Extrapolation: a solution for force feedback ?

Guillaume Picinbono[†] Jean-Christophe Lombardo[‡]

EPIDAURE / INRIA
2004, route des Lucioles, B.P. 93
06902 Sophia Antipolis, Cedex - France

Abstract

Using a force feedback device within the framework of deformable object simulation remains difficult because of the incompatibility between the computation time needed by realistic deformable models and high refresh frequencies necessary for real-time haptic rendering. Giving the user a good haptic sensation requires refreshing the applied forces at least ten times more often than for giving a good visual sensation. Even though visually interactive deformable models exists, they can not be directly used for haptic rendering.
We suggest, in this paper, to extrapolate the forces computed by the deformable model to go beyond interactivity to haptic real-time.

Keywords: simulation, deformable models, real-time, force feedback, extrapolation.

1 Introduction

Whereas force feedback devices have reached technical qualities which allow their use in many simulation applications, the simulation of realistic deformable models remains too slow to be directly used for real-time force feedback, especially when the object geometry is complex. Research on physical deformable models has been very active for ten years[1, 2] and allows us to build visually interactive models. Thanks to retina persistence a visual sensation of continuity is provided with relatively low frequencies (about 25Hz). The sense of touch is much more precise. It requires refresh frequencies ranging from 300Hz for soft objects to 10kHz for rigid contact.
In the literature, two main approaches to reach haptic real-time are developed:

- computing forces empirically [3], for example by using a force proportional to the penetration depth of the tool in the object. In this case the force is not deduced from a physical deformation.

[†]e-mail: Guillaume.Picinbono@sophia.inria.fr
[‡]e-mail: Jean-Christophe.Lombardo@sophia.inria.fr

- using a simplified physical model. The simplification can be done in two ways, either by decreasing the size of the model [4] or by doing as much pre-computation as possible [5]. In the latter case, the model can't endure any topological change. A combination of the two possibilities is also a solution with a large pre-computed part and a small simpler part whose topology can be altered [6].

Our work is placed in the context of minimally invasive surgery simulation, and more precisely in virtual hepatic resection under laparoscopy [7]. The purpose is to give the surgeon the ability to practice on a virtual patient. The simulator must be as realistic as possible both for visual and for haptic rendering, which isn't attainable with the previously mentioned approaches.
In this paper, we propose a solution based on human characteristics. It has been shown ([8]) that if the sense of touch is very precise (we can feel vibrations until 10kHz, and force variations between 30 and 300 Hz), the gesture is slower (from 1Hz for the answer to an unexpected signal and 10Hz for a reflex action). Thus, the applied forces must be refreshed at high rate, but, because it is related to user's action, their evolution is quite slow. The idea is to estimate the force between two time steps of the deformable model simulation.

First, we will quickly describe our simulator architecture. Then we will present various force extrapolation schemes, and finally we will discuss them.

2 Simulator architecture

As the refresh rates of the visual rendering and the haptic rendering are quite different, it seems natural to divide the simulator in two parts. As shown in figure 1, one part manages the force feedback loop, and the other manages the object deformation loop. This latest can be divided into different steps:

- first get the position of the tool, which allows us to
- detect an eventual collision between the tool and the model, and in this case
- compute the deformation of the model;
- from this deformation, deduce the force applied on the tool;
- then, we only need to update the display and to send the force to the extrapolation module, before starting again the loop.

In our case, the object is a tetrahedric mesh representing a human liver. We give it a linear elasticity behavior modeled by the finite element method, which is either solved dynamically with the Tensor/Mass algorithm [6], or quasi-statically using pre-computation [5]. Our model includes 1394 vertices, 8347 edges, and 6342 tetrahedra. This size of mesh permits an interactive simulation on a bi-processor Pentium 333 or on a SGI Onyx2. In our setup, as in most simulators, the force feedback devices are handled by another computer, a Pentium 166 in our case, which drives the two *Laparoscopic Impulse Engines*[1]. The communication between the simulation workstation and the force feedback workstation is done via a classical *Ethernet* connection, using *UDP sockets*.

