2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Identifying Physical Properties of Deformable Objects by Using Particle Filters

Steve Burion*, Francois Conti†, Anna Petrovskaya†, Charles Baur*, Oussama Khatib†

*VRAI-Group
IPR - LSRO
EPFL
CH - 1015 Lausanne, Switzerland
steve.burion@a3.epfl.ch, charles.baur@epfl.ch

†Computer Science Department
Stanford University,
Stanford, California 94305, USA
{conti, anya, ok}@cs.stanford.edu

***Abstract*—This paper presents a new approach for estimating physical properties of deformable models from experimental measurements. In contrast to most previous work, we introduce a new method based on particle filters which identifies the different stiffness properties for spring-based models. This approach addresses some important limitations encountered with gradient descent techniques which often converge towards ill solutions or remain fixed in local minima conditions.**

***Index Terms*—simulation, real-time, deformable bodies, model estimation, particle filter.**

I. Introduction

In recent years important efforts have been devoted towards the development of sophisticated numerical models for describing and simulating the behavior of complex deformable bodies. Among the many techniques developed, finite element models (FEM) have become a gold standard for most simulation applications used today in the industry. With the development of faster computers with enhanced graphics capabilities, these simulation technologies have also been deployed in the movie industry to simulate the dynamic behaviors and interactions between virtual actors and their environments. If such physics engines have enabled animators to program the motion of virtual characters semi automatically, considerable amounts of time are still spent adjusting the physical parameters of each physical model in order to obtain the perfectly desired behavior. Unlike cartoon characters which often display exaggerated and often unrealistic motions, programming the fine motion of a real human in a realistic way remains an extremely difficult task. In a different area such as the medical field, physics engines are also used to improve the capabilities of current imaging technologies such as CT or MRI scanners. For medical and technical reasons such imaging devices can only be used during short periods of time or at different time intervals during an operation. By developing real-time physical models of organs and by using external sensory devices (i.e ultra-sound, force sensors) to update these models, research efforts are aiming at developing various techniques to provide 3D image information to the surgeon in real time. Even further, these same models could also be used for simulation purposes to plan an operation before hand or to train doctors. While the goals addressed by these different applications vary greatly, and despite the fact that very different types of physical models may be used (i.e finite elements versus spring based models), all require some manual or automatic calibration in order to precisely simulate the behavior of the environment being modeled.

To simulate the mechanical behavior of a real object, all physical properties such as stiffness, damping and mass distribution must be identified. In the ideal case, these values may be derived directly from the shape and the material properties composing the object, but due to the complexity of most structures (i.e biological tissues) such information is rarely available directly. In order to identify these parameters, experimental approaches can be used instead; these strategies require applying various force constraints onto the object while measuring the changes of its configuration.

As the physical parameters of the estimated model may be adjusted, the quality of correspondence between the reference model (or real object) and the estimated model is evaluated by measuring the variations of shape under different force constraints. In practice, this may be performed by comparing the average distance between two clouds of points (i.e. markers) which represent the surface of each object. In applications where dynamic properties need to be simulated (damping), velocity measurements may also be used.

To estimate the physical properties of a model, different manual or automatic techniques have been proposed. In [6] and [7] the authors present a genetic based algorithm which estimates various parameters of a deformable model; in [7] they extend their work to identify stiffness properties too. Due to the complexity of their algorithms, their work is limited to 2D models only with homogeneous stiffness.

In [8] a mass-spring model is used in combination with a neural network to simulate the physical behavior of a deformable body in real time. The different nodes composing the neural network are programmed to model the position of the mass units, spring functions and viscosity functions. The authors present a learning method based on a gradient descent approach to tune the neural network according to a reference model.

978-1-4244-1647-9/08/$25.00 ©2008 IEEE.

Optimization search of mass-spring parameters using a simulated annealing algorithm is presented in [5] and address 2D situations only. Another approach presented in [20] refines the mass-springs representation and computes a uniform stiffness value according to the Young Modulus. An FEM representation is used as a reference model.

In [21] experiments using a vision based system and a least square approach are presented.

