
TPA-GEN: A MULTI-MODAL DATA GENERATION METHOD FOR TEXT AND PHYSICS-BASED ANIMATION

Anonymous authors

Paper under double-blind review

A SCENE GRAMMAR PRODUCTIONS

We use an attributed stochastic grammar as a hierarchical and structured representation that determines the scenario’s content with initial physical parameters and appearance settings. The grammar is decomposed into multiple levels of components which are sampled according to the production rules defined in [Table 1](#). The tree structure itself describes the scenario’s content, while the related attributes, which contain numerous features, specialize the content’s characteristics. [Table 2](#) presents a list of the attributes and features designed for each *node*.

Label	Production Rules
Scene	$\text{Scene} \rightarrow \text{TarObjSet} \oplus \text{Environment} \oplus \text{Render}$
Component-0	$\text{TarObjSet} \rightarrow \text{TargetObj}^+ \odot \text{TarObjSet}^*$ $\text{Environment} \rightarrow \text{ColObjSet}$
Component-*	$\text{ColObjSet} \rightarrow \text{CollisionObj}^* \odot \text{ColObjSet}^*$ $\text{TarObjSet}^* \rightarrow \text{TargetObj}^* \odot \text{TarObjSet}^*$

Table 1: **Production rule of the scenario stochastic grammar.** Here, TarObjSet is short for Target Object Set which includes a set of simulated object (TargetObj) with potential relationships; ColObjSet represents a set of non-movable collision objects (CollisionObj) serving as boundary conditions. Moreover, \oplus represents *and* relation, making the child elements mandatory; while \odot refers to *or* relation to connect optional child nodes; $^+$ means one or more and * means zero or more.

B DYNAMIC MODEL AND CONSTRAINTS

B.1 CONSTRAINTS

In practice, we use the following eight constraints to reveal object relationships and the constraints are checked and applied on selected object features. Every constraint consists of a list of operands " $[o_0, o_1, \dots, o_N]$ " and an ID number list " $[n_0, \dots, n_M]$ " which represents the unalterable criteria operand(s). Here, operand o_i refers to either a constant (value or vector that is always unalterable), or a *node-attribute-feature* pair in which case the value of the corresponding feature is fetched for computation. Additionally, the criteria operands must by default satisfy the given constraints to avoid ambiguities.

- *less_eq*($[o_0, \dots, o_N], [n_0, \dots, n_M]$): $o_0 \leq \dots \leq o_N$, with o_{n_0}, \dots, o_{n_M} stay unchanged during resampling.
- *less*($[o_0, \dots, o_N], [n_0, \dots, n_M]$): $o_0 < \dots < o_N$, with o_{n_0}, \dots, o_{n_M} stay unchanged during resampling.
- *larger_eq*($[o_0, \dots, o_N], [n_0, \dots, n_M]$): $o_0 \geq \dots \geq o_N$, with o_{n_0}, \dots, o_{n_M} stay unchanged during resampling.
- *larger*($[o_0, \dots, o_N], [n_0, \dots, n_M]$): $o_0 > \dots > o_N$, with o_{n_0}, \dots, o_{n_M} stay unchanged during resampling.
- *eq*($[o_0, \dots, o_N], [n_0, \dots, n_M]$): $o_0 = \dots = o_N$, with o_{n_0}, \dots, o_{n_M} stays unchanged during resampling.

- *same_dir*($[o_0, \dots, o_N], [n_0]$): o_i must be vectors, and for $\forall i \in [0, \dots, N]$ the angle between o_i and o_{n_0} is zero. Note that only one criterion operand is allowed to be present in this constraint.
- *oppo_dir*($[o_0, \dots, o_N], [n_0]$): o_i must be vectors, and for $\forall i \in [0, \dots, N]$ the angle between o_i and o_{n_0} is 180° . Note that only one criterion operand is allowed to be present in this constraint.
- *similar_dir*($[o_0, \dots, o_N], [n_0, \dots, n_M], \theta$): o_i must be vectors, and for $\forall i \in [0, \dots, N], \forall j \in [0, \dots, M]$ the angle between o_i and o_j is less or equal to θ . Here, o_{n_0}, \dots, o_{n_M} stays unchanged during resampling.

During sampling process, we validate the defined constraints after random sampling all features. The non-criteria operands that violate the constraints will be resampled to guarantee the correctness of the relation. If the criteria operands themselves violate the constraint, the resample process will be terminated and errors will be reported.

B.2 DYNAMIC MODEL

As introduced in the main paper, we have the following dynamic models: JUMP, DROP, THROW, PUSH and STRIKE. We summarize the basic constraints required for each intransitive dynamic model in Table 3 and transitive dynamical verbs in Table 4. In addition to these dynamic models, one can easily define and generate other dynamic models into our codebase with predefined interfaces for constraints. Some example definitions can be found in Table 5.

