SIGGRAPH ASIA 2020 VIRTUAL

Pre-recorded sessions:
From 4 December 2020

Live sessions:
10 – 13 December 2020

SA2020.SIGGRAPH.ORG
#SIGGRAPHAsia | #SIGGRAPHAsia2020

P-Cloth: Interactive Cloth Simulation on Multi-GPU Systems using Dynamic Matrix Assembly and Pipelined Implicit Integrators

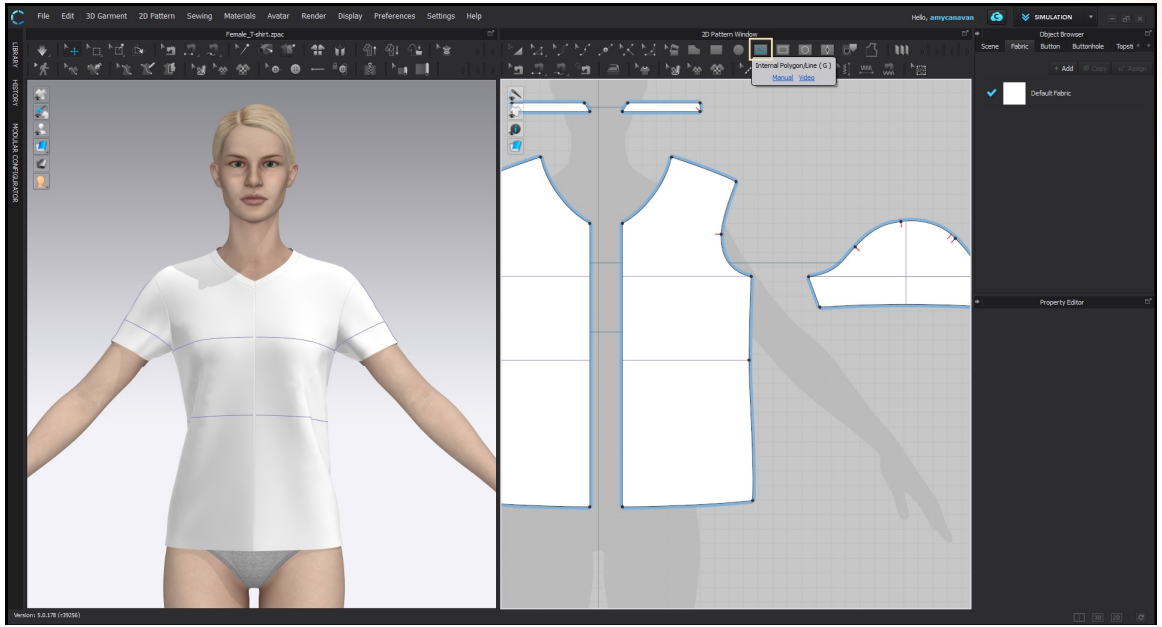
https://min-tang.github.io/home/PCloth/

Cheng Li[1], Min Tang[1], Ruofeng Tong[1], Ming Cai[1], Jieyi Zhao[3], and Dinesh Manocha[2]

[1]Zhejiang University, China
[2]University of Maryland at College Park, USA
[3]University of Texas Health Science Center at Houston, USA

SIGGRAPH ASIA 2020 VIRTUAL

Applications

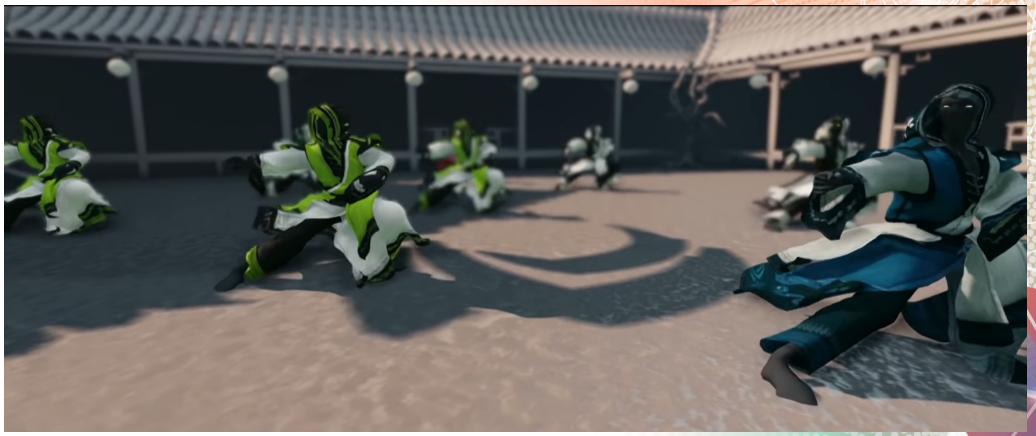


Garment Design

(CLO, Marvelous Designer)