In this paper we present a new strategy based on particle filters to estimate the stiffness properties of a deformable body. Our approach is composed of three stages: In the first step, the behavior of the object is identified by applying force constraints at different locations on the object and by recording the resulting displacements. During the second phase our algorithm randomly selects various sets of stiffness values for each spring composing the estimated model; each set being referred to as a particle. The same external forces that were applied on the real object in step one are then applied on the estimated model under each set of stiffness configurations. The particles (or stiffness configurations) which produced the best results are selected and new particles located within their neighborhood are created. The algorithm halts when the displacements between the reference and the estimated model reach a desired level of correspondence.

II. Background

A. Modeling Deformable Objects

Rendering deformable objects in a realistic manner is a challenging task in computer simulation. While rigid bodies can be accurately described using a limited number of parameters, deformable objects, require a much larger set of variable to express their configuration. During the last two decades, a wide variety of approaches have been presented in soft tissue simulation. Mass-spring system techniques [3] [4] have widely and effectively been used for modeling deformable bodies. Each object is described by a set of mass points dispersed throughout the object and interconnected with each other through a network of springs. These systems are easy to model, to construct and have well understood physics. They are also well suited for parallel computation, making it possible to run complex environments in real-time for interactive simulations. (A surgery simulator for instance). On the other side, mass-spring systems have some drawbacks. Incompressible volumetric objects and high stiffness materials have poor stability requiring small time steps during the numerical integration process, which considerably slows down the simulation. A second category of techniques is finite elements methods [1] [2], which offer a strategy with much higher accuracy to solve continuum models. In FEM, unlike mass-spring methods where the equilibrium equation is discretized and solved at each finite mass point, objects are divided into unitary surface (2D) or volumetric (3D) elements joined at discrete node points where a continuous equilibrium equation is approximated over each element. Compared to mass-spring systems, finite element methods generate a more physically realistic behavior, but at the cost of requiring much more numerical computation. For this reason FEM techniques are difficult to use for real-time simulations.

Since our identification approach requires running a simulator on large amounts of data sets, we opted for a real-time mass-spring skeleton system [16]. Under this model, volumes are approximated with macroscopic elastic sphere which are placed along the medial axis transform of the object. Spheres composing the skeleton are then connected together with three-dimensional elastic links that provide elongation, flexion and torsion properties to the object.

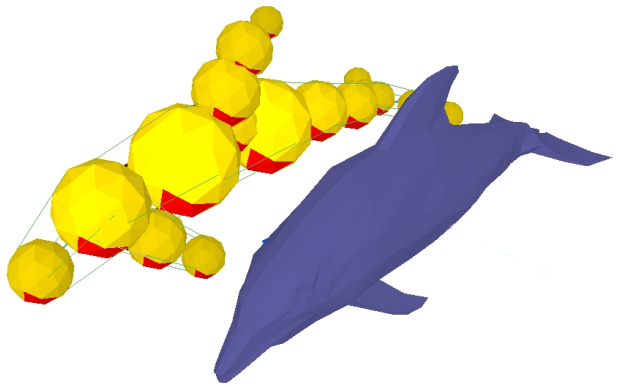

Fig. 1. Skeleton based model: Filling spheres are placed along the medial skeleton of the object and are connected together with elastic links which model elongation, flexion and torsion properties. Each vertex of the mesh is attached to the nearest sphere or link

B. Particle Filters

Particle Filters [12] have already been used successfully in various fields such as mobile robotics [9] [10], feature tracking [11], and medical applications [17]–[19]. They provide a powerful tool to estimate multi-modal parameters in a multi-dimensional problem.

The goal of our work consist in estimating the stiffness parameters (elongation constant k_E, flexion constant k_F and torsion constant k_T) for each spring that composes the physical skeleton of our object. In order to limit the complexity of our system, we shall consider that all damping properties are uniform.

Characterizing the stiffness properties of an object involves estimating their values based on a set of experimental measurements. In practice, such measurements are generally imprecise and noisy due to the limitations of the hardware used to acquire the data. Bayesian estimation provides a systematic approach to parameter estimation under these conditions. One commonly used Bayesian estimation technique is described as the particle filter algorithm, which is a sampling and estimation method that is used to approximate a posterior (i.e. probability distribution) over all states according to the available measurements made on the system and the model evolution. This approach can deal with multi-modal problems where classic optimization algorithms, such as gradient descent, usually fail. The main advantage of Particle Filters comes from the fact that the system maintains a ”global view” of the problem due to the

large number of samples (particles) spread out in the search space.

