L-MCTS Framework of Massive Spatio-temporally Plausible Content Generation

Chia-Hsing Chiu, Yu-Chi Lai, Andrew Chen, Pai-Yi Su, Wei-Yao Ku, Wen-Kai Tai, and Shih-Syun Lin

***Abstract*—Spatio-temporal content, cascades of physical and procedural events, is one of the most indispensable elements in games and animations nowadays. Generally, designing such scenes needs expertise to achieve both procedural rules and targeting criteria such as complexity, appealingness, and difficulty, and hence, to generate various instances is time-consuming and manpower intensive. Therefore, we propose a general framework that combines the L-system and the Monte Carlo Tree Search (MCTS), named L-MCTS, whose automatic workflow, integrating expertise and domain knowledge, is capable of massive production with targeted criteria. More specifically, in order for systematic exploration, we break the content space of near-infinity degrees of freedom into three parts: forming cascading events with L-system, positioning elements with inverse embedding, and activating the content with initial conditions. Furthermore, we adjust Monte Carlo Tree Search (MCTS) to efficiently, systematically, and coherently sample plausible instances through the decomposed space. In order to automatize the work flow with objective evaluations, we empirically describe a metric development procedure derived from experiences and questionnaire postulation to encode the targeted criteria using content-describing tree complexity and parameter variations. Finally, we demonstrate its generality of our framework with three different applications and evaluate their effectiveness and efficiency through several experiments and user studies.**

***Index Terms*—L-system layout representation, Monte Carlo Tree Search generation and evaluation, difficulty evaluation, procedural content generation, layout generation**

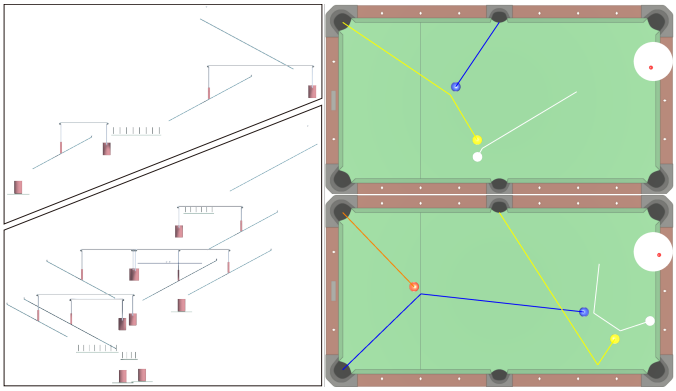


Fig. 1. This shows the examples of our targeting spatio-temporal content, where the left shows the chain reaction that propagates the interaction through various components and the right shows the "artistic pool" that challenges the player to strike all the objects balls in one shot.

I. Introduction

Spatio-temporal content driven by cascading events is commonly seen in games and animations. For examples, the chain reaction is such a visual treat that leaves people breathless when components trigger one another (left of Fig. 1), and the billiard trick shots that strike all object balls into the pockets in one shot is also appealing to the audience hence being referenced as the "artistic pool" (right of Fig. 1).

However, generating such content with physical plausibility is challenging, particularly when additional indicators such as appealingness or difficulties must be considered in practice. Furthermore, designing such content usually requires expertise and domain knowledge, and thus, it is difficult to automate the workflow. All the mentioned issues make the spatio-temporal content generation a manpower-intensive and time-consuming task in the fast-paced multimedia industry. Therefore, this work aims at developing a general framework that automatically generates massive amounts of plausible content without the presence of experts.

Game level design [1]–[4] encodes element distributions as inter-pixel relationship for pattern- and learning- based synthesis. Pixel-based encoding only provides static spatial information which is not sufficient for modeling temporal interactions of cascading events while our framework encodes temporal element-wise interactions with L-system. Furthermore, content modeling and animation synthesis formulate spatial structural requirements, spatial distribution goals, and space-time constraints as optimization problems, and Monte Carlo Optimization (MCO) is one of the successful solver for modeling trees and buildings [5], [6], generating levels and layouts [1], [2], [7]–[9], and synthesizing space-time animations [10]–[12]. However, random exploration of the plausible space is time-consuming. Therefore, our framework systematically parameterizes the spatio-temporal content in three aspects: to describe and deduce the element-wise interactions, to describe the element-wise spatial information with the geometric parameters, and to trigger the cascading events with possible initial conditions. In order to search for plausible spatio-temporal content, we also involve the simulation phase, which verifies validness and objectiveness of the content, resulting in a 4-level Monte Carlo Tree Search, named as L-MCTS. However, it is impractical and inefficient to search the extremely high-dimensional geometric parameters. Thus, we involve the domain knowledge to develop inverse embedding for narrowing parameter searching into a smaller space. Even though our L-MCTS allows an arbitrary function of plausibility, designing a proper metric to reflect objectiveness, such as appealingness, is still an unsolved problem. We free ourselves from domain dependency by using

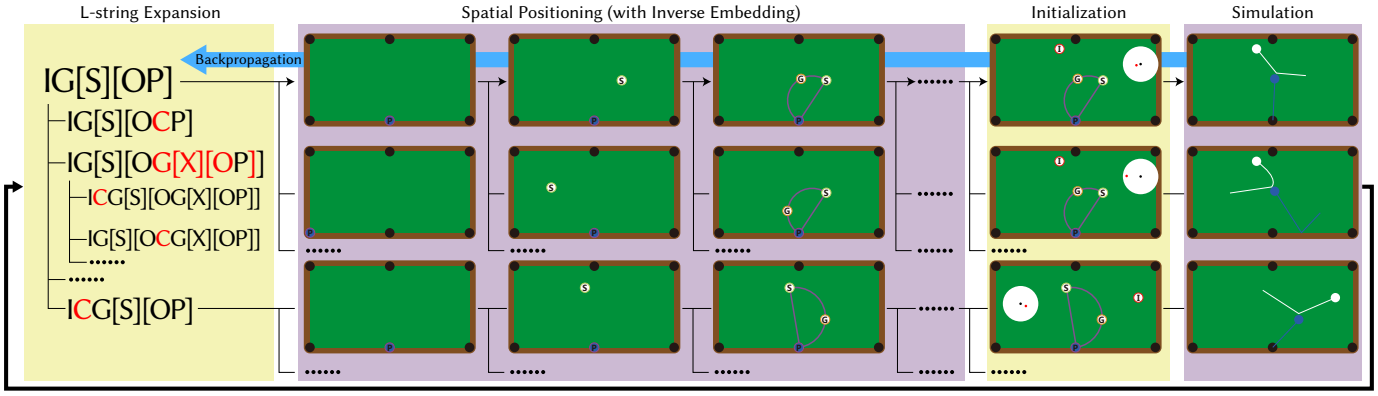


Fig. 2. Our L-MCTS uses MCTS to hierarchically explore the spatio-temporal content space. The system is divided into 4 phases from left to right, **L-string expansion**, from top to bottom, which describes and deduces the sequential interactions, **spatial positioning** that applies inverse embedding for better searching efficiency, **activating initialization** which picks up the initiation conditions, and **validation simulation** that evaluates the generated content, while the evaluation result is backpropagated to the tree for concentrating searching on more promising parameters.

expertise expression and questionnaire postulation to reveal subjective opinions along with domain-free statistics, such as tree complexity and parameter variations.

After conceptual deduction and development, we have applied our framework on three applications, one-shot-all-eliminate breakout clone, chain reaction, and one-shot-all-sink billiard while having experiments and user studies to determine their proper parameters. The implementation can massively generate breakout levels, chain reaction setups, and billiard layouts with the target degree of the level difficulty, visual complexity, and playing difficulty, respectively. While comparing against other baselines, our framework can effectively enhance the search efficiency for content generation. We have also verified the effectiveness of our designed visual complexity and difficulty metric through a set of user studies. Overall, our framework achieves the following three contributions.

1) We propose **L-MCTS**, which incorporates L-system and Monte Carlo Tree Search (MCTS), as a general framework to automatically generate massive amounts of spatio-temporal content that fulfills the user-defined objectiveness.
2) We further enhance the search efficiency by decomposing the spatio-temporal content into the L-system-described cascading sequence, spatially varied geometric parameters, and activated initial conditions for systematic exploration.
3) We develop a mechanism to design evaluation metrics by using questionnaires to collect users' opinions and to postulate user preference while conditioning the evaluation with domain-free information, such as tree complexity and parameter variations. Questionnaire postulation can be easily targeted to extract domain knowledge and preference of other applications along with domain-free formulation. Therefore, both make our procedure general for other applied domains.

After all, our system can effectively and automatically generate spatio-temporal instances under user specified degree of criteria.

II. Related Works

The proposed L-MCTS framework aims at intuitive and controllable automatic generation of various spatio-temporal content. Therefore, the following focuses on the core related works of content generation, which include L-system-based procedural modeling, procedural game level generation, space-time constraint simulation, and Monte Carlo optimization.

L-system-based procedural modeling first comes into our mind. Research [13]–[15] apply rewriting and turtle interpretation of L-system grammars to describe various branching plants, and others [16], [17] extend the concept to describe variant objects. Additionally, there are also research to improve interactivity and intuitiveness of L-system by providing graph-based grammars [18] or visual tools [19], [20]. It is compact and able to construct various models with a small set of derived parameters along with its goodness at systematic and hierarchic deductions of all possible variations. However, to the best of our knowledge, there are only few grammar descriptions and production rules designed for temporal events [8]. Since spatio-temporal content consists of sequences of semantic interactions, for example, the sequence of ball-to-ball and ball-to-cushion collisions in billiard, it seems to be reasonable to derive L-system representations for these interactions because of its easy deduction. However, there are still challenges listed as the following. Traditional L-system only records interactions and components involved, but it does not consider their spatial positioning and temporal actions. Even though its variation, the parametric L-system, is capable of representing the continuous parameters through the production rules, the complex physical phenomena makes the design of effective grammars difficult. Therefore, we adapt Monte Carlo Tree Search (MCTS) to explore the possible spatio-temporal space to seek reasonable solutions. Furthermore, target objectiveness must be numerically described, and thus, we develop an empirical achievement metric to describe users' goals.

Procedural game level generation majorly focuses on static level generation for 2D platform, shooting, and strategy games, and their methods can be categorized as grammar-based, searching-based and learning-based. Grammar-based

methods [3], [21]–[23] first analyze exemplar levels for designing layout patterns and grammars and develop synthesis mechanisms to create various levels accordingly. However, formulation requires expertise and synthesized evaluation requires test plays. Both result in a large amount of manual work. Searching-based algorithms [1], [2], [7], [24]–[26] design evaluation metrics and evolution strategies for searching for possible target goals. The synthesized quality heavily depends on the evaluation metric while the criteria are hard to consider all gaming aspects, especially psychological parts. Learning-based methods [4], [27]–[29] translate elements as various pixels and transform a level into a 2D image for plugging into various image learning frameworks, such as Variational Auto-Encoder (VAE) and Generative Adversarial Network (GAN). To the best of our knowledge, all these works only consider element distribution and inter-element relationship, but different actions in the same layout may result in different interactions. Additionally, they generally use heuristic mechanisms to evaluate the synthesis quality, but we cannot directly apply the heuristics into our generator. Therefore, this work develops a mechanism to hierarchically and systematically represent and deduce all possible interaction sequences while developing an empirical evaluation metric. Furthermore, we also apply MCTS to explore the plausible content space for the target goal.

Space-time constraint simulation aims at achieving key-framed design and control using space-time constraints in order to generate artistically satisfied plausible animations. There are various applications including controlling multi-body trajectories [10]–[12], [30], [31], manipulating skeleton motions [32]–[34], editing deformable objects [35], [36], and controlling fluid [37], [38]. We can categorize the solutions into local greedy control and global optimization. Local control incrementally and consecutively adds external forces to achieve the targeted appearance and configuration [30], [33]–[35], [37], [38]. The control is based on frame-by-frame information but does not consider the entire animation composition. Therefore, global methods formulate it as sequential quadratic programming (SQP) to solve as a linear system using Newton's method [32], [36] and random sampling [10], [11], [31], [33], [34], [38]. Furthermore, a previous work [39] uses graph-based analysis to aid domino placement with user-sketched tracks. Studies [40], [41] parameterize mechanics' geometries and motions as kinematic and dynamic optimization for choosing their layouts. We summarize the above methods as the inverse solutions to model the content distribution with predefined constraints. However, these methods generally require domain knowledge to formulate domain-dependent solutions, i.e., they cannot be easily transferred to other applications as mentioned by Chenney *et al.* [10]. Different from the aforementioned methods, our system first designs forward sampling of cascading interactions, interaction-wise geometric parameters, and initial conditions as a general and domain-free framework. At the same time, we take inverse postulation to derive inverse embedding with simplified procedure models for interaction-wise prediction in order to accelerate the searching procedure. Furthermore, past studies target at generating one artist-satisfied animation, but ours is intended for massive production of animations achieving certain target goals with variation. Also, the solutions were usually limited to a finite domain for tractability and solvability, by contrast our method can sample from the infinite space.