SIGGRAPH ASIA 2020 VIRTUAL

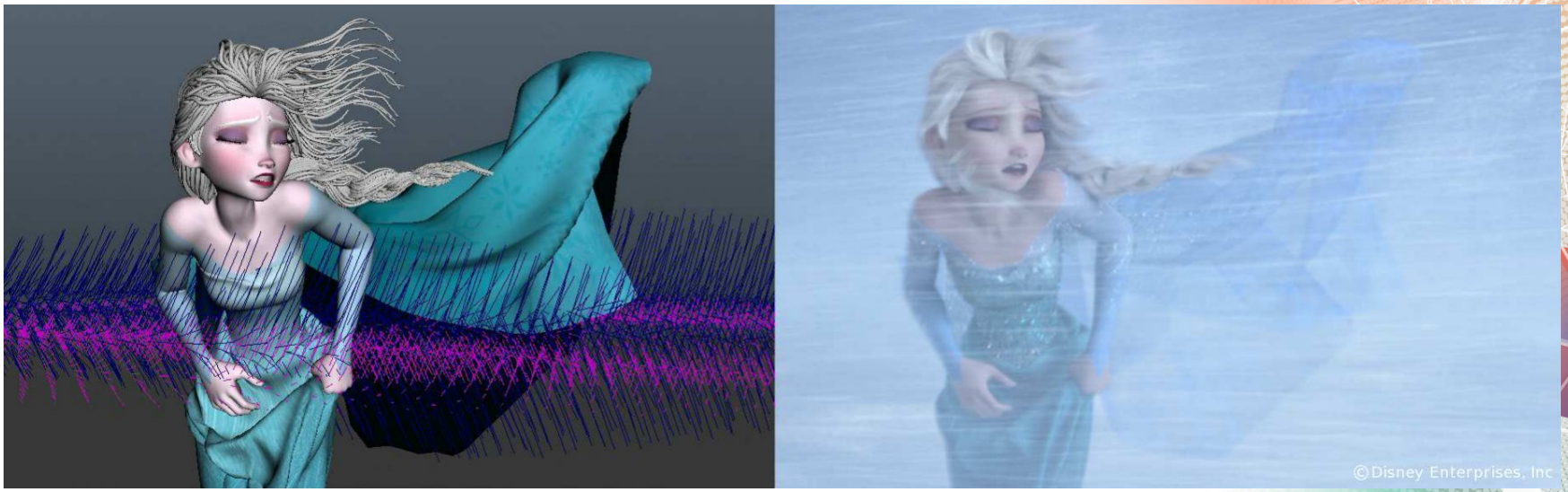
Applications



Video Games

SIGGRAPH ASIA 2020 VIRTUAL

Applications

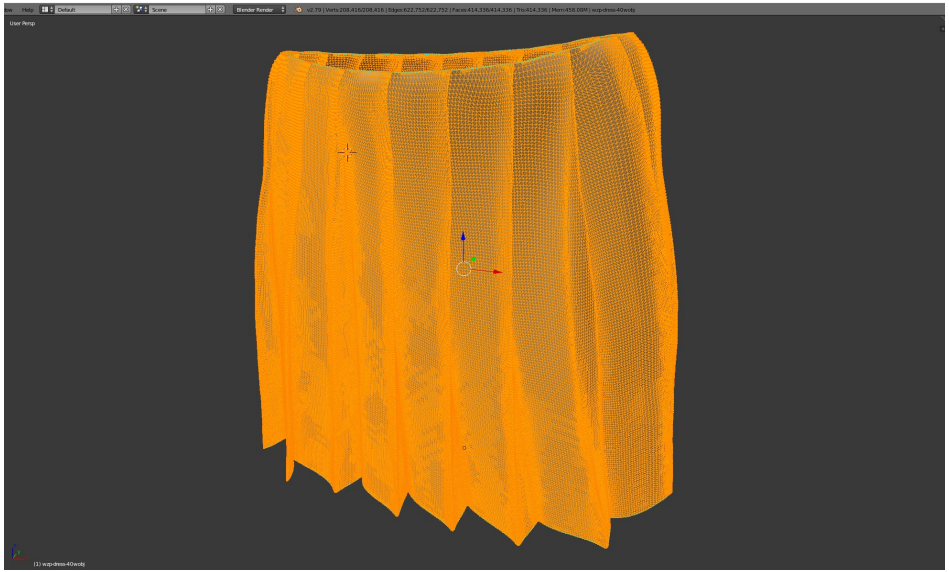


Movies

Challenges

- **Efficiency**

  High fidelity cloth simulation systems use cloth meshes with very high number of vertices.



SIGGRAPH ASIA 2020 VIRTUAL

Challenges

- **Robustness**

  Self-collisions and inter-object collisions could be extremely complex.