Baysian filtering techniques have been widely covered in the literature. A survey of sequential sampling methods for Baysian filtering can be found in [13]. Another review about particle filters can also be found in [12].

The particle filter algorithm works in three phases. In the first stage, the algorithm samples the search space with a finite number of particles spread out randomly and uniformly. In the second stage an associated weight is computed for each particle according to a set of measurements previously acquired. Finally a resampling process selects a new set of particles by favoring previous particles with higher weights. A mutation step can be added at the end to redistribute the selected particles around the estimated solution.

III. Algorithm Description

In the following section, we describe the Particle Filter (PF) approach that we used to characterize the physical properties of our models.

The posterior density represents the probability distribution $p(X|Y_1...Y_j)$, where X is the estimated parameters and Y_j the measurement at time j, is represented by a set of weighted particles with M particles $S_j = \{(s_j^{[0]}, w_j^{[0]}), (s_j^{[1]}, w_j^{[1]})..., (s_j^{[M]}, w_j^{[M]})\}$ where the weights w are proportional to the measurement model:

$$w_j^{[m]} = Cp(Y_j|X = s_j^m) \tag{1}$$

with C a normalizing constant. The weights are normalized, meaning that

$$\sum_{m=1}^{M} w_j^{[m]} = 1 \tag{2}$$

At time j, a new measurement Y_j becomes available using the measurement model $p(Y_j|X)$. Then, the posterior probability can be updated using the Bayes' rule.

In our model, the parameters that we wish to estimate correspond to the stiffness properties of each spring. If we consider that each spring may hold different stiffness properties, the number of parameters needed to describe our model is proportional to the number of links N inside the skeleton model. The state variables X are defined as follow:

$$\begin{aligned} X \quad = \quad & \{(k_E(1), k_F(1)),\ (k_E(2), k_F(2)),\ ..., \\ & ...,\ (k_E(N), k_F(N))\} \end{aligned} \tag{3}$$

The set of particles S_j in the search space V is defined in the same way. Each particle at time j has a dimension of $2N$ and is composed as follow:

$$\begin{aligned} s_j \quad = \quad & \{(k_E(1)_j,\ k_F(1)_j),\ (k_E(2)_j,\ k_F(2)_j),\ ... \\ & ...,\ (k_E(N)_j,\ k_F(N)_j)\} \end{aligned} \tag{4}$$

The main loop of the adapted PF algorithm is presented in Alg. 1. At the end of the algorithm, we find two methods to estimate the state variables given the set of particles: Either by taking the particle with the highest weight, or by computing the mean values for each particle dimensions. With the second method, the standard deviation for each dimension should remain small, meaning that all the particles should be concentrated in the same area to provide relevant results.

A. Measurement model

The measurement model is defined by the probability $p(Y|X)$. In our application, the measurement space is composed by the position in the 3D space of each node $P = \{p_0,\ p_1, ...,\ p_N\}$ with $p_n = \{x_n,\ y_n,\ z_n\}$ the position of the node n, the force applied on the model $f = \{f^{(x)},\ f^{(y)},\ f^{(z)}\}$ and the node on which the force is applied $n^{(f)}$. One measurement at time j is defined by:

$$Y_j = \{P_j,\ f_j,\ n_j^{(f)}\} \tag{5}$$

For each particle, the observation likelihood is represented by a Gaussian computed as follows:

$$p(Y_j|X_m) = \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{e(Y_j, X_m)^2}{2\sigma}) \tag{6}$$

The most expensive operation in the standard Particle Filter algorithm is the evaluation of the likelihood function (6) because it has to be performed once at every algorithm loop for every particle.

After applying the selected particle to the model, the simplest and most instinctive way to compute its likelihood is to define a function $e(Y_j, X_m)$ which expresses the overall distance between the mass nodes of the estimated model and ones of the reference model. This approach turns out to be quite expensive because, for each particle, the simulator needs to compute the final state of the estimated model when a given external force is applied. In practice, the simulator may need to perform a few hundred iterations in order to obtain the final steady state before distances can be computed.

