipsoids having the same height and weight but different height (as in figure 2). Our geometric description is a collection of **simplexes**, representing the Voronoi cells of a given tetrahedron mesh model of the crystalline lens [DWS93]. This representation has been used for 2D surfaces, and it has the advantage of providing a compact description, in which each element has a fixed number of connections. Simplex meshes have also a compact and unambiguous shape description, but some care has to be taken to manage 3D simplexes modifications, because of some particular cases. A simplex is a data structure containing the following topology and geometry information:

- the simplex center geometric position;
- an array containing the neighbor simplices identifiers, i.e. the simplices internally sharing a face with the current simplex;
- an array containing the boundary simplex information; specifically, for each external face, a boundary simplex representation is an array containing the simplices identifiers and face, sharing an edge with the current face;
- optionally, for consistency checks, the original tetrahedron description
- optionally, for consistency checks, an array containing the external face centers, used for the triangulation task.

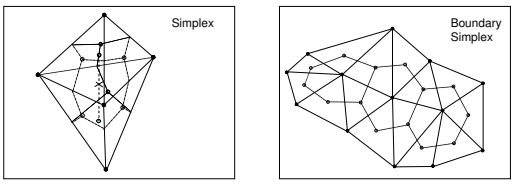


Figure 4: *Simplex geometrical representation and boundary simplex representation.*

Figure 4 demonstrates the simplex structure in a very simple object composed by two tetrahedrons sharing a face, and an example of the boundary simplex representation. According to this representation, each simplex can have a maximum of four neighbors, and in this case, it can be considered internal, and a minimum of zero neighbors, and in this case it is disconnected. When the neighbors are less then four, the simplex is considered external. This representation is very efficient for rendering, since internal simplexes do not provide useful geometry information, and the external surface reconstruction can be obtained by traversing the external boundary simplex connections. Furthermore, the main motivation leading to this kind of description is related to the nature of phacoemulsification task, in which the tool erodes the crystalline lens and breaks it into a number of small fragments. The simplex description is very efficient for handling such a removal primitive, which rapidly updates the geometry and topology, and the correspondent surface triangulation.

5.1. Simplex construction

The simplex construction is obtained by a tetrahedron mesh description, by employing three loop steps:

- a loop over input tetrahedrons, in order to compute center of mass and find shared faces;
- a loop over the shared faces in order to build the simplex internal connections;
- a loop over the non-shared faces, in order to find the boundary simplexes shared edges, and build the simplex external connections.

5.2. Simplex triangulation

The surface reconstruction of a simplex organized model is derived by traversing the boundary simplexes, i.e. the external connections of simplexes, according to the following procedure:

- a loop over the simplexes in order to find the boundary simplex centers and associate a surface node to them;
- for each boundary simplex, loop recursively counterclockwise over its edges in order to get the triangles containing the surface node associated to that boundary simplex;
- a loop over triangles, in order to compute normals.

5.3. Simplex removal primitive

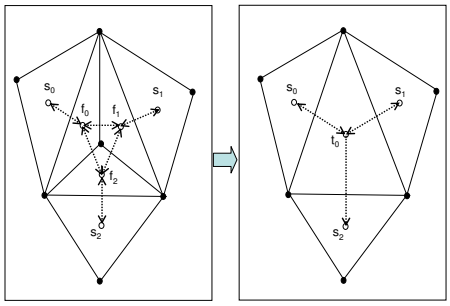


Figure 5: *Boundary surface update involved by the removal of a simplex with three external faces.*

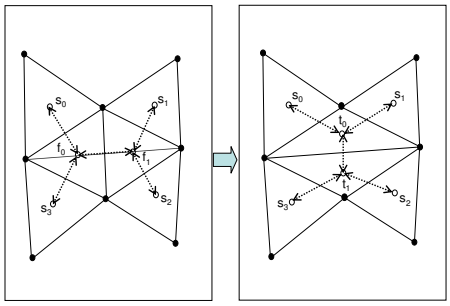


Figure 6: *Boundary surface update involved by the removal of a simplex with two external faces.*

The removal action consists in erasing a single simplex in the structure and updating the topology in order to recover consistency, and to ensure a correct surface reconstruction. The topology update involves the boundary simplices and the internal connections. When the external update is concerned, the procedure is different according to the number of external faces, leading to three possible cases:

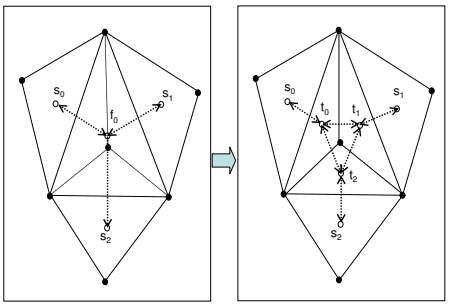
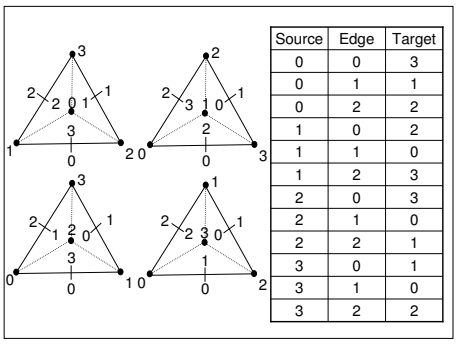


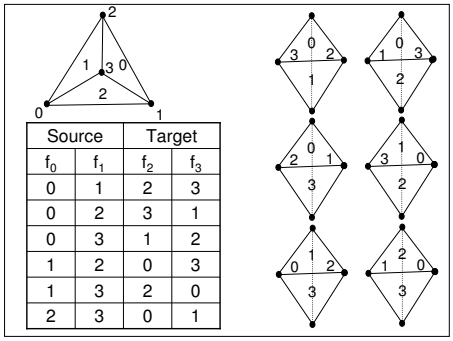
Figure 7: *Boundary surface update involved by the removal of a simplex with one external faces.*



Source	Edge	Target
0	0	3
0	1	1
0	2	2
1	0	2
1	1	0
1	2	3
2	0	3
2	1	0
2	2	1
3	0	1
3	1	0
3	2	2

Figure 8: *Look-up table for finding target faces when removing a simplex with just one external face.*

- **simplex with three faces external**: as shown in figure 5, the boundary surface topology update consists in the following operations:
  - recognizing the target face (the only internal face of the simplex) and the target simplex (t_0 in figure);
  - attaching the source (s_0, s_1, s_2 in figure) to the new target;
  - detaching the source faces (s_0, s_1, s_2 in figure 5) from the old external faces (f_0, f_1, f_2 in figure 5);
- **simplex with two faces external**: as shown in figure 6, the boundary surface topology update consists in the following operations:
  - recognizing the target faces (by using the look-up table in figure 9) and the target simplexes (t_0 and t_1 in figure 6);
  - attaching the target t_0 to the sources s_0 and s_1, the target t_1 to the sources s_2 and s_3, and the target t_0 to the target t_1;
  - detaching the old external face f_0 from sources s_0 and s_3, and the old external face f_1 from the old external faces from sources s_1 and s_2;



Source		Target	
f_0	f_1	f_2	f_3
0	1	2	3
0	2	3	1
0	3	1	2
1	2	0	3
1	3	2	0
2	3	0	1

Figure 9: *Look-up table for finding target faces when removing a simplex with two external faces.*

- **simplex with one face external**: as shown in figure 7, the boundary surface topology update consists in the following operations:
  - recognizing target faces (by using the look-up table in figure 8) and the target simplexes (t_0, t_1 and t_2 in figure);
  - attaching the source s_0 to the target t_0, the source s_1 to the target t_1, and the source s_2 to the target t_2;
  - attaching the target t_0 to the target t_1, the target t_1 to the target t_2, and the target t_2 to the target t_0;
  - detaching the old external face f_0 from the source faces s_0, s_1, and s_2.

The internal connections update simply consists of querying the internal neighbor simplexes of the "delendum" simplex and by removing all connection links. The internal connections deleted are substituted by external connections.