Monte Carlo Optimization (MCO), including Monte Carlo Tree Search (MCTS), obtains the optimal result with designated strategies using random sampling [42]. Past research [5], [6] model the distribution of trees and buildings with manually developed grammars and parameters along with designed evaluation metrics for design and reconstruction. Game content generation also introduces MCO for music composition [8], 2D platform level design [1], [2], and collision layout synthesis [7]. Twigg *et al.* randomly select dynamic parameters to have plausible simulations and let a user pick up his/her desired [11] or choose a backward state based on the current one [12]. [9] apply layout perturbations and progressively approximate the success probability distribution of a given scenario over the layout design space in order to optimize robustness. Research [33], [34], [40], [43] select the top-most from a set of randomly choosing plausible forward-stepping actions, and a study [44] further takes decision variations into consideration by designing a temporal adjustable cost function. Furthermore, [45] facilitate the MCO to decide the consensus threshold of preference relations. However, randomly jumping around the plausible space is inefficient and cannot further explore those possibly important regions for traditional MCO. Therefore, Chenney *et al.* [10] apply Markov Chain Monte Carlo sampling in the parameter space for a plausible motion fulfilling key-framed constraints for automation and better efficiency while Anderson *et al.* [31] extend the concept to flock simulation. MCTS is also introduced for coherent exploration and acceleration. The most famous application lies in the opponent AI of Go for AlphaGo and AlphaGo Zero [46], and it is also widely used in content generation. In addition, Qi *et al.* [47] propose a fast MCTS pruning algorithm to achieve efficient spacecraft trajectory optimization, and Hong *et al.* [48] propose a sampling strategy for decision variables to solve the large-scale multiobjective optimization via MCTS. We summarize the above methods as the forward solutions to model the content distribution based on sampling strategies. It is important to have a systematic and thorough exploration manner, while MCTS further requires it to be a hierarchical searching fashion. As far as we know, there exists no work representing the spatio-temporal content for forward methods. One similar work comes to our vision is the one proposed by [9]. They model interactions with a casual graph and randomly explore possible parameters of all nodes, but do not intend to explore the interaction space formed by all possible configurations, hence lacking of variations in interactions. On the other hand, we develop an L-system representation to encode the interactions and embed geometric and physical parameters to construct the possible spatio-temporal space in order for coherent and thorough exploration. To the best of our knowledge, our framework is the first intending to generalize the MCTS concept into generating spatio-temporal content with targeting goals. Furthermore, we also enhance search efficiency by incorporating the inverse concept to have a hybrid solution, which have the benefits of both the forward

generality and inverse efficiency by reducing the effectively searching domain.

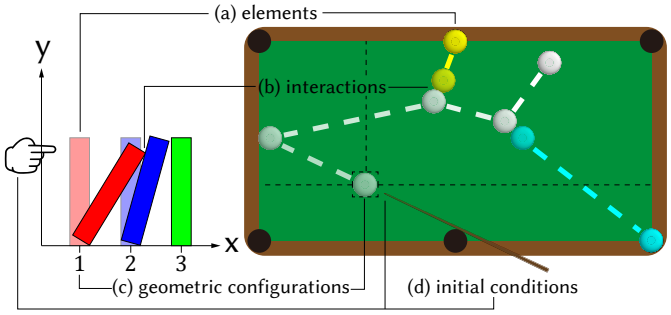


Fig. 3. This shows two spatio-temporal examples of domino toppling and billiard shooting with their corresponding denotations. The content generally consists of 4 parts, which are (a) **element**s, such as dominoes and balls, (b) **interaction**s, such as domino topplings and ball collisions, (c) **geometric configuration**s, such as the position of dominoes and balls, and (d) **initial condition**s, such as the initial force of toppling and striking.

III. Problem Formulation

This work aims to design a fully-automatic framework which is capable of massive generation of spatio-temporal content given a targeted goal. Similar to space-time constraints proposed by [32], this work defines the spatio-temporal content as cascading interactions, denoted as **interaction**s, between objects, referred as **element**s, where **element**s are placed on designated positions, denoted as **geometric configuration**s, and the interactions follow designated procedural rules such as physics while triggering with designed **initial condition**s in the beginning. The degree of goal achievement, $f(\cdot)$, is evaluated based on the process of interactions from triggering until all objects stop moving, and it is fully application dependent while its evaluation criteria, which are usually empirical, requires domain knowledge and expertise. Furthermore, this work sets it to 0 when an interaction fails. Fig. 3 shows two examples, a domino run and a one-shot-all-sink billiard set-up. **Element**s for the domino are the red, blue, and green dominoes with an equally separated x-axis distance, represented as the **geometric configuration**s. It is triggered with the pushing force, **initial condition**, and the **interaction**s are the topplings of the red on the blue, and the blue on the green. The process is defined as a success if all dominoes are toppled in once, or to be a failure vice versa. Similarly, in the billiard set-up, object balls together with the cue ball are referred as the **element**s, and the **geometric configuration**s distribute them on the table. The cascading events begin with the shot, i.e., the **initial condition**, and the **interaction**s, which are collisions among balls, cushions, and pockets, act as trajectories. A successful one-shot-all-sink billiard set-up is determined by whether all object balls are sunk into the pockets in one shot or not. Furthermore, a metric can be designed to determine whether the set-up reaches the target goal or not, such as difficulty for the billiard set-up. Details are given in Section V together with two more examples including the one-shot-all-eliminate breakout clone and chain reaction application. This work denotes the spatio-temporal space, consisting of all plausible instances, $\mathbf{a}^i$, as $\mathcal{A} = \{\cdots, \mathbf{a}^i, \cdots\}$ whose $\mathbf{a}^i = \langle \mathbf{e}^i, \mathbf{t}^i, \mathbf{x}^i, \mathbf{z}^i \rangle$, where $\mathbf{e}^i$, $\mathbf{t}^i$, $\mathbf{x}^i$, $\mathbf{z}^i$ represent the instances of **element**s, **interaction**s, **geometric configuration**s, and **initial condition**s, respectively. Also, the evaluation metric is written as $f(\mathbf{a}^i)$ for any given instance $\mathbf{a}^i$, and the desired content is referred to that satisfies the equation $f(\mathbf{a}^i) > \lambda$, for λ being the targeted goal.

Accordingly, the spatio-temporal content space consists of all plausible instances. As a result, this work treats the targeted generation as sampling from the content space. To be specific, a plausible candidate could be any kind of setting under the application scope, including those that may provide undesired or even failure results. In the above exemplar domino run, a possible solution implies a set-up with any given domino number arranging in any designed pattern, even those fail to topple all dominoes in once or fail to meet the target standard. Since the involved number of objects, the placed locations and properties of each object, the triggering initial conditions, and etc. are all free parameters, the plausible content space is extremely high dimensional. Therefore, this makes it difficult to find solutions for a targeted goal without systematic exploration. As a result, the next section first describes the concept to semantically and parametrically decompose the content space for systematic and coherent exploration. Then, we adapt Monte Carlo Tree Search (MCTS) to efficiently sample the decomposed content space for efficient solution seeking. Finally, we apply the concept on three applications.

IV. Plausible Content Sampling

This work rephrases the targeted problem as a sampling problem which can be clearly stated as "how to sample a desired instance from $\mathcal{A}$ in a massive amount". The preference of an instance is evaluated with $f(\mathbf{a}^i)$ for any given $\mathbf{a}^i$, while such a metric completely depends on the goal to achieve and is not restricted to a specific equation. For such a problem, we follow the approach proposed by Chenney *et al.* [10] and define $p_w(\mathbf{a}^i | f(\mathbf{a}^i) > \lambda)$ to be the probability that the desired content instance $\mathbf{a}^i$ might arise in the simulation environment, where $f(\mathbf{a}^i)$ is the objective function that evaluates how "good" the content is. Due to the intractability of $p_w(\mathbf{a}^i | f(\mathbf{a}^i) > \lambda)$, we model the probability indirectly by

$$p(\mathbf{a}^i) \propto p_w(\mathbf{a}^i) p_f(\mathbf{a}^i) \tag{1}$$

where $p(\mathbf{a}^i)$ models the distribution of the plausible and desired instances in the content space, and $p_f(\mathbf{a}^i)$ depends only on how well the content satisfies objectiveness. According to the definition of $\mathbf{a}^i$, the equation can be further rewritten as

$$p(\mathbf{a}^i) \propto p_w(\mathbf{l}^i) p_w(\mathbf{x}^i) p_w(\mathbf{z}^i) p_f(\mathbf{a}^i). \tag{2}$$

where $\mathbf{l}^i$ is the combination of $\mathbf{e}^i$ and $\mathbf{t}^i$. This work proposes a new framework, named L-MCTS, which incorporates the L-system into our adapted Monte Carlo Tree Search (MCTS), to model $p(\mathbf{a}^i)$. Intuition behind this design is two folds. On one hand, a direct sample to decide $\mathbf{e}^i$ and $\mathbf{t}^i$ seems to be counter-intuitive, while modelling such properties in a semantic way is a more typical solution. That is, both the **element**s and **interaction**s are arranged into the chronological order while

additional **element**s as well as **interaction**s are inserted into the current sequence in an iterative fashion. Such a definition not only helps to sample $\mathbf{e}^i$ and $\mathbf{t}^i$ with unconstrained dimensionality but also better present the progressive process of the content design. On the other hand, modelling each component independently is pointless due to the dependencies between the components. For example, a domino could never topple its next target if it is placed too far away or not on the falling direction of the previous domino. As a result, the L-MCTS is designed to perform semantic and hierarchical exploration of the content by incorporating the L-system representation while combining inverse solutions during the search to greatly improves the search efficiency. More specifically, the search tree is separated into four parts: the **L-string expansion** that models $p_w(\mathbf{l}^i)$, **spatial positioning** that models $p_w(\mathbf{x}^i)$, **activating initialization** that models $p_w(\mathbf{z}^i)$ and **validation simulation** that models $p_f(\mathbf{a}^i)$. With the backpropagation mechanism, L-MCTS concentrates on more promising subtrees and models $p(\mathbf{a}^i)$ efficiently. We show the algorithm of L-MCTS in Algorithm 1.

Algorithm 1 L-system Monte Carlo Tree Search (L-MCTS)

$\mathcal{A}_{desired} = \{\}$
repeat
  $\mathbf{a}^i = SelectToLeaf(tree)$;
  $f^i, reward = Simulate(\mathbf{a}^i)$;
  if $f^i > \lambda$ **then**
    $\mathcal{A}_{desired} \leftarrow \mathbf{a}^i$;
  end if
  $Backpropagate(tree, reward)$;
until Terminate

A. L-system Semantic Interaction Sequence and Tree

L-system is a string rewriting mechanism to systematically expand the sequence of its componential alphabetized symbols using a set of production rules [13], and we think it is perfectly suitable to represent the cascades of **interaction**s due to its strength in describing sequential branching behaviors [8], [16]. Additionally, L-system expands the string in a hierarchical fashion which fits the traditional MCTS usage. In general, L-system is expressed as $\mathcal{G} = \langle \mathcal{V}, \Omega, \mathcal{P} \rangle$ with $\mathcal{V}$ being a set of alphabets for symbolization, Ω being the axiom, i.e., a word, and $\mathcal{P}$ signifying a set of rewriting rules. The produced L-string is composed of a sequence of alphabets where the alphabets reveal $\mathbf{e}^i$ and their order reveals $\mathbf{t}^i$, forming the alternative term $\mathbf{l}^i$ that embeds both the **element**s and **interaction**s. Even though the L-system, $\mathcal{G}$, completely depends on the application and cannot be generalized into a common form, we involve one commonly used rewriting rule to split the interaction sequence into multiple branches with brackets. That is, the sub-string wrapped in the brackets belongs to a single **interaction** path and would not affect other branches, resulting in the form of $str_0[str_1][str_2]\cdots[str_i]$ where str_1 to str_i are the sub-strings branching from the last alphabet of str_0.