[1] http://www.immerse.com

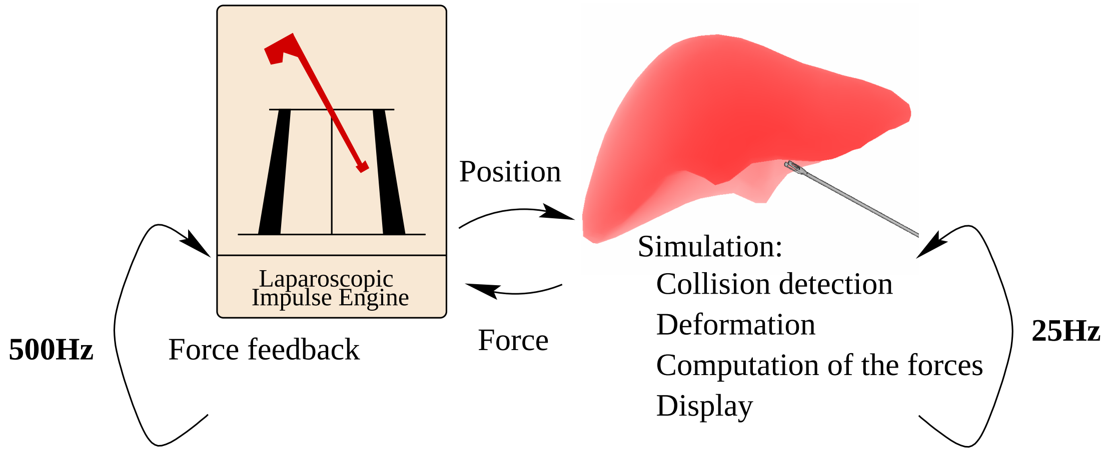

Figure 1: *Simulator architecture*

Our research group [7] works on all the problems encountered in minimal invasive surgery simulation. We can cite the optimization of the collision detection between the tool and the organ [9], the development of a more realistic behavior of the model (non-linear elasticity, anisotropy), the tuning of the behavior model to bio-mechanical data and realistic visual rendering. This paper focuses on the force feedback problem.

3 Force extrapolation

Our aim is to generate forces at a rate of 500Hz from forces computed by the deformable object simulation at a rate of about 30Hz. A solution could have been to introduce a time delay which value is the time needed to compute one simulation step, and to interpolate retrospectively the computed forces. But two major drawbacks prevent us from using this solution. First, this delay is not acceptable for a surgical gesture. Second, the time needed to compute one simulation step can vary somewhat. Thus we choose to estimate the current force by extrapolating the previously computed ones.

The force extrapolation process is described next.

The simulation loop gives us a discrete series of parameters $(t_n, \mathbf{P}_n, \mathbf{F}_n)$ representing the force $\mathbf{F}_n$ applied on the tool in position $\mathbf{P}_n$ at time t_n. The time step between two successive t_n is about 0.04s (visual real-time) and is not necessarily constant. Good quality force feedback can be reached by an update of the force at about 500Hz. So, we must choose an extrapolation function $\mathbf{F}(t)$ providing an estimation of the force to apply to the tool at time t $(t_n \leq t < t_{n+1})$ according to already known data $(t_i, \mathbf{P}_i, \mathbf{F}_i),\ i = 0..n$.

A very popular method for this kind of estimation is Kalman filtering. As defined in preface of [10], Kalman filtering is an optimal state estimation process, which gives a linear, unbiased, and minimum error variance algorithm to estimate the unknown state of a dynamic system from noisy data taken at discrete real-time intervals. But in our case, the absence of random perturbations directs our research towards a simple model of extrapolation.

In order to validate our approach, we have tested different extrapolation functions.