SIGGRAPH ASIA 2020 VIRTUAL

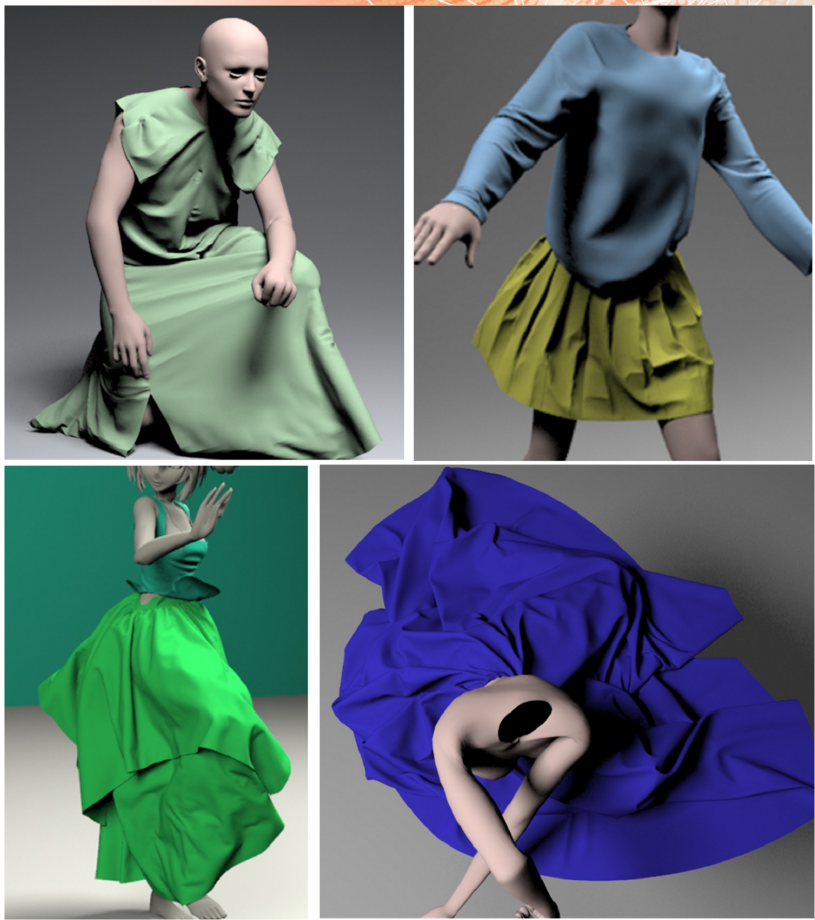
Challenges

- **Multi-GPU**
  - How to assign data and computations to multiple GPUs ?
  - How to reduce data transfers between multiple GPUs ?

SIGGRAPH ASIA 2020 VIRTUAL

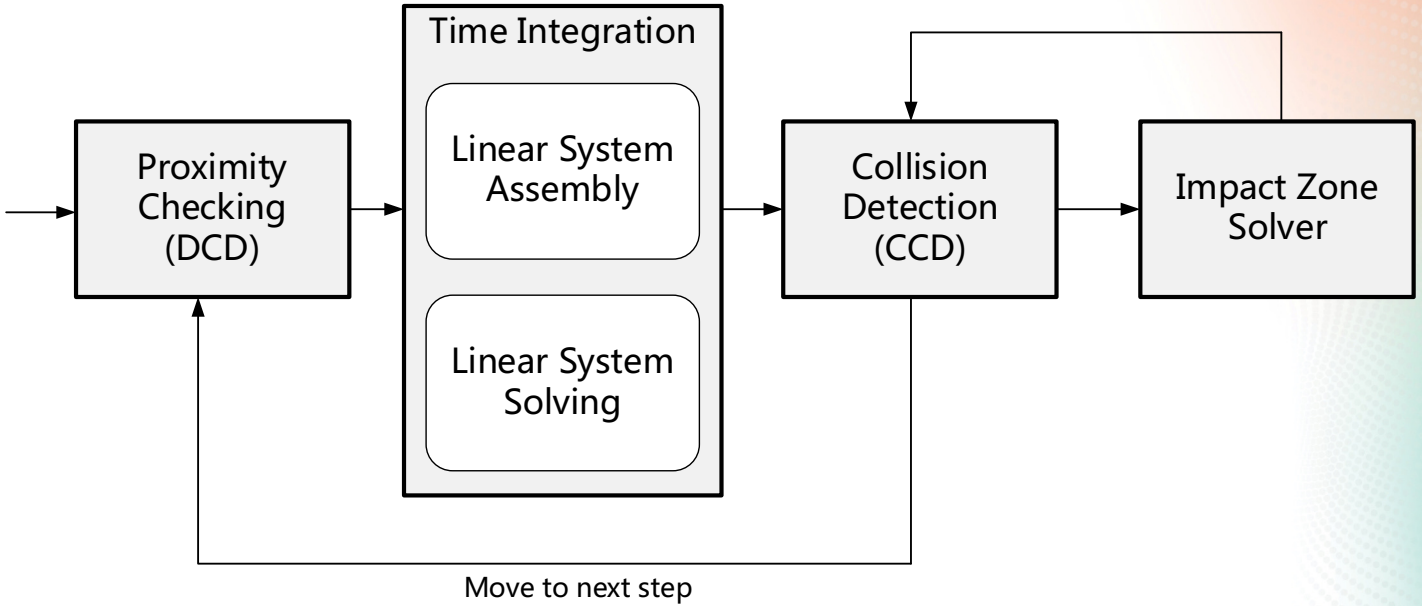
Main Results

- Parallel Multi-GPU cloth simulation
- Scalable parallel algorithms for matrix assembly, sparse linear system solving, and collision handling
- Applicable to cloth meshes with several million triangles on 4/8-GPU systems
- Almost interactive performance on 4 NVIDIA Titan Xp GPUs: 2-5fps
- Up to 8.23X speedups on 8 NVIDIA Titan V GPUs