To derive an L-system from the given application, there exists no single correct answer and the derivation might greatly vary under different objective goals. This work does not aim to provide a common L-system derivation for any arbitrary application and regards such domain knowledge as an essential prerequisite for our system. Instead, we introduce our strategy to design the L-systems proposed in this paper only as an example for the derivation design. Firstly, we identify the possible interactions based on habitually interaction classification. Then, alphabets are assigned to represent the involving elements and production rules are developed by determining how the given interaction can be inserted into the existing string. As an example, we demonstrate the mechanism with a domino application shown in Fig. 4, which the travelling direction is in general restricted to the x-axis but may split into two symmetric paths, forming an angle θ_D ranging between $(0, 60]$ with the x-axis. Therefore, we assign D_h to represent the domino set on the x-axis-aligned path and D_s for the one on the splitting path, where we deterministically have 4 dominoes in a set to simplify the set-up. The production rules are concluded into 2 types, which are

- **Split** inserts two D_s after D_h to form two toppling paths and the additional D_h is appended if it is empty after D_s.
- **Lengthen** inserts an duplication of one alphabet after it to make the domino chain longer.

The system always begins with the axiom, D_h, and extends the set-up by iteratively applying one of the production rules each time. We show one possible extension as well as its evolution path in Fig. 4.

Although the L-string itself can well represent the **element**s and **interaction**s, we would like to further deduce it into the tree form, named as the interaction tree, to describe the spatio-temporal content in a hierarchical fashion. Such a tree not only provides a more intuitive visualization of the **interaction**s but also eases the later process when performing inverse embedding (please refer to Section IV-B for more details). To turn an L-string into a tree, the process starts from the first alphabet and iteratively appends the next alphabet as a child to the current alphabet. The tree splits into branches when the bracket is met, with the sub-string in the bracket forming one single sub-tree. The deducing example is shown in Fig. 4 for each L-string.

B. Inverse Embedding

There are two common ways to derive the corresponding actions from a distribution of $p_w(\mathbf{x}^i)$: one is the forward derivation, and the other is the inverse postulation. Taking the one-shot-all-sink billiard set-up as an example, forward derivation first specifies all possible configurations, including positions and the cue strike conditions, and then runs physical simulation to determine the successfulness, i.e., estimate the probability; on the other hand, inverse postulation uses the collision physics to estimate the possible travelling path of each ball to fulfill the pocketing requirement while deriving the corresponding positions and initial conditions at the same time. The forward, however, necessitates the exploitation of almost infinite space, resulting in computational intensity. On the other hand, the inverse could also be impractical since all path trajectories would be intractable if complex physics involved.

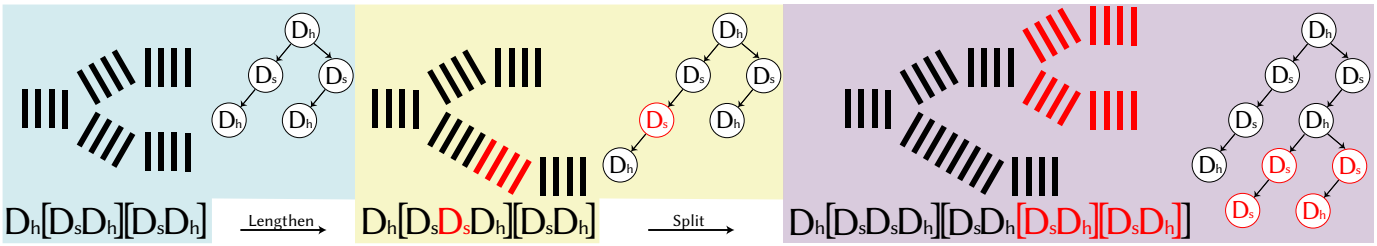


Fig. 4. This shows an example of the L-string evolution of the domino application. Our system decomposes dominoes into slope and horizontal segments and describes them as D_s and D_h using L-system. Initially, interactions start with the horizontal segment splitting into two paths of a slope and horizontal segment. Then, we apply a lengthening on the left path to have two slope segments. Finally, we apply a splitting on the right path to add two additional paths. The corresponding L-strings are provided in the bottom, and the right shows the interaction tree transformed from the L-string by appending the alphabet as the child of is previous alphabet and splitting into branches when the bracket is met.

As a practical solution, coherent sampling algorithms, such as Markov chain Monte Carlo (MCMC) and Monte Carlo Tree Search (MCTS), are proposed to effectively reduce the amount of exploration. However, a pure forward solution seems to be inefficient when the domain knowledge can be applied to greatly reduce the explored space, resulting in a more plausible and efficient algorithm. For example, the possible position of an object ball can be restricted to a sector area in front of the pocket, while the forward strategy requires to sample all possible positions on the entire table. Therefore, we propose to incorporate the inverse concept into MCTS, which is in general a forward solution, as a hybrid method to model $p_w(\mathbf{x}^i)$. More specifically, inverse embedding, derived based on the expertise, is performed to reduce the sample space of $\mathbf{x}^i$ for any given L-string. However, the amount of domain knowledge and the level of procedural factors taken to postulate the forward parameters is a trade-off between the computational complexity and sampling efficiency. In addition, this work does not aim to provide an inverse postulation for any arbitrary application and treats such domain knowledge as the prerequisite of our system. In this paper, we adapt the motion prediction derived from simplified physical models as the key while involving reasonable deviations to increase the content variation. Due to the nature that the inverse embedding mechanism is fully application-dependant, more details are introduced in Section V.

C. L-system Monte Carlo Tree Search (L-MCTS)

Traditional MCTS expands the search tree with four stages of selection, expansion, simulation, and backpropagation as defined in the following:

- **Selection** selects successive children from the root to a leaf that is not yet fully explored.
- **Expansion** appends children with not yet investigated states to a leaf.
- **Simulation** performs a complete playout from the current until a result or predefined state is achieved.
- **Backpropagation** updates the probability distribution based on the evaluated objectiveness and is always performed right after the simulation operation.

However, some modifications are incorporated into the proposed L-MCTS in order to better fit our application. First, we only perform the simulation when all parameters are decided, including $\mathbf{l}^i$, $\mathbf{x}^i$, and $\mathbf{z}^i$, while traditional MCTS performs simulation at any node with the following moves decided randomly. More specifically, we keep performing selection and expansion until simulation is available. The key to such modification is that the number of parameters to sample is relatively constant in our case, compared to traditional cases in which the number of moves to finish the game is usually difficult to predict. Second is the node-choosing strategy. Traditionally, MCTS performs selection based on a designed metric to balance exploration and exploitation. The most popular metric is the Upper Confidence Bounds to Trees (UCT) [49], which in general can be formulated as $c^j = \frac{e^j}{n^j} + \sqrt{2}\sqrt{\frac{ln(N^{j-1})}{n^j}}$, where j stands for any node from the tree, n^j stands for the simulation count of the j-th node, N^{j-1} stands for the simulation count for the parent of the j-th node, and $\frac{e^j}{n^j}$ implies the empirical mean. An intuition is that for an untraversed node, the UCT evaluates to infinity, and therefore, traditional MCTS chooses to perform expansion prior to selection if there are unexplored states. However, as we are sampling parameters from continuous spaces, such behavior would never lead the search to converge. Hence, we slightly modify the definition of **Selection** and **Expansion** into "stepping down the hierarchy by picking one child", while the former only picks the existing child and the latter appends a new one. Also, a new strategy is proposed for deciding whether selection or expansion is performed based on $C^j = \sum c^{j+1}$, where c^{j+1} represents all children of the given j-th node. The concept behind the strategy is a trade-off between exploitation and exploration, i.e., we choose selection if the current parameter sets are promising enough to generate desired results; otherwise choose expansion for exploring other possibilities. Technically, we choose to perform selection if $C^j \geq C_\alpha$. Such a threshold C_α might vary under different search scenarios, and hence, it is decided empirically for different applications proposed in this paper. We introduce the experiment involved to decide the threshold in Section VI-A1

Another difference is that the simulation result of L-MCTS ranges between $[0, 1]$ to imply the preference instead of simple win or lose. More specifically, we have $\Delta e = Max(1 - \frac{|\lambda - f(\mathbf{a}^i)|}{\beta}, 0)$ and add Δe to e^j for nodes on the path from root to leaf when performing backpropagation. λ reflects the objectiveness, but its value is application-dependent and not normalized to the range $[0, 1]$. Therefore, this work uses β to

normalize the application-dependent objectiveness deviation, and its value is decided empirically.

The process to sample $\mathbf{a}^i$ can be separated into three phases: L-string expansion, spatial positioning, and activating initialization while each distributes in the search tree from top to down respectively, and different actions are taken to sample a new parameter set during different phases.

- **L-string expansion** aims to sample the L-strings by inserting alphabets into the string iteratively. For any given node in this phase, the number of its child is one plus the number of its L-string variants with one rewriting rule applied, where the additional node represents to step down to spatial positioning with the current string. Since the node number in one layer is limited in this phase, we follow the traditional strategy to select a child with the largest c^j. Also, to avoid the string growing infinitely, we simply restrict the string length to n_{max}, which depends on the search goal and application.
- **Spatial positioning** aims to place the elements that come from L-string expansion into the space. Any given node in this phase represents the position of one corresponding element, where its expansion is conditioned by the inverse embedding mechanism. According to the propagation direction, we have a forward solution that determines positions from root to leaves given the interaction tree transformed from the string and the backward solution that reverses such order. In general, both approaches are acceptable, but the better one is application-dependent. More specifically, the backward solution better fits the applications with hard constraints on element positioning while using the forward one might struggle in such cases. For example, it is easier to use the fixed position of 6 pockets as hard constraints for inverse postulation in the billiard game.
- **Activating initialization** aims to decide initial conditions to trigger the spatio-temporal content. Practically, this part is relatively simple as the initial conditions are independent with each other in common cases. As a result, it is similar to have a single-layer node for randomly sampling all parameters in once. Furthermore, the phase could be optional if a sequence does not require any explicit triggering action. For example, a chain reaction generally start from a free falling ball without the need of triggering.

D. Objective Function

The objective function directly defines the search goal of L-MCTS, and it could be either objective or subjective and usually fully application-dependent. This paper shows the possible derivation of objectiveness by incorporating objective metrics, such as the animation elapsing time, i.e., application-independent, as well as subjective metrics, such as the visual complexity, i.e., application-dependent. We demonstrate a procedure to derive the subjective evaluation based on questionnaire postulation of user preferences in Section V-B4 and Section VI-A2. Even though our L-MCTS does not constrain it to any specific form, it should be positively correlated with the number of interactions, i.e., the length of the sequence, to facilitate coherent exploration of MCTS. Generally, each interaction should have its own contribution while consecutive interaction may introduce extra inter-interaction contributions. Since our L-system representation of cascading events has been chosen based on habitual and simulation completeness, inter-interaction contributions should be minor and negligible. Therefore, we can first derive the interaction-wise contribution for each type based on its configured and geometric parameters and then sum up all interaction contributions with empirical weightings. All objective functions proposed in this paper follow the above design concept, and details are given in their corresponding sections in Section V.

V. Applications

Our L-MCTS is designed to be a general framework of massive production for various applications. To demonstrate its generality, we apply it on three different applications, including one-shot-all-eliminate breakout clone, chain reaction, and one-shot-all-sink billiard. The following sections introduce them respectively along with their specific L-system, inverse embedding, activating initiation, and objective function derived from their unique domain knowledge. For brevity, we omit the unit of length if it is measured in meter.

A. One-shot-all-eliminate Breakout Clone

The breakout clone is a famous video game that a player controls the moving paddle located at the bottom to make the ball hit the bricks and destroy them. The ball bounces when hitting either the paddle, a brick, or the top, left, and right wall, and the player loses when the ball hit the bottom. As shown in the left of Fig. 5, we derive a simple toy example from it as our first validation attempt. We let the ball destroy all bricks without another bounce from the paddle, i.e., once the ball is emitted from the paddle, it never lands but destroys all bricks aligned horizontally.

