# Tellimation: Just-in-time Scene Element Animation for Supporting Children's Storytelling

Anonymous Author(s)

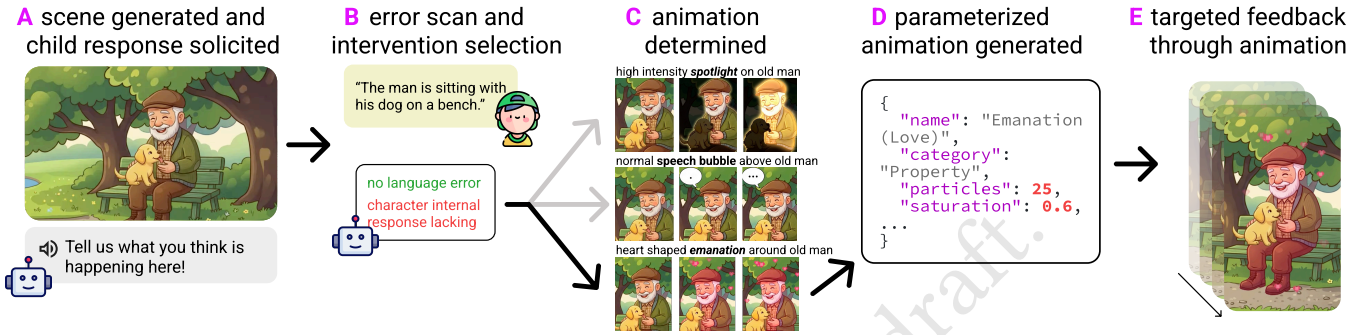


**Figure 1: A Tellimation interaction cycle. (A) The child sees an illustrated scene and is invited to narrate. (B) The child describes the scene but does not mention how the man feels about his dog; the pipeline detects no factual error but identifies a missing internal response. (C) The system consults the animation grammar for candidates that can scaffold this narrative gap. (D) The pipeline selects Emanation (Love) and parameterizes it. (E) Heart particles appear around the man, making the emotion visible and inviting the child to verbalize it.**

## Abstract

Illustrated scenes are widely used to elicit expressive narratives from children, yet children with language difficulties often struggle to verbalize visual details, omitting actions, relations between entities, and markers of coherence. We introduce Tellimation, a system that provides real-time visual feedback on children's narration by animating scene elements the child has omitted or described incorrectly. At its core is a grammar of 20 animations organized into 8 semantic categories, grounded in classical animation principles and visual conventions from comics. These animations are driven by a pipeline that detects discrepancies between the child's utterance and the scene, selects feedback based on the child's history and the scene's narrative potential, and monitors whether the child incorporates prior feedback. We validate the grammar's interpretability (N=80) and evaluate Tellimation's effect on narrative richness in a within-subjects study (N=8). Children in the animation condition produced a broader range of narrative elements and self-corrected more frequently than in a control condition.

**ACM Reference Format:**
Anonymous Author(s). 2018. Tellimation: Just-in-time Scene Element Animation for Supporting Children's Storytelling. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX


*Conference acronym 'XX, Woodstock, NY*

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

## 1 Introduction

Illustrated scenes are widely used in clinical and educational settings to elicit narratives from children. A clinician or teacher shows the child a picture and asks them to tell a story about what they see. The richness of the child's narration (the characters they name, the actions they describe, the causal links they draw, the emotions they attribute) is one of the most reliable indicators of language development. Yet children with language difficulties often struggle to verbalize what they see: they omit descriptive adjectives, skip over actions and spatial relations, fail to express how characters feel, and produce narratives that lack sequence and coherence. The illustrated scene contains all the information the child needs, but the gap between what is visible and what the child says remains invisible to both of them.

We set out to make that gap visible. We introduce Tellimation, a system that provides real-time visual feedback on a child's narration by animating the scene elements the child has omitted or described incorrectly. When a child says "the cat" but the scene shows a fluffy orange cat perched on a fence, the system does not tell the child what to say: it animates the cat's color to make the missing detail perceptible, and trusts the child to incorporate it. When the child has described the characters but has not mentioned that the old man loves his dog, heart particles appear around him, making the emotion visible. The child remains the author of the story; the system's role is to make the picture listen.

At the core of Tellimation is a grammar of 20 animations organized into 8 semantic categories, derived from a bottom-up analysis of children's oral language errors and grounded in classical animation principles and visual conventions from cartoons, comics, and

video games. These conventions (motion lines for speed, speech bubbles for dialogue, particle effects for emotions) are ones that children already know from illustrated media, which means the animations are interpretable without training. The grammar covers the 15 elements of the MISL narrative proficiency framework [22], spanning both macrostructure (characters, settings, causal links) and microstructure (conjunctions, adverbs, tense consistency). This design inverts the direction of Kineticons [17], which define a fixed lexicon mapping motion patterns to semantic meanings that users must learn. Tellimation maps meaning to motion: the system starts from a semantic objective (a narrative element the child should produce) and selects the visual transformation most likely to make the corresponding gap perceptible. The child does not need to learn what each animation means, because the animations exploit existing visual literacy to make discrepancies self-evident.