In order to speed up this process, we established a different measurement model by looking at the force domain. Instead of comparing the overall distance between the estimated and reference models, we analyze the resulting forces at each mass node. When the initial measurements are performed on the reference model, external forces are applied at different locations of the object. Since the measurements are performed once the reference model reaches a steady state (zero velocity at each node), we can conclude that the resulting forces at each mass node is equal to zero. If we position the mass nodes of the estimated model in the same configuration as the reference model and apply the same external forces, we can now measure the likelihood of a particle by computing the resulting forces at each mass node. If the sum of these forces is near zero, this means that the stiffness of the springs are very similar to the reference model. On the other side, if

the resulting forces are important, we can conclude that the estimated stiffness values are far from the reference model.

For each particle, there are two observation likelihoods: one for forces and one for torques. These quantities are presented in equation (6) with $e(Y_j, X_m)$ defined as follow:

$$e_F(Y_j, X_m) = \sum_{n=1}^{N} (\sum \overline{F_i})_n \tag{7}$$

$$e_T(Y_j, X_m) = \sum_{n=1}^{N} (\sum \overline{T_i})_n \tag{8}$$

Where $e_F(Y_j, X_m)$ and $e_T(Y_j, X_m)$ denote the error functions to compute the resulting forces and torques on each node. $(\sum \overline{F_i})_n$ represents the sum of all forces (internal and external) on node n and $(\sum \overline{T_i})_n$ represents the sum of torques on node n.

B. Weights computation

The last step of the algorithm consists in estimating the weight for each particle in order to define their likelihood:

$$w_m = \prod_{j=1}^{J} p(Y_j^F | X_m) \prod_{j=1}^{J} p(Y_j^T | X_m) \tag{9}$$

The weights are then normalized as shown in equ. (2).

PF_loop (V, S, M, σ_r, σ_F, σ_T)
// Sample M particle uniformly in the search space V
$\{X_m\} \leftarrow$ Uniform_Sample(M, V)
for k = 1 to K **do**
  for j = 1 to J **do**
    // Compute estimated forces and torques
    $(\overline{F_n}, \overline{T_n})_j \leftarrow$ evol_model$\{S_k, Y_j\}$
  end for
  // Compute likelihood weights
  $w_m \leftarrow p(Y_j^F | X_m) p(Y_j^T | X_m)$
  // Normalize weights
  $w_m = w_m / \sum w_m$
  // Resample with Low Variance Sampling Method
  $\{X_m\} \leftarrow$ low_var_resample($\{X_m\}$, $\{w_m\}$, M)
  // Mutate
  X_m = X_m + white_noise(σ_r)
end for

Alg. 1: PF algorithm to characterize deformable objects.

C. Algorithm Parameters Definitions

The number of particles used in our PF system directly affects the precision at which we can estimate the physical parameters of our model and also the speed at which the estimation can be performed.

If the model we wish to evaluate is describes by N variables, the number of necessary particles is defined by M^N. By including a mutation step to add noise in the particle set after resampling, it becomes possible to recenter the particles around the found solutions, and therefore we can refine our search while still maintaining a reasonable number of particles.

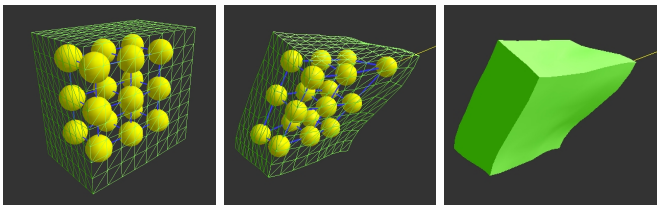

Fig. 2. The deformable object used for the first experiment. A cube in the skeleton representation made with 3x3x2 spheres. On the left, the cube is in a steady position under gravity. In the middle and on the right, an external force is applied.

IV. Results under simulation

To validate our approach, several experiments were performed on different sets of 3D models of various complexity and size. The application that we developed was programmed in C++ and integrated the open source framework CHAI 3D [23] which provides graphic and dynamic rendering modules. For each experiment. we defined a reference model on which our application could perform various measurements. An estimated model for which the stiffness properties need to be estimated was interfaced to the Particle Filter for identification. All experiments were performed on a 2.1 GHz single core laptop computer running Windows XP.

In the first experiment, we present the results performed on an homogeneous deformable cube consisting of a 3x3x2 network of elastic spheres. The model is illustrated in figure 2. Since every links contains the same stiffness properties, the complexity of the problem is reduced to two dimensions where k_E and k_F are to be estimated. 600 particles were used for this experiment and six measurements were performed on the reference model. The evolution of the particle set, starting with the initial distribution, is plotted in figure 3. We observe that the particles converge towards a unique solution only after a few iterations. The algorithm estimates the stiffness values with a final error of only 0.8% for k_E and 0.6% for k_F. The standard deviation for k_E and k_F was 0.023N/mm and 6.7Nmm/rad respectively. It took 7.8s to obtain this result.

