Spectral Animation Compression

Chao Wang, Yang Liu, Xiaohu Guo, Zichun Zhong,
Binh Le and Zhigang Deng

Outline

- Introduction
- Overall scheme
- Spectral animation compression (SAC)
- Result and demo

Introduction

- Dynamic mesh animation
  - Every mesh has the same connectivity
  - No sudden changes in the subsequent frames

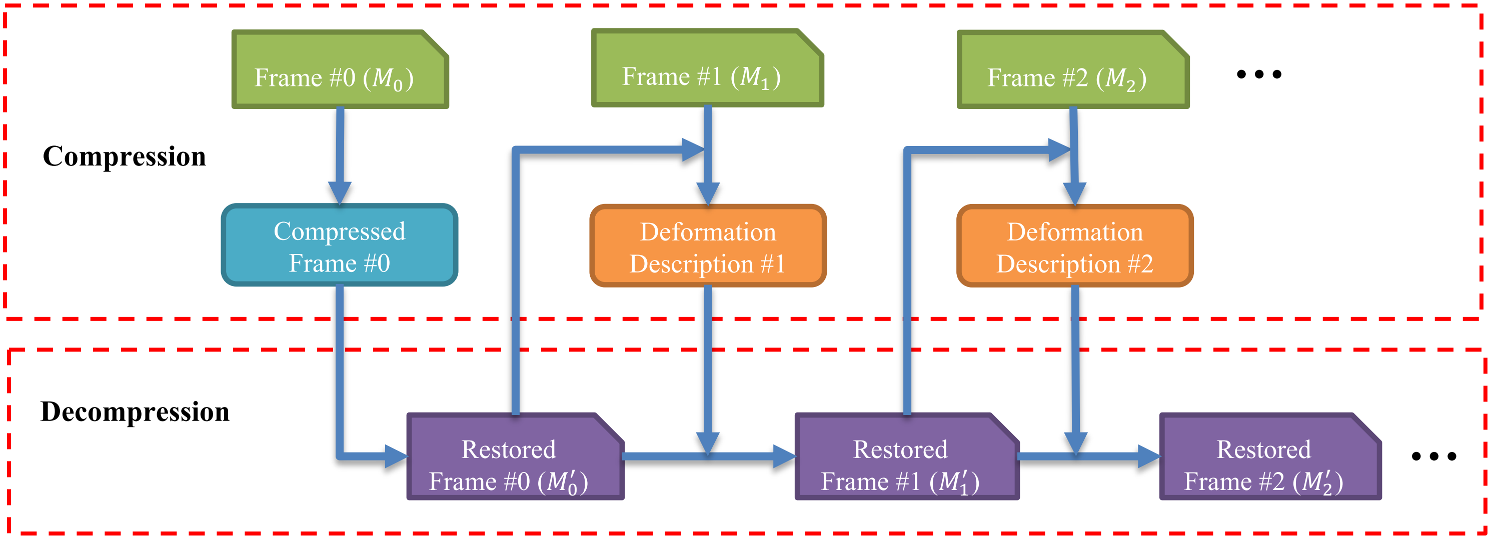
Introduction

- Dynamic animation compression
  - Filter-based method: Principal component analysis (PCA)-based method
  - Predictor-based method
  - Method combining filter and predictor: CoDDyaC [Vasa *et al.*, 3DTV07]

Our framework

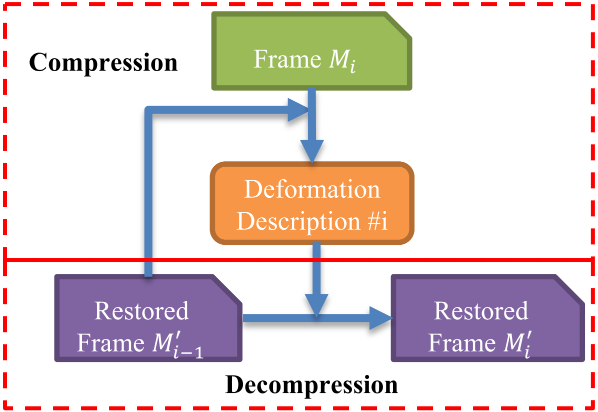
- Spectral animation compression (SAC)
  - Filter: Manifold Harmonics Bases (MHB) [Vallet and Levy, CGF08]
  - Deformation gradient
  - Compress the deformation gradient by decomposing it into rotation and stretching parts
  - Achieve a good balance between the reconstruction quality and compression ratio

