Comparison Supplement to Incremental Potential Contact: Intersection- and Inversion-free, Large-Deformation Dynamics

MINCHEN LI, University of Pennsylvania & Adobe Research
ZACHARY FERGUSON and TESEO SCHNEIDER, New York University
TIMOTHY LANGLOIS, Adobe Research
DENIS ZORIN and DANIELE PANOZZO, New York University
CHENFANFU JIANG, University of Pennsylvania
DANNY M. KAUFMAN, Adobe Research

ACM Reference Format:
Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Comparison Supplement to Incremental Potential Contact: Intersection- and Inversion-free, Large-Deformation Dynamics. *ACM Trans. Graph.* 39, 4, Article 49 (July 2020), 7 pages. https://doi.org/10.1145/3386569.3392425

1 COMSOL

Scene Setup. We set up a simple scene to illustrate issues we encounter for contact modeling in COMSOL. See Figures 1 top and middle, for our initial positions where we have, from left to right, a fixed rectangle (0.2×0.7m), two free squares (0.5×0.5m), and a moving rectangle (0.2×0.7m) with symmetric boundary conditions in the y-direction. We impose displacement boundary conditions on the right-most rectangle of -0.6m, use densities of 1000kg/m^3 and a linear material model with $E = 1000$Pa and $\nu = 0.4$. We experimented with different discretizations and applications of the boundary conditions. With a quad model we apply the boundary conditions incrementally ($-0.6s$) with s going from zero to one, and steps of size 0.01, while with a triangle mesh we applied steps of 0.005. The first four figures in Figure 1 show respectively the initial configurations and the solutions for these two experiments, the last figure shows increased failure if we simply apply the boundary conditions directly without incremental loading.

Discussion. In COMSOL, we initially attempted to simulate dynamic contacts. However, the contact formulation is documented as being strictly valid only for stationary problems [**?**, Time-Dependent Contact Analysis, Page 199]. We then attempted to solve a simpler stationary problem. We take two cubes compressed by two rectangle, again, as above, one fixed, and one displaced. After an extensive correspondence with COMSOL technical support (see Additional Material), we were told to apply a few necessary "tricks": (i) since the problem is symmetric, it is better to cut it in half and impose symmetry boundary conditions; (ii) the mesh on the destination object should be twice as fine as the mesh on the source object along all contact interface boundaries; (iii) the scene should be designed so that there are no initial gaps between the objects that are in contact (no changing contact sets); (iv) the system should be solved applying an auxiliary sweep with small parametric steps, and so apply the displacement incrementally. Applying all of these additional customized tricks we were able to solve the simulation with both COMSOL's penalty and augmented Lagrangian options. Our experiments, agreeing with COMSOL technical support, show that the number of parametric steps that must be applied depend on the mesh size, with different values (larger or smaller) leading to reasonable solutions or not that must be discovered per example. For both simulations penetration is expected, more, we observe for the penalty method, as intersection is how contact pressure develops in the solver. Additionally, we learned that it is never a good idea to have corners being a part of a contact boundaries, as the corners will be numerically singular and cause penetration between the contact boundaries. Thus, according to COMSOL guidelines[1] corners should be filleted.

Authors' addresses: Minchen Li, University of Pennsylvania & Adobe Research, minchernl@gmail.com; Zachary Ferguson; Teseo Schneider, New York University; Timothy Langlois, Adobe Research; Denis Zorin; Daniele Panozzo, New York University; Chenfanfu Jiang, University of Pennsylvania; Danny M. Kaufman, Adobe Research, dannykaufman@gmail.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2020 Association for Computing Machinery.
0730-0301/2020/7-ART49 $15.00
https://doi.org/10.1145/3386569.3392425

2 ANSYS

Scene Setup. We again set up two simple problems to test contact modeling in ANSYS. For the first example we generate a "C"-shaped mesh (formed from 5 $10 \times 10 \times 10$mm cubes) supported on the bottom and a dropped cube ($10 \times 10 \times 10$mm) subject to gravity 981mm/s^2. For the second example we placed a large cube ($14 \times 14 \times 14$mm) with fixed support on the bottom and placed a smaller cube on top ($10 \times 10 \times 10$mm), beveled at 1mm; both were subject to gravity of 100mm/s^2. Both set ups apply densities of 10^{-6}kg/mm^3 and neo-Hookean material with $E = 0.01$MPa and $\nu = 0.4$. Figure 2 shows the two set ups and the resulting simulation failures. Note for the second experiment the spacing between the two objects is 1mm, from the plot on the bottom we see that the top, falling cube has a maximum displacement of more than 1.4mm for an implicit time step, and 1.02mm for an explicit time step.