Background

- **Overview of Cloth Simulation**



Background

- **Time Integration**

$$\mathbf{M}\ddot{\mathbf{u}} = \mathbf{f},$$

$$\mathbf{f}(\mathbf{u}_{t+\Delta t}) = \mathbf{f}(\mathbf{u}_t) + \mathbf{J} \cdot (\mathbf{u}_{t+\Delta t} - \mathbf{u}_t),$$

Considering the formulas:

$$(\mathbf{M} - \Delta t^2 \mathbf{J}) \cdot \Delta \mathbf{v} = \Delta t \mathbf{f}(\mathbf{u}_t + \Delta t \mathbf{v}_t),$$

SIGGRAPH ASIA 2020 VIRTUAL

Background

- **Collision Detection and Response**
  - Collision Detection

    Broad Phase:

    High-level culling using spatial acceleration data structure

    Narrow Phase:

    Intersection Test
  - Collision Response

    Nonlinear Optimization

SIGGRAPH ASIA 2020 VIRTUAL

Contributions

- Pipelined SpMV
- Dynamic Matrix Assembly
- Parallel Collision Handling

Pipelined SpMV

- **Recalling CG Solver**

1 **procedure modified-pcg**
2 $\Delta \mathbf{v} = \mathbf{z}$
3 $\delta_0 = \mathbf{filter}(\mathbf{b})^T \mathbf{P}\, \mathbf{filter}(\mathbf{b})$
4 $\mathbf{r} = \mathbf{filter}(\mathbf{b} - \mathbf{A}\Delta\mathbf{v})$
5 $\mathbf{c} = \mathbf{filter}(\mathbf{P}^{-1}\mathbf{r})$
6 $\delta_{new} = \mathbf{r}^T\mathbf{c}$
7 **while** $\delta_{new} > \epsilon^2\delta_0$
8 $\quad \mathbf{q} = \mathbf{filter}(\mathbf{Ac})$
9 $\quad \alpha = \delta_{new}/(\mathbf{c}^T\mathbf{q})$
10 $\quad \Delta\mathbf{v} = \Delta\mathbf{v} + \alpha\mathbf{c}$
11 $\quad \mathbf{r} = \mathbf{r} - \alpha\mathbf{q}$
12 $\quad \mathbf{s} = \mathbf{P}^{-1}\mathbf{r}$
13 $\quad \delta_{old} = \delta_{new}$
14 $\quad \delta_{new} = \mathbf{r}^T\mathbf{s}$
15 $\quad \mathbf{c} = \mathbf{filter}(\mathbf{s} + \frac{\delta_{new}}{\delta_{old}}\mathbf{c})$

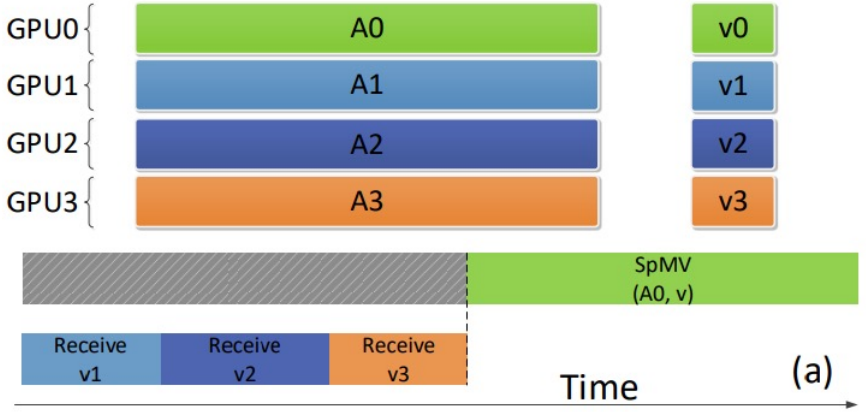
- Vector operations can be easily implemented on multiple GPUs.
- What about matrix-vector multiplication?