3.1 Constant extrapolation

The easiest way to produce the unknown data from the known one is to use the latest computed force as a guess for the current one.

$$\mathbf{F}^{cst}(t) = \mathbf{F}_n \qquad\qquad t_n \leq t < t_{n+1}$$

This extrapolation method has several advantages. First, it needs no computation. Then, as the applied force results from the deformation computation, such an extrapolation scheme ensures us to apply only valid forces. There is no risk of applying too large a force which could damage the device. In most force feedback devices, this extrapolation scheme is a part of the hardware and is automatically applied. This is the case of the *Laparoscopic Impulse Engine* hardware. Obviously, this method takes into account only the last computed force and produces a false estimation of the force as soon as it changes to another step. The main problem with this method is the discontinuity of the applied force which gives the sensation of touching a rough surface as soon as the refresh rate becomes too low (under about 300Hz).

3.2 Linear extrapolation over time

Another way to estimate the current value of a signal changing over time is to extrapolate it over time. As our deformable model sketches a linear elastic behavior, we only consider linear extrapolation.

Let us call t^e_{n+1} and $\mathbf{F}^e_{n+1}$ the time and the force estimated for the next data triplet, extrapolating the force linearly is equivalent to interpolating linearly on the interval of time $[t_n, t^e_{n+1}]$ the forces $[\mathbf{F}_n, \mathbf{F}^e_{n+1}]$.
Assuming that the simulation runs at a constant frequency, we have:

$$t^e_{n+1} = t_n + (t_n - t_{n-1}).$$

If we suppose that the force variation is linear in time, we have the following definition of the force function:

$$\mathbf{F}^t(t) = \frac{t - t_n}{t_n - t_{n-1}}(\mathbf{F}_n - \mathbf{F}_{n-1}) + \mathbf{F}_n \qquad\qquad t_n \leq t < t_{n+1}.$$

This method gives better results than the previously described one. The force discontinuities are less noticeable. But we must face a new problem, as the applied forces are not the ones that the simulation of the deformable model computes, they can be arbitrarily large. These force amplitude peaks occur especially when simulation time step increases $(t_n - t_{n-1} \ll t_{n+1} - t_n)$.

3.3 Linear extrapolation over position

The force changes are mainly due to the tool movement. In the case of a dynamic deformable model, the force also changes slightly over time, even if the tool doesn't move, when the model evolves towards its rest position. The tool position is thus vital for guessing what is the force applied to it by the deformable model. Although computing physically based deformation at the needed frequency is not yet possible, nothing prevents us from querying the force feedback device to know the tool position at such a high frequency. These observations lead us to develop a force estimator based on the tool position.

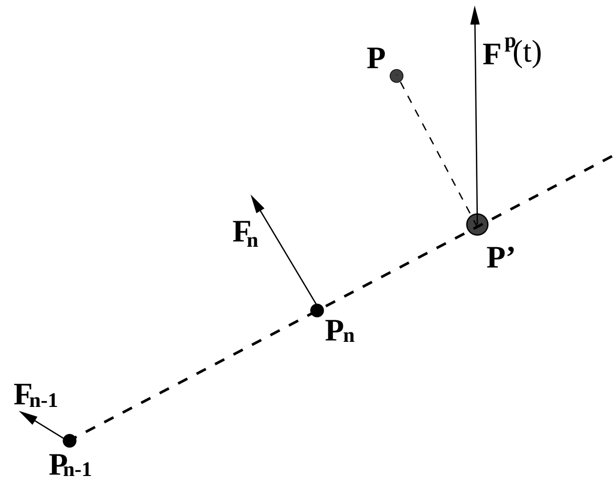

Figure 2: *Tool position projection for extrapolation over position*

As the tool position is not a scalar value, we can not directly apply the algorithm previously presented. A way to obtain comparable scalar values is to project the current tool position $\mathbf{P}$ in $\mathbf{P}$' on the line defined by the two precedent tool position $\mathbf{P}_{n-1}$ and $\mathbf{P}_n$ (figure 2). We can then consider the norm ratio for extrapolation:

$$\mathbf{F}^p(t) = \mathbf{F}_n + \frac{\|\mathbf{P}' - \mathbf{P}_n\|}{\|\mathbf{P}_n - \mathbf{P}_{n-1}\|}(\mathbf{F}_n - \mathbf{F}_{n-1}) \qquad t_n \leq t < t_{n+1}$$

We can notice that the error induced by the tool position projection is null, when $\mathbf{P}_{n-1}$, $\mathbf{P}_n$ and $\mathbf{P}$ are aligned, in other words when the tool trajectory is a line.