Discussion. ANSYS provides explicit (through its explicit dynamics modules) and implicit (through its transient structural module)

[1] https://www.comsol.com/support/knowledgebase/1102

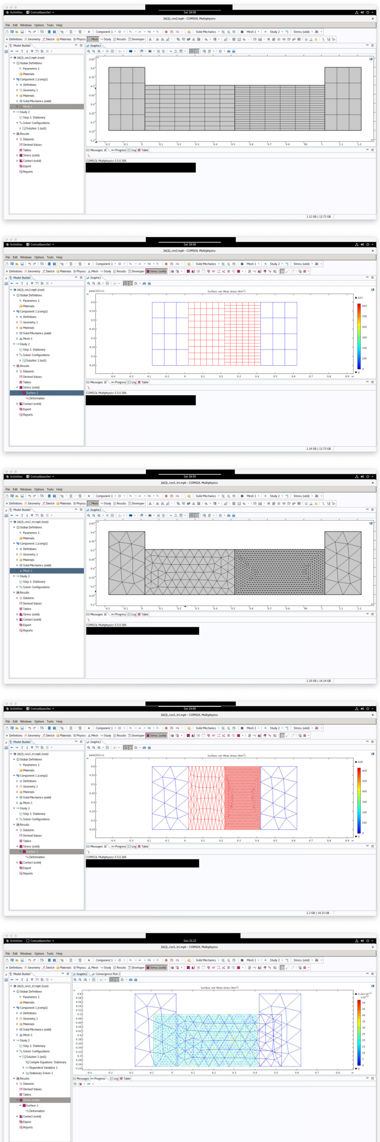


Fig. 1. Experiment setup (first and third figure) and results (second and fourth figure) of our COMSOL experiment. Disabling the incremental application of the boundary conditions leads to massive penetrations (last figure).

numerical time-integration options. The two models are fundamentally different in the way they handle collisions, but they both allow for small inter-penetrations. The explicit module is mostly automatic both in how it detects possible contact pairs and then resolves them. The results of the simulations are reasonable, although they do contain some small penetrations. The implicit module, is more complex, requires identifying contact pairs and much hand- tweaking of many exposed parameters in order to gain a successful simulation. This is true even for a simple scenario where two cubes are colliding. Here the default parameters do not produce a reasonable solution.

