GausSim: Foreseeing Reality by Gaussian Simulator for Elastic Objects

Supplementary Material

Table A1. We list the mathematical symbols mentioned in the main manuscripts and supplementary materials as follows.

Symbols	Descriptions
$\mathcal{G}$	Set of Gaussian kernels.
$\boldsymbol{x}_k$	The position of the k-th kernel.
$\boldsymbol{\sigma}_k, \boldsymbol{\Sigma}_k$	The covariance of the k-th kernel in deformed space and material space respectively.
$\boldsymbol{c}_k$	The color of the k-th kernel.
α_k	The opacity of the k-th kernel.
ρ_k	The trainable density of k-th kernel.
$\boldsymbol{a}_k$	The trainable vector for attributes of k-th kernel.
t	The timestamp.
$\psi_\theta(\cdot)$	GausSim, a neural network parameterized by θ, designed to predict the future states of Gaussian kernels.
I_t^o	The ground truth image at time t for view direction o.
$\boldsymbol{X}_k$	The position of k-th kernel in material space.
$\boldsymbol{x}_r, \boldsymbol{X}_r$	The position of the static kernel, which can be the root of the flower.
$\boldsymbol{F}_k$	The deformation gradient for k-th kernel to compute the deformed states at any timestamp.
$\boldsymbol{U}_k, \boldsymbol{V}_k$	The components of "Polar SVD", representing the rotations.
$\boldsymbol{R}_k$	The rotation matrix computed by $\boldsymbol{U}_k, \boldsymbol{V}_k$, representing the closest rotation to the deformation gradient.
$\boldsymbol{\Lambda}, \boldsymbol{\lambda}$	The diagonal matrix in "Polar SVD" decompositions and the 3D vector representing the singular values respectively.
p, q, r	Non-negative real numbers.
$\boldsymbol{d}$	The 3D vector of view direction.
$g(\cdot)$	The function to compute Gaussian kernel's color given view direction $\boldsymbol{d}$.
m_k, V_k	The mass and volume for k-th kernel.
m_{c_l}, V_{c_l}	The mass and volume for c_l-th Center of Mass System.
c_l	The index indicating the Center of Mass System at l-th level of hierarchical structure.
L	Total number of hierarchical level.
$\hat{\boldsymbol{x}}_k^{h-1}, \hat{\boldsymbol{x}}_{c_h}^{h-1}$	The predicted positions given h-th level's simulation for the center of k-th kernel and c_h-th Center of Mass System respectively.
$\hat{\boldsymbol{\sigma}}_k^{h-1}, \boldsymbol{c}_k^{h-1}$	The predicted covariance and color for k-th kernel respectively.
$\mathcal{F}(\cdots)$	The function to render Gaussian kernels for given view direction.
$G_k(\cdot)$	The Gaussian function for k-th Gaussian kernel.

A1. Symbols

As shown in Table A1, we list the symbols used in the main manuscript and supplementary materials.

A2. Methodology Proofs and Details

In this paper, we treat each Gaussian kernel as continuous pieces of matter and formulate GausSim based on continuum mechanics. The hierarchical structure is built following the rules of Center of Mass Systems (CMS). Most importantly, the Gaussian kernels themselves are already Center of Mass Systems as shown in Section A2.2, which are constructed for the areas of continuous volume.

A2.1. Proof of Hierarchical Structure

Suppose we already have the simulated results $\hat{\boldsymbol{x}}_k^{h+1}$ at level $h+1$. By regarding the results $\hat{\boldsymbol{x}}_k^{h+1}$ as a special kind of "template states" mentioned in Equation 2, the formulations of simulating the lower levels in a recursive manner are as follows:

$$\hat{\boldsymbol{x}}_k^h = \hat{\boldsymbol{x}}_{c_{h+1}}^{h+1} + F_k^{h+1}(\hat{\boldsymbol{x}}_k^{h+1} - \hat{\boldsymbol{x}}_{c_{h+1}}^{h+1}), \quad (21)$$

$$\hat{\boldsymbol{x}}_k^{h-1} = \hat{\boldsymbol{x}}_{c_h}^{h} + F_k^{h}(\hat{\boldsymbol{x}}_k^{h} - \hat{\boldsymbol{x}}_{c_h}^{h}), \quad (22)$$

where $\hat{\boldsymbol{x}}_{c_{h+1}}^{h+1}, \hat{\boldsymbol{x}}_{c_h}^{h}$ are the barycenters of the Center of Mass Systems at level $h+1, h$ respectively.