6. Physical simulation

The crystalline lens physical simulation involves a quasi–rigid motion, according to the geometrical constraints and the tools interactions (hook, phaco–emulsificator and aspirator). The lens is simulated as a collection of simplex cells, and can be broken in many small independent and interacting fragments. Therefore the physical simulation can be subdivided into the following steps:

- **environment interaction**: gravity and viscosity are applied to cells by a first-order Euler integration scheme;
- **environment constraints**: the cornea, the anterior camera and the posterior camera are geometrically modeled as half–spheres in order to compute simple point-sphere collisions and to constrain the cells positions;
- **tools interactions**: the phaco–emulsificator can remove simplexes falling inside an influence cone with a distance threshold according to a Russian roulette scheme, an aspirator attracts simplexes falling in an influence sphere, while the hook drags away simplexes falling inside an influence cylinder;

- **fragments interactions**: the interaction of the fragments created by the action of phaco–emulsificator is handled by employing a Smoothed Particle Hydrodynamics scheme in a spatial ordering description based on Morton numbers. The fragments are recognized by a recursive painting algorithm that traverses all simplexes internal connections;
- **shape matching correction**: the fragments undeformed corrections are restored by applying a quasi–static shape matching correction algorithm based on [MHTG05]. The technique is improved and further stabilized by solving the absolute orientation problem in a closed form [Hor87]. In this way, the simulation is able to handle the motion of fragments composed of very few simplexes (less than 4), and fragments in degenerated configurations (coplanar points).

6.1. Collision detection

The fragments collision detection is handled by employing the following approach:

- a SPH scheme employing a 6th order smoothing kernel in a support h, as described in [MCG03]
- spatial ordering based upon Morton curves in order to reduce the computational complexity. Grid cells are chosen to be of size $2h$.

The environmental collision detection consists in treating geometric intersections and inelastic collision: the interacting cells are force to assume a very big mass, and the shape-based mesh–less correction scheme adjusts the configuration according to the constraints.

6.2. Quasi static shape-matching correction

Particles evolution schemes applied to the simulation of the crystalline do not consider the original rigid shape, but they lead to a deformed configuration which is far removed from the original shape. A way to solve this problem is to apply a correction scheme based on a shape–matching approach, as proposed by Mueller et al. [MHTG05]. Given a mesh-less description, in which a deformable object is considered as a point system, the shape–matching technique tries to find the rigid body transformation that maps the configuration as in the undeformed state to the configuration in the deformed state. Since in our physical simulation system the crystalline lens is fractured in many very small fragments, which can be composed of one or very few cells, the approach described in [MHTG05] cannot be directly applied, but it needs slight modifications. Furthermore, in [MHTG05] they solve the problem by finding the best-fit linear transformation between the two configurations, and then by finding the nearest orthonormal matrix. It has been demonstrated that this approach involves more work and does not produce the solution to the original problem [Hor88]. We solve these problems by employing the closed–form solution described in [Hor87]. In fact, the problem of finding a rigid body map between two coordinate systems is a classic photogrammetry task, called **absolute orientation**. To our knowledge, that technique has not yet been applied in the context of dynamic simulation. The approach consists in solving a least square error problem, where unit quaternions are used to represent rotations. The solution of the problem is proven to be the eigenvector of a symmetric 4X4 matrix associated with the most positive eigenvalue. In the following subsections we detail the technique.

6.2.1. Problem definition

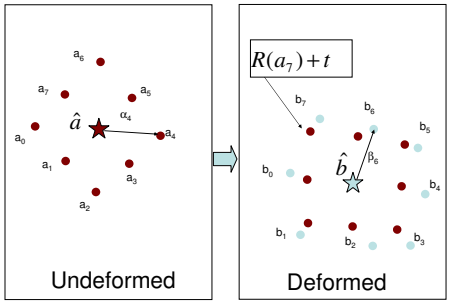


Figure 10: *Point-based deformable object passing from an undeformed to a deformed configuration.*