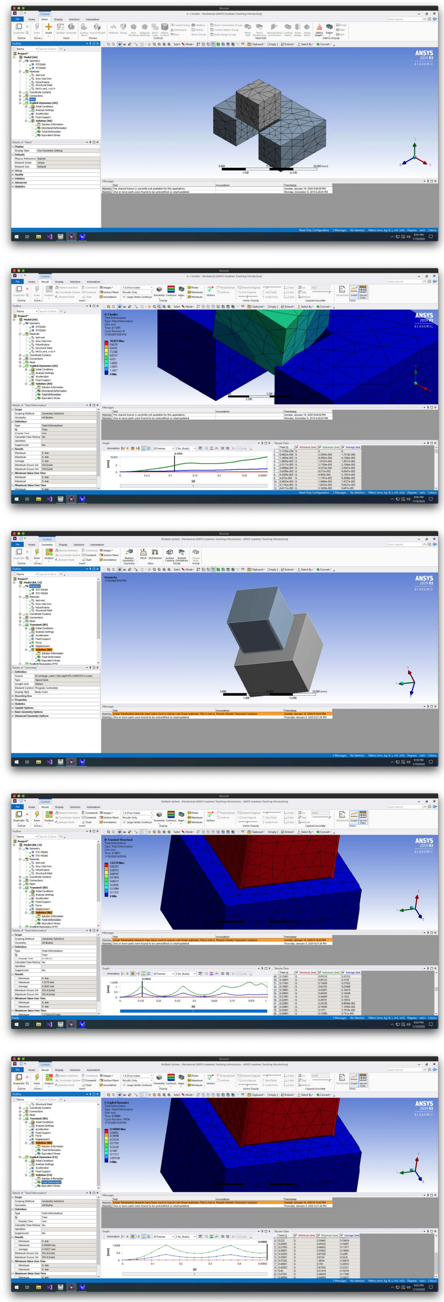


Fig. 2. First and second Ansys experiment (first and third image). All experiments all have inter-penetration. The second-to-last image applies implicit time-stepping, and the last one explicit time-stepping.

For such simple scenarios ANSYS technical support advised us to apply explicit dynamics and did not provide us with a reasonable set of parameters to fix the implicit simulation.

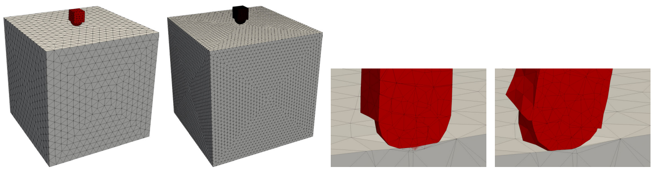


Fig. 3. Utopia setup coarse and fine mesh (first two image). Closeup on the two failures (third image has self-penetration because of large time step and fourth image has oscillations that prevent the solver the continue) for the coarse mesh.

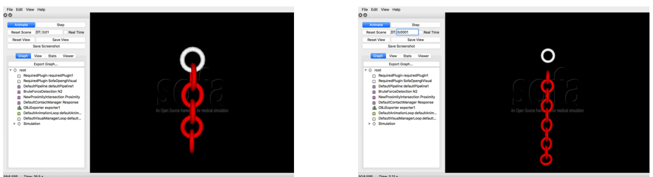


Fig. 4. Left: SOFA is able to simulate a five link even with large timesteps. Right: As we add more links, decreasing time steps down to 10^{-4}s, we are unable to find a time step that will not break the chain.

3 UTOPIA [?]

Scene Setup. To investigate Utopia we set up a simulation with a large $10 \times 10 \times 10$m block with fixed displacement on the bottom and a small cube $1 \times 1 \times 1$m with a half-sphere on the bottom and prescribed -1.5N force on the top flat face. See Figure 3. Both objects have the same neo-Hookean material parameters: density 1kg/m^3, $\lambda = 10$Pa, $\mu = 10$Pa. in Figure 3 we show the initial set up and the resulting penetration and oscillations for the coarse mesh version. Note that oscillations disappear for the fine mesh and penetrations decrease as we go to smaller time steps. We tested time steps of 0.5, 0.05, 0.01, and 0.001s.

Discussion. Utopia applies a parallel mortar algorithm for FE contact modeling. Like several other engineering FEM codes, UTOPIA requires a priori manual marking of all possible paired mesh boundary regions that can be treated for contact resolution. Such marking could be automated, see e.g, Yang and Laursen [?]). Note that even with automated pairing the mortar approach does not yet extend for more general collisions in regions that would involve more than two distinct surface regions. We tested the UTOPIA implementation [?] with a simple didactic experiment in which we drop an elastic sphere under gravity upon an elastic cube with its bottom DOF fixed. At larger time steps, we observe large scale volume intersections. These contact errors then reduce but do not disappear as time step decreases. These issues are not surprising as mortar methods enforce contact constraints in a weak sense and so allow intersections.

