A GPU-friendly Method for Haptic and Graphic Rendering of Deformable Objects

M. de Pascale, G. de Pascale, D. Prattichizzo, and F. Barbagli

Siena Robotics and Systems Laboratory, University of Siena, Siena, 53100, Italy
[mdepascale, gdepascale, prattichizzo, barbagli]@dii.unisi.it

Abstract. In this paper we present a new method for real-time interactive haptic and graphic rendering of complex objects locally deformed by multiple contacts. Core algorithms have been designed to be executable also on videoboard's GPU, thus taking advantage of parallel matrix and vector computational power. Although complex physical simulation has been simplified to run on GPUs, results are characterized by high visio-tactile realism perceived by users. Graphical rendering algorithms can be easily added to pre-existing vertex shaders/programs. The proposed method makes use of common triangular meshes, thus making the method a good choice when adding haptic feedback to existing graphical applications.

1 Introduction

One of the most valuable benefits offered by the increasing computational power of PCs is the possibility to improve virtual simulations of real environments through the haptic rendering, that use force feedback-enabled devices to simulate touch[1, 2]. Realistic real-time haptic rendering is a heavy-computation task, and the required computational power is usually taken away from graphics. The haptic rendering load is mainly due to the use of complex mathematical models, needed to accurately simulate physical behaviour, instead of simple approximations that can result satisfactory for 3D rendering. In this paper we present a new method able to realistically simulate visio haptic interaction of locally deformable objects, even on mainstream PCs. Two are the main ideas at the base of the proposed method:

- trading off between exact physical simulation and realistic user perception
- using modern GPUs parallel and vector computational power[3, 4]

The proposed model simulates deformable objects as a set of uncoupled elements, thus avoiding any possible stability issues due to element interaction. Moreover it uses a highly customizable non-linear dynamic local-model able to simulate also objects with different compliance and dynamic behaviours. Such model is perceived by the user as more realistic than the simple elastic model. The absence of links between the dynamic elements modeling the object is compensated by the use of *force fields* to propagate deformations from contact points to the

object surface. Deformations are fully customizable, both geometrically than dynamically. The proposed approach is also well suited to handle more than one contact point. A relevant attention is devoted to the computational aspects. At each cycle, only a small stream of data is sent down to the video board instead of the whole deformed mesh. This saves bandwidth and allows for very complex meshes to be used[5]. Moreover any existing graphic shader[1] code can be used, by simply adding few instructions. All these features allow running of real-time visio/haptic simulations with a high level of realism also on commonly diffused PCs.

2 Related Works

At the present the most diffused methods for visio-haptic simulation of deformable objects are springs-masses systems and the many finite element based variations such as FEM[6], BEM[7], LEM[8] and REM[9]. The two main approaches used for such simulations are dynamic and quasi-static. In the dynamic approach the simulation is performed by real-time integration of the motion differential equations. In the quasi-static approach, instead, the equilibrium points of such equations are computed. In the springs-masses systems objects are modelled as a set of point like masses, linked together with springs and dampers. In the finite-elements based methods the entities are modeled as a finite set of elements interacting each other while preserving physical invariants such as masses, energy, volume and others. Differences between various implementations of these methods consist in the shape of elements and relations between them. Springs-masses systems are approached dynamically, since the corresponding equations are easily integrable. Unfortunately they may present stability problems, mainly due to the choice of visco-elastic parameters and of step used for time discretization. On the other side, finite elements systems are approached in a quasi-static setting, due to the computational complexity of the resulting differential equations. Luckily enough finite elements method can be partially precomputed or used for off-line computation of elementary deformations which can than be used at runtime[10].
Another issue in visio-haptic rendering is the mathematical representation of deformable objects surfaces. Meshes and implicit surfaces are the main methods. Mesh-based representations present the great advantage of one-to-one correspondence with the elements of the physical model, thus the triangular mesh needed by 3D graphical adapters is easily built from elements data. On the other hand, thanks to collision detection and haptic feedback easy computation, implicit surfaces are one of the preferred methods when dealing with haptic rendering only. Unfortunately recomputing the triangular mesh for visualization from an implicit surface can be a heavy task.

[1] We use the convention of calling the code executable by the GPU Vertex Shaders and Pixel Shaders. It's quite diffused also another nomenclature calling them Vertex Programs and Fragment Programs (as in OpenGL).