In all the above mentioned tables, "DM" refers to Dynamic Model, and the "sub" is used to denote the subjective object on which the verb in dynamic model focuses. And as shown in the last two columns of the table, objective objects (represented by "obj") are introduced with `from` and `to` directional relations.

C DATASHEET FOR DATASET

We answer the questions that applied to our work:

C.1 MOTIVATION

- **For what purpose was the dataset created?** We propose a method to generate Text and Physics-based Animation (TPA) for multi-modal model training. The goal is to expand the current problem domain of multi-modal learning from image-text understanding to vision-world dynamics understanding. We believe that this is the one of the beginning steps to enable multi-modal model's capability to understand our world from a model fundamental perspective.
- **Who created the dataset and on behalf of which entity?** The authors are from three different institutes. Yuxing Qiu, Minchen Li and Chenfanfu Jiang are from UCLA Multi-Physics Lagrangian-Eulerian Simulation Lab (MultiPLES). Feng Gao, was a Ph.D. student at UCLA when the project was initiated. Yin Yang is from University of Utah, Utah graphics lab. Govind Thattai joins the project as a individual contributor.
- This work is fully funded by UCLA computer science department and UCLA MultiPLES lab.

C.2 COMPOSITION

- **What do the instances that comprise the dataset represent?** Each generated instance consists of the following elements: a video of the rendered physics-based animation, a set of text descriptions of the animation, a 3D material points (position) of the object ID at frame ID.
- **How many instances are there in total?** As mentioned in the main paper, we propose a method to generate TPA data. In theory, one can generate as much as possible of TPA data with sufficient objects, types of materials and pre-defined motion types. To better illustrate the details of the data, we provide a sample set of TPA data which contains 500 instances.

-
- **Does the dataset contain all possible instances or is it a sample of instances from a larger set?** No.
 - **Are there recommended data splits?** We don't specify any splits. One can configure the split accordingly.

C.3 COLLECTION PROCESS

As we mentioned in the main paper, we propose a method to generate text and physics-based animation data. Since the data is synthetic, we don't require any human annotators to involve. Besides, in order to improve the quality of the generated texts, we take advantages of the most popular large language model, ChatGPT, to rewrite the generated description of the animation. Beside, we also use ChatGPT to help proposing label names of the 3D objects in a TPA instance. The 3D objects are generated with a SoTA text-3D generation model. Specifically we use the checkpoint of GPT-3.5-turbo-0301 as the backbone model.

C.4 PRE-PROCESSING/CLEANING/LABELING

- **3D object representation transformation:** The 3D objects are spatially discretized to material points at each time step. The positions of the material points representing each object (including object-of-interests and collision object)d are stored in separate PLY files.
- **Description rewriting:** We use ChatGPT to help cleaning the generated text description by rewriting the sentences without changing the meaning of it.

C.5 DISTRIBUTION

- **How will the dataset be distributed?** The implementation of the propose TPA data generation method contains three parts: 1, a scene sampling process code; 2, a binary execution of a physics-based simulation engine compiled from a set of source code. The source code of this simulation engine is based on an open-source version of Material Point Method (MPM); 3, data cleaning codes. Above implementations will be made public on GitHub.
- **When will the dataset be distributed?** The sample dataset and the implementation code will be distributed after the NeurIPS dataset track review process.

C.6 MAINTENANCE

- **Who will be supporting/hosting/maintaining the dataset?** UCLA MultiPLES lab will be the host of the proposed dataset generation method.
- **How can the owner of the dataset be contacted?** Yuxing Qiu (yuxqiu[AT]g.ucla.edu) and Chenfanfu Jiang (cffjiang[AT]math.ucla.edu) are the main contacts of the proposed data generation method.
- **Will the dataset be updated?** UCLA MultiPLES lab will keep updating the generation method including adding more 3D objects and types of motions into the generation process.

D DATASET SUBMISSION REQUIREMENTS

D.1 IMPACT AND CHALLENGES

We expect our proposed method and the data generated by this method can make a board impact to both multi-modal and computer graphics community.

- In terms of multi-modal understand, as discussed in the main paper, we aim to help this community to expand the problem domain from shallow vision-language alignment to deep comprehension of the knowledge space of vision-language-world dynamics. It could boardly impact specific research domain such as Text-to-Video/Simulation (T2V/S), robotics and intuitive physics.

-
- Our method could also make a large impact in the conventional computer graphics domain by reformulating the 3D animation creation process. Our generation process provides a rule-based generation of 3D animation scenes. On the other hand, once we have a reliable model that can generate 3D animation from human language, it could save huge amount of efforts to create 3D animation from scratch. The traditional pipeline usually requires very experienced artist and computer graphics researchers and engineers to collaborate even for a simple scene. Our method initialize the very first step towards fully automatic generation of 3D physics-based animations from text description.