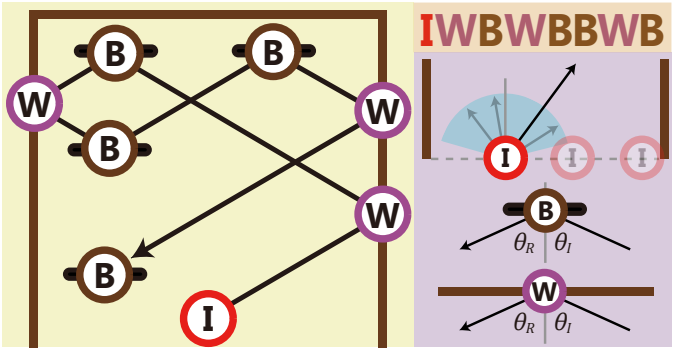


Fig. 5. This shows an example of our one-shot-all-eliminate breakout clone with its representative L-string, $IWBWBBWB$. The ball, I, releases to hit the wall, W, bounces to a brick, B, follows by bounces of W, B, B and W, and finally ends at B. Furthermore, ball's release position and angle are the initial conditions and the reflection rule at the brick and wall is used for inverse embedding.

1) L-string Expansion: Based on the bouncing behaviors, we can semantically categorize them into two bouncing types, wall bounces, W, and brick bounces, B. We also use I to denote the emission position of the ball. The system always begins with the axiom, IB, and extends the level by iteratively inserting consecutive bounces into the current string:

- The **wall bounce** inserts a W into any position of the current string, except for the very front and the very back.
- The **brick bounce** inserts a B into any position of the current string, except for the very front.

An example string as well as its possible level is shown in Fig. 5.

2) Spatial Positioning: Accordingly, the face normal at the bouncing point is the only affecting factor of ball movement, i.e., the outgoing direction is based on the reflection of perfect elastic collision where the incidence angle equals the reflection angle. To decide the element positions, we start from the root of the interaction tree and propagate to its descendant in a forward order. More specifically, we decide both the element position, p^i, and outgoing direction, $\vec{v}^i$, for the i-th node based on the element type:

- **I**: p^i represents the emission point randomly chosen along the paddle moving line, while $\vec{v}^i$ is the emission direction whose angle between the vertical axis is sampled within the range of $[-65°, 65°]$.
- **W**: p^i represents the bouncing point of the wall located as the first intersection of the ray $p^{(i-1)} + \vec{v}^{(i-1)}$ to the three walls, while $\vec{v}^i$ is the bouncing direction derived from the reflection rule, i.e., to have the reflection angle θ_R equal to the incidence angle θ_I.
- **B**: p^i represents the bouncing point of the brick randomly chosen along the outgoing path of the previous node, while $\vec{v}^i$ is the bouncing direction derived from the reflection rule.

After processing all nodes, we place the paddle as well as other bricks with their center locating at p^i. We demonstrate the spatial positioning mechanism in the right bottom of Fig. 5.

3) Initialization: To initiate the break-out clone, the ball is emitted into the scene from the center of the paddle with an outgoing direction. Since the paddle position and outgoing direction is already determined in the spatial positioning phase, there are no extra factors sampled in the initialization phase.

4) Objective Functions: We evaluate the breakout clone level in terms of difficulty to achieve the one-shot-all-eliminate goal. Because the bouncing mechanism is easy to predict, we consider the difficulty as answering the question of "how difficult to figure out the correct bouncing path to eliminate all bricks in once for the given level". The farther two bouncing point are, the harder to plan the bouncing path, i.e., the difficulty should be proportional to the distance between two bouncing points. If the number of bounces increases, the difficulty increases accordingly, i.e., the difficulty should be the accumulation of difficulty at each bouncing. As a result, we can express the difficulty as $f_d = \sum \alpha^i \cdot (1 + 0.2 \cdot \frac{l^i}{l_{diag}})$ for $i \neq 0$, where l represents the distance between two consecutive bounces, l_{diag} denotes the diagonal length of the scene, and α is the bouncing weighting which is set to 15 for W and 10 for B due to the fact that a brick is smaller and harder to predict its bounce. Furthermore, we have $\beta = 0.25 \cdot \lambda$ to normalize the reward.

B. Chain Reaction

Chain reaction is the most classic application of spatio-temporal content because of its cascading events to propagate interactions from one component to another. In general, elements involved in chain reaction can be anything as long as it makes cascading propagation continue, and the allowable interactions between elements are totally implementation dependent. Here, we choose several commonly seen components, such as balls and dominoes, to produce chain reaction animations of targeting visual complexity. As shown in Fig. 6, we restrict the interaction path to a 2D plane in general, while the path might split into multiple ones where each lies on a 2D plane paralleling to each other.

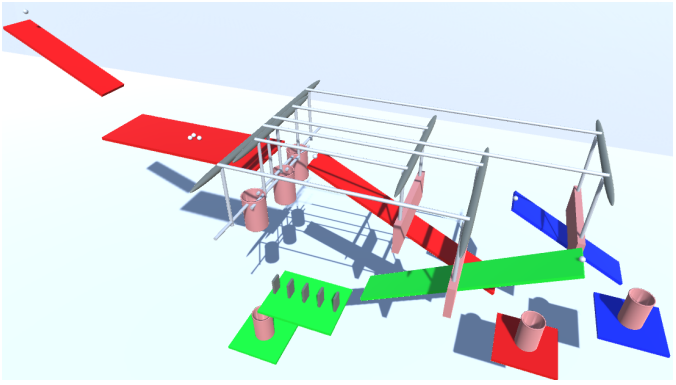


Fig. 6. This demonstrates an example of our chain reaction set-up with its representative L-string, $BL[PBC][PBDC][PBC]$, where the interaction paths are designed to lie on 3 2D planes marked with red, green, and blue, respectively. The action starts from the ball that rolls down the slope, B, and collides to strike the 3 balls on the platform, L. All three balls fall into the corresponding cups to activate their respective pulley sets, P, for splitting the path into three. While two of them drop into the cup to end their sequences, the other topples the domino run, D, which ends up with the last domino falling into the cup, C.

1) L-string Expansion: Designers generally plan chain reactions based on sets of objects which have similar physical actions or are assembled to act as a whole such as domino sets. In other words, interactions propagate between elements inside a set and between sets. We denote these sets as components. As a result, we semantically derive the L-system representations by assigning an alphabet to each component while the production rules are then carefully designed for plausible propagation. To demonstrate our framework, we select 5 different types of components visualized in Fig. 7. The first component, B, is composed of a ball placed on a slope, and the ball rolls down the slope. Second, we have the domino set, D, that is straightly placed along the x-axis and can be toppled from the first domino. Third, we have the pulley set, P, with one side being the cup and the other being the baffle, while the baffle is heavier than the cup to block the ball from rolling down the slope. In other words, the pulley is only triggered when something falls into the cup and makes the ball start rolling down the slope because the baffle gets removed. Additionally,

a platform with multiple balls placed on it, referred as L, is included to perform the branching behavior. More specifically, balls on the platform can be struck into different directions and split the interaction path into multiple ones where each lies on a new 2d plane. Lastly, we have the single cup, C, to be the end of the chain reaction, i.e., the cascading events end when the previous component falls into the cup. For each component, we propose a corresponding production rule to insert a new instance into the current set-up as following:

- The **dominoes**, **D**, can only be toppled by the rolling ball or another domino set and cannot trigger multiple balls on the platform, which means D can only be inserted after B or D and not before L.
- The **pulley set**, **P**, and the **ball on slope**, **B**, are grouped together as PB because the baffle blocks the ball from rolling down the slope while the pulley can be triggered by falling components including the domino and ball. Specifically, PB must be inserted after B, D or L.
- The **platform with multiple balls**, **L**, can only be triggered by the rolling ball and only be used to trigger the pulley set, i.e., we insert L after B and before P. Also, due to the branching behavior of L, the interaction path splits into n paths with n balls placed on the platform, with one sub-path being the original sub-string after the insertion point and others being initialized to be C. To simplify the set-up, we restrict n to be 3 and the branching behavior can only be inserted to the main branch as illustrated in the top left of Fig. 7.
- The **cup**, **C**, can only be placed at the end of the string and no production rule inserts C into the string directly.

The chain reaction always begins with the minimum set-up, BC, and iteratively applies the insertion rules to extend the interaction paths.

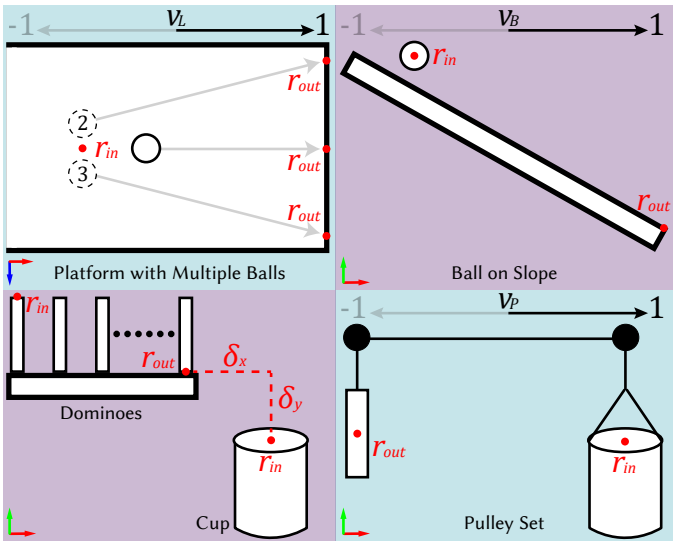


Fig. 7. This visualizes the 5 components involved in our chain reaction application, where the V_L, V_B, and V_P decide the placing directions along the x-axis, of the component L, B, and P. Also, we define the **reference input**, r_{in}, and the **reference output**, r_{out}, for each component to align two components with additional offset δ_x and δ_y for the inverse embedding mechanism.

2) Spatial Positioning: If we fix the behavior inside each component, such as the number of dominoes and the distance between consecutive ones, the dimension of plausible content space becomes small and the animation become boring and dump. Thus, we would like to increase the behavior variety of each component by let them have more control parameters. Without loss of generality, we use the x-axis to denote the movement direction of the chain reaction since we assume the interaction path be linear on a 2D plane.

- The **dominoes** are parameterized by the domino number, n_D, within the range $[4, 10]$, and the interval between each domino, l_D, within the range $[0.2, 0.3]$.
- The **pulley** is parameterized by the distance between two pulleys, l_P, within the range of $[2, 5]$, the distance between the pulley and its carrying object, l_{cup} or l_{baffle}, within the range of $[0.5, 1]$, and the facing direction v_P to be either $\{-1, 1\}$ to represent the alignment along the x-axis.
- The **ball on slope** is parameterized by the slope length, l_B, within the range of $[1, 5]$, the elevation angle between the slope and the x-axis, θ_B, within the range $[26°, 30°]$, and the facing direction v_B to be either $\{-1, 1\}$ to represent the alignment along the x-axis.
- The **platform with multiple balls** is parameterized by the facing direction v_L to be either $\{-1, 1\}$. The first ball would be put on the position which offsets $4 \cdot r \cdot v_L$ from the center of the platform along the x-axis, where r represents the ball radius, and the followings are put on the position which offsets $\frac{4}{3} \cdot r \cdot \alpha_L$ from the center along the z-axis, where α_L is -1 for the second ball and 1 for the third ball.
- The **cup** is not parameterized.

Generally speaking, we arbitrarily choose some parameters to have an adequate sampling range for compact illustration. For example, n_D is chosen to have a reasonable length of domino toppling; l_P is selected to make the pulleys not too far away. At the same time, the others are decided empirically to provide acceptable success rate. For example, l_B and θ_B are chosen to provide adequate energy applied to the ball for triggering the next component. Due to the length limitation, those parameters, such as mass and dimensions, chosen for producing the results are summarized in Appendix A.

