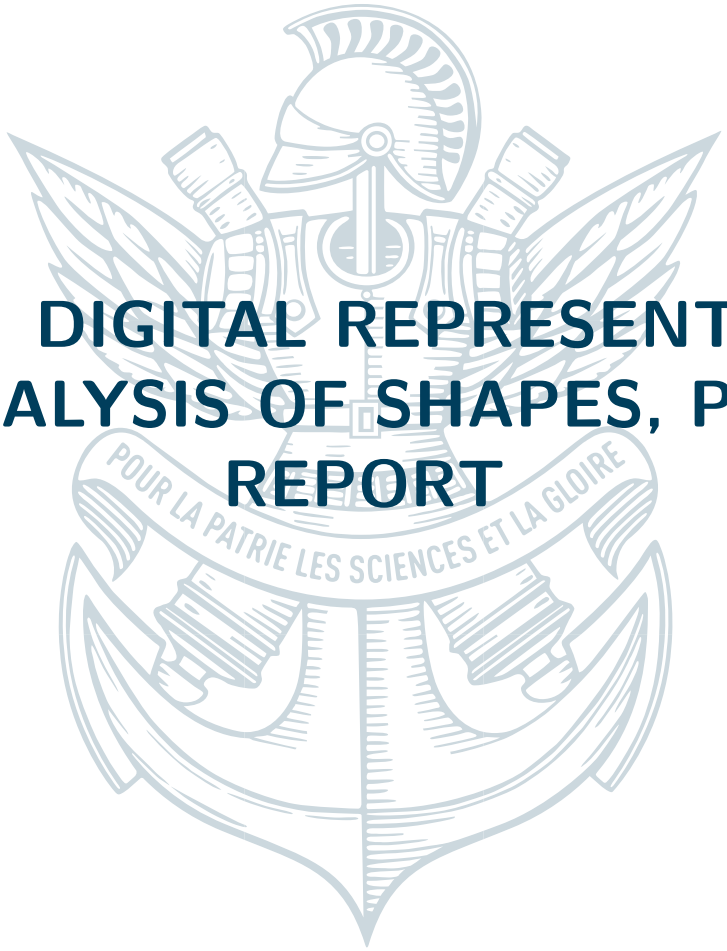

INF574 : DIGITAL REPRESENTATIONS AND ANALYSIS OF SHAPES, PROJECT REPORT

Décembre 2021

Thibaut Vacek, Gaspard De Batz

SUMMARY

1 INTRODUCTION

Our project consist in the implementation of a method to compute the deformations of objects expected to be very efficient. Unlike most of the methods used to do so, this one does not base the deformation on the estimation of the external forces on the objects and the intern constraints to deduce its evolution over time, the method we studied links to each mesh a form to refer to and around which the mesh oscillates elastically when deformed by an external constraint. The result is not realistic but allow to create scenes with numerous objects that interact at a reasonable cost which can be really useful for example in video games. Starting from this general principle, we implemented different methods depending on expected results, for example, the "quadratic" one transmits the deformation wave more uniformly than the "linear" method. We also implemented the cluster method which goal is to represent subdivided structures that are therefore not linearly elastic.

2 STATE OF THE ART

The paper proposing the method we implemented is well documented and refers to numerous other papers dealing with the elastic shape animation, with various results in terms of accuracy, efficiency, controllability, or stability. It is difficult to say which searcher published the best and more recent paper, knowing that this type of research is a key aspect of the video game company's work. Those companies also protect the knowledge in this field, but we see nowadays that the video games go further than only rigid bodies, the most famous ones being FIFA and NBA 2k that propose moving realistic jerseys on the players.

3 PROJECT PRESENTATION

Let us present our work in this section: you will find a first part explaining how the code we wrote should be understood, and then a bit more details about the diverse methods we implemented.

3.1 Code's structure

Starting from a previous project from the course, we chose to use a structure with external functions to limit the calculus in the main program so that our code remains flexible and we can easily jump from one method to another. To limit the space needed, most of the operations are done in place (except during the integration, where they are stocked in a temporary matrix, the values we modify are not reused afterward). For more clarity, we divided the code in different files each one corresponding to a specific method we implemented. The Auxiliary.cpp file provides numerous functions useful during the integration of the positions, and more generally for the whole process of elasticity calculus. The Forcesext.cpp file provides the two functions that calculates the gravity force and the reaction force. The "main2.cpp" file is here to give you a taste of how we manage to code the cluster-based method.

3.2 Methods/ extensions

3.2.1 • Linear deformation

The linear deformation is the basic implementation of the paper, defining the elasticity of the shape point by point by their reference to the original shape. For general deformations such as a fall, this technique can be relevant but the fact that the deformation does not extend to close points makes it difficult to use for local deformations. as just the few initially deformed points would move as separated strings.

3.2.2 • Quadratic deformation and damping

The quadratic deformation can be seen as the extension of the linear deformation. As described in the paper, we use the same principle of a deformation matrix that pushes the points towards their destinations. Unlike the linear method, we use crossed coordinates to get a non linear deformation. As a result of this change, the deformation extends to the whole shape which makes the movement of the object more natural.