Overall Scheme of SAC

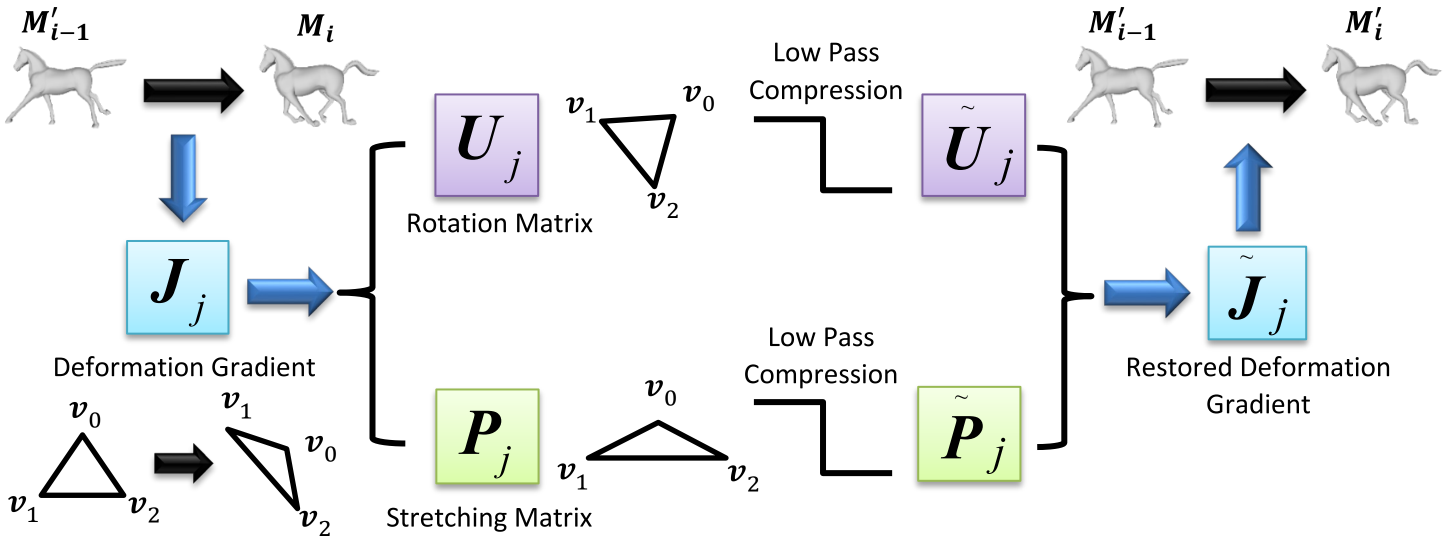


Compress and decompress frame #i CVM 2015

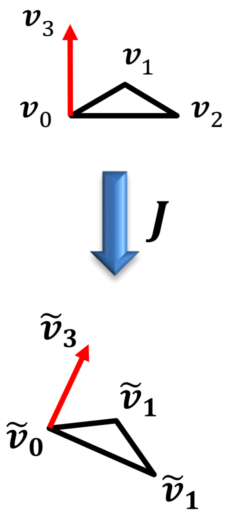
- Input: M'_{i-1}(restored frame #(i-1)) and M_i (original raw data of frame #i)
- Output: M'_i
- Aim: the deformation between M'_i and M'_{i-1} is as close to that between M_i and M'_{i-1} as possible



Compress and decompress frame #i



Deformation Gradient



$$v_3 - v_0 = \frac{(v_1 - v_0) \times (v_2 - v_0)}{|(v_1 - v_0) \times (v_2 - v_0)|}$$

$$V = [v_1 - v_0 \quad v_2 - v_0 \quad v_3 - v_0]$$

$$\tilde{V} = [\tilde{v}_1 - \tilde{v}_0 \quad \tilde{v}_2 - \tilde{v}_0 \quad \tilde{v}_3 - \tilde{v}_0]$$

$$VJ = \tilde{V}, \qquad J = \tilde{V}V^{-1}$$

Polar Decomposition

$$\boldsymbol{J}_j = \boldsymbol{U}_j \boldsymbol{P}_j$$

$\boldsymbol{U}_j$: rotation matrix (3 Degrees of freedom (Dof))

$\boldsymbol{P}_j$: stretching matrix (symmetric, 6 Dof)

Totally 9 Dof for each triangle, present all of these values as 9 m-by-1 column vectors (functions) $\boldsymbol{f}_t$, $t = 1, \ldots, 9$, m is the number of triangles

Manifold Harmonics Bases (MHB)

- MHB is orthogonal to each other with inner product
- MHB can be used as a low-pass filter on user-defined functions on the mesh
- Problem: the functions must be defined on vertices.