D.2 LICENSING AND ACCESS

We would like to specify that we intend to utilize the **MIT license** for the method proposed in this paper to generate TPA data. This open-source license grants users the freedom to use, modify, and distribute the dataset while providing clear attribution to the original creators. By choosing the MIT license, I aim to foster collaboration, encourage innovation, and ensure that the generated data the code of the proposed method to generate remains accessible to the wider community for further exploration and development.

E DATA SAMPLES

Figure 1, **Figure 2**, **Figure 3**, and **Figure 4** demonstrate more sample examples of the proposed generative algorithm. For more demos, please check out this google drive: https://drive.google.com/drive/folders/1IbPJBmPLlzB4DPmXVx1eQhSr42WYjLU_?usp=sharing. One can check out the demos in the following file hierarchy (DM refers to concrete dynamic model names):

- DM_scene:
 - label_out.json (labels of all nodes, attributes and features)
 - value_out.json (quantitative values of corresponding setups)
 - sentence_original.json (sentence sampled from the proposed language model)
 - sentence_rewrite.json (sentence rewritten by ChatGPT)
 - render (a folder contains rendered results)
 - * FID.png (FID refers to frame ID)
 - * out.mp4 (video of rendering results)
- DM_3d: (3D object data)
 - _0_0_0_target_OID_FID.ply (OID refers to object ID; 3D material points (position) of object OID at frame FID)
 - _0_0_0_collision_FID.ply (point positions of collision objects)


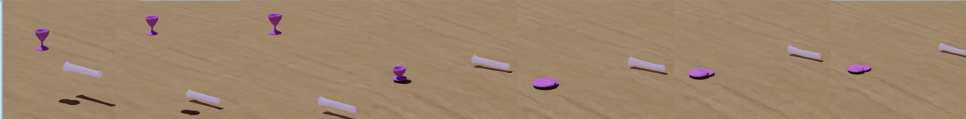

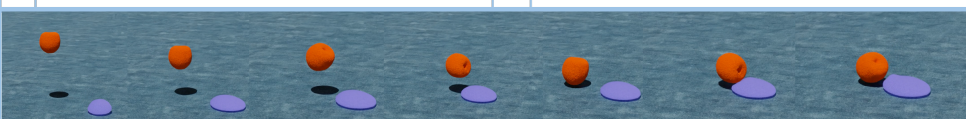

Video				
1	<p>Text Descriptions</p> <p>Sampled</p> <ol style="list-style-type: none"> Green Açaí berry which is mediumweight and large up in the air jump up and slow from lime Apricot that is medium-sized up in the air. Large Açaí berry which is rough jump upward and slow from light and emission Apricot. <p>ChatGPT</p> <ol style="list-style-type: none"> The Açaí berry, which is medium-weight, jumps upward from a light Apricot that emits light and has a lime color. The diffuse Açaí berry jumps upward from a smooth Apricot that emits light. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: green, - render_material: diffuse - shape: Açaí berry - size: large - vel_value: slow - vel_dir: up - pos: up in the air - physics_material: elastic - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: lime, - render_material: emission - shape: Apricot - size: super medium-sized - vel_value: fast - vel_dir: up - pos: up in the air - physics_material: sand - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: jump - sub: target 0 - obj: target 1 - dir_relation: to 		
Video				
2	<p>Text Descriptions</p> <p>Sampled</p> <ol style="list-style-type: none"> Violet fluid, and small Goblet which is mediumweight jump from purple, elastic, and mediumweight Telescope. Even surface and diffuse Goblet jump fast and upward from Telescope that is large, purple, and elastic. <p>ChatGPT</p> <ol style="list-style-type: none"> A mediumweight goblet, which is small and violet in color, jumps from a purple and elastic telescope.. A diffuse goblet with an even surface jumps quickly and upward from a large, elastic, and purple telescope. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: violet - render_material: diffuse - shape: Goblet - size: small - vel_value: fast - vel_dir: up - pos: up in the air - physics_material: fluid - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: purple - render_material: emission - shape: Telescope - size: super large - vel_value: slow - vel_dir: right - pos: on the ground - physics_material: elastic - density: mediumweight <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: jump - sub: target 0 - obj: target 1 - dir_relation: from 		
Video				
3	<p>Text Descriptions</p> <p>Sampled</p> <ol style="list-style-type: none"> Dandelion light Banana which is elastic jump up to glossy, even surface, and sand Apple which is green. Banana jump to Apple which is super large and green. <p>ChatGPT</p> <ol style="list-style-type: none"> The banana is light and elastic, and it jumps up to a glossy, even surface. The apple is green and sandy. The banana jumps to the super large and green apple. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: blue - render_material: diffuses - shape: Banana - size: super large - vel_value: medium-speed - vel_dir: up - pos: on the ground - physics_material: elastic - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: green - render_material: glossy - shape: Apple - size: super super large - vel_value: slow - vel_dir: up - pos: up in the air - physics_material: sand - density: heavy <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: jump - sub: target 0 - obj: target 1 - dir_relation: to 		
Video				
4	<p>Text Descriptions</p> <p>Sampled</p> <ol style="list-style-type: none"> Rough Apricot which is large and elastic medium-speed jump to Cantaloupe which is snow on the ground. Rough Apricot which is red and mediumweight jump to Cantaloupe which is snow on the ground. <p>ChatGPT</p> <ol style="list-style-type: none"> A rough and elastic Apricot, which is large in size, jumps at a medium speed and lands on a Cantaloupe covered in snow on the ground. A red and medium-weight Apricot, which is rough in texture, jumps and lands on a Cantaloupe covered in snow on the ground. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: blue - render_material: emission - shape: Cantaloupe - size: medium-sized - vel_value: medium-speed - vel_dir: backward - pos: on the ground - physics_material: snow - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: red - render_material: diffuse - shape: Apricot - size: super large - vel_value: medium-speed - vel_dir: horizonly - pos: up in the air - physics_material: elastic - density: mediumweight <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: jump - sub: target 1 - obj: target 0 - dir_relation: to 		
Video				
5	<p>Text Descriptions</p> <p>Sampled</p> <ol style="list-style-type: none"> Megaphone up in the air jump up to large Stopwatch that is rough and light up in the air. Pink and glossy Megaphone jump to Stopwatch up in the air. <p>ChatGPT</p> <ol style="list-style-type: none"> The megaphone in the air jumped up towards a rough and light large stopwatch. The pink and glossy megaphone jumped up into the air towards the stopwatch. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: red - render_material: glossy - shape: Stopwatch - size: large - vel_value: slow - vel_dir: forward - pos: up in the air - physics_material: fluid - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: pink - render_material: glossy - shape: Megaphone - size: medium-sized - vel_value: medium-speed - vel_dir: up - pos: up in the air - physics_material: sand - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: jump - sub: target 1 - obj: target 0 - dir_relation: to 		