4 SOFA

We modify the the chain example provided with SOFA, replacing the ring mesh with simple, low-resolution tori (~500 tetrahedra). Table 1 shows the material settings used. Changing any of these

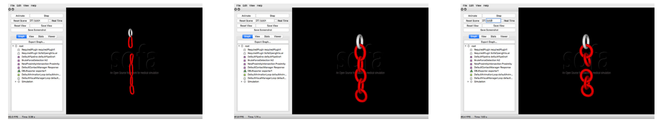


Fig. 5. We test SOFA on a five link chain with three varying stiffnesses, all with a fixed timestep of 10^{-3}s. Left: Young's Modulus of 100 fails, Middle: Young's Modulus of 1000 works, Right: Young's Modulus of 10000 fails.

parameters can result in different levels of failure (see Figure 5 for an example of changing material stiffness).

Elasticity Model	linear
Young's Modulus	1000
Poisson's Ratio	0.3
Mass	5.0

Table 1. SOFA FEM parameter values

5 HOUDINI

We test Houdini with the simple chain example. Each link is a torus with outer radius 1.25 and inner radius 0.28. The links are duplicated with a transform downwards of 1.62 and a rotation of 90 degrees. One ring at the top is added as a static surface collider. Parameter values that were changed from default Houdini parameters for the FEM solver are shown in Table 2 and values for the vellum solver are shown in Table 3. The scene was set to 24 FPS.

Damping Ratio	0.5
Shape Stiffness	387
Volume Stiffness	384
Repulsion	1e11
Friction	0.1
Substeps	14
Collision Passes	16

Table 2. Houdini FEM parameter values

6 SQP BENCHMARK

The table at the bottom of this document summarizes the results of our benchmark comparison of different SQP-type contact handling methods. For each scene we vary the time step size, constraint offset, constraint type, and solve the problem as both a fully nonlinear problem at each time step and a quadratic approximation (Linearized elasticity) of the energy with nonlinear constraints. See our main paper for details.

Additionally, we also initially tested two active-set update strategies: the first being to clear the active set and rebuild it upon every iteration and the second, applied successfully in recent methods [??] incrementally adds newly detected collisions to the active set.

Cloth Node		Struts Node		Solver Node	
Mass	Calculate Varying	Direction Jitter	2	Substeps	40
Thickness	Calculate Uniform	Constraints Per Point	80		
Stretch Stiffness	1000000	Stretch Stiffness	1e-9		
Bend Stiffness	1e-4	Compression Stiffness	10000		

Table 3. Houdini Vellum parameter values that were changed from default

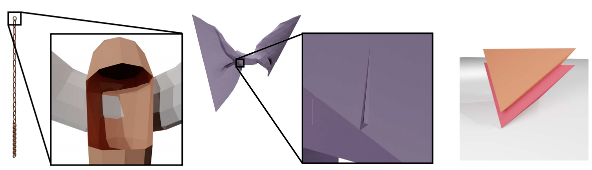


Fig. 6. **Intersections.** Severity of intersections affect usability of simulation results. Left: Examining a chain drop the results look correct until we look inside the top ring. It is clear the fixed ring has started to intersect the top FEM ring. These small intersections can build in to larger constraint drift and eventually pass-through of the rings. Middle: At first glance the results of this mat twist appear fine, but when we look closely there are several small intersections hidden in the folds of the mat (22 in total). Right: An extreme case of intersections where the two tetrahedron meshes have almost completely overlapped.

We found that the latter performed significantly better across our benchmark so we restrict our results here to the latter.

In the table we document results of each simulation as either: 1) intersecting: the simulation fails with some amount of intersection (see Figure 6 for examples of minor and major intersections); 2) blow-up: the simulation blows up due to rapid growth in energy and/or displacement during the optimization (see Figure 7 for examples); 3) incomplete: the simulation is unable to complete in the given time limit of four hours, or 4) successful: the simulation successfully completes with none of the aforementioned failures. Note that IPC is the only method to succeed across all simulations.

6.1 Intersections

SQP-type optimizations can result in intersection for a variety of reasons, but they are commonly generated in cases where the time step is too large (leading to poor linearization of constraints) or the constraint offset is too small (allowing slight solver inaccuracies to generate drift and intersections). While possible in lucky cases, recovering from intersections is generally rare on it sown and of course challenging to fix algorithmically.

Not all intersections are alike, however. Minor intersections while not physical are commonly ignored as they do not affect visual quality. Figure 6 shows examples of both minor and egregious intersections.