To set up a chain reaction that can successfully propagate the interactions to the end, it is difficult not to perform trial-and-error even for the experts because the involving physics could be extremely complex. In fact, the experts usually place the component based on the intuition in the first place and make small modifications according to the simulation result. Following such a concept, we define r_{in} and r_{out} for each component to represent the point of **reference input** and **reference output**, visualized in Fig. 7, where the former one indicates the point to successfully trigger the component, and the latter one indicates the point that the component propagates the interaction to its descending path at. More specifically, the component i is positioned by aligning r^i_{in} and r^{i-1}_{out}, where the decision of r_{in} and r_{out} is component-wise and summarized as following:

- The **domino set** has r_{in} to be the entry point to topple its first domino, and we define it as the top position of the first domino. On the other hand, we have r_{out} to reveal the end position of the dominoes and defining it as the bottom position of the last domino.
- The **pulley** has r_{in} to be the point that the previous component is supposed to pass and fall into the cup, located as the center of the cup mouth, and has r_{out} to be the center position of the baffle.
- The **ball on slope** sets r_{in} as the initial position of the ball while having r_{out} to be the leaving point of the ball from the slope, i.e., the lowest point of the slope.
- The **platform with multiple balls** has r_{in} to be the entry point to collide the balls on the platform, defined to be the center of the platform, and has multiple r_{out}s where each represents the falling point of the ball from the platform.
- The **cup** only have r_{in} which shares the same definition with the pulley.

When aligning two reference points, in order to avoid component overlapping as well as to handle the unpredictable behavior of physics, we add an extra 2D random offset, (δ_x, δ_y), both sampled in the range of $[0.2, 1]$. The overall workflow of spatial positioning begins from the root and processes nodes in a forward order, while each component is first parameterized and placed to the scene by aligning its r_{in} to previous r_{out} using the above process. The only exception is the root position which is randomly sampled from the 3D space.

3) Initialization: The initial conditions determine how much energy the chain reaction system has in the beginning. While we always have the first component to be a B according to our production rules, the initial conditions fully depend on the initial height of the first ball. More specifically, we sampled a distance ranging between $[10, 20]$ that represents the height difference between the ball and the slope.

4) Objective Function: Chain reaction is evaluated based on two different metrics: f_t simply reveals the elapsed time from begin to end, and f_c evaluates the visual complexity of the set-up. f_t helps to validate the effectiveness of the proposed framework, and f_c generally involves subjective opinions and consensus. To the best of our knowledge, no existing literature has ever formulated and postulated it. Therefore, this work formulates it empirically based on the questionnaire methodology introduced in Section VI and proposes to model f_c as

$$f_c = \sum H(i) + \alpha_g G(\cdot), \tag{3}$$

where $H(i)$ evaluates the component-wise complexity for the i-th component, $G(\cdot)$ evaluates the layout complexity as a global term, and $\alpha_g = 0.4$ is the weighting to balance between the local and global term. The component term, $H(i)$, is defined as $H(i) = c^i \cdot k^i$, where c is the scaling factor to normalize the complexity contribution into the same range for each type, and k is the complexity contribution. We have $k = \frac{K}{R_{max}}$ where K is the value of the control parameter which has the most dominant affection in complexity, and R_{max} is the maximal allowable value of the control parameter. Accordingly, we set c and K as follows:

- $\mathbf{B}$ has K to be the slope length, and sets $c^i = 1.00$.
- $\mathbf{D}$ has K to be the domino number, and sets $c^i = 2.75$.
- $\mathbf{P}$ chooses K to be the distance between two pulleys, and sets $c^i = 1.22$.
- $\mathbf{L}$ chooses K to be the branch number, and sets $c^i = 2.81$.
- $\mathbf{C}$ has $K = 1$ because the cup is not parameterized, and sets $c^i = 0.1$.

The global term is modelled as $G(\cdot) = w_b \cdot g_b + w_d \cdot g_d + w_v \cdot g_v$, where g_b represents the tree balance, g_d represents the depth of the interaction tree, g_v represents the number that movement direction changes, and w_b, w_d, and w_v are the weightings of each term set to be $1.88, 2.25$, and 1.88, respectively. The intuition behind the balance term is to reveal the number of interactions happened simultaneously because the system looks more complex when there are more interactions happened in the same time. Specifically we have $g_b = \sum \frac{d^j_{avg}}{d^j_{max}}$, where d^j_{avg} and d^j_{max} represents the average and maximal branch depth at the j-th branching point where the branch depth is defined as the depth from the branching point to either the leaf node or the next branching point. All the above mentioned weightings are decided based on the user study discussed in Section VI-A2. The only exception is cup's weight set to 0.1 because it remains static and unchanged throughout the animation. Also, we set $\beta = 2 \cdot \lambda$ to normalize the reward.

C. One-shot-all-sink Billiard Layout

Billiard is a well-known sport that strikes balls into the pockets with a cue. Among such a category, there is a special variant that challenges the player to sink all object balls in only one shot, which is referred as the one-shot-all-sink billiard. To design a plausible one-shot-all-sink layout such as the one shown in Fig. 8, we define several terminologies at first hand to systematically explore the domain. Ghost ball is an abstract concept that we use to represent the event of ball-to-ball collision, and a kiss shot or a borrow shot is referred to the pocketing event while the object ball from the former shot directly goes into the pocket and the one from latter shot collides with other balls before pocketing. We apply the proposed content generation framework on Pool Ace, which is the commercial game product provided by the International Gaming System (IGS), with Unity 2019.

1) L-string Expansion: Conceptually, experts use the collisions among the cue, objects, cushions, and pockets to describe billiard interactions. Therefore, to semantically derive the L-system with the billiard application, we conclude two collision types, which are the ball-to-ball collision denoted as G and the ball-to-cushion collision denoted as C, as well as two pocketing strategies, which are the kiss shot referred as P and the borrow shot referred as X. Also, we involve some alphabets to represent the ball position, which are I, S, and O for the begin position of the cue ball, the end position of the cue ball, and the begin position of the object ball respectively. We design the production rules that insert shot combination, including the cushion and kiss shots, into the existing trajectory, while the borrow shot is always transformed from the kiss shot in specific situation. More specifically, the production rules are listed as below:

- **Cushion** inserts an extra cushion shot, C, on any moving trajectory, i.e., after I, O, C, or G but not between GO.
- **Kiss** inserts an extra kiss shot, GOP, on any moving trajectory and results in a splitting branch. If the insertion happens before any kiss shot, i.e., P, change it into X as the shot becomes a borrow shot.

The system always begins with the axiom, $IG[S][OP]$, and extends the set-up by iteratively applying one of the two production rules each time. When transform the L-string into the interaction tree, O is omitted because GO always show up together. An example string and its possible layout is shown in Fig. 8, where its deducing interaction tree is placed on the right.

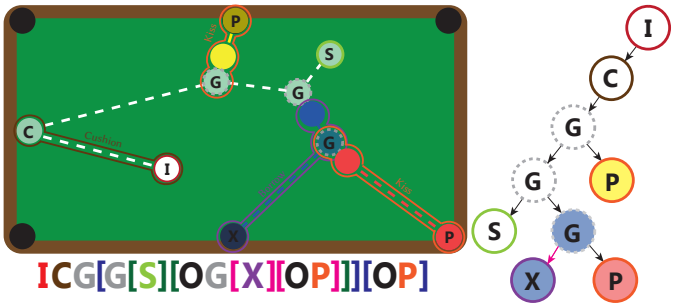


Fig. 8. This shows an exemplar of our one-shot-all-sink billiard application, where the layout is shown on the left top. The cue ball, I, initiates the shot to hit the cushion, C, which is called the cushion shot. The cue collides with the yellow object ball, O, to induce a collided ghost ball, G, which is a kiss shot while the yellow sinks in the pocket, P. The cue follows through to collide with the blue, O, induced another colliding ghost, G, before it stops at its stopping point, S. The blue collides with the red to induce G, before the blue sinks into a pocket, X, referring as a borrow shot, while the red ball forming another kiss shot. The entire sequential events can be expressed as the L-string of $ICG[G[S][OG[X][OP]]][OP]$ and its corresponding interaction tree is shown on the right.

2) Spatial Positioning: In Pool Ace, all the balls are forced to remain on the table and no bouncing happens during the play, so we design the inverse embedding techniques based on such assumption to postulate the initial positions of the cue and objects. We first decide the end position and propagate to the rest of the balls in a backward order, due to the fact that pockets are constrained to fixed positions on the table. That is, the positions are decided by processing each node from leaves to root, given the interaction tree transformed from an L-string. To decide the position of leaf nodes, we randomly sample from the 6 pockets for P and X, and from the circle region centered at its sibling for S, with the radius set to $\frac{1}{3}$ of the table diagonal. On the other hand, positions of the non-leaf node are decided based on the inverse embedding mechanism to narrow the sampling space into a more reasonable subspace. While precisely predicting the ball trajectory is computationally expensive when taking the properties such as rotation or friction into account and the collision may even not be elastic, we perform inverse embedding by simply assuming particle balls and elastic collisions without frictions to avoid costly computation. More specifically, position of a non-leaf node is sampled based on the following two constraints. The first one is the circle constraint that is only applied when there exists two child nodes. One circle is constructed by connecting its two children as the diameter, and the node is constrained to the circle. The second one is the direction constraint that is applied to each individual child node separately. A fan area is constructed by spanning θ_p degrees to two sides of the direction $\vec{n}$, and the node is constrained to be inside the area, where θ_p and $\vec{n}$ vary for the different child node:

- $\mathbf{P/X}$: $\vec{n}$ represents the angle bisector and $\theta_p = 22.5$ for pockets on the corner and $\theta_p = 45$ for others.
- $\mathbf{G}$: $\vec{n}$ represents the outgoing direction passing the child node from the circle that G is sampled from and $\theta_p = 5$.
- $\mathbf{C}$: $\vec{n}$ represents the reflection path and $\theta_p = 0$.

When multiple constraints are applied, the node is sampled from the intersection of all constraints. Additionally, the sampling is supposed to not violate the characteristic of the node. For example, the point should lie on the table edge if the current node is C. The mechanism is demonstrated in Fig. 9 with the given L-string and the final spatial distribution is shown on the right top.

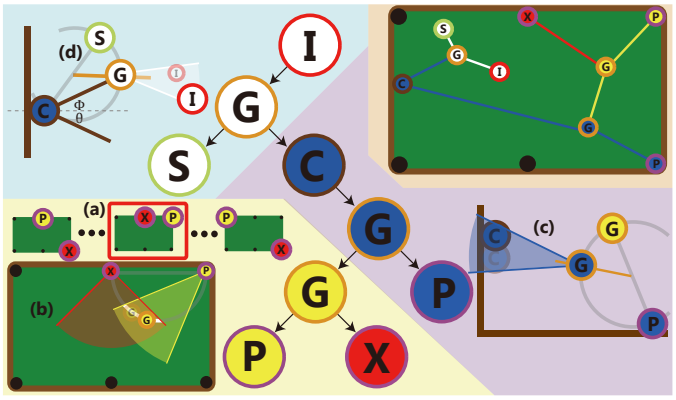


Fig. 9. This shows an exemplar spatial positioning process of deciding the ball positions with a given interaction tree by traversing nodes in the backward breadth-first order. (a) The position of leaf nodes are sampled from their candidate space, which being the 6 pockets for X and P, (b) while the ghost ball G with two children is placed on the intersection of the circle and the fan areas. (c) The cushion node C with one single child is constrained by the fan area as well as its characteristic to be on the table edge. (d) The ghost ball G with one cushion child is placed on the intersection of the circle and the reflection direction that satisfies the equation of $\phi = \theta$.

3) Initialization: The initial condition refers to the cue hitting behavior and is defined as $H(\vec{f}, p)$ in our billiard application, where $\vec{f}$ represents the directional applied force and the p reveals the hitting point on the ball surface. In general, the cue ball draws a curved trajectory before the first collision if the applied force does not pass the ball center, while the rotation velocity usually vanishes quickly due to the friction, leaving the movement to be straight on the remaining path.

4) Objective Function: After examining billiard instructions and books [50], [51], experts generally evaluate the difficulty by considering the path planning as well as the striking execution. Furthermore, the striking effect is exponentially waded off by the friction, so the initial cue hitting effects become negligible for later collisions. Therefore, the difficulty metric, f_d, is empirically designed to be a summation of the striking term and the collision term:

$$f_d = \alpha G(\cdot) + \sum H(i), \tag{4}$$

where $G(\cdot)$ denotes the cue-hitting difficulty and $H(i)$ denotes the collision-wise difficulty measurement for each collision during the set-up. In addition, α, which is a user specified weight, is involved to balance these two factors and set to 0.5 in all billiard experiments in this paper.

