Supplement to Guaranteed Globally Injective 3D Deformation Processing

YU FANG*, University of California, Los Angeles & University of Pennsylvania & Adobe Research
MINCHEN LI*, University of California, Los Angeles & University of Pennsylvania
CHENFANFU JIANG, University of California, Los Angeles & University of Pennsylvania
DANNY M. KAUFMAN, Adobe Research

ACM Reference Format:
Yu Fang*, Minchen Li*, Chenfanfu Jiang, and Danny M. Kaufman. 2021. Supplement to Guaranteed Globally Injective 3D Deformation Processing. *ACM Trans. Graph.* 40, 4, Article 75 (August 2021), 4 pages. https://doi.org/10.1145/3450626.3459757

1 IDP PSEUDO CODE

In this section we provide pseudo code for the core time-stepping and Newton solver components of IDP. Algorithm 1 covers the time stepping through a moving BC process while Algorithm 2 demonstrates the filtered projected Newton solver.

Algorithm 1 A Single Stage of IDP

```
procedure IDP(x^0, 𝓑, x̃, ε_d)
    [x̃_k^1, x̃_k^2, ...] ← ProcessBC(x^0, 𝓑, x̃)              ▹ Section 4.3
    x ← x^0
    for time step t do
        x ← BarrierAwareProjectedNewton(x, 𝓑, x̃^t, ε_d)
        Update plasticity                                  ▹ Section 4.5
    Remesh                                                 ▹ Section 4.5
```

Line-search filtering in Algorithm 2 applies continuous collision detection (CCD), when the IPC barrier is in the objective, in order to avoid intersections along descent steps taken. When barrier distortion energies are minimized, we also apply Smith and Schaefer's [2015] root-finding in the line-search filter to similarly prevent inversion in descent. At the end of each Newton iteration in Algorithm 2, we update the barrier stiffness κ and the BC penalty stiffness κ_B following Li et al. [2020].

Authors' addresses: Yu Fang*, University of California, Los Angeles & University of Pennsylvania & Adobe Research, squarefk@gmail.com; Minchen Li*, University of California, Los Angeles & University of Pennsylvania, minchernl@gmail.com; Chenfanfu Jiang, University of California, Los Angeles & University of Pennsylvania, chenfanfu.jiang@gmail.com; Danny M. Kaufman, Adobe Research, danny.kaufman@gmail.com.

0730-0301/2021/8-ART75 $15.00
https://doi.org/10.1145/3450626.3459757

Algorithm 2 Barrier Aware Projected Newton

procedure $\textsc{BarrierAwareProjectedNewton}(x^t, \mathcal{B}, \tilde{x}^t, \boldsymbol{\epsilon}_d)$
  $x \leftarrow x^t$
  $S_{\text{prev}} \leftarrow S_t(x)$
  $x_{\text{prev}} \leftarrow x$
  do
    $H \leftarrow \text{SPDProject}(\nabla_x^2 S_t(x))$
    $p \leftarrow -H^{-1}\nabla_x S_t(x)$
    // line search filtering:
    $\alpha \leftarrow \min(1, \text{StepSizeUpperBound}(x, p))$
    do
      $x \leftarrow x_{\text{prev}} + \alpha p$
      $\alpha \leftarrow \alpha/2$
    while $S_t(x) > S_{\text{prev}}$
    $S_{\text{prev}} \leftarrow S_t(x)$
    $x_{\text{prev}} \leftarrow x$
    Update κ_B and κ ▷ Li et al. [2020]
  while $\frac{1}{h}\|p\|_\infty > \boldsymbol{\epsilon}_d$
  return x

2 EXPERIMENT STATISTICS

Here we provide performance and set up details for all tests and applications. in Figure 1, 2, 3, 4, 5.