By expanding the variables in Equation 22 using Equation 21, we have:

$$\hat{\boldsymbol{x}}_k^{h-1} = \hat{\boldsymbol{x}}_{c_h}^{h} + F_k^{h}F_k^{h+1}(\hat{\boldsymbol{x}}_k^{h+1} - \hat{\boldsymbol{x}}_{c_h}^{h+1}). \quad (23)$$

Therefore, through the expansion of the variables within the brackets, the results $\hat{\boldsymbol{x}}_k^{h-1}$ at level $h-1$ can be traced back till the top level L as follows:

$$\hat{\boldsymbol{x}}_k^{h-1} = \hat{\boldsymbol{x}}_{c_h}^{h} + \prod_{j=h}^{L} F_k^{j}(\hat{\boldsymbol{x}}_k^{L} - \hat{\boldsymbol{x}}_{c_h}^{L}), \quad (24)$$

where we define $\hat{\boldsymbol{x}}_k^L, \hat{\boldsymbol{x}}_{c_h}^L$ as $\boldsymbol{X}_k, \boldsymbol{X}_{c_h}$, which are from the original material space as the initial conditions for the recursive formulations, respectively. Specifically, $\boldsymbol{X}_{c_L}$ is defined as the static position $\boldsymbol{x}_r$. Further replacing $\hat{\boldsymbol{x}}_k^L, \hat{\boldsymbol{x}}_{c_h}^L$ by $\boldsymbol{X}_k, \boldsymbol{X}_{c_h}$, we have:

$$\hat{\boldsymbol{x}}_k^{h-1} = \hat{\boldsymbol{x}}_{c_h}^h + \prod_{j=h}^{L} F_k^j (\boldsymbol{X}_k - \boldsymbol{X}_{c_h}), \tag{25}$$

which is exactly the Equation 8 in the main manuscript.

Equation 9 can be obtained in the same manner, where the only difference is the subscript. For illustration, we show an example of expanding the equation by two steps from level h to level $h+2$ as follows:

$$\begin{aligned}
\hat{\boldsymbol{x}}_{c_h}^h &= \hat{\boldsymbol{x}}_{c_{h+1}}^{h+1} + F_k^{h+1}(\hat{\boldsymbol{x}}_{c_h}^{h+1} - \hat{\boldsymbol{x}}_{c_{h+1}}^{h+1}), && (26)\\
&= \hat{\boldsymbol{x}}_{c_{h+2}}^{h+2} + F_k^{h+2}(\hat{\boldsymbol{x}}_{c_{h+1}}^{h+2} - \hat{\boldsymbol{x}}_{c_{h+2}}^{h+2}) && (27)\\
&\quad + F_k^{h+1} F_k^{h+2}(\hat{\boldsymbol{x}}_{c_h}^{h+2} - \hat{\boldsymbol{x}}_{c_{h+1}}^{h+2}) && (28)\\
&= \boldsymbol{x}_r + \sum_{i=h}^{L}\prod_{j=i}^{L} F_k^{j+1}\left(\boldsymbol{X}_{c_j} - \boldsymbol{X}_{c_{j+1}}\right). && (29)
\end{aligned}$$

As for the covariance in Equation 10 in the main manuscript, we start from the Gaussian kernel at material space:

$$G_k(\boldsymbol{X}) = e^{-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{X}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{X}-\boldsymbol{X}_k)} \tag{30}$$

According to Equation 25, we have

$$\begin{aligned}
\boldsymbol{X} - \boldsymbol{X}_{c_h} &= \left(\prod_{j=h}^{L} F_k^j\right)^{-1} (\hat{\boldsymbol{x}}^{h-1} - \hat{\boldsymbol{x}}_{c_h}^h), && (31)\\
\boldsymbol{X}_k - \boldsymbol{X}_{c_h} &= \left(\prod_{j=h}^{L} F_k^j\right)^{-1} (\hat{\boldsymbol{x}}_k^{h-1} - \hat{\boldsymbol{x}}_{c_h}^h), && (32)\\
\boldsymbol{X} - \boldsymbol{X}_k &= (\boldsymbol{X} - \boldsymbol{X}_{c_h}) - (\boldsymbol{X}_k - \boldsymbol{X}_{c_h}) && (33)\\
&= \left(\prod_{j=h}^{L} F_k^j\right)^{-1} (\hat{\boldsymbol{x}}^{h-1} - \hat{\boldsymbol{x}}_k^{h-1}), && (34)
\end{aligned}$$