Figure 1: Data samples of **DROP** dynamics.

<i>Node</i>	Attribute	Feature	Label Candidates
Env	Boundary Condition	BOUNDARY	Box, Floor
		TYPE	Sticky, Slip
		FRICITION FACTOR	Smooth, Even Surface, Rough, Extremely rough
	External Force	FORCE TYPE	Gravity, Wind
		<i>Force value</i>	<i>Dependent on</i> FORCE TYPE
Temporal	TOTAL FRAME	Short, Medium, Long	
Object	Appearance	COLOR	White, Red, Blue, Green, Lime, Orange, Yellow, Pink, Purple ...
		MATERIAL	Glossy, Matte
	Shape	SHAPE	Cube, Sphere, Cylinder, Mesh
		SIZE	Small, Medium-sized, Large, Super large
	Motion	VELOCITY VALUE	Slow, Medium-speed, Fast
		VELOCITY DIRECTION	Up, Down, Right, Left, Forward, Backward, Horizontal, Vertical
		INITIAL POSITION	On the ground, In the sky
	Physics	MATERIAL	Elastic, Rigid, Fluid, Snow, Mud, Sand, Granular
		<i>Young's Modulus</i>	<i>Dependent on</i> MATERIAL; Soft, Moderate-hardness, Hard, Rigid
		<i>Poisson Ratio</i>	<i>Dependent on</i> MATERIAL; Elastic, Rigid
DENSITY		Light, Medium-weight, Heavy	
FRICITION FACTOR		Smooth, Even Surface, Rough, Extremely rough	
Render	Background	LIGHT	Bright, Dark
		TEXTURE	Preset texture list or "random"
	Camera	CAMERA POSITION	Preset camera position
		<i>Viewpoint</i>	<i>Dependent on</i> CAMERA POSITION

Table 2: Attributes **with** features **associated for each scene node**. In the table, independent features are highlighted with the SMALL CAPS font style, whereas dependent features are labeled with *italic*. The final column lists examples of candidate labels for each feature. Each label is mapped to a specific value or range of values based on its semantics. Additional labels can be easily appended by providing a mapping between the label name and corresponding value ranges.