Examples	# nodes	# tets	E (Pa)	dhat (m)	contacts avg. (max.) per frame	CPU	avg. t (min), iters per frame	# frame
Bunny in hourglass (IDP)	19.1K, 4.5K	74.4K	1E+04	1E-03	66.2K (83.7K)	9700	6.5, 237.0	60
Bunny in hourglass (SCAF)	13.5K, 4.5K		/	/	/	9700	0.1, 1.0	77
Octocat in hourglass (IDP)	23.3K, 4.5K	89.3K	1E+04	1E-03	105.0K (121.3K)	8700K	1.3, 37.3	60
Octocat in hourglass (SCAF)	17.1K, 4.5K		/	/	/	8700K	9.3, 1.0	88
Tube in tube (IDP)	4.4K, 5.5K	18.8K	1E+05	1E-03	13.1K (24.4K)	8700K	0.2, 20.8	60
Tube in tube (SCAF)	2.8K, 5.5K		/	/	/	8700K	0.01, 1.0	57

Fig. 1. **Cages Examples.** Here all IDP examples use tetrahedral meshes.

Examples	# nodes	# tets	E (Pa)	PNTol	dhat (m)	contacts avg. (max.) per frame	CPU	avg. t (min), iters per frame	# frame
Bunny in box (IDP,SD)	78.9K	315.5K	1E+05	0.5	1E-03	62.2K (81.5K)	8700K	1.0, 13.1	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+05	0.5	1E-03	111.3K (142.3K)	8700K	1.2, 14.9	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+04	0.5	1E-03	57.5K (80.0K)	8700K	0.5, 6.7	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+04	0.5	1E-03	108.8K (144.4K)	8700K	0.7, 8.3	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+03	0.5	1E-03	22.3K (34.9K)	8700K	0.2, 2.8	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+03	0.5	1E-03	50.6K (72.9K)	8700K	0.4, 4.9	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+05	1	1E-03	57.2K (77.4K)	8700K	1.0, 11.5	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+05	1	1E-03	106.3K (137.9K)	8700K	1.2, 13.7	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+04	1	1E-03	45.5K (70.8K)	8700K	0.4, 4.6	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+04	1	1E-03	95.9K (132.4K)	8700K	0.6, 7.2	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+03	1	1E-03	9.6K (21.9K)	8700K	0.1, 1.4	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+03	1	1E-03	31.2K (46.8K)	8700K	0.3, 2.9	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+05	2	1E-03	47.0K (66.8K)	8700K	0.9, 10.3	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+05	2	1E-03	95.2K (127.3K)	8700K	1.2, 13.7	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+04	2	1E-03	24.4K (40.9K)	8700K	0.3, 3.4	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+04	2	1E-03	67.8K (98.4K)	8700K	0.5, 5.3	60
Bunny in box (IDP,SD)	78.9K	315.5K	1E+03	2	1E-03	3.3K (11.2K)	8700K	0.1, 1.0	60
Bunny in box (IDP,NH)	78.9K	315.5K	1E+03	2	1E-03	5.4K (15.2K)	8700K	0.1, 1.0	60
Bunny in box (IDP,SD)	19.1K	74.4K	1E+04	1	1E-03	11.1K (17.1K)	8700K	0.08, 4.5	60
Bunny in box (IDP,NH)	19.1K	74.4K	1E+04	1	1E-03	24.7K (35.0K)	8700K	0.1, 6.4	60
Bunny in box (IDP,SD)	4.6K	17.3K	1E+04	1	1E-03	3.5K (5.3K)	8700K	0.01, 4.1	60
Bunny in box (IDP,NH)	4.6K	17.3K	1E+04	1	1E-03	7.7K (11.4K)	8700K	0.02, 5.5	60
Bunny in box (IDP,SD)	0.9K	2.9K	1E+04	1	1E-03	1.2K (1.8K)	8700K	0.002, 2.8	60
Bunny in box (IDP,NH)	0.9K	2.9K	1E+04	1	1E-03	2.8K (4.1K)	8700K	0.003, 3.9	60
Bunny in box (SCAF, SD)	54.0K	\	\	\	\	\	8700K	0.73, 1	395
Bunny in box (SCAF, SD)	13.5K	\	\	\	\	\	8700K	0.12, 1	314
Bunny in box (SCAF, SD)	3K	\	\	\	\	\	8700K	0.02, 1	118
Bunny in box (SCAF, SD)	0.7K	\	\	\	\	\	8700K	0.002, 1	111

Fig. 2. **Bunny in box comparisons.** Here all IDP examples use tetrahedral meshes.