For the striking term, we assume that spin and power are the two most effective factors to predict the ball trajectory. That is, when applying more spin and power on the cue, it is harder to predict its travelling path and its consecutive collisions. As a result, we have $G(\cdot) = \|\vec{f}\|l_G^2$ where $\|\vec{f}\|$ is the applied power in newton (N), and l_G is the distance between the center and hit point in cm. For the collision-wise term, the intuition is that the more consecutive of collisions happen before a specific collision or the longer the distance a ball travels from the previous collision, the harder it is to predict and execute the desired process. Moreover, while the exiting direction deviates more from the entering, it is also harder to aim. All three factors pull together to determine the measurement of the collision i as $H(i) = l_H^i n^i A(\cdot)$, where l_H^i represents the traveling distance from collision $i-1$ to i, n^i represents the accumulated number of consecutive collisions before i, and $A(\cdot)$ is measured based on the angle information depending on the following collision types:

- **Cushioning** happens between a ball and a cushion. $A(\cdot)$ is measured as $\sin(\theta_I^i)Max(\frac{tan(\theta_I^i)}{tan(\theta_R^i)}, \frac{tan(\theta_R^i)}{tan(\theta_I^i)})$, where θ_I^i and θ_R^i represent the incident and reflective angles. In general, it is more difficult and unpredictable when θ_I^i is larger and the difference between θ_I^i and θ_R^i is larger. Furthermore, the difficulty increases non-linearly with deviations.
- **Colliding** happens between two balls. We use the cut angle θ_c^i, which is the angle between the incoming direction and the reflection axis, and the tail angle θ_t^i, which is the angle between the outgoing direction and the extension of the incoming direction, as $\frac{1}{\cos(\theta_c^i)\cdot\cos(\theta_t^i)}$ because it is easier to aim while the enter and exit directions are near parallel.
- **Pocketing** refers to a ball entering a pocket. We use the pocket angle, θ_P^i, which represents the angle between the angle bisector of the pocket and the entering direction, to measure $A(\cdot)$ as $\frac{1}{cos(\theta_P^i)}$, due to the intuition that it is easier to hit the corner of the pocket and pop out if θ_P^i is larger.

The lengths and angles involved in the collision-wise difficulty measurement are visualized in Fig. 10. We set $\beta = 0.25 \cdot \lambda$ to normalize the reward.

VI. Experiments and Results

After implementing three targeting applications, we conduct several experiments and user studies to decide their applied parameters. Furthermore, we compare our proposed method to three baselines in the aspect of content generation with the targeted goal. Finally, since objective functions are important and abstract, we conduct user studies to validate their effectiveness. Due to length limitation, readers can refer to the supplemental Website[1] for more details.

[1]Website: http://graphics.csie.ntust.edu.tw/pub/LMCTS/

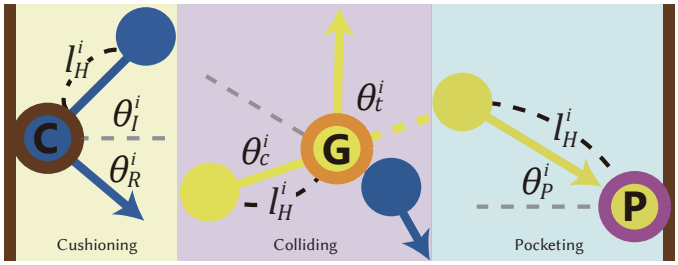


Fig. 10. This illustrates the collision-wise difficulty metric calculated based on the traveling lengths and collision angles where the lengths l_H^i is the distance between parent and node collisions, the incident angle θ_I^i and the reflective angle θ_R^i are used for cushioning, the cut angle θ_c^i and tail angle θ_t^i are selected for ball-to-ball collision, and the pocket angle θ_P^i is chosen when pocketing.

A. Parameter Selection

Generally, our applications have several user-tuning parameters of either an exploration/exploitation trade-off or user's subjective preferences, which cannot be directly decided. As a result, we conduct experiments and user studies for proper selection.

1) Selection Threshold: Our L-MCTS chooses to perform **selection** or **expansion** depends on whether C^j is greater than the selection threshold C_α. The choice of C_α itself is, however, a trade-off between exploration and exploitation. A smaller C_α makes the system focus more on the previously explored results while a larger one tends to exploit other new parameter spaces. To decide the threshold, we first evaluate the search with different C_α's to figure out its actual impact on the search task. More specifically, we perform the search tasks with different C_α's and evaluate the desired rate ρ_γ, which is defined as the number of desired instances out of the total number of searching. A desired instance is defined as the $\mathbf{a}^i$ that satisfies $\frac{f(\mathbf{a}^i)-\lambda}{\lambda} < \gamma$, where the left term represents the error percentage and the right is the error tolerance. We test with 5 different C_α's, which are $\{2.0, 2.5, 3.0, 3.5, 4.0\}$, on the chain reaction with its error tolerance γ set to 0.05. For each, we search 5000 iterations, and set the maximum string length n_{max} to be 8 and the desired goal λ to be 8 seconds. The left of Fig. 11 shows the desired rate under different C_α's across the search, while $C_\alpha = 2$ generally fails to perform the search, and $C_\alpha = 3$ outperforms others when the iteration number increases. The right of Fig. 11 shows the exploration rate across the search, which is defined as the none-repeated rate of the parameters we sampled from the continuous space. In general, such rate indicates the number of explored parameters and decreases more when the C_α is smaller. Accordingly, we choose $C_\alpha = 3.0$ for the chain reaction due to its significant performance gap against others and its acceptable exploration rate after 5000 iterations. Similarly, we apply the same process to set C_α to 3.0 and 1.85 for the breakout and billiard, respectively.

2) Subjective Weightings: In Section V-B4, we select several factors to be our indicator of the visual complexity and formulate them into Eq. 3 as our complexity metric. However, the weight of each factor remains undetermined because they are subjective preferences. Therefore, we designed a user study

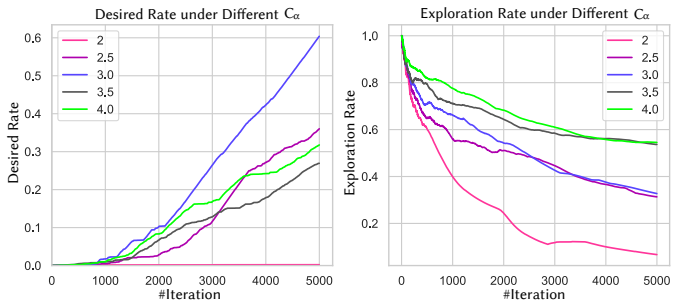


Fig. 11. We visualize the desired rate (left) and the exploration rate (right) under different C_α while searching. The iteration number is denoted along the x-axis. For the y-axis, we have the desired rate, defined as the number of instances whose error percentage less then 0.05 out of the total number of searching, in the left, and have the exploration rate, defined as the none-repeated rate of the parameters we sampled from the continuous space, in the right. The experiment result of different C_α is visualized in different colors denoted in the legend.

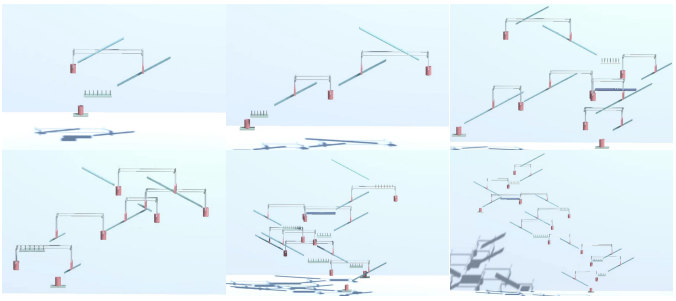


Fig. 12. This shows the chain reaction setups used for the factor ordering study. Its actual animation can be found in the supplementary materials.

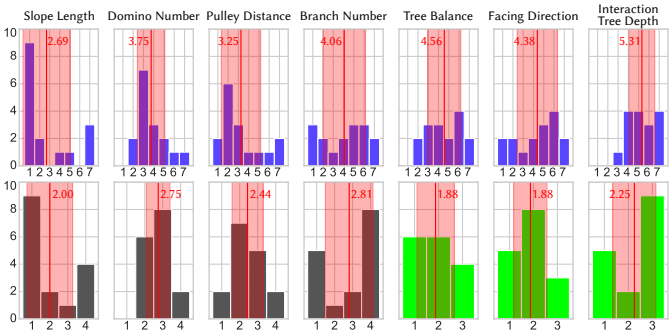


Fig. 13. This shows the histograms of our subjective weighting study for the chain-reaction visual complexity metric along with their average (marked with the red line) and standard deviation (marked with the red zone). The top histograms visualized in blue represent the ***raw score***s, while the left bottom ones visualized in black represent the ***local score***s which are the order after excluding the global factor, and the right bottom ones visualized in green represent the ***global score***s vise versa.

to ask users to sort them based on visual complexity from the least dominant to the most dominant. It was conducted on 16 subjects, aged from 20 to 29. Among them, there were 15 males and 1 female, and also there were 7 majoring in computer graphics and 2 having experience in animation design. At the beginning of the user study, we showed 6 chain reaction set-ups from Fig. 12 and explains the meanings of the 7 factors to the users. Factors of the 4 components, including B, D, P, and L, are explained as the definition listed in Section V-B4. Besides, g_b is given as "whether the branches split from same origin are equally long or not", g_d as "the longest path in the scene", and g_v as the "direction change among the set-up". Then, the users were asked to perform the sorting via the factor dominance on visual complexity according to their own subjective opinions. Because we intended to evaluate the complexity of the entire scene, instead of single component, users were asked to rate all factors at once. Accordingly, we applied the direct rating [52] of asking the users to assign the rank of each factor and defined the ranks as the ***raw score***s, and summarize them as subjective weighting [53]. Furthermore, we categorized g_b, g_d, and g_v as global factors because their scopes involved the entire content generation, for others being the local factors, and compute the ranks among the local/global factors to acquire the ***local score***s and the ***global score***s. We demonstrate the scores under different domains in Fig. 13 along with their averages and standard deviations. We use the average of ***local score***s and ***global score***s as the weightings, while the corresponding weighting of B and P is further multiplied with 0.5 to half the providing contribution since they are always bounded together when inserting to string, and the weighting of C, which is not involved in the user study, is directly set to 0.1 since it remains static during the chain reaction. Except for the weights of the 7 factors, we also involve the weighting to balance between the local and global contribution, which is denoted as α_g and set to 0.4 empirically. The intuition behind the scale is to let the global term have the effective value of their weightings fall into the same range of $[0, 1]$ as those local parameters, k's. Later, Section VI-C validates their effectiveness.

B. *Efficiency Enhancement in Content Generation*

Since the L-MCTS is claimed as the forward solution incorporating the inverse mechanism, we evaluate the system in two aspects:

1) **The choice of the sampling strategy**: We compare the MCTS with the random sampling to show the strength in systematical and coherent exploration.
2) **The effectiveness of the inverse embedding**: We compare the search efficiency of the framework with inverse embedding against the one without the mechanism, i.e., randomly placing elements into the scene.

Accordingly, we construct three baselines based on the combinations of (1) and (2) and denote them as **(a) random sampling with inverse embedding (RSIE)**, **(b) MCTS without inverse embedding (MCTS)** and **(c) random sampling without inverse embedding (RS)**. Technically, random sampling also samples a solution based on the 3 phases of L-string expansion, spatial positioning, and initialization, while L-string expansion samples one single string from all possible L-strings restricted by the maximum string length of n_{max}, and the rest of two phases performs random sampling based on the selected L-string. Those, that do not involve inverse embedding, choose elements' position from the entire space, while in chain reaction, the infinitely large space is restricted to a cube with a dimension of $[100, 100, 2]$. We apply all 4 methods to the three applications introduced in Section V,

TABLE I
THIS SHOWS THE STATISTICS OF GENERATION EFFICIENCY UNDER SPECIFIC ERROR PERCENTAGES, γ, FOR BREAKOUT CLONE, CHAIN REACTION, AND THE BILLIARD WHILE USING OUR ALGORITHM (L-MCTS), MCTS WITHOUT INVERSE EMBEDDING (MCTS), RANDOM SAMPLING WITH INVERSE EMBEDDING (RSIE), AND RANDOM SAMPLING WITHOUT INVERSE EMBEDDING (RS).