6.2 Optimization blow-up

Optimization “blow-up” occurs when the objective energy or search direction of the optimization increases at alarming rates due to numerical difficulties. Studying the results of our sweep, we find that blow-ups more often occur with large time steps where a large number of constraints may be activated at once. This can create a difficult, if not numerically infeasible, optimization problems and likewise instability due to lack of line search.

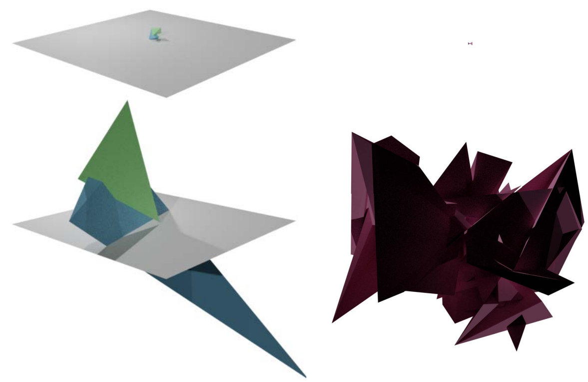


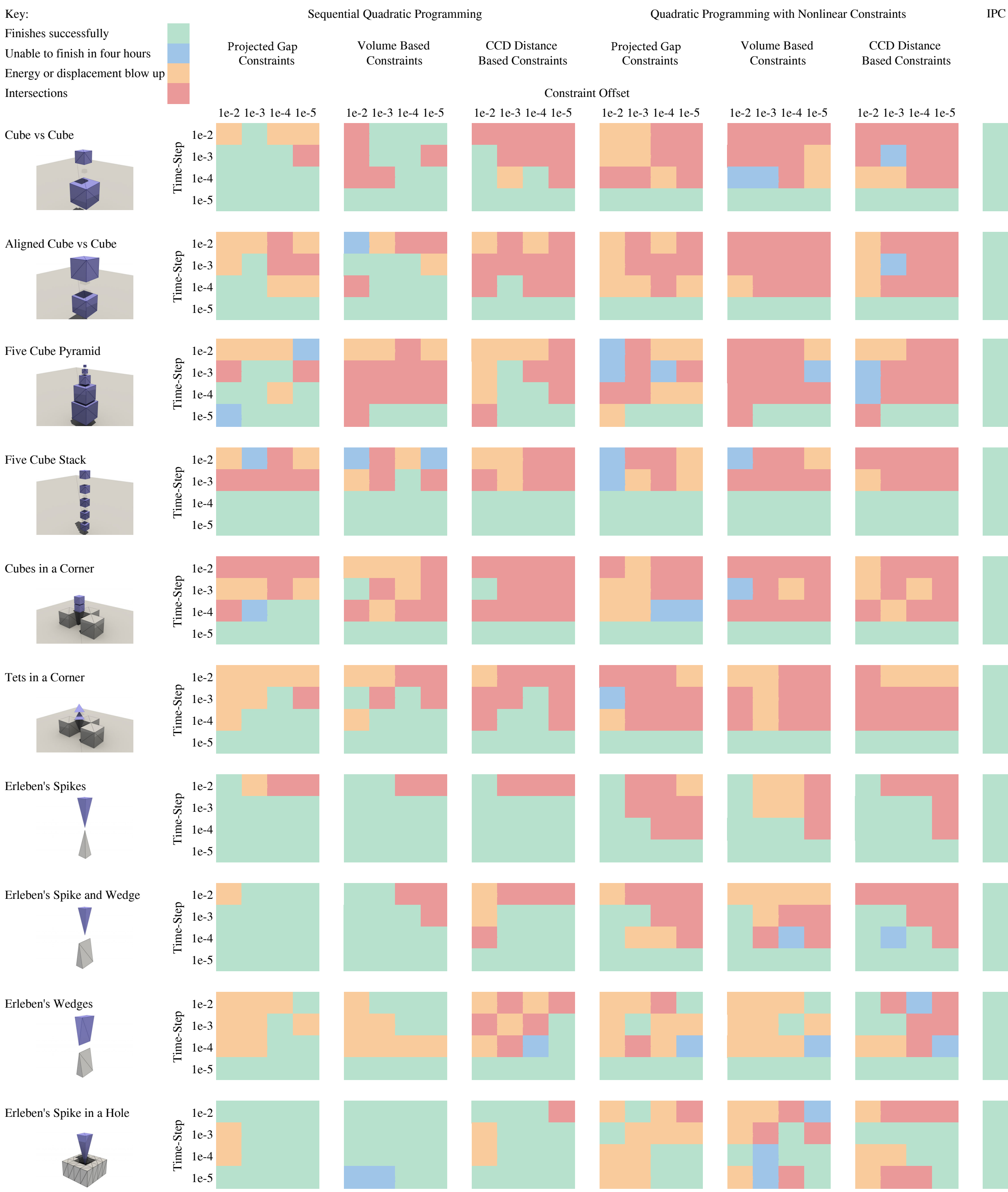
Fig. 7. **Explosions.** Top: Two different examples (left and right) with their time steps right before the SQP optimization starts to fail with increasingly large displacements. Bottom: An intermediate state for each example during the SQP optimization “blow-up”.