[Baraff and Witkin 1998]

Pipelined SpMV

- All-to-all data transfers are expensive
- Significantly reduce GPU loads

$$\mathbf{Av} = \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_{n-1} \end{bmatrix} \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{n-1} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0(\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}) \\ \mathbf{A}_1(\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}) \\ \vdots \\ \mathbf{A}_{n-1}(\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}) \end{bmatrix}.$$

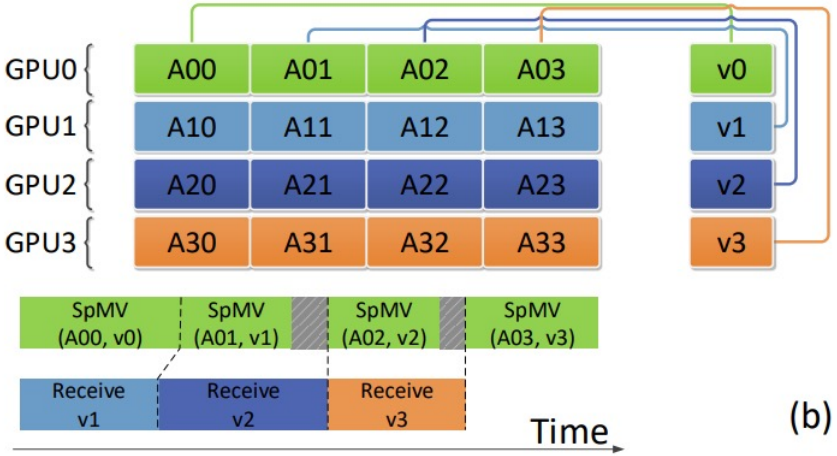


Pipelined SpMV

- **Novel Sparse Matrix-vector (SpMV) Multiplication**
  - Pipeline-based multiplication
  - Inter-leave computation and data transfer

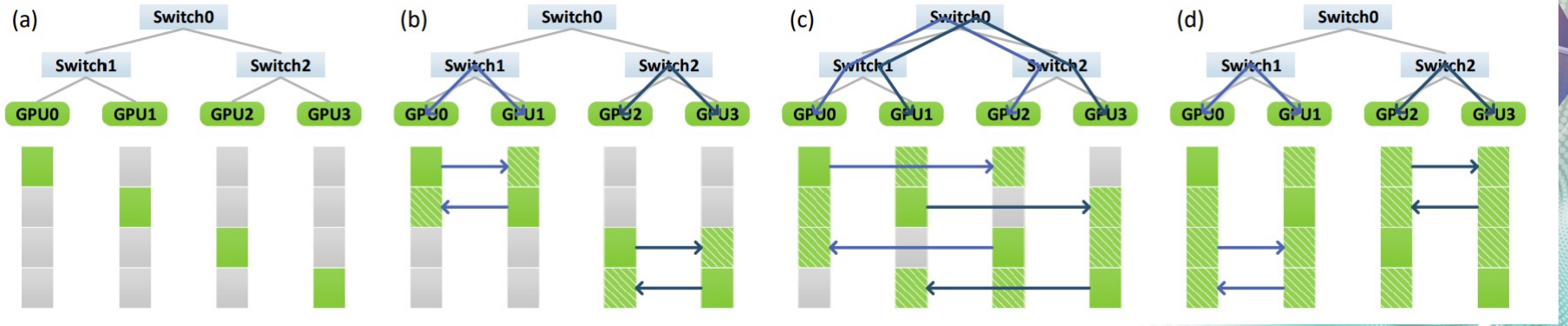
$$\mathbf{Av} = \begin{bmatrix} \mathbf{A}_{00} & \mathbf{A}_{01} & \cdots & \mathbf{A}_{0,n-1} \\ \mathbf{A}_{10} & \mathbf{A}_{11} & \cdots & \mathbf{A}_{1,n-1} \\ \vdots & & \ddots & \vdots \\ \mathbf{A}_{n-1,0} & \mathbf{A}_{n-1,1} & \cdots & \mathbf{A}_{n-1,n-1} \end{bmatrix} \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_{n-1} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{A}_{00}\mathbf{v}_0 + \mathbf{A}_{01}\mathbf{v}_1 + \ldots + \mathbf{A}_{0,n-1}\mathbf{v}_{n-1} \\ \mathbf{A}_{10}\mathbf{v}_0 + \mathbf{A}_{11}\mathbf{v}_1 + \ldots + \mathbf{A}_{1,n-1}\mathbf{v}_{n-1} \\ \vdots \\ \mathbf{A}_{n-1,0}\mathbf{v}_0 + \mathbf{A}_{n-1,1}\mathbf{v}_1 + \ldots + \mathbf{A}_{n-1,n-1}\mathbf{v}_{n-1} \end{bmatrix}.$$