Let us consider a deformable object described as a collection of n point masses m_i. During a dynamic simulation, this object starts from an undeformed configuration represented by positions $\mathbf{a_i}$, and reaches a deformed state represented by $\mathbf{b_i}$, due to external forces, constraints and tool interaction (as in figure 10). The **absolute orientation** problem consists in finding the optimal rigid body map between the two representations, such that

$$\mathbf{b} = R(\mathbf{a}) + \mathbf{t} \tag{1}$$

where R is a rotation mapping and $\mathbf{t}$ is a translational offset. The transformation is optimal in the sense that it is the solution to the following MLS problem: given a residual error

$$\mathbf{e_i} = \mathbf{b_i} - R(\mathbf{a_i}) - \mathbf{t}$$

we want to find R and $\mathbf{t}$ such that minimizes the functional

$$\Psi = \sum_{i=0}^{n} \|\mathbf{e_i}\|^2$$

6.2.2. Translational offset

Given the center of mass of the deformable object, in the undeformed and deformed state,

$$\hat{\mathbf{a}} = \frac{\sum_i m_i \mathbf{a_i}}{\sum_i m_i} \quad \hat{\mathbf{b}} = \frac{\sum_i m_i \mathbf{b_i}}{\sum_i m_i}$$

we can derive the point offsets in respect of center of mass,

$$\alpha_\mathbf{i} = \mathbf{a_i} - \hat{\mathbf{a}} \quad \beta_\mathbf{i} = \mathbf{b_i} - \hat{\mathbf{b}}.$$

With these definitions, the residual error can be expressed as

$$\mathbf{e_i} = \beta_\mathbf{i} - R(\alpha_\mathbf{i}) - \tau$$

where

$$\tau = \mathbf{t} - \hat{\mathbf{b}} + R(\hat{\mathbf{a}}).$$

Now

$$\begin{aligned} \sum_i \|\mathbf{e_i}\|^2 &= \sum_i \|\beta_\mathbf{i} - R(\alpha_\mathbf{i}) - \tau\|^2 \\ &= \sum_i \|\beta_\mathbf{i} - R(\alpha_\mathbf{i})\|^2 - 2\tau\sum_i(\beta_\mathbf{i} - R(\alpha_\mathbf{i})) + n\|\tau\|^2. \end{aligned}$$

where the middle term is zero, since $\sum_i \beta_\mathbf{i} = \sum_i \alpha_\mathbf{i} = \mathbf{0}$.

So, the sum of square residual errors is the minimum, when $\tau = 0$, and

$$\mathbf{t} = \hat{\mathbf{b}} - R(\hat{\mathbf{a}}) \tag{2}$$

is the optimal translation offset.

6.2.3. Rotation

If the optimal translation offset is obtained with $\tau = 0$, then, the sum of square errors becomes

$$\begin{aligned} \sum_i \|\mathbf{e_i}\|^2 &= \sum_i \|\beta_\mathbf{i} - R(\alpha_\mathbf{i})\|^2 \\ &= \sum_i \|\beta_\mathbf{i}\|^2 + \sum_i \|R(\alpha_\mathbf{i})\|^2 - 2\sum_i \beta_\mathbf{i} R(\alpha_\mathbf{i}) \\ &= \sum_i (\alpha_\mathbf{i}{}^2 + \beta_\mathbf{i}{}^2) - 2\sum_i \beta_\mathbf{i} R(\alpha_\mathbf{i}). \end{aligned}$$

Hence, in order to minimize the sum of square errors, R has to maximize the quantity $\sum_i \beta_i R(\alpha_i)$. Now, the rotation R can be represented by means of a unit quaternion $\tilde{q}$, so

$$\sum_i \beta_i R(\alpha_i) = \sum_i \tilde{\mathbf{q}} \alpha_\mathbf{i} \tilde{\mathbf{q}}^* \beta_\mathbf{i}$$

and the product can be expressed as a matrix product, leading to

$$\sum_i \beta_i R(\alpha_i) = \tilde{\mathbf{q}}^T N \tilde{\mathbf{q}}$$

where $N = \sum_i N_i$ and $N_i = \sum_i A_i^T B_i$, with

$$A_i = \begin{bmatrix} 0 & -\alpha_{x,i} & -\alpha_{y,i} & -\alpha_{z,i} \\ \alpha_{x,i} & 0 & \alpha_{z,i} & -\alpha_{y,i} \\ \alpha_{y,i} & -\alpha_{z,i} & 0 & \alpha_{x,i} \\ \alpha_{z,i} & \alpha_{y,i} & -\alpha_{x,i} & 0 \end{bmatrix}$$

and

$$B_i = \begin{bmatrix} 0 & -\beta_{x,i} & -\beta_{y,i} & -\beta_{z,i} \\ \beta_{x,i} & 0 & -\beta_{z,i} & \beta_{y,i} \\ \beta_{y,i} & \beta_{z,i} & 0 & -\beta_{x,i} \\ \beta_{z,i} & -\beta_{y,i} & \beta_{x,i} & 0 \end{bmatrix}.$$