All of these three extrapolation methods were implemented in our surgery simulator. In order to compare and to evaluate them, several experiments were performed. The results are presented in the next section.

4 Evaluation

The first experiment was simply performed by using our surgery simulator with one of the three extrapolation schemes activated. The constant extrapolation gives us the sensation of touching a rough surface. The extrapolation according to time is an improvement, but we sometimes face to unexpected large forces. As soon as the tool movement is slow enough, the sensation given by the extrapolation over position are smooth. Of course, this is a very biased evaluation, and we tried to compare the three methods more objectively.

4.1 Reference data set

We want to compare the forces produced by the different extrapolation schemes to a reference force. We also want to study the impact of the simulation frequency on these errors. The time, the tool position, and the force computed by the simulation of the deformable model were recorded during several surgery simulation sessions. A simplified mesh was used to model the deformable object. It allows the simulation to reach a frequency of 80Hz. From this high frequency data, we generate lower frequency data by keeping only one sample over n. For example, keeping one sample over 4 gives us a data set at 20Hz. To evaluate the extrapolation, we need a reference force value for each extrapolated value.

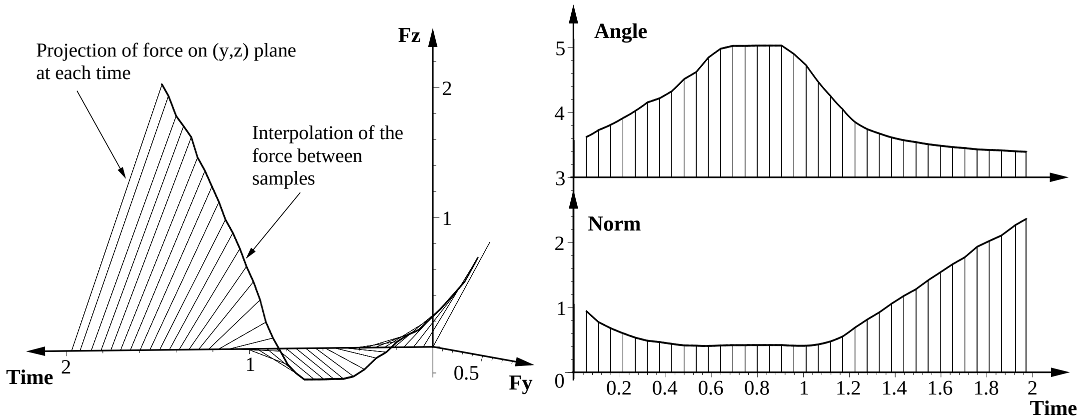

Figure 3: *A reference data set (20Hz) with interpolation (500Hz)*

We produced this reference force by linearly interpolating the forces computed by the simulation. This is shown in figure 3. Impulses represent the input forces, the line shows the interpolation. For an easier understanding of the shown figures, we plot 2D experiments. We also prefer a polar representation of the forces (right isde of the figure 3).

4.2 Error measurement

For each extrapolated force, we measure the differences between the interpolated and the extrapolated forces.

The left column shows the original data set with impulses and the extrapolated one with a line. These plots are compared to the reference one in figure 3. The difference between the extrapolated and the interpolated forces, which is taken as a measure of the error, is plotted, also in a polar fashion, in the right column of the figure 4. Theses plots shows the extrapolation results for a input data set at 20Hz.

We can note that the linear extrapolation over position gives very interesting results (very few discontinuities and no singular force). We tried the same kind of experiment with different simulation frequencies and with different tool movement. The position linear extrapolation always gave the best results, which is confirmed by sensation received during utilization.