In this second experiment, we apply the same algorithm on a non-homogeneous object consisting of 3x6 spheres. The object is split in 3 areas with different stiffness; the model is illustrated in figure 4. In this experiment, six unknown parameters need to be estimated (k_E and k_F for each zone). 900 particles were spread out randomly in the entire search space. The evolution of the particle set, starting with the initial distribution, is plotted in figure 5. Once that the particle model had converged, the maximal error was measured at 4.6% for k_E and 10.1% for k_F. The standard deviation did not exceed 0.069N/mm for the elongation parameters and 15.1Nm/rad for the flexion parameters. The experiment was performed in 42.6 seconds.

To evaluate our particle filter method, we also ran a series of experiments using a non-linear least-squares optimization

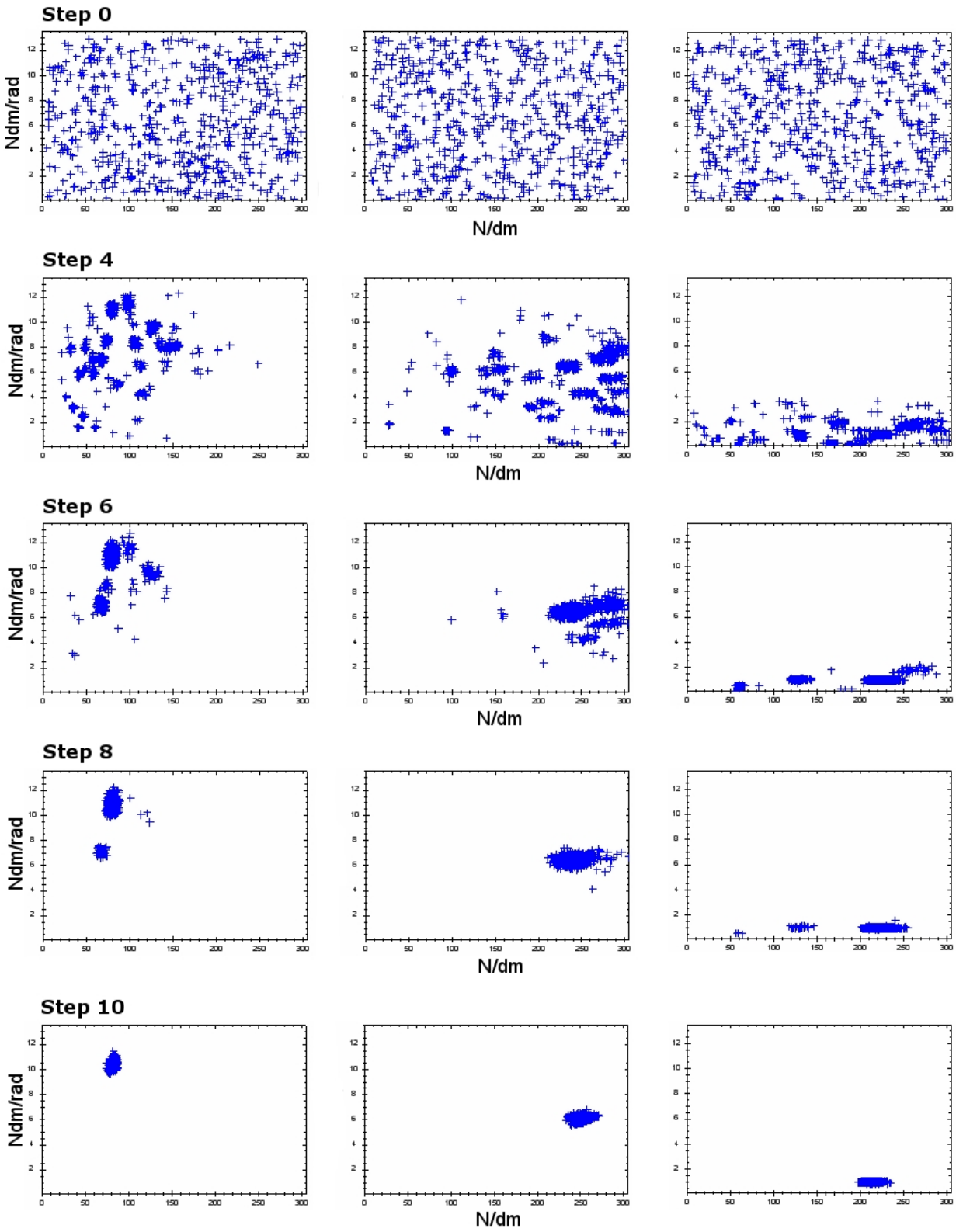

Fig. 5. Experiment II: the objet to characterize is a structure of 6x3 spheres in a network of springs defined by 3 homogeneous zones. These graphs represent the particle distribution at the end of each step of the PF algorithm. It starts with the initial distribution and then a selection of steps are shown. Each step contains 3 graphs that represent the 3 homogeneous zones. Each of them include elongation stiffness k_E [N/mm] for horizontal axes and flexion stiffness k_F [Nmm/rad] for vertical axes. 900 particles were used to estimate the 6 parameters k_{E1}, k_{E2}, k_{E3} and k_{F1}, k_{F2}, k_{F3}

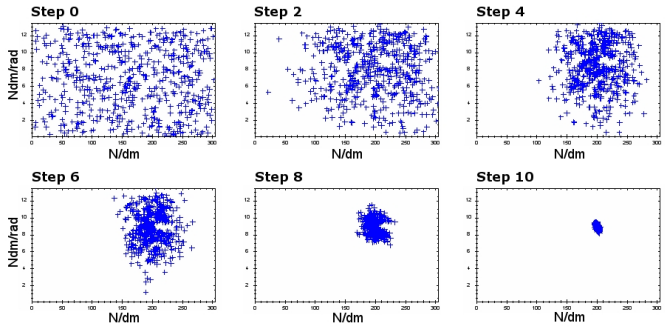

Fig. 3. Experiments I: the object to characterize is a deformable cube with homogeneous stiffness consisting of 3x2x3 sphere. Every link carried the same stiffness parameters. These graphs represent the particle distribution at the end of each step of the PF algorithm. It starts with the initial distribution and then each even steps are shown. Each graph include elongation stiffness k_E and flexion stiffness k_F for horizontal and vertical axes respectively. 600 particles were used to estimate the two parameters k_E and k_F