where $\boldsymbol{X}$ and $\boldsymbol{X}_k$ share the same deformation sequences since $\boldsymbol{X}$ belongs to the k-th kernel and they are always deformed together. Therefore, combining Equation 34 with Equation 30, the Gaussian kernel can be represented by

$$\begin{aligned}
G_k(\boldsymbol{X}) &= e^{-\frac{1}{2}(\hat{\boldsymbol{x}}^{h-1} - \hat{\boldsymbol{x}}_k^{h-1})^\top (\hat{\boldsymbol{\sigma}}_k^{h-1})^{-1}(\hat{\boldsymbol{x}}^{h-1} - \hat{\boldsymbol{x}}_k^{h-1})}, && (35)\\
\hat{\boldsymbol{\sigma}}_k^{h-1} &= \left(\prod_{j=h}^{L} F_k^j\right) \boldsymbol{\Sigma}_k \left(\prod_{j=h}^{L} F_k^j\right)^\top, && (36)
\end{aligned}$$

which is the Equation 10 in the main manuscript.

As for the color in Equation 11 from the main paper, we provide an analysis below. As mentioned above, for the given simulated results $\hat{\boldsymbol{x}}_k^{h+1}$ at level $h+1$, we treat them as a special kind of "template states", based on which we predict the new deformation gradients to deform the kernels or Center of Mass Systems in the lower levels. Thus, we can always apply Equation 5 recursively to compute the new color after each level's simulation, leading to the form in Equation 11.

A2.2. Gaussian Kernel Is CMS

Equivalent Volume. Suppose we have a kernel

$$G(\boldsymbol{x}) = e^{\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}, \tag{37}$$

where we assume that for a given position $\boldsymbol{x}$, $G(\boldsymbol{x})$ is proportional to the density with ratio ρ, which is invariant overtime. Thus, the total mass of this kernel, which is the integration of the Gaussian function, is known as:

$$\begin{aligned}
m &= \int \rho G(\boldsymbol{x}) d^3x && (38)\\
&= \rho\sqrt{\det(2\pi\Sigma)}, && (39)
\end{aligned}$$

where we obtain the volume for the kernel equivalently as $V = m/\rho = \sqrt{\det(2\pi\Sigma)}$, or the volume is proportional to the root of determinant of covariance matrix as $V \propto \sqrt{\det(\Sigma)}$ since the trainable ρ can learn the coefficient.

Gaussian kernel is a Center of Mass System for continuous area. The barycenter for the Gaussian kernel locates at the mean position $\boldsymbol{\mu}$. To illustrate, we have the barycenter's position on x-axis, which should be $\boldsymbol{\mu}_x$, as follows:

$$\begin{aligned}
&\frac{1}{m}\int x dm && (40)\\
=\ &\int \frac{x}{m}\cdot \rho G(\boldsymbol{x}) dxdydz && (41)\\
=\ &\int \frac{x-\boldsymbol{\mu}_x}{m}\rho G(\boldsymbol{x}) d(x-\boldsymbol{\mu}_x) d(y-\boldsymbol{\mu}_y) d(z-\boldsymbol{\mu}_z) && (42)\\
&+ \int \frac{\boldsymbol{\mu}_x}{m}\rho G(\boldsymbol{x}) dxdydz && (43)\\
=\ &0 + \boldsymbol{\mu}_x, && (44)
\end{aligned}$$

where Equation 42 equals to 0 since it is an odd function with symmetric integral domain. The same applies to $\boldsymbol{\mu}_y, \boldsymbol{\mu}_z$.

A3. Input Details of GausSim

We adopt the MeshGraphNet [31] as our backbone and model the interactions of Center of Mass Systems to predict the corresponding gradient deformations. The inputs of GausSim includes node features, which denote the states of

Table A2. The radii used for clustering kernels to construct the Hierarchical Center of Mass Systems at level 1 and level 2.