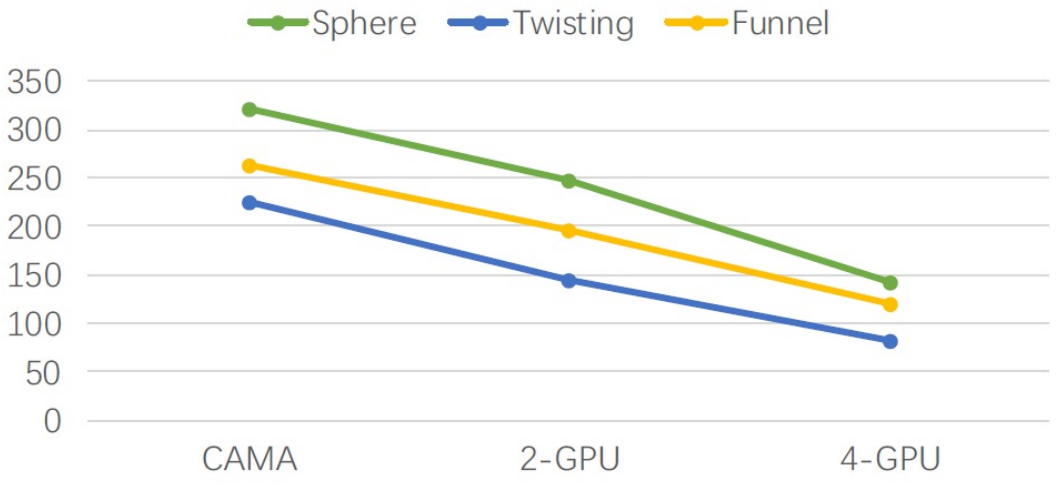
Pipelined SpMV

- **Novel Work Queue Generation Algorithm for Fat-tree Interconnect Topology**
  - Optimize data transfer between multiple GPUs
  - High throughput on complex meshes



Matrix Assembly

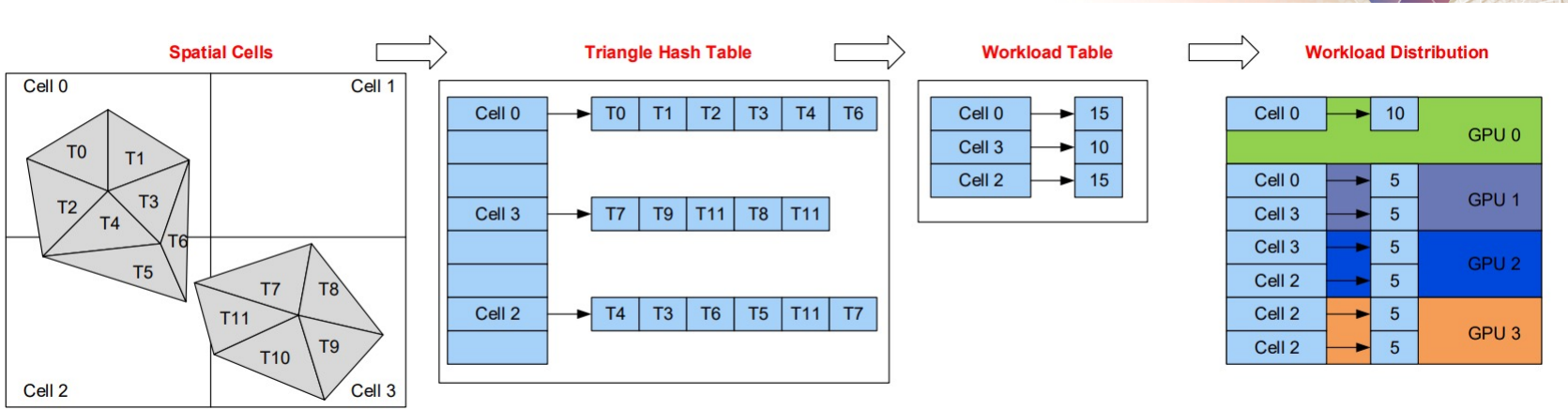
- Compute new contact forces and internal forces at each frame
- Assembly Elements Distribution
- Sparse Matrix Filling
- Use with a preconditioned conjugate gradient (PCG) solver on multiple GPUs



Performance Comparison
with CAMA[Tang et al. 2016]

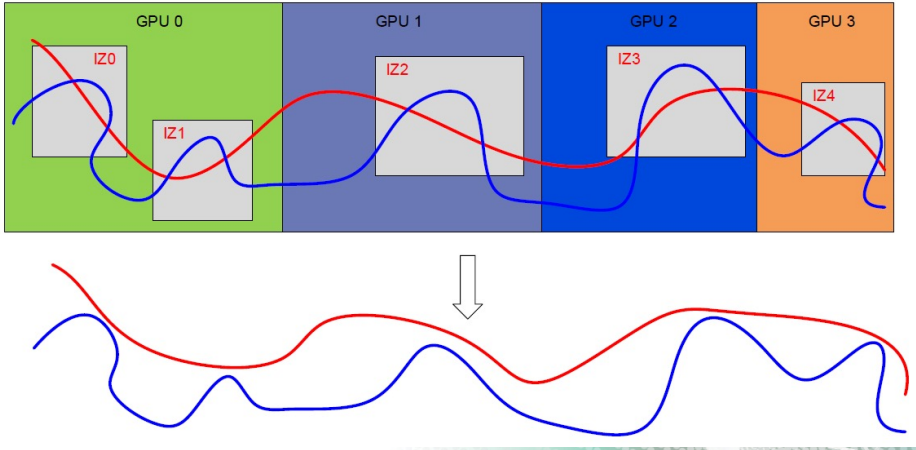
Collision Handling

- **Spatial Hashing**
  - Take hash table into parts and assign them to multiple GPUs