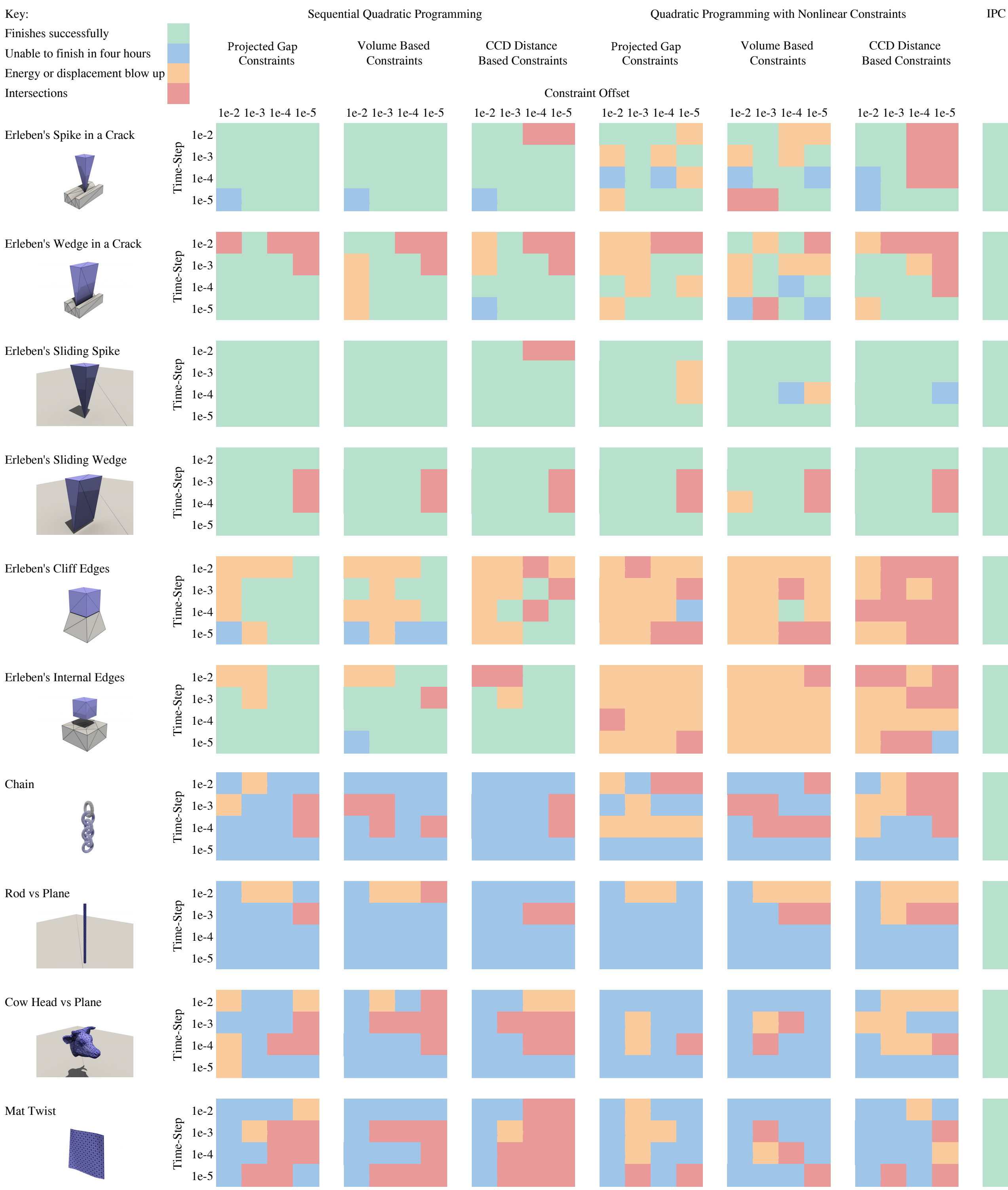
We also note the large amount of blow-ups when applying the single quadratic approximation per time step. While faster to optimize, this method is a poor approximation of the nonlinear elasticity energy and leads to increased energy values. Again, the problem is worsened in all cases by the lack of line-search to find a step length that sufficiently decreases the energy.

