CGGS-CW2 Report

s2150204

Section 1-2 Observations

This report evaluates the performance of linear FEM for soft-body simulation, where deformations arise solely from initial velocity impulses without external forces. All scenes that are referred to can be found in `<submission>/videos/<scene>.mp4`.

In the bunny scene, a model with low initial deformation behaves as expected. The overall deformation remains minimal, with only a slight oscillation of the ears until the system stabilizes. This result confirms that linear FEM works well when deformations are small and that no significant artifacts occur under these conditions.

The cube scene tests two cubes with different material stiffness under medium initial deformation. The compliant cube deforms more noticeably than the stiff one, and although both resolve their deformations, a slight rotation, flattening, and expansion are observed afterward. These outcomes align with predictions: while linear FEM resolves the deformation, its linear approximation introduces predictable artifacts in moderate deformation scenarios.

In the epcot scene, where a high initial deformation is applied, the object flattens and expands as it relaxes. Despite the pronounced deformation, the method successfully brings the system to a resolved state. However, the observed flattening and expansion confirm that linear FEM struggles to accurately capture the behavior under large deformations.

The fertility scene features two instances of the same mesh with differing initial deformations. As anticipated, the instance with high initial deformation exhibits significant early distortion, whereas the low-deformation instance remains relatively stable. Both eventually settle into a resolved configuration, demonstrating that the initial conditions strongly influence the simulation outcome.

A newly added torsion scene further illustrates the method's behavior. In this scene, a box is subjected to a torsional impulse that causes it to twist. As a result, the box expands along the axis of rotation. This expansion is a predictable artifact of the linear approximation when handling torsional loads, reinforcing the observation that, while linear FEM is efficient and stable, it may not fully capture the complex behavior associated with rotational deformations.

In conclusion, these simulations confirm that linear FEM is computationally efficient and produces stable, plausible results for small to moderate deformations. However, for larger deformations, predictable artifacts such as rotation, flattening, and expansion emerge. These effects are inherent in the linear approximation, suggesting that alternative approaches like the corotational method should be considered when higher accuracy is required under large deformations.

Extension

Implementation

My implementation of corotational elements consisted of a tweak to the `create_element_stiffness` function I had previously implemented, and a tweak to the `update_scene` function. Starting with the scene class, when the scene is updated the large scene FEM matrix is reinitialised, and then the solver is factorised. This follows almost the exact same steps as the original scene file, but without initialisation steps. Then, in the mesh class, the code shown in Listing 1 is added after K_e is initialised. This directly implements the lecture slides into code, and then transforms each K_e individually.

```
[..]
138   // ======= Corotational Element ========
139
```

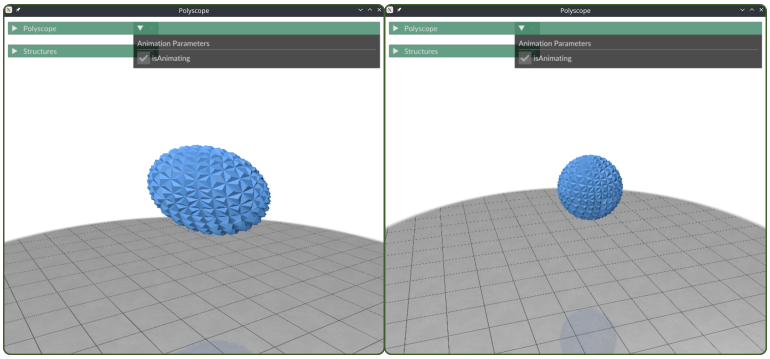
```
140   // Construct stacked matrices P and Q
141   Matrix<double,3,3> P, Q;
142
143   for (int i = 0; i < 3; i++) {
144     // Original undeformed edges
145     Vector3d x0 = origPositions.segment<3>(3 * tet(0));   // x1
146     Vector3d xi = origPositions.segment<3>(3 * tet(i+1)); // xn
147     P.col(i) = x0 - xi;
148
149     // Current deformed edges
150     x0 = currPositions.segment<3>(3 * tet(0));   // x1'
151     xi = currPositions.segment<3>(3 * tet(i+1)); // xn'
152     Q.col(i) = x0 - xi;
153   }
154
155   // Compute rotation matrix S and use SVD to get R
156   Matrix3d S = P.transpose() * Q;
157   JacobiSVD<Matrix3d> svd(S, ComputeFullU | ComputeFullV);
158   Matrix3d R = svd.matrixV() * svd.matrixU().transpose();
159
160   // Ensure R is a rotation (det(R) positive)
161   if (R.determinant() < 0.0) {
162     Matrix3d V = svd.matrixV();
163     V.col(2) *= -1.0;
164     R = V * svd.matrixU().transpose();
165   }
166
167   // Construct the larger rotational element so it applies it to all
168   //  vertices within the tet
169   Matrix<double, 12, 12> Re = Matrix<double, 12, 12>::Zero();
170   for (int i = 0; i < 4; ++i) {
171     Re.block<3,3>(3*i, 3*i) = R;
172   }
173   // Corotational stiffness matrix (from lecture slides: K' = R*K*R^-1)
174   // R tranpose is the same as the inverse
175   Matrix<double, 12, 12> Ke_corot = Re * Ke_linear * Re.transpose();
176
177   return Ke_corot.sparseView();
[..]
```

Listing 1: Source file: `mesh_section3.h`, Function: `create_element_stiffness()`

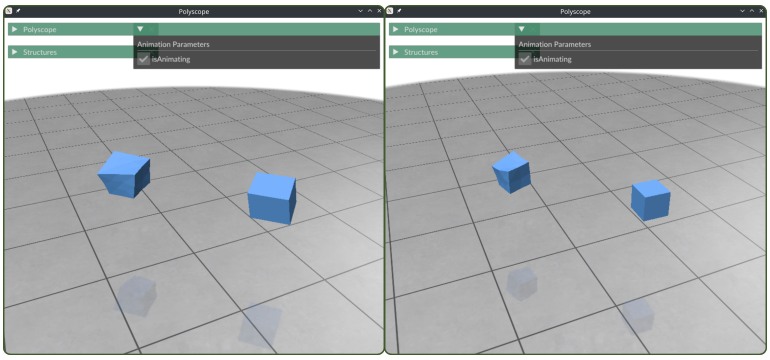
Observations

We can see the benefits afforded by the co rotational element in a variety of scenes. Figure 1 shows a sequence of comparisons before and after the co rational element was added. Figure 1a shows the difference at frame 25 of the Epcot scene before and after the co rotational element was added. Figure 1b shows a similar difference at frame 25 the cube86 scene. In both scenes the objects initially deform and then slowly tilt and expand outwards in the original implementation. In the updated implementation, the objects instead deform and then resolve correctly. This is due to rotations being present in the deformation. When the linear system is used, it cannot account for rotational deformation, so the inconsistencies with the deformation add up. This leads to artifacts such as these. In comparison, when rotational elements are taken into account, these scenes correctly deform, and resolve correctly.

Despite these benefits, the improvements come with a marked cost. The performance on most scene falls by about x% after the co rotational element is added. This is due to the solver being recomputed each frame, which causes each frame to take more processing power to resolve. Moreover, there are still artifacts in some scenes. For example the fertility scene deforms massively with and without the co rotational element, leading to an unrealistic simulation. This indicates that improvements can still be made to this method of soft-body simulation to make it more realistic.



(a) Epcot scene with and without the corotational element. On the left is the linear system and on the right is the corotational system



(b) Cube86 scene with and without the corotational element. On the left is the linear system and on the right is the corotational system

Figure 1: Comparison with and without the corotational element