DM	Type	Constraint
JUMP	Basic	$similar_dir([0, 1, 0], (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})), [0], \theta_0)$
		$less_eq([v_{min}, (\text{sub}, \text{Motion}, \text{VELOCITY VALUE})], [0])$ (v_{min} defined by user)
	from	$eq([\mathbf{p}^{gt}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})], [0])$, with $\mathbf{p}^{gt} = [p_0^{gt}, p_1^{gt}, p_2^{gt}]$ Here, $p_i^{gt} = p_i^{obj} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i \in [0, 2]$
to	$similar_dir([\mathbf{d}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})], [0], \theta_1)$ Here, $\mathbf{d}^{gt} = (\mathbf{p}^{obj} - \mathbf{p}^{sub}) + \alpha \cdot [0, 1, 0]$ (α defined by user)	
DROP	Basic	$similar_dir([0, -1, 0], (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})), [0], \theta_0)$
		$larger_eq([v_{small}, (\text{sub}, \text{Motion}, \text{VELOCITY VALUE})], [0])$ (v_{small} defined by user)
		$less_eq([\mathbf{p}^{gt}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})], [0])$, $\mathbf{p}^{gt} = [p_0^{gt}, p_1^{gt}, p_2^{gt}]$; p_1^{gt} is user-defined threshold; p_0^{gt} and p_2^{gt} are the global minimum position
	from	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $p_i^{gt} = p_i^{sub} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i \in [0, 2]$
to	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $p_i^{gt} = p_i^{sub} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i = 0, 2$; $p_1^{gt} = s_1^{obj} + C$	
THROW	Basic (UP)	$similar_dir([\mathbf{v}_{dir}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})], [0], \theta_0)$, $\mathbf{v}_{dir}^{gt} = [C_0, 1, C_1]$
	Basic (DOWN)	$similar_dir([\mathbf{v}_{dir}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})], [0], \theta_0)$, $\mathbf{v}_{dir}^{gt} = [C_0, -1, C_1]$
	from	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$. Here, $p_i^{gt} = p_i^{sub} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$
	to	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $\mathbf{p}^{gt} = \mathbf{p}^{sub} + \mathbf{s}^{sub} + v_{value}^{sub} \cdot \mathbf{v}_{dir}^{sub} \cdot C$

Table 3: **Constraints in each single-object dynamic model.** In the table, \mathbf{s} , \mathbf{p} , v_{value} and \mathbf{v}_{dir} refers to object size, initial position, velocity value and velocity direction, separately; $C \geq 0$, $C_0 \in [-1, 1]$, $C_1 \in [-1, 1]$, $C_{small} \in [0, 0.1 * s_{max}^{sub}]$ represents random noise, and \pm means +/- are chosen randomly in practice. All the dynamic model has the basic constraints applied to the objects, with randomly sampled from, to, or NONE relation. Note that from and to relation will be chosen only when there are enough objects in the scenario. Specially, for THROW model, we will first sample to decide if it is "Throw UP" or "Throw DOWN" before defining the basic constraints.

DM	Type	Constraint
PUSH	Basic	$similar_dir(\mathbf{v}_{dir}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})), [0], \theta_0)$ Here $\mathbf{v}_{dir}^{gt} = \mathbf{p}^{obj} - \mathbf{p}^{gt-sub}$
		$less_eq(v_{large}, (\text{sub}, \text{Motion}, \text{VELOCITY VALUE})), [0])$ (v_{large} defined by user)
		$larger_eq(v_{small}, (\text{obj}, \text{Motion}, \text{VELOCITY VALUE})), [0])$ (v_{small} defined by user)
	Basic (Not from)	$eq(\mathbf{p}^{gt-sub}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})), [0])$ Here, $p_i^{gt-sub} = p_i^{obj} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i \in [0, 2]$
	from	$eq(\mathbf{p}^{gt-sub}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})), [0])$ Here, $p_i^{gt-sub} = p_i^{obj-extra} \pm (s_i^{sub} + s_i^{obj-extra} + C) * 0.5$ for $i \in [0, 2]$
		$eq(\mathbf{p}^{gt-obj}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})), [0])$ Here, $\mathbf{p}^{gt-obj} = \mathbf{p}_i^{gt-sub} + K * \frac{\mathbf{p}^{gt-sub} - \mathbf{p}^{obj-extra}}{\ \mathbf{p}^{gt-sub} - \mathbf{p}^{obj-extra}\ }$ $K = 0.5 \cdot \max_{i=0,1,2}(s_i^{sub} + s_i^{obj}) + C$
	to	$eq(\mathbf{p}^{gt-obj-extra}, (\text{obj-extra}, \text{Motion}, \text{INITIAL POSITION})), [0])$ Here, $\mathbf{p}^{gt-obj-extra} = \mathbf{p}^{obj} + K * \mathbf{v}_{dir}^{sub}$ $K = 0.5 \cdot \max_{i=0,1,2}(s_i^{obj} + s_i^{obj-extra}) + C$
STRIKE	to	$\mathbf{p}^{to} = \mathbf{p}^{obj-extra}$
	Not to	Random sample \mathbf{p}^{to}
	Basic	$similar_dir(\mathbf{p}^{to} - \mathbf{p}^{sub}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})), [0], \theta_0)$
		$similar_dir(\mathbf{p}^{to} - \mathbf{p}^{obj}, (\text{obj}, \text{Motion}, \text{VELOCITY DIRECTION})), [0], \theta_0)$
		$less_eq(v_{large}, (\text{sub}, \text{Motion}, \text{VELOCITY VALUE})), [0])$ (v_{large} defined by user)
		$less_eq(v_{large}, (\text{obj}, \text{Motion}, \text{VELOCITY VALUE})), [0])$ (v_{large} defined by user)
from	$eq(\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})), [0])$ Here, $p_i^{gt} = p_i^{sub} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i \in [0, 2]$	

Table 4: **Constraints in each multiple-object dynamic model.** These dynamic models are applied to at least two objects, one representing the subjective and the other representing the objective ("sub" and "obj" in the table). *from* and *to* relation will include another objective object, named "obj-extra" in the table. $C \geq 0$ represents random noise, and \pm means +/- are chosen randomly in practice.