Collision Handling

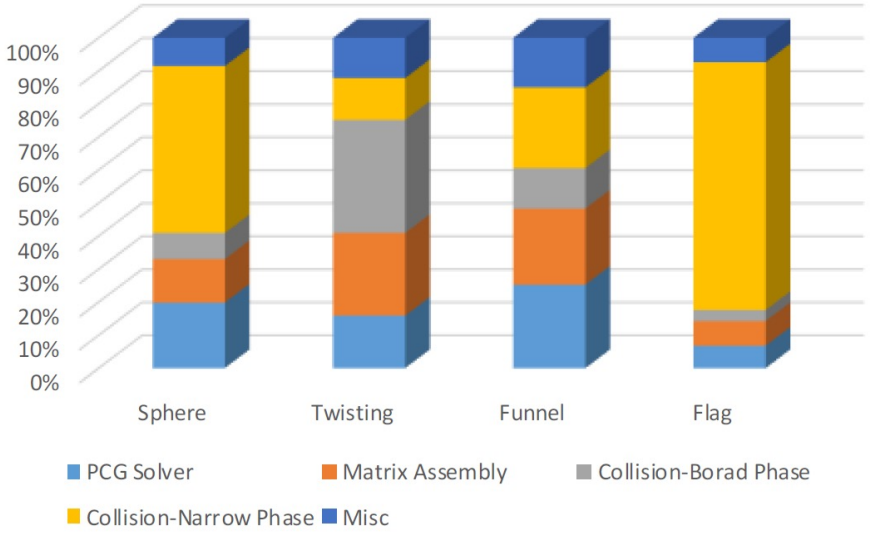
- Parallel algorithms for discrete and continuous collision detection
- Use spatial hashing over multiple GPUs for collision culling
- Scale linearly with number of GPUs
- Reduce memory overhead of each GPU
- First interactive approach for self-collision detection on meshes with 1+M triangles



Parallel Penetration Handling on Multi-GPUs

Results

- P-Cloth: Faster GPU-based simulator for complex meshes
- CUDA 10.0/Ubuntu 16.04
- Handles complex models with 1+ M triangles
- 2-5 fps on 4 NVIDIA Titan Xp GPUs
- Up to 5.1X on 4 NVIDIA Titan Xp GPUs
- Up to 8.23X on 8 NVIDIA Titan V GPUs



Runtime Ratios for Stages

Benchmark: Miku

- 1.33M triangles
- Time step: 1/1000s
- Multiple layers and self-collisions
- Average FPS: 2.59
- 5.4X on 4 GPUs

Benchmark: Miku

- Real-time Playback

SIGGRAPH ASIA 2020 VIRTUAL

Benchmark: Zoey

- 569K triangles
- Time step: 1/500s
- Multiple layers and self-collisions
- Average FPS: 2.04
- 5.37X on 4 GPUs

Benchmark: Zoey

- Real-time Playback

Benchmark: Kimono

- 1M triangles
- Time step: 1/1000s
- Multiple layers and self-collisions
- Average FPS: 1.11
- 4.73X on 4 GPUs

Benchmark: Flag

- 1.2M triangles
- Time step: 1/250s
- Multiple layers and self-collisions
- Average FPS: 5.45
- 4.66X speedup on 4 GPUs
- 8.23X speedup on 8 GPUs

Comparison on Resolutions

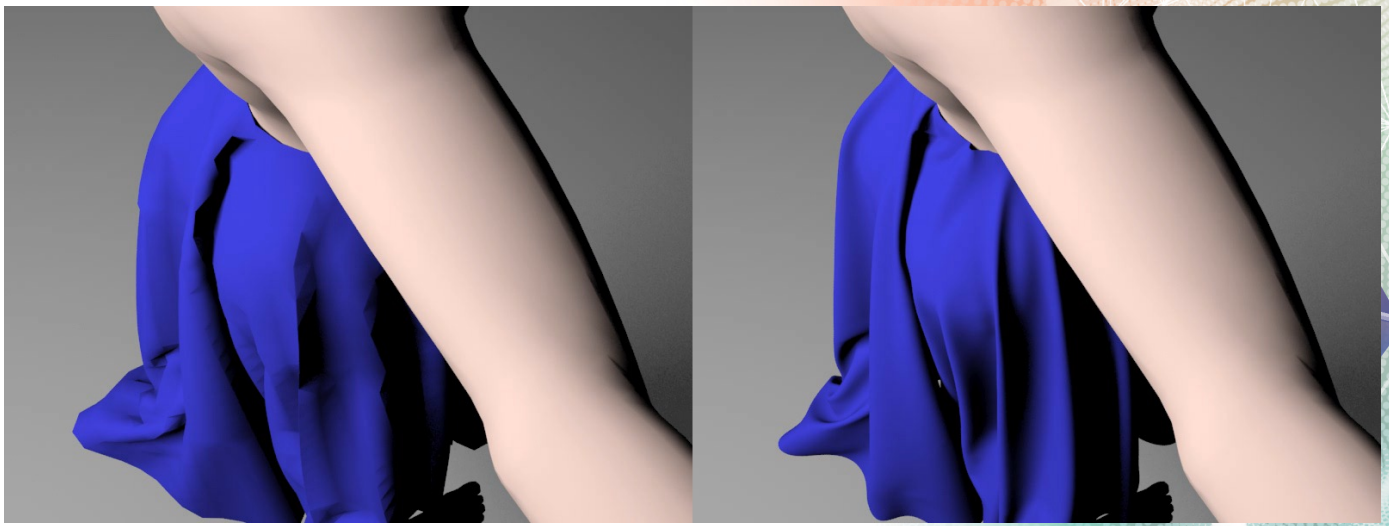
- More wrinkles and folds with higher triangle resolution
- 2.48fps on 4 GPUs with 1.65M triangles