Hierarchical Level	Mothorchids	Carnation	Pudding	Duck	Bunny
l=1	0.04	0.03	0.04	0.035	0.03
l=2	0.5	0.3	0.4	0.35	0.3

the Center of Mass Systems, and edge features, which represent the interaction information between neighbor nodes. Specifically, the node feature includes:

- The equivalent acceleration of the local system, *i.e*. Center of Mass System: $-\frac{\boldsymbol{x}_{c,t}-2*\boldsymbol{x}_{c,t-1}+\boldsymbol{x}_{c,t-2}}{dt^2}$.
- The local velocity of each component k within the Center of Mass System: $\frac{(\boldsymbol{x}_{k,t}-\boldsymbol{x}_{c_t})-(\boldsymbol{x}_{k,t-1}-\boldsymbol{x}_{c,t-1})}{dt}$.
- The learnable attribute for each component k: $\boldsymbol{a}_k$.

We assign an edge feature between nodes when they are close to each other in both material space and deformed states. And the edge feature with source node $\boldsymbol{x}_s$ and destination node $\boldsymbol{x}_d$ includes:

- The ratio of the deformation: $\|\boldsymbol{x}_{d,t}-\boldsymbol{x}_{s,t}\|/\|\boldsymbol{X}_d-\boldsymbol{X}_s\|$.
- The direction from source node to target node: $(\boldsymbol{x}_{d,t}-\boldsymbol{x}_{s,t})/\|\boldsymbol{x}_{d,t}-\boldsymbol{x}_{s,t}\|$.
- The relative velocity: $\frac{(\boldsymbol{x}_{d,t}-\boldsymbol{x}_{s,t})-(\boldsymbol{x}_{d,t-1}-\boldsymbol{x}_{s,t-1})}{dt}$.
- The relative angles given the barycenter c: $< \boldsymbol{x}_{d,t}-\boldsymbol{x}_{c,t}, \boldsymbol{x}_{s,t}-\boldsymbol{x}_{c,t} > - < \boldsymbol{X}_d-\boldsymbol{X}_c, \boldsymbol{X}_s-\boldsymbol{X}_c >$.

The radii to cluster kernels and construct the Center of Mass Systems are as shown in Table A2.

A4. Experiment Details

A4.1. Dataset READY

To capture the dynamic motions with less blurred details, we increase the camera's shutter speed, which inevitably results in darker images. Since the brightness does not affect our study, we adopt a post-processing step to increase the brightness for all images shown in the paper and supplementary for better visual quality. In addition, we segment out the foreground and focus on the motions of the objects. Training and evaluations are conducted using the segmented images without post-processing. As shown in Table A3, we report the averaged color values normalized between 0 to 1 for the foreground objects. Notice that the black background's color value is $(0, 0, 0)$, and the contrast between the foreground color and black background is larger in "Bunny". This phenomenon results in larger variations of absolute $\ell 2$ errors in the quantitative comparisons.

Differences from Video Diffusion-based Data. Instead of distilling priors from Video Diffusion Model, our data is completely from real world, without concerning about the realism of the videos generated by diffusion models. GausSim trained on our dataset can directly replicate and foresee the dynamics in real world, closing the gap between experimental settings and real-world scenarios.

A4.2. Implementation Details

GausSim. We adopt the MeshGraphNet [31] consisting of 16 graph neural layers as GausSim's backbone to handle the kernel-wise interactions. The output deformation gradients are obtained through a three-layer MLP, with the output dimension being set to 11, where 8 for two quaternions and 3 for the diagonal matrix. All hidden vectors are of size 128. Moreover, we train our GausSim on all objects with 25 epochs both separately and jointly, as analyzed in the Section 4.2. The length of predictions T in Equation 20 increases after every epoch, which is capped at $T = 16$. We adopt the Adam optimizer with an initial learning rate of 0.0008, which starts to decrease with a factor of 0.5 after 16 epochs. All experiments are conducted on four NVIDIA A800 GPUs with a batch size of 4, taking 22 hours to converge during training.

DreamGaussian4D. We make sure to use the same amount of images to train DreamGaussian4D (DG4D) and other models. We vary the hyperparameters for training, including the number of iterations, the learning rate, the number of sampled views for score distillation sampling, etc. We also vary the black background and white background to find the best settings for training. Notice that the dynamic results reconstructed by DG4D tend to be slightly more blurred than the reference images, which aligns with the fact in the official website.