Furthermore, given a quadratic form $\Phi = \tilde{\mathbf{q}} N \tilde{\mathbf{q}}$, where N is a symmetric 4x4 real matrix, the unit quaternion maximizing Φ is the eigenvector associated to the most positive eigenvalue of the matrix N. In fact, if N is a symmetric matrix, it has four real eigenvalues $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ with corresponding orthogonal unit eigenvectors $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}, \mathbf{e_4}$ such that

$$N\mathbf{e_i} = \lambda_i \mathbf{e_i} \quad with \quad i = 1,2,3,4. \tag{3}$$

The set of unit eigenvectors span the four dimensional space, so an arbitrary quaternion $\tilde{\mathbf{q}}$ can be written as a linear combination of the form $\tilde{\mathbf{q}} = \sum_{i=1}^4 \rho_i \mathbf{e_i}$. Hence, $\tilde{\mathbf{q}}^T \tilde{\mathbf{q}} = \sum_i \rho_i^2 = 1$, and $N\tilde{\mathbf{q}} = \sum_i \rho_i \lambda_i \mathbf{e_i}$, so we can write $\tilde{\mathbf{q}}^T N \tilde{\mathbf{q}} = \tilde{\mathbf{q}}^T (N\tilde{\mathbf{q}}) = \sum_i \lambda_i \rho_i^2$. If we order the eigenvalues with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$, then

$$\tilde{\mathbf{q}}^T N \tilde{\mathbf{q}} \leq \lambda_1 \sum_i \rho_i^2 = \lambda_1 \tag{4}$$

so the maximum value of the quadratic form cannot be bigger than the maximum eigenvalue, and this value of the quadratic form can be obtained by using $\rho_1 = 1$ and $\rho_2 = \rho_3 = \rho_4 = 0$, so $\tilde{\mathbf{q}} = \mathbf{e_1}$ is the solution of the maximization problem. This formulation leads to a full closed solution, since eigenvalues can be found by solving an algebraic quartic equation, and it works for degenerated configurations (coplanar points). No approximation is involved, and it works also with objects composed of just three points.

7. Cataract surgery system: architecture and components

The phaco–emulsification simulation approach is integrated into a real–time virtual reality training system for cataract surgery. In the following subsections we briefly describe the architecture of the system, and the other simulation components.

Decoupled simulation. In order to provide realistic images to the user, the system needs to simulate very different interactions. This difference is exploited by modeling the training simulator as a collection of loosely coupled concurrent components. Logically, the system is divided in a fast subsystem, responsible for the surgical instrument tracking (100 Hz), and a slower one, essentially dedicated to the production of data for visual feedback. The system runs on two multiprocessor machines connected with a 100 Mbit Ethernet link. The first machine is dedicated to the high-frequency devices tracking tasks (100 Hz), while the second machine concurrently runs the low-frequency task (20-25 Hz): eye simulation, anterior camera simulation, and crystalline lens simulation.

Tool-eye interaction simulation. Since force feedback returned by surgical instruments is nearly imperceptible, the

main way for the surgeon to understand the effort really exerted on the patient eye is the visual feedback provided by the tool position in relation to the environment and the eye globe displacement. In order to evaluate tool-eye interaction, we use a two-step algorithm. In the first step, a simple conjugate gradient descent is employed to minimize the total deformation energy of the following constraints: the corneal tunnels (both translational and rotational constraint), the internal cornea surface, and the external crystalline surface (unidirectional radial interaction) which mark the boundaries of the tool working area. The result is an equilibrium position, to be visualized during simulation. In the second step, the eye globe is rotated in order to reduce the deformation of the tunnel. The new globe position is thus the resultant of the force applied to the tunnel borders by the tool and the globe muscles reaction. All constraints are simply modeled as linear springs.