Manifold Harmonics Bases (MHB)

- Given a per-vertex function $\boldsymbol{g}_t$, a per-triangle function $\boldsymbol{f}_t$ can be obtained by averaging the function values on the vertices:

$$\boldsymbol{f}_t = C\boldsymbol{g}_t$$

$$\boldsymbol{g}_t = (C^T C)^{-1} C^T \boldsymbol{f}_t$$

where C is m-by-n matrix with

$$c_{ij} = \begin{cases} \frac{1}{3}, & vertex\ j\ is\ in\ triangle\ i \\ 0, & otherwise \end{cases}$$

Compression with MHB

- Computing spectral descriptors of function $\boldsymbol{g}_t$ on MHB $\{\boldsymbol{H}_k\}$ by projecting $\boldsymbol{g}_t$ on each $\boldsymbol{H}_k$:

$$r_{k,t} = < \boldsymbol{H}_k, \boldsymbol{g}_t >$$

- In compression, drop out the high frequency data by only using the first l bases where $l \ll n$ (number of vertices)

Quantization and Arithmetic Coding CVM 2015

- Further reduce the compression size of the spectral descriptors $\{r_{k,t}\}$

Decompression

- Restore the spectral descriptors $\{\tilde{r}_{k,t}\}$
- Restore the function $\tilde{\boldsymbol{f}}_t$

$$\tilde{\boldsymbol{f}}_t = C\tilde{\boldsymbol{g}}_t = C\sum_{k=1}^{l}\tilde{r}_{k,t}\boldsymbol{H}_k$$

- Deformation gradient

$$\tilde{\boldsymbol{J}}_j = \tilde{\boldsymbol{U}}_j\tilde{\boldsymbol{P}}_j$$

Decompression

- Input: $\{\tilde{\boldsymbol{J}}_j\}$ after decompression, and one vertex $\boldsymbol{v_0} \in M_i$
- Reconstruct M_i' by optimizing:

$$\min_{\boldsymbol{v}_0',\dots,\boldsymbol{v}_{n-1}'} \sum_{j=0}^{m-1} ||\boldsymbol{S}_j - \tilde{\boldsymbol{J}}_j||_F^2$$

subject to $\boldsymbol{v}_0' = \boldsymbol{v}_0$

m, n: number of triangles and vertices, respectively

$\tilde{\boldsymbol{J}}_j$: decompressed deformation gradient between M_i and M_{i-1}'

$\boldsymbol{S}_j$: deformation gradient between M_i' and M_{i-1}'

Decompression

$$\min_{v'_0,\dots,v'_{n-1}} \sum_{j=0}^{m-1} ||\boldsymbol{S}_j - \tilde{\boldsymbol{J}}_j||_F^2$$

subject to $\boldsymbol{v}'_0 = \boldsymbol{v}_0$

- Convert it to a linear system [Sumner and Popovic, TOG04]:

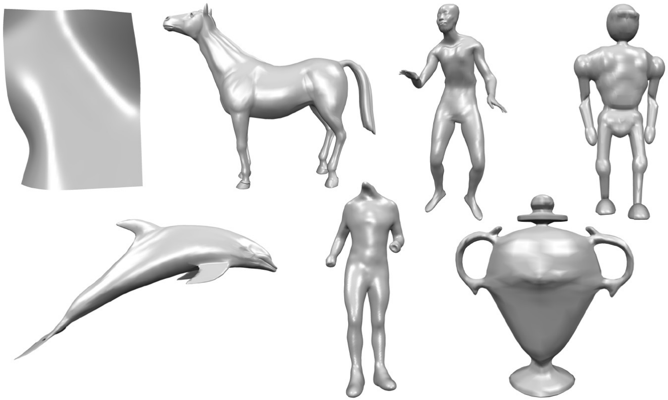
$$\boldsymbol{AX} = \boldsymbol{b}$$

$$\boldsymbol{X} = [\boldsymbol{v}'_0, \dots, \boldsymbol{v}'_{n-1}]^T$$

Experimental Results

- Restored Error w.r.t. compression size/ratio
- Error metrics:
  - KG error [Karni *et al*., C&G04]
  - Spatial-Temporal Edge Difference (STED) error [Vasa *et al*., TVCG11]
- Compression size: bit per vertex per frame (bpvf)

Experimental Data and Time Cost



Animation	# of Frames	# of Vertices	# of triangles	Running Time (s) per frame
Cloth	199	5525	10752	0.4821
Horse-collapse	53	8431	16843	0.5554
Dance	200	7061	14118	0.6541
Humanoid	153	7646	15288	0.5340
Dolphin	100	6179	12278	0.5849
Jump	221	15826	31648	1.7808
Vase	70	2502	5008	0.2378