Both of the linear and quadratic methods use a semi explicit Euler method to find the position of the points at each time step. To make the movement more natural and as suggested in the video referred in the paper, we tried to implement a damping in the movement. To do so, we added a parameter damping (D) and changed the integration formula :

$$\begin{cases} v_i(t+h) = (1-D)v_i(t) + \alpha\frac{g_i(t)-x_i(t)}{h} + h\frac{f_{ext}}{m_i} \\ x_i(t+h) = x_i(t) + hv_i(t+h) \end{cases} \tag{1}$$

We can see when launching the algorithm that the object stabilize at a point. Therefore the previous system has a limit and we can get from the equations that v tend toward zero and therefore x tend to g so the damping does not changes the final result.

3.2.3 • Splitting the shape in clusters

In a bid to widen the range of possibilities using this method for elastic) shape-deformation the paper proposes a “Cluster Based Deformation”. The principle is as follows: “subdivide the space around a given surface mesh into overlapping cubical regions” means that you will split your vertices matrix into several overlapping clusters (meaning that the vertices at the border of a cluster will be in the adjacent cluster too). In the integration of the speed and position, a term specific to the cluster is added. This term is the same as the original elastic term, except the goal positions are computed with respect to the cluster, not the general shape. We tried to implement this method by taking the example proposed in the paper; the cluster are computed on a rectangular parallelepiped, split in 2, 3 or more smaller paralepidids. We had force to create the vertices and the adjacency matrix; we could choose the number of points, the height of the parallepiped and the number of clusters.

The computation of the clusters is as following: we take the cluster from the general vertices’ matrix, we integrate it, then we rewrite the new positions in the general vertices matrix. One of the problems encountered was the overlapping points; for them, two different integrations are calculated, we decided to take the average of the two. Furthermore, when we take an overlapping point form the general matrix, the previous already integrated the point, this is not the position we want. We had to store the next cluster’s positions BEFORE integrating the overlapping points. Another question was the choice of the goal points; for the general shape, the goal points are the positions from the straight stick, but for the clusters, we can choose the corresponding points from the straight stick but rotated so as to fit better the bent stick. You can see later the different results with different choices.

3.2.4 • Exterior forces implementation

To test the realism of our model, we wanted to create a scene where elastic objects bounce on the ground and deform depending on the shock they have undergone. This idea required us to implement the forces applying on those objects. The exterior forces are taken in account in the integration formula proposed by the paper.

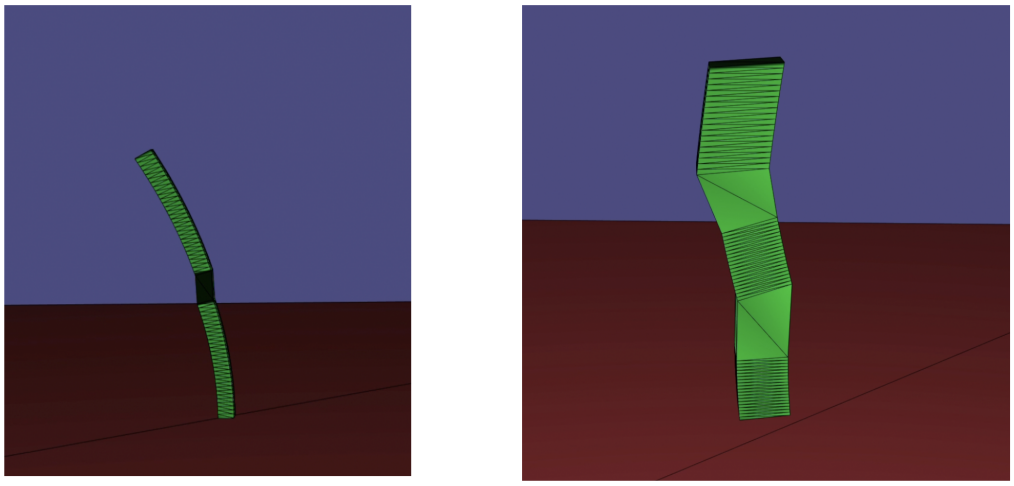

Figure 1: A bent parallepiped split into two then three clusters.

We chose to create a force matrix of dimension $number_of_vertices \times 3$ as the exterior force can change between two vertices of the same mesh.

This fact was especially helpful when we implemented the reaction force when the mesh is hitting the ground at $z = 0.0$. We modeled the force by the following formula:

$$\begin{cases} if\ z > 0.0,\ F_{reaction} &= (0.0, 0.0, 0.0) \\ if\ z < 0.0,\ F_{reaction} &= -1.9v_i + g \end{cases} \tag{2}$$

Indeed, the vertices hitting the ground are given the same speed they had, but in the uplifting direction. The little coefficient before the speed is here to model a little amortization (different to the amortization due to the elasticity of the shape), because we had better results when simply delete the gravity factor in the final exterior force (what explains the g factor, here to compensate the gravity still present). This force is applied on any vertex going under $z = 0.0$.