Corneal tunnel simulation. At the moment the cutting feature is intended as a geometric tool to select the corneal insertion point and the port geometric features, such us orientation and width. Tunnel section profile control is currently not implemented. The cut is modeled as a circular arc and stored as its extrema radial versors. During the cutting stage, the extrema are constantly recomputed, in order to perform the cut enlargement and its partial reorientation.

Capsulorhexis simulation. For the capsulorhexis procedure, we geometrically model the anterior camera as a mesh composed of triangular facets. This model is used for the physical simulation as well as the rendering stage. The physical simulation is obtained by mapping a mass-spring network over the triangular mesh [WSM02b, HPH96], where mass particles are mapped over the mesh vertices, and linear springs are mapped over the mesh edges. Tearing is obtained by breaking the spring exerting the largest force on a vertex if it is overextended, and by propagating cuts along the most stressed directions. All these physical effects rely on a number of parameters, which need to be adequately tuned in order to get a membrane physically similar to a real anterior camera.

8. Results

The prototype of the cataract extraction training simulation system was developed in C++ on top of the OpenSceneGraph toolkit [ope05]. A complete and detailed eye model was built according to the indications of [LCRB03]. Our current configuration is the following:

- a single-processor PIV/1.5 GHz for the high-frequency device tracking task; two threads run in parallel: one for the tracking loop (100 Hz), and one for sending instruments position updates to the other machine;
- a dual-processor PIV/2.2 GHz with 2 GB RAM and a NVIDIA GeForce 6800, and running a 4.4 linux kernel, for the low frequency tasks (25 Hz). Two threads are continuously running on this machine: one to receive position updates, one for simulations and visual rendering;
- a Phantom Desktop haptic device for the dominant hand; the device is connected to the single processor PC. It provides 6DOF tracking;
- a Phantom 1.0 haptic device for the non-dominant hand; the device is connected to the single processor PC. It provides 6DOF tracking;
- a N-vision VB30 binocular display for presenting images to the user. The VB-30 contains a small high-resolution LCD display and is connected to the S-VGA output of the dual processor PC.

A number of training sessions have been performed, and recorded on video. The performance of the prototype is sufficient to meet the timing constraints for display, even though the computational and visualization platform is made only of affordable and accessible components. Early impressions by professional ophtalmologists are positive. Subjective input is currently being used to tune the parameters that control capsulorhexis and phacoemulsification simulations. Figure 11 shows a sequence from a virtual phacoemulsification, where the cross-shaped incision is clearly visible. Our approach is able to perform a real–time simulation of a virtual phaco–emulsification on a machine equipped with a dual-processor PIV/2.2 GHz with 2 GB RAM and a NVIDIA GeForce 6800 with a frame rate of 25 fps on a tetrahedron model composed by 2100 cells. Since a crystalline lens has an approximate volume of $60mm^3$, a cell is about $0.003mm^3$, and the resolution can be considered adequate for the physical simulation.

9. Conclusions and future work

We have described an approach for the simulation of the phaco–emulsification task in a virtual reality system for cataract surgery training. The main contribution of this work is a real-time scheme that combines the following methods:

- a mesh-less shape–based dynamic simulation algorithm based on a photogrammetry technique originally derived for solving the absolute orientation problem;
- a simplex geometry representation in order to efficiently handle rendering and the continuous modifications involved by the surgical tool interaction;
- a smoothed particle hydrodynamics based scheme with spatial ordering for handling fragments interactions.

An implementation of this method is integrated in a simulator running on a multiprocessing PC platform. The methods employed were proved able to sustain interactive rates. Current work is concentrating on validating the system by mean of professional surgeons subjective input.

Acknowledgments. We acknowledge the NIH National Research Resource in Molecular Graphics and Microscopy at the University of North Carolina at Chapel Hill for the VRPN library. We also aknowledge Annalisa Brughitta for her precious technical support.