Other tests were performed with different speeds for the tool movement. They show that for a high speed tool movement or for a too weakly sampled movement (too low simulation frequency), the error becomes important. An important feature to get a good quality force feedback is the sampling of the movement. When the sampling is too sparse, the assumption that the tool follows a linear path is no longer hold.

In order to complete our study, we have also performed a long simulation experience (several minutes) on our liver mesh. During this experiment, the simulation loop ran at about 30Hz. Some statistical data was computed on this experiment, including the tool speed. This time we only consider the norm error. This allows us to give the errors on the applied forces as a percentage of the interpolated force. Results are given in the following table:

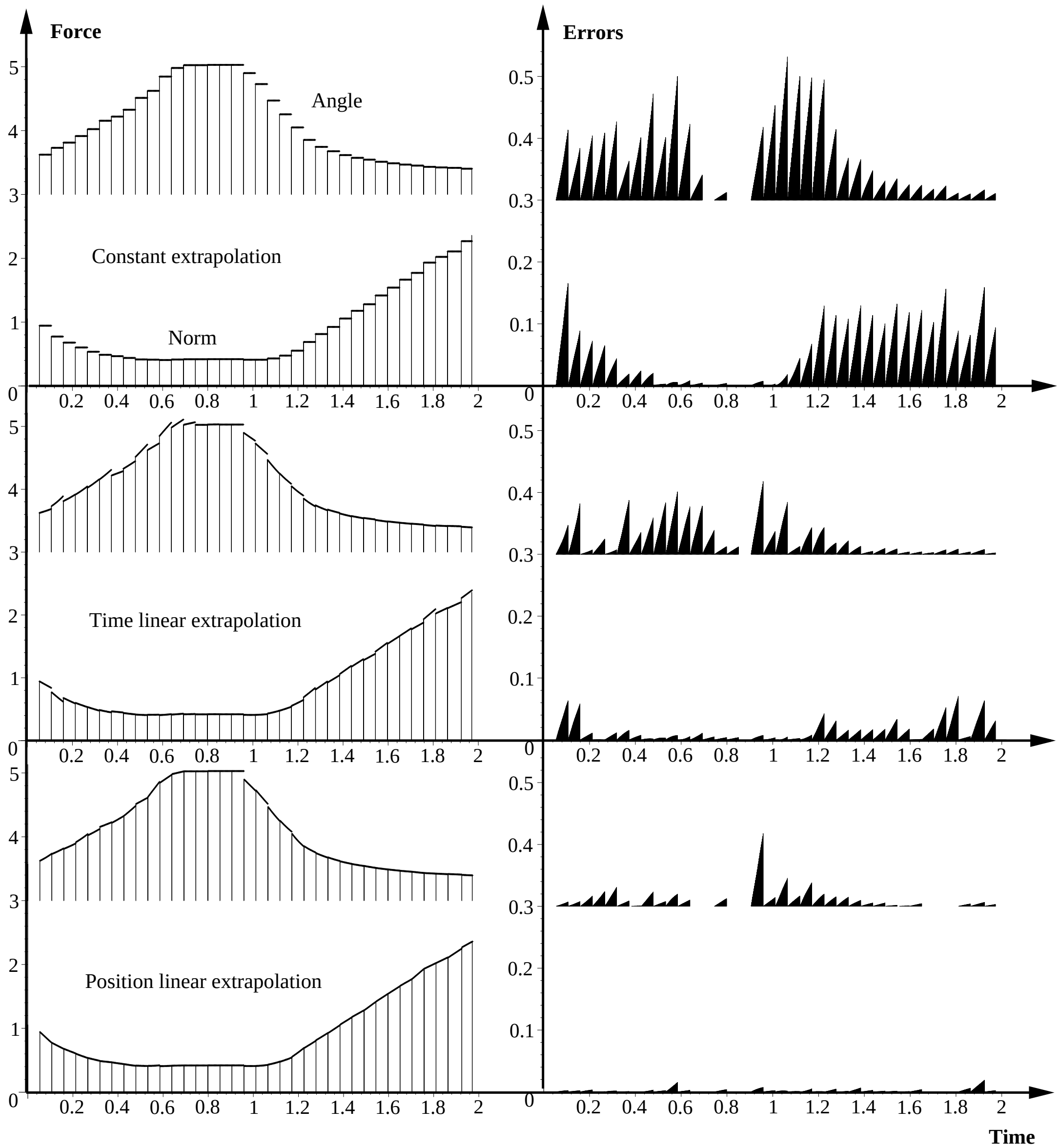

Figure 4: *Evaluating the different extrapolation methods*

Average speed	0.017 ms^{-1}		
Max speed	0.078 ms^{-1}		
Frequency	33 Hz		
Spatial sampling	0.5 mm		
method	average error	maximal error	maximal force
constant	1.1 %	56 %	133 %
linear in time	0.3 %	9 %	109 %
linear in position	0.1 %	7 %	106 %

This experiment confirms the precision of the linear extrapolation over position and the importance of the spatial sampling of the movement. Studies have shown that the surgeon's gesture is performed at about 0.01 $m.s^{-1}$. With such a speed and a simulation running at a visual real-time rate (about 20Hz), the linear extrapolation over position gives very good results.

5 Conclusion

We have developed a method to reconcile the computation time of realistic physically based deformable models and the high refresh frequencies needed for haptic rendering. This method takes into account the sensibility of the human touch, and the characteristics of the human gesture. It is based on two asynchronous loops, one for the deformable model running at visual real-time frequency, the other for the extrapolation of the forces computed by the simulation loop. We have tried three extrapolation functions: constant, linear over time extrapolation, and the more