DM	Type	Constraint
FLY	Basic	$similar_dir([\mathbf{v}_{dir}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})], [0], \theta_0),$ $\mathbf{v}_{dir}^{gt} = [C_0, 0, C_1]$
		$less_eq([v_{large}, (\text{sub}, \text{Motion}, \text{VELOCITY VALUE})], [0])$ (v_{large} defined by user)
		$less_eq([\mathbf{p}^{gt}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})], [0]),$ $\mathbf{p}^{gt} = [p_0^{gt}, p_1^{gt}, p_2^{gt}], p_1^{gt}$ is user-defined threshold p_0^{gt} and p_2^{gt} are the global minimum position
	from	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $p_i^{gt} = p_i^{sub} \pm (s_i^{sub} + s_i^{obj} + C) * 0.5$ for $i \in [0, 2]$
to	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $\mathbf{p}^{gt} = \mathbf{p}^{sub} + \mathbf{s}^{sub} + v_{value}^{sub} \cdot \mathbf{v}_{dir}^{sub} \cdot C$	
SLIDE	Basic	$similar_dir([\mathbf{v}_{dir}^{gt}, (\text{sub}, \text{Motion}, \text{VELOCITY DIRECTION})], [0], \theta_0),$ $\mathbf{v}_{dir}^{gt} = [C_0, 0, C_1]$
	Basic (Not from)	$less_eq([\mathbf{p}^{gt-min}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION}), \mathbf{p}^{gt-max}], [0, 2])$ Here $\mathbf{p}^{gt-*} = [p_0^{gt-*}, p_1^{gt-*}, p_2^{gt-*}], p_1^{gt-min} = s_1^{sub}, p_1^{gt-max} = s_1^{sub} + C_{small}$ p_i^{gt-*} refers to global * position for $i = 0, 2$ and * refers to min/max
	from	$eq([\mathbf{p}^{gt}, (\text{sub}, \text{Motion}, \text{INITIAL POSITION})], [0]),$ with $\mathbf{p}^{gt} = [p_0^{gt}, p_1^{gt}, p_2^{gt}]$ Here, $p_1^{gt} = p_1^{obj} + (s_1^{sub} + s_1^{obj} + C_{small}) * 0.5$ Random sample $p_i^{gt} \in [p_i^{obj} - s_i^{obj} * 0.5, p_i^{obj} + s_i^{obj} * 0.5]$ for $i = 0, 2$
	to	$eq([\mathbf{p}^{gt}, (\text{obj}, \text{Motion}, \text{INITIAL POSITION})], [0])$ Here, $\mathbf{p}^{gt} = \mathbf{p}^{sub} + \mathbf{s}^{sub} + v_{value}^{sub} \cdot \mathbf{v}_{dir}^{sub} \cdot C$

Table 5: **Example of other possible dynamic models.** One can easily define and implement additional dynamic models with in our codebase.

6	Video			Simulation Specs	Target 0: - color: yellow, - render_material: glossy - shape: Stopwatch - size: small - vel_value: fast - vel_dir: up - pos: up in the air - physics_material: snow - density: light Target 1: - color: purple, - render_material: matte - shape: Goblet - size: super medium-sized - vel_value: fast - vel_dir: up - pos: up in the air - physics_material: elastic - density: heavy Dynamic Model: - verb: jump - sub: target 0 - obj: target 1 - dir_relation: to
	Text Descriptions	Sampled	ChatGPT		
7	Video			Simulation Specs	Target 0: - color: green, - render_material: glossy - shape: Stopwatch - size: large - vel_value: medium-speed - vel_dir: down - pos: up in the air - physics_material: fluid - density: mediumweight Target 1: - color: lime - render_material: glossy - shape: Trinket box - size: medium-sized - vel_value: medium-speed - vel_dir: up - pos: up in the air - physics_material: fluid - density: light Dynamic Model: - verb: jump - sub: target 1 - obj: target 0 - dir_relation: from
	Text Descriptions	Sampled	ChatGPT		
8	Video			Simulation Specs	Target 0: - color: violet - render_material: matte - shape: Binoculars - size: small - vel_value: slow - vel_dir: up - pos: up in the air - physics_material: fluid - density: light Target 1: - color: red - render_material: diffuse - shape: Apple - size: super super large - vel_value: medium-speed - vel_dir: backward - pos: up in the air - physics_material: elastic - density: heavy Dynamic Model: - verb: jump - sub: target 0 - obj: target 1 - dir_relation: to
	Text Descriptions	Sampled	ChatGPT		
9	Video			Simulation Specs	Target 0: - color: purple - render_material: emission - shape: Top hat - size: large - vel_value: medium-speed - vel_dir: up - pos: up in the air - physics_material: elastic - density: light Target 1: - color: lime - render_material: matte - shape: Globe - size: super large - vel_value: fast - vel_dir: up - pos: on the ground - physics_material: sand - density: mediumweight Dynamic Model: - verb: jump - sub: target 1 - obj: target 0 - dir_relation: to
	Text Descriptions	Sampled	ChatGPT		
10	Video			Simulation Specs	Target 0: - color: violet - render_material: diffuse - shape: Sundial - size: large - vel_value: fast - vel_dir: horizontally - pos: on the ground - physics_material: sand - density: light Target 1: - color: blue - render_material: emission - shape: Goblet - size: large - vel_value: fast - vel_dir: down - pos: up in the air - physics_material: snow - density: mediumweight Dynamic Model: - verb: jump - sub: target 0 - obj: target 1 - dir_relation: to
	Text Descriptions	Sampled	ChatGPT		