	Breakout Clone			Chain Reaction			Billiard		
	$\gamma = 0.05$	$\gamma = 0.025$	$\gamma = 0.01$	$\gamma = 0.05$	$\gamma = 0.025$	$\gamma = 0.01$	$\gamma = 0.05$	$\gamma = 0.025$	$\gamma = 0.01$
L-MCTS	**4519**	**4347**	**683**	**2515**	**1468**	**501**	**2963**	**2775**	**854**
RSIE	195	101	24	27	10	3	13	6	3
MCTS	4	2	2	0	0	0	4	2	1
RS	9	3	2	0	0	0	1	1	0

with the targeted goal and maximum string length configured as following:

- **One-shot-all-eliminate Breakout Clone**: we set target difficulty to 75, $C_\alpha = 3.0$ and $n_{max} = 6$.
- **Chain Reaction**: we set desired time duration to 10 seconds, $C_\alpha = 3.0$ and $n_{max} = 10$.
- **One-shot-all-sink Billiard**: we set target difficulty to 1000, $C_\alpha = 1.85$ and $n_{max} = 6$

We use the same C_α for both our L-MCTS and the **MCTS**. For each method, we randomly generate 5000 sets and record the number achieved under the targeting error percentage, γ, of 0.05, 0.025, and 0.01. As shown in Table I, our L-MCTS outperforms others with a significant gap. Accordingly, we can conclude with two folds. On one hand, 3 compared methods show low searching effectiveness to demonstrate the difficulty to sample goal-fulfilling instances while L-MCTS can enhance this search efficiency using coherently hierarchical exploration. On the other hand, the inverse embedding technique shows significant impact on improving the search efficiency, but the choice of the forward sampling strategy also plays an important role to produce massive amounts of desired results.

C. Chain Reaction Complexity Metric

To evaluate whether the proposed subjective metric is capable of revealing the viewers' opinion, we conducted a user study that asked the subjects to grade the chain reaction scene based on the visual complexity. First of all, we categorized the scenes into 5 complexity levels whose mean complexity is $32.6 \cdot 1.75^i, i \in \{1, 2, 3, 4, 5\}$. The category was designed to be equally distributed within the minimum and maximum complexity, which were both chosen based on authors' experiences. We generated 2 scenes from each level with the error percentage threshold γ set to 0.05 as the ***reference scene***s. Also, we generated 2 scenes for each level, resulting in 10 scenes in total, and denoted them as the ***evaluation scene***s. Due to the length limitation, here, we only show one scene for $i = 1, 3, 5$ from the ***reference scene***s and the ***evaluation scene***s in Fig. 14 as an example. Please refer to the supplementary materials for the actual videos shown to the user. During the user study, we first showed the ***reference scene***s to the subjects as well as their designated visual complexity levels. Then, all participants were asked to assign their subjective complexity level for all ***evaluation scene***s arranged in a random order in a 5-point Likert scale manner, where 1 represented the least complex and 5 represented the most complex. We conducted the user study on 16 subjects who were aged between 20 and 29, with 15 males and 1 female, and there were 7 majoring in computer graphics and 2 had experience in animation design. We sum up all user assigned levels for each scene as its complexity scores, shown in Fig. 15. It is obvious to see the user assigned level is highly correlated to the designated level. To further justify the assumption, we calculate the Spearman's rank correlation coefficient [54] between i and $\bar{i}$, for $\bar{i}$ being the user assigned level, and acquire $\rho = 0.9833, p \approx 0.0$. The detailed equation is provided in Appendix B. The interpretation of $\rho > 0$ shows that the user assigned level is positively correlated to the designated level. In addition, $|\rho| \to 1$ indicates the degree of correlation and $|\rho| > 0.7$ is usually to be seen as a strong correlation evidence [55]. Also, the p-value indicates the probability to observe such result under the null hypothesis being $\rho = 0$, and since $p \approx 0.0 < 10^{-3}$, it is statistically significant to conclude that the proposed complexity metric can reveal the viewers' opinion.

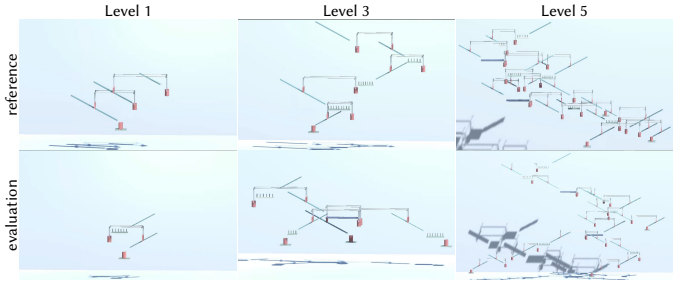


Fig. 14. This shows several examples of the ***reference scene***s and the ***evaluation scene***s for the complexity study of the chain reaction. Complete scenes and their animations can be found in the supplementary materials.

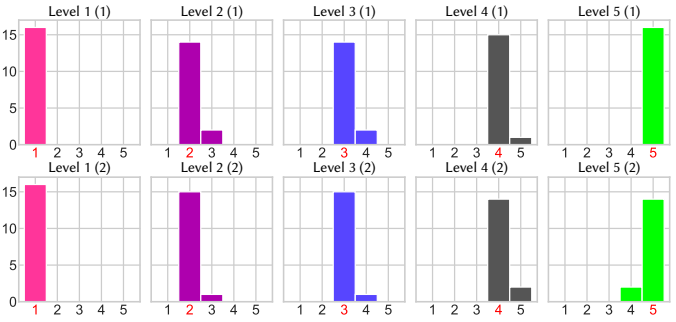


Fig. 15. This shows the user assigned complexity levels of chain reaction scenes, while the designated complexity level is highlighted in red in the legend and the bars are visualized in different color according to the level for clarity.

D. One-Shot-All-Sink Billiard Layout Difficulty Metric

We have also conducted a user study to validate whether the proposed difficulty metric is capable of revealing the players' opinions. We first categorized the layouts into 5 difficulty intervals of $[150 + (i-1) \cdot 210, 150 + i \cdot 210], i \in \{1, 2, 3, 4, 5\}$ denoted as very simple, simple, moderate, difficult, and very difficult based on authors' playing experience. Then, we generated 2 layouts from each interval as the ***reference layout***s

and 3 layouts from each interval as the ***evaluation layout***s. Due to the length limitation, Fig. 16 shows one selected layout of ***reference***s and ***evaluation***s for the level of $i = 1, 3, 5$, and the complete layouts for both sets are provided in the supplementary materials. Prior to the user study, we told participants the goal of one-shot-all-sink as well as the hitting interfaces of the billiard application, and let them play the game for 2 minutes to make sure all participants be familiar with the game. We started the user study by showing the all five-scaled ***reference layout***s to the user as well as their designated difficulty levels. Then, they were asked to assign the difficulty level for all ***evaluation layout***s arranged in a random order in a 5-point Likert scale manner. At each layout evaluation, we showed both the ball trajectory and the striking video to the user, and let them have three trial plays to reproduce the solution. After that, they gave the difficult level. Such an experiment design intended to prevent the user from solving the layout with a different solution. The user study was conducted on 16 subjects who were aged between 22 and 26, with 15 males and 1 female, and there were 14 having experiences in playing the billiard in real life and 2 had never played any billiard-related video games. We show the result in Fig. 17, and calculate the Spearman's rank correlation coefficient to get $\rho = 0.9191, p \approx 0.0$. Since $\rho > 0.7$ and $p \approx 0.0 < 10^{-3}$, we can conclude that there exists strong positive correlation between the user assigned level and the designated level, which indicates that the proposed difficulty metric is capable of revealing the player's opinions.

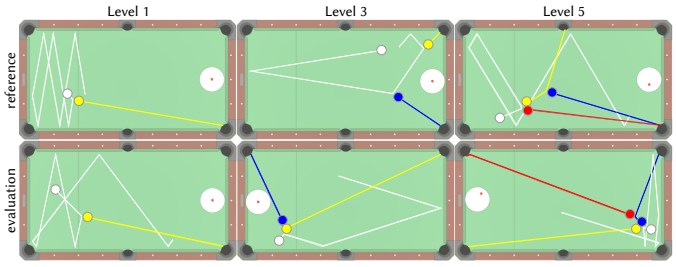


Fig. 16. This shows several examples of the ***reference layout***s and the ***evaluation layout***s for the difficulty study of the billiard. We enlarge the balls twice for better visualization. Complete scenes and their animations can be found in the supplementary materials.

VII. Conclusion and Future Work

This work proposes the L-MCTS as a general framework to produce spatio-temporal content under the user-specified metric, while experts are excluded from the workflow to achieve the goal of automatic generation and massive production. The framework involves the L-system to semantically model the elements and interactions of the content, makes use of the inverse embedding technique to improve the search efficiency, and effectively explores the content space with the MCTS sampling strategy. As experiments shown, the proposed method is capable of generating massive amount of desired content comparing to other baselines, and the generality is also demonstrated by applying the framework to 3 applications. However, there exists a few issues which are planned to be solved in the future. First, we assume that the branches

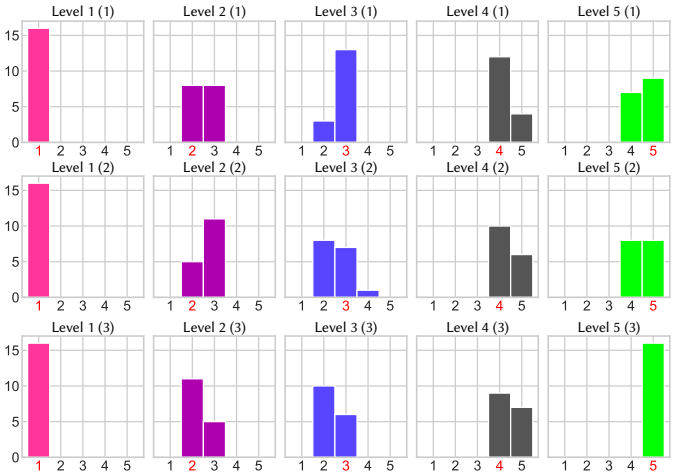


Fig. 17. This shows the user assigned difficulty levels of billiard layouts, while the designated difficulty level is highlighted in red in the legend and the bars are visualized in different color according to the level for clarity.

are totally independent and do not affect each other, while such assumption limits the possible interactions as well as the content variation. For example, one single objective ball can be hit with the cue ball multiple times before pocketing, while such behavior cannot be represented by our system. One possible solution might apply graph-based modeling for interactions, but we would need to further develop the semantic L-system denotation and deduction for it. Another issue comes from the choice of the objective metric. This paper chooses all involved metrics to be positively related to the string length, while this assumption may not always be true in practice. For example, measuring the similarities between two chain reaction scenes is in general irrelevant with the string length. Therefore, it is intriguing to validate the effectiveness of the framework on more arbitrary metrics.

Appendix A
Component Property in Chain Reaction

Due to the fact that we simulate the chain reaction with the unity built-in physical engine, we summarize the fixed properties of each component for reproducibility.

All the balls involved in the application are identical to have the radius being $0.075m$ and the mass being $10kg$. The dimensionality of each single domino is $0.18m \times 0.04m \times 0.3m$ and the mass is $5kg$. The radius of the cup mouse is $0.2m$, and its mass is $2kg$, which is same with the baffle used in the pulley set.

Appendix B
Spearman's Rank Correlation Coefficient

The Spearman's rank correlation coefficient between two variables, X and Y, is calculated as

$$\rho = \frac{\sum_i^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i^n (x_i - \bar{x})^2 \sum_i^n (y_i - \bar{y})^2}}, \tag{5}$$

where the x_i and y_i are the rank of the raw data X_i and Y_i, and n is the sample size.

REFERENCES

[1] S. Snodgrass and S. Ontañón, "A hierarchical approach to generating maps using markov chains," in *Proceedings of the Tenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, ser. AIIDE'14. AAAI Press, 2014, p. 59–65.

[2] S. Summerville, S. Philip, and M. Mateas, "Mcmcts pcg 4 smb: Monte carlo tree search to guide platformer level generation," *Artificial Intelligence in the Game Design Process*, vol. 11, no. 1, pp. 68–74, 2015.

[3] E. Hauck and C. Aranha, "Automatic generation of super mario levels via graph grammars," in *Proceedings of the 2020 IEEE Conference on Games (CoG)*, ser. CoG 2020. New York, NY, USA: IEEE, 2020, pp. 297–304.