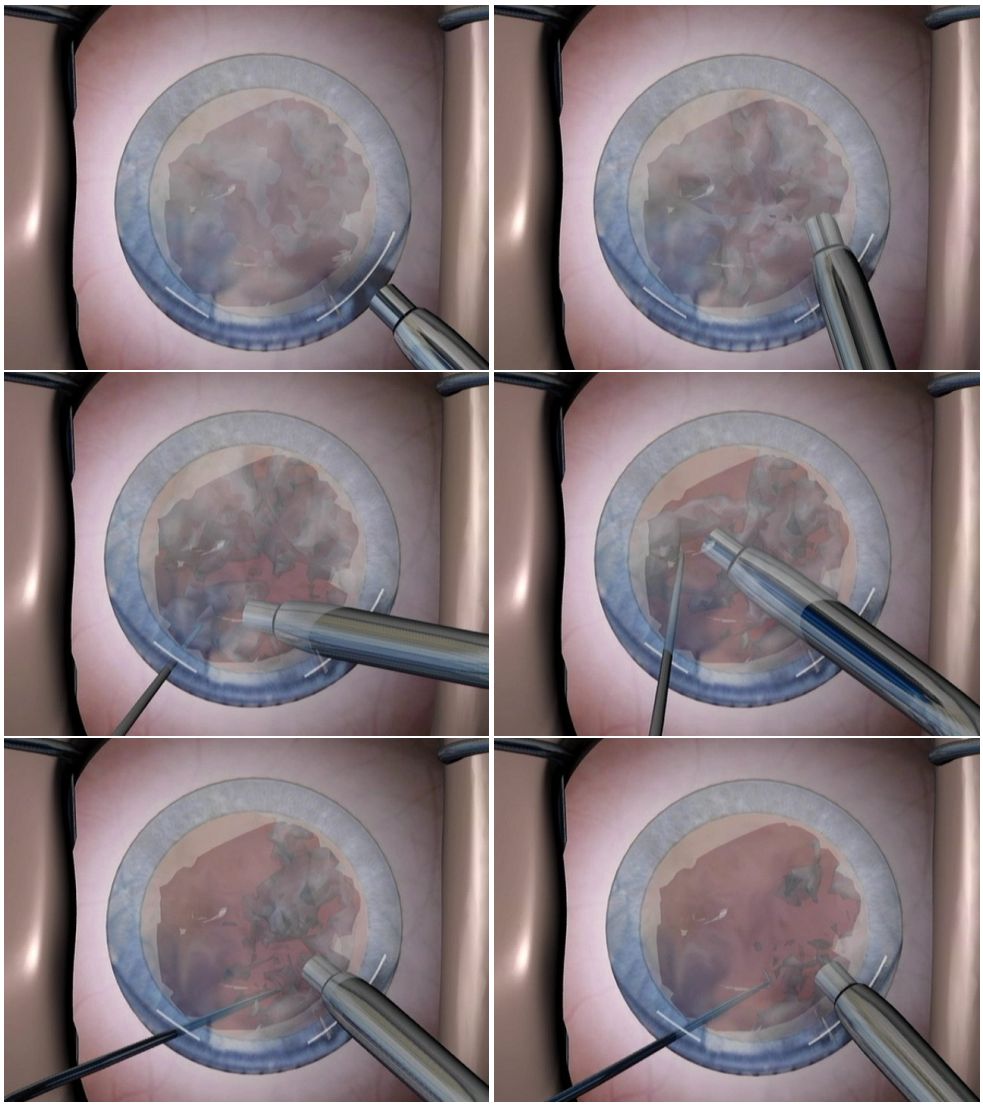


Figure 11: *A virtual phacoemulsification sequence. Here we show a typical lens rupture and extraction performed by a phacoemulsificator tool. The typical cross-shaped cut is clearly visible.*

References

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1998), ACM Press, pp. 43–54. 92

[CCI*05] CHEN M., CORREA C., ISLAM S., JONES M. W., SHEN P.-Y., SILVER D., WALTON S. J., WILLIS P.: Deforming And Animating Discretely Sampled Object Representations. Chrysanthou Y., Magnor M., (Eds.), Eurographics Association, pp. 113–140. Eurographics 05 Star. 92

[CDA00] COTIN S., DELINGETTE H., AYACHE N.: A hybrid elastic model allowing real–time cutting, deformations and force–feedback for surgery training and simulation. *The Visual Computer 16*, 8 (2000), 437–452. 92

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic real-time deformations using space & time adaptive sampling. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), ACM Press, pp. 31–36. 92