original linear over position extrapolation, which uses the fact that the position can be read on the force feedback device anytime we need it. We found that this last method gives good results as soon as the spatial sampling of the tool movement is dense enough, or, in other words the the ratio between the simulation frequency and the tool speed is large enough. This leads us to conclude that for the speed of the surgeon's gesture, a simulation at a visual real-time frequency (about 20Hz) is enough to produce high quality force feedback (about 500Hz) thanks to the linear extrapolation over position.

References

[1] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 205–214, July 1987.

[2] Marie-Paule Cani-Gascuel and Mathieu Desbrun. Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):39–50, January – March 1997. ISSN 1077-2626.

[3] C. Bosdogan, C. Ho, M. A. Srinivasan, S. D. Small, and S. L. Dawson. Force Interaction in Laparoscopic Simulation: Haptic Rendering Soft Tissues. *Proceedings of the Medecine Meets Virtual Reality (MMVR'6) Conference*, pages 28–31, January 1998.

[4] G. Burdea, G. Patounakis, V. Popescu, and R. E. Weiss. Virtual Reality Training for the Diagnosis of Prostate Cancer. In *IEEE International Symposium on Virtual Reality and Applications (VRAIS'98), Atlanta, Georgia*, pages 190–197, March 1998.

[5] S. Cotin, H. Delingette, and N. Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics*, (in press), 1998.

[6] S. Cotin, H. Delingette, and N. Ayache. Efficient linear elastic models of soft tissues for real-time surgery simulation. Technical Report RR-3510, INRIA, 1998.

[7] Action Incitative SIMulation de chirurgie. http://www.inria.fr/epidaure/AISIM.

[8] T.L. Brooks. Telerobotic response requirements. *Proceedings of the IEEE International Conference on systems, Man and Cybernetices*, pages 113–120, 1990. Los Angeles.

[9] Jean-Christophe Lombardo, Marie-Paule Cani, and Fabrice Neyret. Real-time collision detection for virtual surgery. In *Computer Animation*, Geneva - Switzerland, May 26 - 28 1999.

[10] Charles K. Chui and Guanrong Chen. *Kalman filtering with real-time applications*. Springer, 1987.