[4] R. Rodriguez Torrado, A. Khalifa, M. Cerny Green, N. Justesen, S. Risi, and J. Togelius, "Bootstrapping conditional gans for video game level generation," in *Proceedings of the 2020 IEEE Conference on Games (CoG)*, ser. CoG 2020. New York, NY, USA: IEEE, 2020, pp. 41–48.

[5] J. Talton, Y. Lou, S. Lesser, J. Duke, R. Měch, and V. Koltun, "Metropolis procedural modeling," *ACM Trans. Graph.*, vol. 30, no. 2, pp. 11:1–11:14, 2011.

[6] O. Stava, S. Pirk, J. Kratt, B. Chen, R. Mundefinedch, O. Deussen, and B. Benes, "Inverse procedural modelling of trees," *Comput. Graph. Forum*, vol. 33, no. 6, p. 118–131, 2014.

[7] L. Ferreira and C. Toledo, "A search-based approach for generating angry birds levels," in *Proceedings of the 2014 IEEE Conference on Computational Intelligence and Games*, ser. CIG 2014. New York, NY, USA: IEEE, 2014, pp. 1–8.

[8] M. Fridenfalk, "Algorithmic music composition for computer games based on l-system," in *Proceedings of the 2015 IEEE Games Entertainment Media Conference (GEM)*, ser. GEM 2015. New York, NY, USA: IEEE, 2015, pp. 1–6.

[9] R. Roussel, M.-P. Cani, J.-C. Léon, and N. Mitra, "Designing chain reaction contraptions from causal graphs," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 43:1–43:14, jul 2019.

[10] S. Chenney and D. Forsyth, "Sampling plausible solutions to multi-body constraint problems," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '00. New York City, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 219–228.

[11] C. Twigg and D. James, "Many-worlds browsing for control of multi-body dynamics," *ACM Trans. Graph.*, vol. 26, no. 3, p. 14–es, 2007.

[12] ——, "Backward steps in rigid body simulation," in *ACM SIGGRAPH 2008 Papers*, ser. SIGGRAPH '08. New York, NY, USA: ACM, 2008.

[13] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*. Heidelberg, Berlin, Germany: Springer-Verlag, 1990.

[14] P. Prusinkiewicz, M. James, and R. Měch, "Synthetic topiary," in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '94. New York, NY, USA: ACM, 1994, pp. 351–358.

[15] P. Prusinkiewicz, L. Mündermann, R. Karwowski, and B. Lane, "The use of positional information in the modeling of plants," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 289–300.

[16] Y. Parish and P. Müller, "Procedural modeling of cities," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 301–308.

[17] J.-E. Marvie, J. Perret, and K. Bouatouch, "The fl-system: A functional l-system for procedural geometric modeling," *The Vis. Comput.*, vol. 21, no. 5, pp. 329–339, 2005.

[18] B. Ganster and R. Klein, "An integrated framework for procedural modeling," in *Proceedings of the 23rd Spring Conference on Computer Graphics*, ser. SCCG '07. New York, NY, USA: ACM, 2007, pp. 123–130.

[19] M. Lipp, P. Wonka, and M. Wimmer, "Interactive visual editing of grammars for procedural architecture," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 102:1–102:10, 2008.

[20] G. Patow, "User-friendly graph editing for procedural modeling of buildings," *IEEE Comput. Graph. Appl.*, vol. 32, no. 2, pp. 66–75, 2012.

[21] G. Smith, M. Treanor, J. Whitehead, and M. Mateas, "Rhythm-based level generation for 2d platformers," in *Proceedings of the 4th International Conference on Foundations of Digital Games*, ser. FDG '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 175–182.

[22] N. Shaker, J. Togelius, G. Yannakakis, B. Weber, T. Shimizu, T. Hashiyama, N. Sorenson, P. Pasquier, P. Mawhorter, G. Takahashi, G. Smith, and R. Baumgarten, "The 2010 mario ai championship: Level generation track," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 4, pp. 332–347, 2011.

[23] N. Shaker, G. Yannakakis, J. Togelius, M. Nicolau, and M. O'Neill, "Evolving personalized content for super mario bros using grammatical evolution," in *Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, ser. AIIDE'12. AAAI Press, 2012, p. 75–80.

[24] J. Togelius, M. Preuss, N. Beume, S. Wessing, J. Hagelbäck, and G. N. Yannakakis, "Multiobjective exploration of the starcraft map space," in *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, ser. CIG 2010. New York, NY, USA: IEEE, 2010, pp. 265–272.

[25] J. Togelius, G. Yannakakis, K. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 172–186, 2011.

[26] J. Togelius, M. Preuss, N. Beume, S. Wessing, J. Hagelbäck, G. Yannakakis, and C. Grappiolo, "The 2010 mario ai championship: Level generation track," *Genetic Programming and Evolvable Machines*, vol. 14, pp. 245–277, 2013.

[27] S. Dahlskog, J. Togelius, and M. J. Nelson, "Linear levels through n-grams," in *Proceedings of the 18th International Academic MindTrek Conference: Media Business, Management, Content & Services*, ser. AcademicMindTrek '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 200–206.

[28] M. Traichioiu, S. Bakkes, and D. Roijers, "Grammar-based procedural content generation from designer-provided difficulty curves," in *Proceedings of the 10th International Conference on the Foundations of Digital Games*, ser. FDG 2015. New York, NY, USA: Society for the Advancement of the Science of Digital Games, 2015, pp. 1–5.

[29] V. Volz, J. Schrum, J. Liu, S. M. Lucas, A. Smith, and S. Risi, "Evolving mario levels in the latent space of a deep convolutional generative adversarial network," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 221–228.

[30] J. Popović, S. Seitz, M. Erdmann, Z. Popović, and A. Witkin, "Interactive manipulation of rigid body simulations," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. USA: ACM Press/Addison-Wesley Publishing Co., 2000, p. 209–217.

[31] M. Anderson, E. McDaniel, and S. Chenney, "Constrained animation of flocks," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '03. Goslar, DEU: Eurographics Association, 2003, p. 286–297.

[32] M. Gleicher, "Motion editing with spacetime constraints," in *Proceedings of the 1997 symposium on Interactive 3D graphics*, ser. I3D '97. New York, NY, USA: ACM, 1997, pp. 139–ff.

[33] Y. Ye and C. Liu, "Synthesis of detailed hand manipulations using contact sampling," *ACM Trans. Graph.*, vol. 31, no. 4, 2012.

[34] L. Liu, K. Yin, and B. Guo, "Improving sampling-based motion control," *Computer Graphics Forum*, vol. 34, no. 2, pp. 415–423, 2015.

[35] R. Kondo, T. Kanai, and K.-i. Anjyo, "Directable animation of elastic objects," in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '05. New York, NY, USA: ACM, 2005, p. 127–134.

[36] J. Barbič, M. da Silva, and J. Popović, "Deformable object animation using reduced optimal control," *ACM Trans. Graph.*, vol. 28, no. 3, jul 2009.

[37] A. Treuille, A. McNamara, Z. Popović, and J. Stam, "Keyframe control of smoke simulations," *ACM Trans. Graph.*, vol. 22, no. 3, p. 716–723, 2003.

[38] Z. Pan, J. Huang, Y. Tong, C. Zheng, and H. Bao, "Interactive localized liquid motion editing," *ACM Trans. Graph.*, vol. 32, no. 6, nov 2013.

[39] Y. Peng, Y. Mishima, Y. Igarashi, R. Miyauchi, M. Okawa, H. Xie, and K. Miyata, "Sketch2domino: Interactive chain reaction design and guidance," in *2020 Nicograph International (NicoInt)*. Los Alamitos, CA, USA: IEEE Computer Society, 2020, pp. 32–38.

[40] L. Zhu, W. Xu, J. Snyder, Y. Liu, G. Wang, and B. Guo, "Motion-guided mechanical toy modeling," *ACM Trans. Graph.*, vol. 31, no. 6, 2012.

[41] Y. Cheng, Y. Sun, P. Song, and L. Liu, "Spatial-temporal motion control via composite cam-follower mechanisms," *ACM Trans. Graph.*, vol. 40, no. 6, dec 2021.

[42] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton,

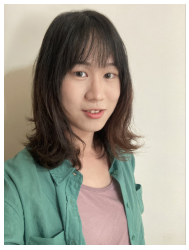
"A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.
[43] K. Yin, L. Liu, and M. de Panne, *Simulation for Control*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2018, pp. 1–44.
[44] S. Agrawal, S. Shen, and M. van de Panne, "Diverse motion variations for physics-based character animation," in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '13. New York, NY, USA: ACM, 2013, p. 37–44.
[45] F.-Y. Meng, J. Tang, W. Pedrycz, and Q.-X. An, "Optimal interaction priority calculation from hesitant fuzzy preference relations based on the monte carlo simulation method for the acceptable consistency and consensus," *IEEE Transactions on Cybernetics*, vol. 51, no. 12, pp. 5871–5882, 2020.
[46] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Driessche, J. S., I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 2016.
[47] N. Qi, Z. Fan, M. Huo, D. Du, and C. Zhao, "Fast trajectory generation and asteroid sequence selection in multispacecraft for multiasteroid exploration," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 6071–6082, 2020.
[48] H. Hong, M. Jiang, and G. G. Yen, "Improving performance insensitivity of large-scale multiobjective optimization via monte carlo tree search," *IEEE Transactions on Cybernetics*, 2023.
[49] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *Machine Learning: ECML 2006*. Berlin, Heidelberg, Germany: Springer Berlin Heidelberg, 2006, pp. 282–293.
[50] D. G. Alciatore, *The illustrated principles of pool and billiards*. Dr. Dave Billiards Resources, 2004.
[51] R. Byrne, *Byrne's Treasury of Trick Shots in Pool and Billiards*. Houghton Mifflin Harcourt, 1983.
[52] A. Arbel, "Approximate articulation of preference and priority derivation," *European Journal of Operational Research*, vol. 43, no. 3, pp. 317–326, 1989.
[53] G. Odu, "Weighting methods for multi-criteria decision making technique," *Journal of Applied Sciences and Environmental Management*, vol. 23, no. 8, pp. 1449–1457, 2019.
[54] C. Spearman, "The proof and measurement of association between two things." 1961.
[55] L. Leclezio, A. Jansen, V. H. Whittemore, and P. J. de Vries, "Pilot validation of the tuberous sclerosis-associated neuropsychiatric disorders (tand) checklist," *Pediatric Neurology*, vol. 52, no. 1, pp. 16–24, 2015.

Chia-Hsing Chiu is currently a Ph.D student at Department of Computer Science and Information Engineering, the National Taiwan University of Science and Technology, Taiwan. His research interests include computer graphics and multimedia.

Yu-Chi Lai is currently a professor in Computer Science and Information Engineering with the National Taiwan University of Science and Technology, Taiwan, and his research interests include computer graphics and multimedia. He received the M.S. and Ph.D. degrees in CS from UW-Madison, Madison, WI, USA, in 2004 and 2010, respectively.

Andrew Chen is currently pursuing his graduate degree in Computer Science at the University of Illinois Urbana-Champaign, IL, USA. He completed his undergraduate studies at National Taiwan University of Science and Technology, Taiwan, where he earned dual bachelor degrees in both Computer Science and Information Engineering & Electrical Engineering. His research interests primary lie in computer graphics and human-computer interaction, with a focus on creativity-supporting design tools.

Pai-Yi Su is currently a M.A student at Department of Computer Science and Engineering, the National Taiwan Ocean University, Taiwan. His research interests include computer graphics and image processing.



Wei-Yao Ku completed a master's in 2022, and is currently a Ph.D. student in Computer Science and Engineering of the National Taiwan University of Science and Technology, Taiwan. His research interests involve computer graphics and physical simulation.

Wen-Kai Tai received his M.S. and Ph.D. degrees in Computer Science from National Chiao Tung University in 1989 and 1995, respectively. He is currently a professor in the Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology, Taiwan. His research interests include animation, real-time rendering, and gaming.

Shih-Syun Lin (Member, IEEE) received the Ph.D. degree in computer science and information engineering from National Cheng-Kung University (NCKU), Tainan City, Taiwan, in 2015. Since 2016, he has been a Faculty Member with the Department of Computer Science and Engineering, National Taiwan Ocean University (NTOU), Keelung City, Taiwan, where he is currently an Associate Professor with the Department of Computer Science and Engineering. He also leads the Intelligent Graphics Laboratory. His research interests include computer graphics, information visualization, information platform, and pattern recognition.