Of course, we implemented the gravity, which is simply the following formula:

$$F_{gravity} = (0.0, 0.0, -g) \tag{3}$$

Finally, when integrating the positions, the exterior force was obtained by adding up the two terms:

$$F_{ext} = F_{gravity} + F_{reaction} \tag{4}$$

4 LIMITS AND RESULTS

We illustrated our implementation through several videos. Let's present what you can find in those videos, and how they are representative of our results and their limitations.

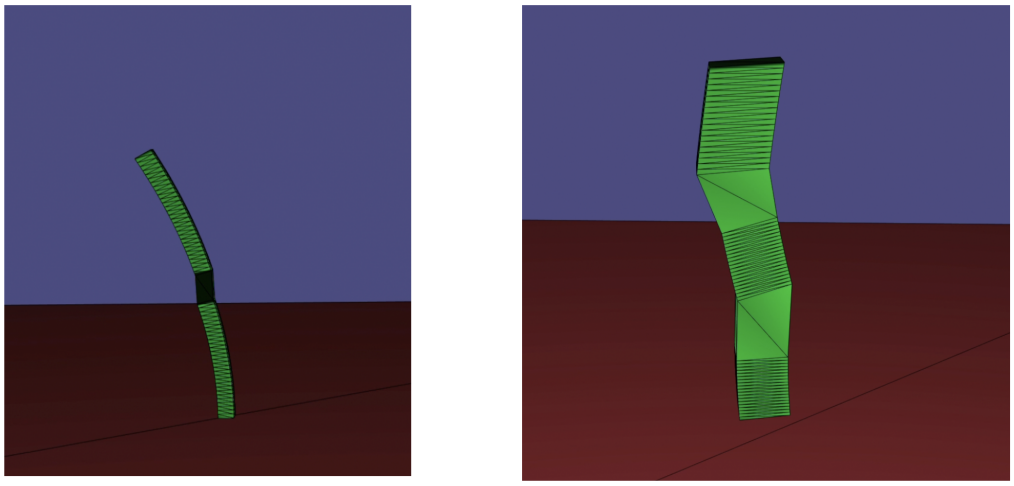

Figure 2: Objects crushing on the ground, and deforming.

The first video in the demo shows three different shapes falling and hitting the ground, deforming in accordance with the shock. The three different shapes show the range of reaction and deformation allowed by our model. For the video, we used the amortization and the quadratic deformation in order to have a more realistic rendering. We see in the video that the computed deformations are quite close to those showed in the video from the paper we studied, which is already a good achievement. Furthermore, the video is fluid, what shows that the computation is fast, with an easy enough complexity for the computer to give a real time rendering. However, few limitations are visible. The realism of the method is quite good, but stays quite far from the reality, especially for objects as complex as heads, which have a wide diversity of hardness all over its surface. In addition, the mesh-ground interaction shows a few liabilities; the evolution of the positions is sometimes unrealistic, the objects are sometimes going under the ground. We could possibly improve our model for the $F_{reaction}$ to solve this problem.

This also aims to show the same object, a rabbit, but with more or less extensions. Thanks to this video, we can compare the interest of quadratic deformations, plasticity, or amortization. The utility of a method must be seen as a way of judging as much the realism added thanks to the extension as the creativity it brings to a person that wants to represent an elastic shape.

-The amortization makes the deformation more realistic as the object finds its original shape again in the end.

-The interest of the quadratic is more theoretical, because it simply allows a shape to extend the range of its possible deformation to fit any goal shape. If the goal shape is quite close to the actual shape, the linear deformations will be enough, contrary to the situation where bending or twisting are needed. This video shows how quadratic deformations extend the range of deformations allowed by the algorithm.

-The plasticity is not hard to compute, but its interest is hard to get. We hardly see a difference compared to the basic elastic-shape animation.

The second video proposes a result for the cluster-based deformation presented earlier. The deformation is not as well done as in the paper, but the result is still quite convincing, and could be easily generalized to any shape. As for the scientific paper, the code for the cluster based method is way more complex than the initial model, what reduces the performance of the computation. We finally didn't achieve to code the overlapping points; the difficulty is that when you are calculating the new positions for a cluster, you need to have the positions at the time t ti give them at the time $t + dt$. The thing is that if you already updated the overlapping points from the previous cluster, you don't have all he positions you need. Unluckily, the different storage solutions we tried did not work.

Several thing we didn't have the time implement could be explored to go further :

We didn't see the influence of the mass matrix over the elasticity. Even though we implement it, we always set it to one. The mass could be at the origin of some particular shape movement. Then, we neither had the time to model the interactions between multiple meshes, as we saw in complex scenarios presented in the paper. This would be a very interesting but also a very difficult problem to optimize, as many positions would need to be calculated at the same time, what is not always possible.

5 CONCLUSION

To conclude, we hope to have shown in this report and in our demo that we have implemented almost all the techniques proposed in the article, with more or less success. Above all, we hope to have proved that we have understood the functioning and the aims of these different techniques. More time would be needed to reach the complexity of the scenarios proposed in the paper, but this project was a good first approach to animation for elastic shape deformation.