6.3 Poor convergence and subsequent timeout

In our benchmark, SQP-type methods struggle to handle moderately large scenes. We attribute this poor performance to 1) large numbers of constraints when the constraint offsets are enabled – this, in turn, can lead to infeasibility in the QP solve and so subsequent solve fails; and 2) oscillating optimization iterations largely resulting from lack of line-search. When the underlying QP solver fails, we choose to clear and build a new active set. While this helps in some cases, it often results in more QP fails and a stagnant optimization. Convergence could be improved with a line-search along the constrained search directions, but suitable merit-functions for line search here remain an open and ongoing area of research.


Key:
Finishes successfully
Unable to finish in four hours
Energy or displacement blow up
Intersections
Sequential Quadratic Programming
Quadratic Programming with Nonlinear Constraints
IPC
Projected Gap Constraints
Volume Based Constraints
CCD Distance Based Constraints
Projected Gap Constraints
Volume Based Constraints
CCD Distance Based Constraints
Constraint Offset
1e-2 1e-3 1e-4 1e-5
Time-Step
1e-2
1e-3
1e-4
1e-5
Mesh Point vs CO Triangle
Mesh Triangle vs CO Point
Mesh Edge vs CO Edge
Mesh Point vs Mesh Triangle
Mesh Edge vs Mesh Edge
Tet vs Tet
Cube vs Plane
Cube vs Cube CO
Tet vs Cube
Aligned Tet vs Cube


Key:
Finishes successfully
Unable to finish in four hours
Energy or displacement blow up
Intersections
Sequential Quadratic Programming
Quadratic Programming with Nonlinear Constraints
IPC
Projected Gap Constraints
Volume Based Constraints
CCD Distance Based Constraints
Projected Gap Constraints
Volume Based Constraints
CCD Distance Based Constraints
Constraint Offset
1e-2 1e-3 1e-4 1e-5
Time-Step
1e-2
1e-3
1e-4
1e-5
Cube vs Cube
Aligned Cube vs Cube
Five Cube Pyramid
Five Cube Stack
Cubes in a Corner
Tets in a Corner
Erleben's Spikes
Erleben's Spike and Wedge
Erleben's Wedges
Erleben's Spike in a Hole


Key:
Finishes successfully
Unable to finish in four hours
Energy or displacement blow up
Intersections
Sequential Quadratic Programming
Quadratic Programming with Nonlinear Constraints
IPC
Projected Gap Constraints
Volume Based Constraints
CCD Distance Based Constraints
Constraint Offset
1e-2 1e-3 1e-4 1e-5
Time-Step
Erleben's Spike in a Crack
Erleben's Wedge in a Crack
Erleben's Sliding Spike
Erleben's Sliding Wedge
Erleben's Cliff Edges
Erleben's Internal Edges
Chain
Rod vs Plane
Cow Head vs Plane
Mat Twist