Figure 2: Data samples of **DROP** dynamics.

1	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> Glossy and blue Top hat drop slowly to emission , smooth , and medium-sized Fedora on the ground . Fluid and rough Top hat drop to emission Fedora which is light , medium-sized , and white on the ground . <p>ChatGPT</p> <ol style="list-style-type: none"> A blue and glossy top hat fell slowly to the ground, while a smooth and medium-sized fedora lay nearby. A fluid and rough top hat dropped to the ground, where it met a light, medium-sized, and white fedora. 	Simulation Specs	<p>Target 0:</p> <ul style="list-style-type: none"> - color: white - render_material: emission - shape: Fedora - size: medium-sized - vel_value: slow - vel_dir: right - pos: on the ground - physics_material: elastic - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: blue - render_material: glossy - shape: Top hat - size: medium-sized - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: fluid - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to
2	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> Medium-sized Avocado that is pink , light , and emission up in the air down and slow drop to Tumbler on the ground. Fluid Avocado which is medium-sized up in the air drop slow to Tumbler which is violet and even_surface on the ground. <p>ChatGPT</p> <ol style="list-style-type: none"> The avocado is a medium size and has a pink color. It was thrown up in the air and slowly dropped into a tumbler on the ground . The avocado was fluid and medium-sized. It was thrown up in the air and slowly dropped into a violet, even-surfaced tumbler on the ground. 	Simulation Specs	<p>Target 0:</p> <ul style="list-style-type: none"> - color: violet - render_material: matte - shape: Tumbler - size: super large - vel_value: slow - vel_dir: left - pos: on the ground - physics_material: fluid - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: pink - render_material: emission - shape: Avocado - size: super medium-sized - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: elastic - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to
3	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> Rough Compass drop slowly and down to diffuse and elastic Stopwatch that is even_surface on the ground. Light , diffuse , and super large Compass up in the air down drop to super large , elastic , and mediumweight Stopwatch. <p>ChatGPT</p> <ol style="list-style-type: none"> The compass is rough and drops slowly, eventually landing on the ground where the elastic stopwatch is diffuse and has an even surface. A light, diffuse, and super large compass is dropped from the air, eventually landing on the ground where the elastic, medium-weight stopwatch is waiting. 	Simulation Specs	<p>Target 0:</p> <ul style="list-style-type: none"> - color: red - render_material: diffuse - shape: Compass - size: super large - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: elastic - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: violet - render_material: diffuse - shape: Stopwatch - size: super super large - vel_value: slow - vel_dir: right - pos: on the ground - physics_material: elastic - density: mediumweight <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 0 - obj: target 1 - dir_relation: to
4	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> Fluid and extremely rough Binoculars up in the air drop to Locket which is green on the ground . Binoculars drop slow and down to light Locket which is green. <p>ChatGPT</p> <ol style="list-style-type: none"> Fluid and extremely rough binoculars fall from the air and land on a green locket on the ground. The binoculars fall slowly and downward onto a lightweight green locket. 	Simulation Specs	<p>Target 0:</p> <ul style="list-style-type: none"> - color: pink - render_material: emission - shape: Binoculars - size: medium-sized - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: fluid - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: green - render_material: diffuse - shape: Locket - size: super large - vel_value: medium-speed - vel_dir: backward - pos: on the ground - physics_material: elastic - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 0 - obj: target 1 - dir_relation: to
5	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> Light Top hat which is orange and extremely rough up in the air down drop to heavy Cantaloupe which is matte. Top hat drop to Cantaloupe on the ground . <p>ChatGPT</p> <ol style="list-style-type: none"> The light orange top hat, which was rough, was thrown up in the air and then dropped down onto the heavy, matte cantaloupe on the ground. The top hat fell onto the cantaloupe on the ground. 	Simulation Specs	<p>Target 0:</p> <ul style="list-style-type: none"> - color: pink - render_material: matte - shape: Cantaloupe - size: super large - vel_value: slow - vel_dir: up - pos: on the ground - physics_material: fluid - density: heavy <p>Target 1:</p> <ul style="list-style-type: none"> - color: orange - render_material: emission - shape: Top hat - size: small - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: elastic - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to

Figure 3: Data samples of **JUMP** dynamics.