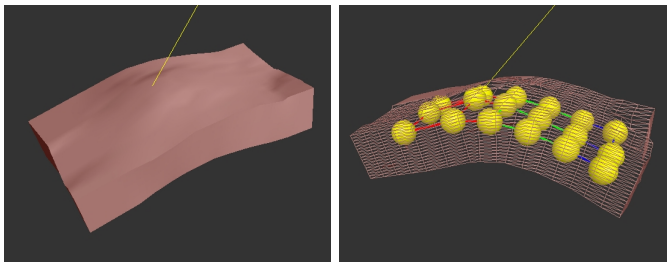

Fig. 4. The deformable object for the second experiment. A sheet of 6x3 spheres split in 3 areas with different stiffness. Each link colors represent an area with a different stiffness.

algorithm. The experiments were performed under Matlab with the non-linear optimization tool box. Despite the fact that the algorithm was able to estimate correctly the stiffness parameters for a simple homogeneous object, the results were less accurate and the computation time was 10 to 20 times slower compared to the Particle Filter approach. Moreover, the computation time increased drastically with the number of parameters to estimate. The algorithm also fell more often in local minima when computing solutions on larger objects.

V. Conclusion and Future Work

In this paper we introduced a new method based on particle filters to identify stiffness properties of deformable objects. Through experiments performed on different 3D models, we showed that by selecting an adequate set of particles, it is possible to develop an effective algorithm which converges towards a solution that describes the physical properties of the object. Our approach also addressed some problems encountered with gradient descent techniques.

Future work will include refining the Particle Filter algorithm in order to reduce the necessary number of particles when more complex deformable objects are used. Other research aspects will include using such algorithm on real objects; this will require developing different sensing techniques to perfectly capture the configuration of the object under various force constraints.

References

[1] L. Segerlind, *Applied Finite Element Analysis*, John Wiley and Sons, New York, 1984.