Examples	# nodes	E (Pa)	dhat (m)	contacts avg. (max.) per frame	CPU	avg. t (min), iters per frame	# frame
Dumpling Compress	4.9K	1E+05	1E-03	0.5K (4.7K)	8700K	6.3, 26.1	30
Dumpling Fold	14.4K	1E+05	1E-03	5.7K (10.8K)	8700K	0.6, 35.8	20
Dumpling Push	14.4K	1E+04	1E-03	5.4K (8.5K)	8700K	2.0, 64.5	20
Dumpling Pinch	14.4K	1E+04	1E-03	3.0K (21.0K)	8700K	1.4, 46.4	12
Twist armadillo	43K	1E+06	1E-03	0.3K (4.3K)	8700K	12.4, 28.6	100
Twist bar (Plastic + Barrier)	5.5K	1E+05	1E-03	2.0K (6.2K)	8700K	0.07,12.7	150
Twist bar (Barrier)	5.5K	1E+05	1E-03	1.7K (5.7K)	8700K	0.1, 22.6	150
Twist bar (-)	5.5K	1E+05	1E-03	/	8700K	0.02, 9.2	150
Rose	8.1K	1E+04	1E-03	0.6K (12.3K)	9900K	0.03, 5.4	120
Rhododendron	61.4K	1E+04	1E-03	775.0K (1569.5K)	7700K	2.0, 16.0	20
Justicia	76.3K	1E+04	1E-03	892.0K (1279.4K)	8700K	0.9, 6.9	113

Fig. 3. **Statistics of modeling examples** using tetrahedral meshes.

Examples	# nodes	# frames	contacts avg. (max.) per frame	Machine	avg. t (min), iters per frame
Brain	70.8K	8	42.8K (92.1K)	Mac	2.7, 35.8
Cat	7.4K	10	6.3K (7.5K)	Mac	0.04, 8.6
Hand	5.0K	3	5.4K (7.4K)	Mac	0.03, 9.7
Bunny	3.1K	50	5.1K (11.8K)	Mac	0.02, 7.2
Feline	10.3K	50	36.0K (52.7K)	Mac	0.06, 7.0
“Tao”	9.9K	10	1.7K (5.2K)	Mac	0.02, 5.7
“Peng”	9.2K	5	0.3K (1.0K)	Mac	0.01, 5.2
Delicious	28.3K	12	4.1K (12.5K)	Mac	0.05, 5.0
Seriously	40.1K	50	1.4K (2.7K)	Mac	0.02, 3.0

Fig. 4. **Normal flow examples** are performed with 3D triangle meshes.

Examples	# nodes	E (Pa)	dhat (m)	contacts avg. (max.) per frame	CPU	avg. t (min), iters per frame
Rumba	12.8K	1E+03	1E-02	49.7K (55.1K)	8700K	0.10, 7.1
Kick	12.8K	1E+03	1E-02	50.2K (52.4K)	8700K	0.13, 7.6
Break dancing	12.8K	1E+03	1E-02	49.7K (55.5K)	8700K	0.15, 9.4
Mousey	6.1K	1E+05	1E-03	1.5K (2.6K)	8700K	0.15, 29.1

Fig. 5. **Animation repair examples.** All repairs are performed on 3D triangle meshes with $\rho = 1000 kg/m^3$ and thickness $1cm$ to compute per element volume and mass weighting.