6	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> 1. White Anvil up in the air slowly drop to Top hat that is matte on the ground. 2. Mediumweight and even_surface Anvil that is sand drop slow to Top hat that is fluid. <p>ChatGPT</p> <ol style="list-style-type: none"> 1. A white anvil was dropped from the air and slowly landed on the ground next to a matte top hat. 2. An even-surfaced and sand-colored anvil was dropped slowly and turned into a fluid, landing next to a top hat. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: green, - render_material: matte - shape: Top hat - size: small - vel_value: medium-speed - vel_dir: backward - pos: on the ground - physics_material: fluid - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: white, - render_material: glossy - shape: Anvil - size: super large - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: sand - density: mediumweight <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to 	
7	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> 1. Elastic and pink Tumbler which is heavy and even_surface up in the air drop to smooth Quill that is sand on the ground. 2. Pink , large , and elastic Tumbler drop down and slow to smooth , sand , and light Quill on the ground. <p>ChatGPT</p> <ol style="list-style-type: none"> 1. The pink tumbler, which is elastic and heavy, falls from the air onto a smooth surface, landing on a pink quill. 2. A pink, large, and elastic tumbler falls slowly to the ground, landing on a smooth and light quill that is covered in sand. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: pink - render_material: glossy - shape: Quill - size: medium-sized - vel_value: medium-speed - vel_dir: horizonly - pos: on the ground - physics_material: sand - density: light <p>Target 1:</p> <ul style="list-style-type: none"> - color: pink - render_material: glossy - shape: Tumbler - size: super large - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: elastic - density: heavy <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to 	
8	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> 1. Monocle that is small , rough , and red up in the air drop to elastic and smooth Anvil that is mediumweight on the ground. 2. Monocle drop down and slowly to Anvil which is smooth and yellow. <p>ChatGPT</p> <ol style="list-style-type: none"> 1. A small, rough, and red monocle fell slowly from the air onto a smooth and elastic surface. 2. A glossy and elastic anvil, which was super large and smooth, caught the slowly dropping monocle. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: red - render_material: matte - shape: Monocle - size: small - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: snow - density: heavy <p>Target 1:</p> <ul style="list-style-type: none"> - color: yellow - render_material: glossy - shape: Anvil - size: super large - vel_value: fast - vel_dir: up - pos: on the ground - physics_material: elastic - density: mediumweight <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 0 - obj: target 1 - dir_relation: to 	
9	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> 1. Orange , glossy , and super large Pocket watch that is rough drop slowly and down to super large , glossy , and smooth Cantaloupe on the ground. 2. Pocket watch that is rough and super large up in the air down drop to glossy , red , and super large Cantaloupe. <p>ChatGPT</p> <ol style="list-style-type: none"> 1. A rough and super large pocket watch falls from the air, eventually landing on a super large, glossy, and red cantaloupe. 2. A pocket watch with a glossy and fluid appearance drops slowly from the air, eventually landing on a super large, glossy, and red cantaloupe that is also fluid. The watch is orange and mediumweight with a rough texture. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: orange - render_material: glossy - shape: Pocket watch - size: super large - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: fluid - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: red - render_material: glossy - shape: Cantaloupe - size: super large - vel_value: slow - vel_dir: forward - pos: on the ground - physics_material: fluid - density: light <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 0 - obj: target 1 - dir_relation: to 	
10	Video			
	Text Descriptions	<p>Sampled</p> <ol style="list-style-type: none"> 1. heavy , green , and rough Compass which is large down drop to red , diffuse , and small Apple that is mediumweight. 2. Heavy and rough Compass which is green up in the air drop to mediumweight , diffuse , and rough Apple on the ground. <p>ChatGPT</p> <ol style="list-style-type: none"> 1. The big compass fell slowly and landed on an apple that is medium-weight and has a diffuse texture. 2. The heavy and rough green compass dropped from the air and landed on a medium-weight, diffuse, and rough-textured apple on the ground. 	<p>Simulation Specs</p> <p>Target 0:</p> <ul style="list-style-type: none"> - color: red - render_material: diffuse - shape: Apple - size: small - vel_value: slow - vel_dir: horizonly - pos: on the ground - physics_material: fluid - density: mediumweight <p>Target 1:</p> <ul style="list-style-type: none"> - color: green - render_material: matte - shape: Compass - size: large - vel_value: slow - vel_dir: down - pos: up in the air - physics_material: sand - density: heavy <p>Dynamic Model:</p> <ul style="list-style-type: none"> - verb: drop - sub: target 1 - obj: target 0 - dir_relation: to 	

Figure 4: Data samples of JUMP dynamics.