3.1K triangles

1.65M triangles

Comparison on Resolutions

- More wrinkles and folds with higher triangle resolution
- 5.12fps on 4 GPUs with 510K triangles



10.1K triangles

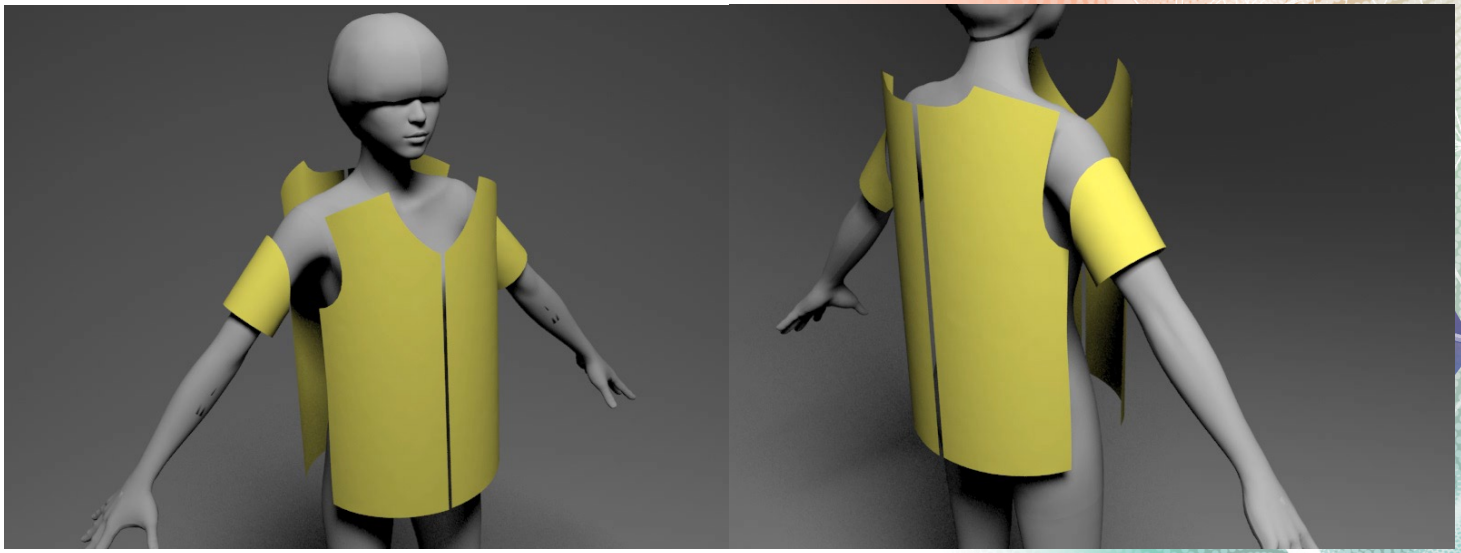
510K triangles

Comparison with I-Cloth

- Similar simulation accuracy
- 5.66X faster on 4 GPUs

Interactive Stitching

- Interactive rate (10+ fps)
- 316K triangles



SIGGRAPH ASIA 2020 VIRTUAL

Limitations

- Collision detection remains a bottleneck.
- Pipelined SpMV is a general solution for all distributed systems, but work queue and data transfer algorithms are limited to fat-tree topologies.
- Speedup can vary depending on the cloth configuration and data synchronization overhead.
- Robustness is governed by contact force computation and a non-linear impact zone solver for penetration handling.

SIGGRAPH ASIA 2020 VIRTUAL

Acknowledgements

- ✓National Key R&D Program of China (2017YFB1002703)
- ✓NSF China (61972341, 61832016, 51775496, 61732015)
- ✓Zhiyu Zhang and Xiaorui Chen for helping on the benchmarks
- ✓Momo Inc. for the Benchmark Kimono
- ✓Zhijiang Lab for the 8-GPU workstation
- ✓Anonymous referees for their valuable comments and helpful suggestions

Thanks!

Q&A