[2] K-J. Bathe, *Finite Element Proceedures*, Prentice Hall, Englewood Clis, NJ, 1996.

[3] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, *Elastically deformable models*, Computer Graphics (Proc. SIGGRAPH87), 21(4):205214, 1987.

[4] K.Waters and D. Terzopoulos, *Modeling and animating faces using scanned data*, Visualization and Computer Animation, 2:123-128, 1991.

[5] O. Deussen, L. Kobbelt, and P. Tucke, *Using Simulated Annealing to Obtain Good Nodal Approximations of Deformable Objects*, In Computer Animation and Simulation, pages 3043, 1995.

[6] G. Bianchi, M. Harders, and G. Szekely, *Mesh topology identification for massspring Models*, In MICCAI 2003, volume 1, pages 5058, 2003.

[7] G. Bianchi, B. Solenthaler, G. Szekely, and M. Harders, *Simultaneous Topology and Stiffness Identification for Mass-Spring Models Based on FEM Reference Deformations*, In MICCAI (2), pages 293301, 2004.

[8] A. Nrnberger, A. Radetzky, and R. Kruse, *A Problem Specific Recurrent Neural Network for the Description and Simulation of Dynamic Springs Models*, IEEE IJCNN98, pages 468473, 1998.

[9] S. Thrun, *Particle Filters in Robotics*, In Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI), 2002.

[10] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, *FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges*, In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI), Acapulco, Mexico, 2003. IJCAI.

[11] S. Hamlaoui, F. Davoine, *Facial action tracking using particle filters and active appearance models*, In Proceedings of the 2005 Joint Conference on Smart Objects and Ambient intelligence. Grenoble, France, October 12 - 14, 2005.

[12] S. Arulampalam, S. Maskell, N. J. Gordon, and T. Clapp, *A Tutorial on Particle Filters for On-line Non-linearNon-Gaussian Bayesian Tracking*, IEEE Transactions of Signal Processing, Vol. 50(2), pages 174-188, February 2002.

[13] A. Doucet, *On Sequential Simulation-Based Methods for Bayesian Filtering*, Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.

[14] A. Petrovskaya, O. Khatib, S. Thrun, A.Y. Ng, *Bayesian Estimation for Autonomous Object Manipulation Based on Tactile Sensors*, Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on, vol., no.pp. 707- 714, May 15-19, 2006.

[15] D. Fox, *Adapting the Sample Size in Particle Filters Through KLD-Sampling*, The International Journal of Robotics Research.2003; 22: 985-1003.

[16] F. Conti, O. Khatib, and C. Baur, *Interactive Rendering Of Deformable Objects Based On A Filling Sphere Modeling Approach*, In IEEE International Conference on Robotics and Automation, 2003.

[17] B.A. Ma and R.E. Ellis, *Surface-Based Registration with a Particle Filter*, In Medical Image Computing and Computer-Assisted Intervention - MICCAI 2004, Part 1, 566-573. Springer Lecture Notes in Computer Science #LNCS 3216, 2004.

[18] C. Florin, J. Williams, A. Khamene, N. Paragios, *Registration of 3D Angiographic and X-Ray Images Using Sequential Monte Carlo Sampling*, 427-436 Electronic Edition, BibTeX.

[19] C. Florin, N. Paragios and J. Williams, *Monte-carlo Sampling, Particle Filters and Segmentation of Coronaries*, Technical Report RR0503 CERTIS, 2005.

[20] V. Baudet, M. Beuve, F. Jaillet, B. Shariat, F. Zara, *New Mass-Spring System Integrating Elasticity Parameters in 2D*, Research Report RR-LIRIS-2007-003, quipe SAARA, Universit Lyon 1, 2007.

[21] D. K. Pai, K. van den Doel, D. L. James, J. Lang, J. E. Lloyd, J. L. Richmond, and S. H. Yau, *Scanning physical interaction behavior of 3D objects*, In Computer Graphics (ACM SIGGRAPH 2001 Conference Proceedings), 2001.

[22] J. Lang, D. K. Pai, and R. J. Woodham, *Robotic Acquisition of Deformable Models*, ICRA 2002.

[23] Conti F, Barbagli F, Morris D, Sewell C, *CHAI: An Open-Source Library for the Rapid Development of Haptic Scenes*, Demo paper presented at IEEE World Haptics, Pisa, Italy, March 2005.