Assignment 6: Shape Optimization

Handout date: 11.05.2018
Submission deadline: 25.05.2018 at 09:00

In this exercise you will build an interactive editor that uses shape optimization to help the user create objects that can stand. Inspiration for this task was drawn from [1]. A simple way to tell if an object will fall over or not, is to check if its center of mass projected onto the ground lies within the object's support polygon. Assuming a flat ground, gravity in y direction and a two-dimensional space, the following inequality must hold:

$$x_{\mathsf{s,min}} \leq x_{\mathsf{COM}} \leq x_{\mathsf{s,max}} \tag{1}$$

where x_{COM} is the x-coordinate of the center of mass, $x_{\mathsf{s,min}}$ and $x_{\mathsf{s,max}}$ are the minimum and maximum of the support of the object. With m_i being the mass of a node and $M = \sum_i m_i$ being the total mass respectively, the center of mass is defined as $\mathbf{x}_{\mathsf{COM}} = \frac{1}{M}\sum_i m_i \mathbf{x}_i$.

We now want to create a "Make it Stand"-editor, where the user can load a 2D object and specify the region on which the object should stand on. The editor will then deform the object in order to make it stand on the specified region. We can describe this task as a constrained optimization problem:

$$\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{X}) \text{ s.t. } x_{\mathsf{s,min}} \leq x_{\mathsf{COM}}(\mathbf{x}) \leq x_{\mathsf{s,max}} \tag{2}$$

where $\mathbf{x}$ and $\mathbf{X}$ are the node positions of the deformed and undeformed mesh, and $E(\mathbf{x}, \mathbf{X})$ is some measure of deformation.

1. Sequential Quadratic Programming

To solve the above problem, we will use the Sequential Quadratic Programming method. Let's first try it out on a simpler problem:

$$\min_{x_1, x_2} x_1^2 + x_2^2 \text{ s.t. } x_1 + 2x_2 = 3 \tag{3}$$

Implement the above problem using the `SQPMinimizer`. Derive from `ObjectiveFunction` and `FunctionConstraints` to implement the objective function and the equality constraint. Test it using `test_SQP.cpp`.

Report. Report the output of `test_SQP`. Explain the number of iterations it took to converge.

2. "Make It Stand" - Editor

Now that you know how to implement constraints, we can make the interactive editor. As the deformation measure $E(\mathbf{x}, \mathbf{X})$ use the Neo-Hookean deformation energy from the previous assignement.

2.1. **Equality Constraint.** In this exercise, we want to maximize the stability and thus we want the center of mass to be exactly in the middle of the support. In addition, we need constraints that keep the support nodes $\mathcal{T}$ in place. To allow the user to change the shape of the object while still satisfying the stability constraints, we will add the possibility to constrain user-selected nodes $\mathcal{T}$ to a target position. In summary, the optimization problem is:

$$\begin{aligned} &\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{X}) \\ &\text{s.t. } x_{\text{COM}}(\mathbf{x}) = \frac{1}{2}(x_{\text{s,max}} + x_{\text{s,min}}) \text{ and } \mathbf{x}_i = \mathbf{x}_{i,\text{target}} \forall i \in \mathcal{S} \cup \mathcal{T} \end{aligned} \tag{4}$$

Solve the above constrained optimization problem in the function SimulationMesh::solveShape using the SQP minimizer.

In the class SimulationMesh, the set supportNodes contains all the support nodes ($= \mathcal{S}$) and the constraintNodes maps the index of node i to its target (support) position. constraintNodes contains the nodes of both sets $\mathcal{S}$ and $\mathcal{T}$.

When the option "Set Support" is active in the combo box "Mouse Mode", support nodes can be added by clicking on nodes and removed while holding the CTRL key. If "Deform Handles" is selected, nodes are added to constraintNodes, but not to the set supportNodes.

Report. Show a screenshot of `data/woody-lo.off` balancing on one of its feet.

2.2. **Inequality Constraint.** For the object to stand, the center of mass doesn't have to be exactly in the middle of the support. It's enough if the center of mass projected onto the ground lies within the support region:

$$\begin{aligned} &\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{X}) \\ &\text{s.t. } x_{\text{s,min}} \leq x_{\text{COM}} \leq x_{\text{s,max}} \text{ and } \mathbf{x}_i = \mathbf{x}_{i,s} \forall i \in \mathcal{S} \cup \mathcal{T} \end{aligned} \tag{5}$$

Solve the above constrained optimization problem in the function SimulationMesh::solveShape using the SQP minimizer.

Depending on the selected option in the "CoM Constraint" combo box, the corresponding type of constraint shall be used. In the function SimulationMesh::solveShape, make sure the correct set of constraints is used: If COMconstraintType==0 use the constraints from 2.1, and if COMconstraintType==1 the constraints from this exercise (2.2).

Report. Show a screenshot of `data/woody-lo.off` balancing on one of its feet.

2.3. **Make the most amazing balancing object! (Bonus).** Play around with the editor and create the most amazing balancing object. There will be a vote in the lecture and the object with the most votes will be printed - it better not fall!

Feel free to change the Neo-Hookean material properties, load in different meshes or even add your own new type of handle.

Report. A screenshot and the mesh in `.off` format.

References

[1] Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. Make It Stand: Balancing shapes for 3D fabrication. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 32(4):81:1–81:10, 2013.