PhysDreamer. We adopt 768 sub-steps between adjacent video frames with a duration of 4.34×10^{-5} seconds per sub-step. Since PD's performance is *highly dependent on hyperparameters and initialization configurations*, we try our best to find the best hyperparameters and initial values for each domain. We also try to change the grid size that maximizes the performance during training and test. Notice that though we vary the initial young's modulus from 1.0 to 5×10^8 to find the best value to represent the stiffness, PD cannot support objects with high stiffness and swaying frequency, such as the synthetic "Bunny" and real-world "Duck". Moreover, we observe that PD tends to predict rapid dissipation, which aligns with the phenomenon in the official website.

A4.3. Rendering Results from Different Views

In Figure A1, we exhibit more rendering results on different views. GausSim achieves superior performance regardless of the view directions of the cameras, suggesting the effectiveness in capturing the underlying physics laws.

A4.4. SSIM and LPIPS Evaluations

In Table A4 and Table A5, we report the SSIM for structural similarities and LPIPS for perceptual evaluations respec-

Table A3. The average color values normalized between 0 to 1 for foreground objects. Notice that the black background during training has a value of $(0, 0, 0)$, leading to larger contrast of color with the "Bunny". We also report the maximum value of $\ell 2$ errors on sampled data.

	Mothorchids	**Carnation**	**Pudding**	**Duck**	**Bunny**
Avg Color	(0.22, 0.14, 0.14)	(0.21, 0.23, 0.18)	(0.24, 0.15, 0.06)	(0.35, 0.31, 0.17)	(0.78, 0.73, 0.67)
Avg Color's $\ell 2$	0.30	0.37	0.29	0.50	1.26
Max $\ell 2$ Loss	0.67	0.91	0.87	0.95	1.63