[DG96] DESBRUN M., GASCUEL M.-P.: Smoothed particles: a new paradigm for animating highly deformable

bodies. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96* (New York, NY, USA, 1996), Springer-Verlag New York, Inc., pp. 61–76. 92

[DWS93] DELINGETTE H., WATANABE Y., SUENAGA Y.: Simplex based animation, 1993. 93

[Hor87] HORN B.: Closed–form solution of absolute orientation using unit quaternions. *Journal of Optical Society of America 4* (April 1987), 629–642. 92, 96

[Hor88] HORN B.: Closed–form solution of absolute orientation using orthonormal matrices. *Journal of Optical Society of America 4* (July 1988), 1127–1136. 96

[HPH96] HUTCHINSON D., PRESTON M., HEWITT T.: Adaptive refinement for mass/spring simulations. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96* (New York, NY, USA, 1996), Springer-Verlag New York, Inc., pp. 31–45. 98

[LCRB03] LEFOHN A., CARUSO R., REINHARD E., BADGE B.: An ocularistś approach to human iris synthesis. *IEEE Computer Graphics and Applications 23*, 6 (November-December 2003). 98

[LNS*01] LAURELL C., NORDH L., SKARMAN E., ANDERSSON M., NORDQVIST P.: Computer-simulated phacoemulsification. In *SPIE Proceedings* (2001), pp. 174–176. 91

[MCG03] MULLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 154–159. 92, 96

[MDM*02] MULLER M., DORSEY J., MCMILLAN L., JAGNOW R., CUTLER B.: Stable real-time deformations. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2002), ACM Press, pp. 49–54. 92

[MHTG05] MULLER M., HEIDELBERGER B., TESCHNER M., GROSS M.: Meshless deformations based on shape matching. *ACM Trans. Graph. 24*, 3 (2005), 471–478. 92, 96

[MKN*04] MULLER M., KEISER R., NEALEN A., PAULY M., GROSS M., ALEXA M.: Point based animation of elastic, plastic and melting objects. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2004), ACM Press, pp. 141–151. 92

[MSKG05] MULLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM Press, pp. 237–244. 92

[MST04] MULLER M., SCHIRM S., TESCHNER M.: Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics. *Technol. Health Care 12*, 1 (2004), 25–31. 92

[NMK*05] NEALEN A., MULLER M., KEISER R., BOXERMAN E., CARLSON M.: Physically based deformable models in computer graphics. In *Eurographics 2005 STAR - State of The Art Report* (2005), pp. 71–94. 92

[ope05] Openscenegraph, 2005. Available from http://www.openscenegraph.org. 98

[PDA01] PINCINBONO G., DELINGETTE H., AYACHE N.: Non–linear and anisotropic elastic soft tissue models for medical simulation. In *Proceedings of the IEEE International Conference on Robotics and Automation* (2001). 92

[PKA*05] PAULY M., KEISER R., ADAMS B., DUTRE P., GROSS M., GUIBAS L. J.: Meshless animation of fracturing solids. *ACM Trans. Graph. 24*, 3 (2005), 957–964. 92

[SLA*02] SODERBERG P. G., LAURELL C., ARTZÉN D., NORDH L., SKARMAN. E., ANDERSSON M., NORDQVIST P.: Computer-simulated phacoemulsification, improvements. In *SPIE Proceedings* (2002), pp. 76–80. 91

[VBB*04] VIDAL F. P., BELLO F., BRODLIE K., JOHN N., GOULD D., PHILLIPS R., AVIS N.: Medical Visualization And Virtual Environments. Schlick C., Purgathofer W., (Eds.), Eurographics Association, pp. 1–35. Eurographics 04 Star. 92

[WSM02a] WAGNER C., SCHILL M., MANNER R.: Intraocular surgery on a virtual eye. *Communicactions of the ACM 45*, 7 (2002), 45–49. 91

[WSM02b] WAGNER C., SCHILL M. A., MANNER R.: Collision detection and tissue modeling in a vr-simulator for eye surgery. In *EGVE '02: Proceedings of the workshop on Virtual environments 2002* (Aire-la-Ville, Switzerland, Switzerland, 2002), Eurographics Association, pp. 27–36. 98