Experiment on Polar decomposition

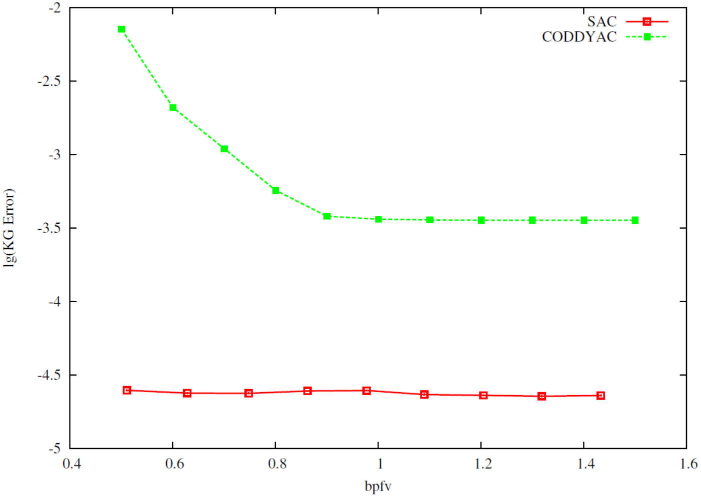
- SAC (red) and CoDDyaC [Vasa *et al.*, 3DTV07] (green)

KG error

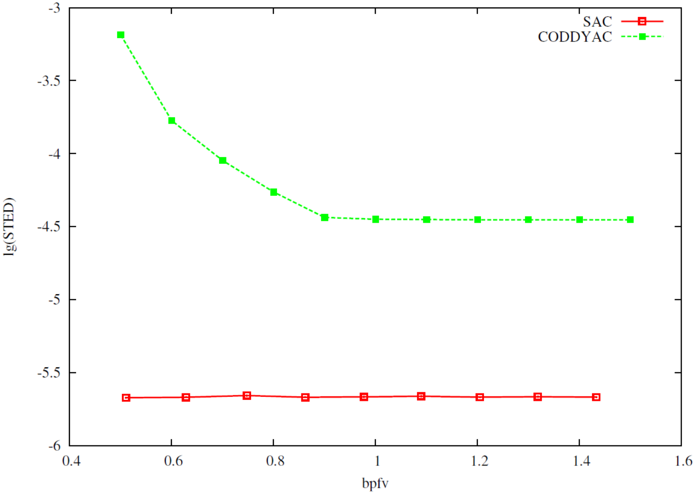
STED

Result on Vase

- SAC (red) and CoDDyaC [Vasa *et al.*, 3DTV07] (green)



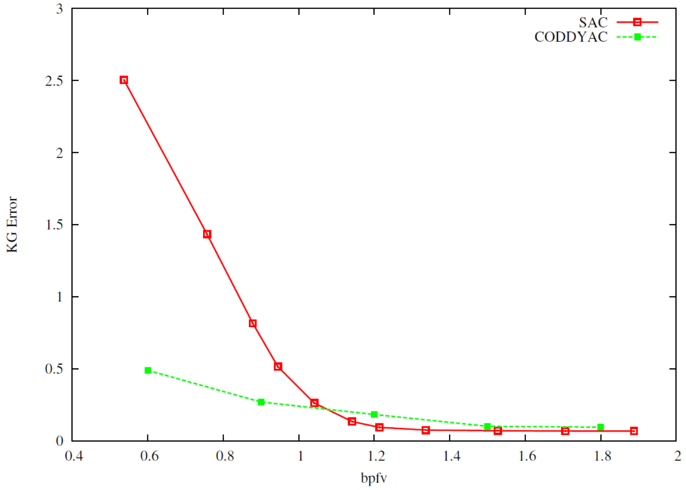
KG error



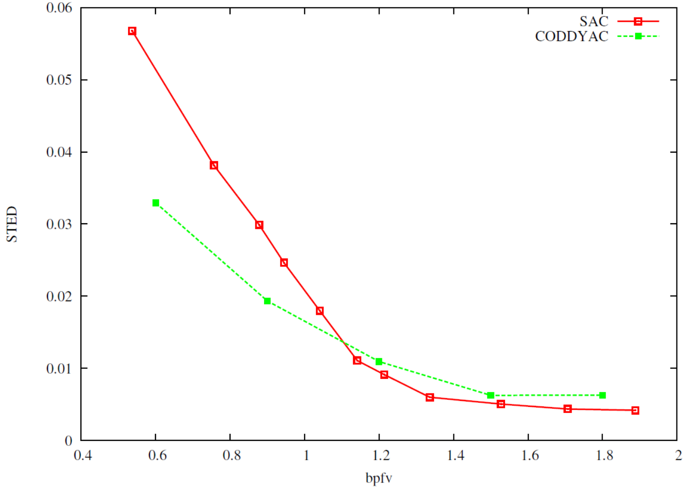
STED

Result on Cloth

- SAC (red) and CoDDyaC [Vasa *et al.*, 3DTV07] (green)



KG error



STED

Thanks!