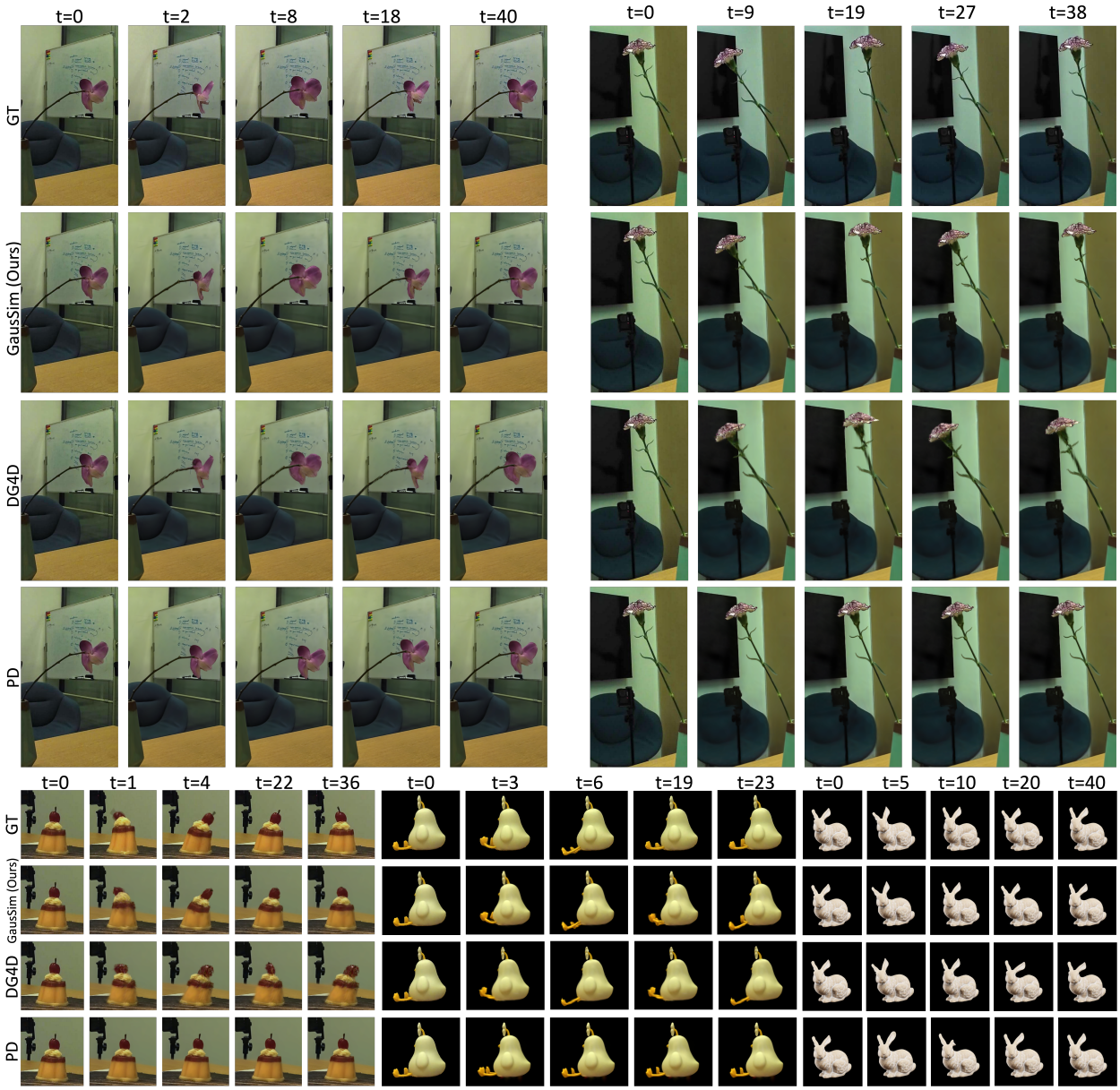

Figure A1. More qualitative results from different views. GausSim exhibits realistic deformations even with challenging initial conditions, indicating the effectiveness of our methods.

Table A4. SSIM (1×10^{-1}) results on our real dataset consisting of "Mothorchids", "Carnation", "Pudding", and "Duck", as well as synthetic "Bunny". For the unseen frames, we report the evaluations on 83 frames from $t = 16$ to $t = 99$ on "Mothorchids", "Carnation", and "Pudding", and 34 frames from $t = 16$ to $t = 49$ on "Duck" and "Bunny". Our GausSim achieves superior performance in all cases.

SSIM (↑)	**Mothorchids**		**Carnation**		**Pudding**		**Duck**		**Bunny**	
	Seen	Unseen 83	Seen	Unseen 83	Seen	Unseen 83	Seen	Unseen 34	Seen	Unseen 34
DG4D [30]	9.89±0.02	9.82±0.03	9.80±0.01	9.65±0.03	9.91±0.00	9.88±0.01	9.93±0.00	9.92±0.00	9.96±0.00	9.91±0.00
PD [45]	9.76±0.02	9.86±0.02	9.71±0.02	9.73±0.02	9.91±0.00	9.92±0.00	9.91±0.00	9.92±0.01	9.89±0.00	9.86±0.00
GausSim(Ours)	**9.90±0.01**	**9.90±0.01**	**9.81±0.01**	**9.74±0.02**	**9.95±0.00**	**9.95±0.00**	**9.94±0.01**	**9.95±0.01**	**9.97±0.00**	**9.93±0.01**

Table A5. LPIPS (1×10^{-2}) results on our real dataset consisting of "Mothorchids", "Carnation", "Pudding", and "Duck", as well as synthetic "Bunny". For the unseen frames, we report the evaluations on 83 frames from $t = 16$ to $t = 99$ on "Mothorchids", "Carnation", and "Pudding", and 34 frames from $t = 16$ to $t = 49$ on "Duck" and "Bunny". Our GausSim achieves superior performance in all cases.

LPIPS (↓)	**Mothorchids**		**Carnation**		**Pudding**		**Duck**		**Bunny**	
	Seen	Unseen 83	Seen	Unseen 83	Seen	Unseen 83	Seen	Unseen 34	Seen	Unseen 34
DG4D [30]	2.54±0.21	3.41±0.54	4.00±0.19	7.00±0.61	1.42±0.03	1.72±0.05	1.52±0.13	1.82±0.14	0.27±0.02	0.69±0.01
PD [45]	3.62±0.24	2.77±0.11	4.56±0.27	5.66±0.54	1.38±0.02	1.23±0.04	1.41±0.15	1.32±0.17	1.67±0.19	0.89±0.13
GausSim(Ours)	**2.46±0.10**	**2.47±0.07**	**3.91±0.16**	**5.10±0.58**	**1.13±0.03**	**1.17±0.04**	**1.18±0.18**	**1.23±0.19**	**0.21±0.02**	**0.49±0.08**

tively. Our GausSim achieves superior performance comparing with baselines.

A4.5. Interactive Applications

As shown in Figure A2, our method supports interactive dynamics by applying customized external forces f on selected Gaussian kernels. In practice, we convert the forces as displacements of Gaussian kernels' positions to adapt to our simulation pipeline in Equation 1 as follows:

$$\Delta \boldsymbol{x} = \frac{1}{2}\frac{f}{m}t^2, \tag{45}$$

$$\mathcal{G}_{t+1} = \psi(\mathcal{G}_t + \Delta \boldsymbol{x}, \mathcal{G}_{t-1}), \tag{46}$$

where t is the time interval between two simulation steps and m is the mass of the kernels as illustrated in Section 3.3.

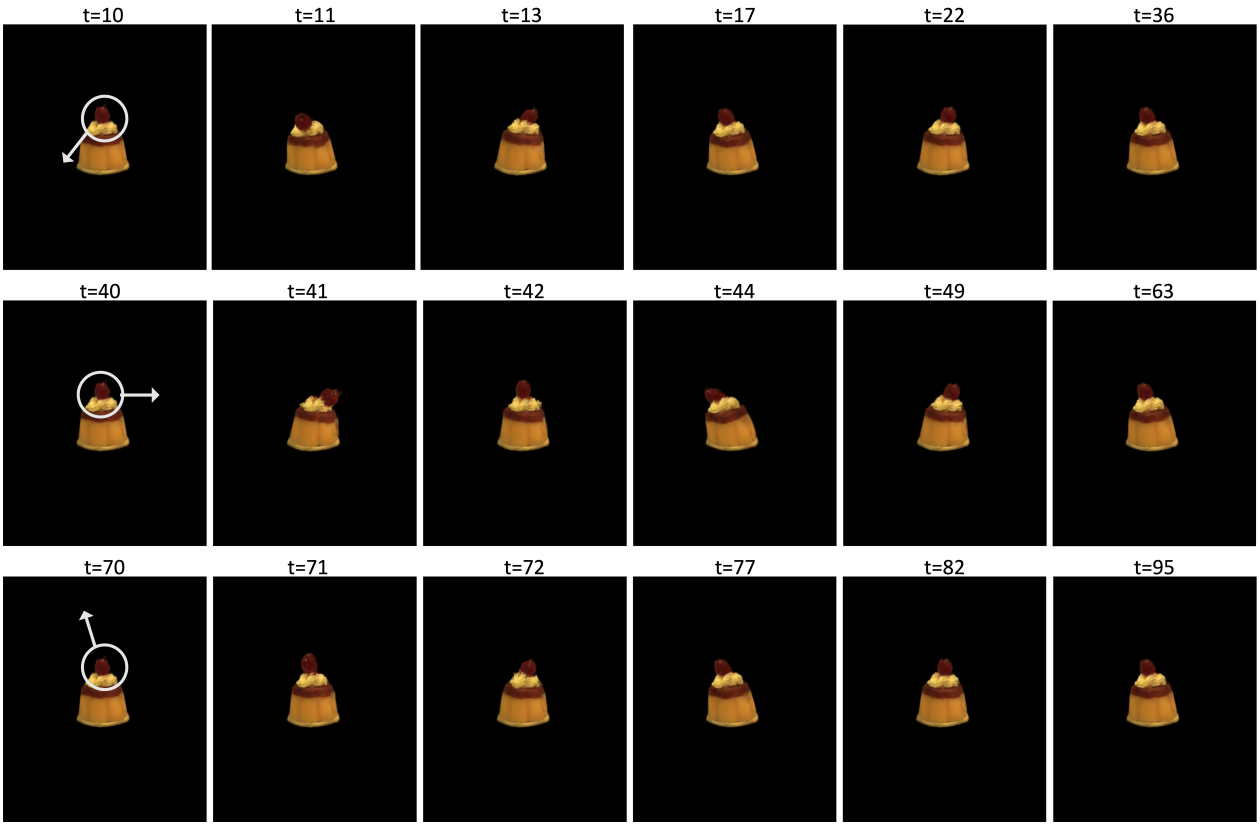

Figure A2. Interactive dynamics on pudding by GausSim. We randomly drag the pudding along the arrows' directions. GausSim can vividly simulate the dynamics given external forces, suggesting the effectiveness and robustness of our method.