The animations are driven by a hybrid pipeline that runs after each utterance. A speech-to-text model transcribes the child's utterance, after which two concurrent paths combine model calls with deterministic selection rules: one detects factual errors, the other consults the child's mention history to identify enrichment opportunities. A compatibility check filters out the model's occasional hallucinations before they reach the child. If a previous animation went unaddressed, the system records it and amplifies the animation's parameters the next time the same type of gap arises. The pipeline completes a full cycle in a median of 5.2 seconds, fast enough for the system to respond before the child's attention shifts.

We make three contributions:

(1) An **animation design space and grammar** of 20 animations organized into 8 semantic categories, derived from oral language error classification and grounded in visual conventions from animated media.
(2) A **real-time pipeline** that detects discrepancies between the child's utterance and the scene, selects an animation from the grammar, validates it against the scene's entity registry, and parameterizes it based on the child's interaction history.
(3) An **evaluation** in three stages: a pipeline validation with 200 synthetic utterances rated by 80 adult participants, an interpretability study (N=80) showing that 22 of the 25 tested animations are correctly interpreted above chance, and a within-subjects study with 8 children showing that animations increase the number of narrative elements produced and the number of discrepancies the child addresses.

## 2 Background and Related Work

### 2.1 Narrative Storytelling and Assessment

Oral narratives are among the most widely used measures of children's language development [3, 31]. When asked to tell a story from a picture, a child must name characters, describe actions, establish spatial and temporal relations, attribute emotions, and connect events causally. The richness and coherence of the resulting narratives predict reading comprehension and academic achievement [4, 16]. These skills develop substantially between ages 5 and 12, with the critical transition from simple event descriptions to causally structured multi-episode narratives occurring around ages 7 to 9 [2, 37, 43]. This is why our target population begins at age 7: younger children are still developing the foundational narrative competencies that the system is designed to scaffold.

Children with developmental language disorder (DLD) and other language difficulties produce narratives that are shorter, less cohesive, and structurally simpler than those of their peers [3, 15, 28, 29]. They tend to omit initiating events, internal responses, and causal connectives, producing sequences of loosely related events rather than structured plots [18, 33]. Narrative intervention programs such as SKILL [12] have shown that structured instruction can improve these skills, but they require direct clinician involvement and cannot scale beyond the therapy session.

Several frameworks exist for assessing narrative proficiency. Story grammar analysis, rooted in schema theory [43], evaluates the presence of canonical story elements (setting, problem, resolution). The Monitoring Indicators of Scholarly Language (MISL) rubric [13, 22], grounded in discourse theory [14, 23], scores 15 elements across two subscales. Macrostructure captures story grammar: Character (CH), Setting (S), Initiating Event (IE), Internal Response (IR), Plan (P), Action (A), and Consequence (CO). Microstructure captures linguistic devices: coordinating conjunctions (CC), subordinating conjunctions (SC), mental verbs (M), linguistic verbs (L), adverbs (ADV), elaborated noun phrases (ENP), grammaticality (G), and tense consistency (T). Together these elements distinguish proficient narrators from emerging ones. We adopt MISL as the scaffolding target for Tellimation because it provides a fine-grained, element-level description of what the child should produce, which maps directly onto specific visual feedback strategies.

Digital storytelling tools for children exist in many forms, from open-ended creation environments [41] to robot-mediated interactions [24] and AI-assisted co-authoring systems [6, 8, 10, 35, 46, 47]; a scoping review surveys this growing landscape [21]. These systems support story creation (generating content with or for the child) rather than story assessment (analyzing and scaffolding the child's own production). The closest system to ours is OSOS [26], which generates personalized storybooks targeting vocabulary gaps identified through pervasive profiling of the child's language environment. Like Tellimation, it is co-developed with speech-language pathologists and uses generative AI to adapt to individual children. But OSOS generates stories for the child to read, whereas Tellimation responds to stories the child tells: the feedback is visual, immediate, and contingent on what the child just said. On the clinical side, card-based intervention programs such as Story Champs [42] and picture description tasks [11, 39] provide no real-time feedback: the clinician scores after the session and plans the next intervention manually. Tellimation closes this loop by analyzing the child's speech and responding visually within the same storytelling session.

### 2.2 Animation as Communication

Animation has a long history as a communication medium in user interfaces. Baecker and Small [1] argued early on that animation could convey state, affordance, and transition in ways that static graphics cannot. Chang and Ungar [5] demonstrated that cartoon animation techniques (squash-and-stretch, anticipation, slow-in slow-out) could make interface operations more legible, drawing directly on the classical animation principles formalized by Thomas

and Johnston [44]. More recently, Chevalier et al. [7] and Heer and Robertson [19] showed that animated transitions in information visualization help users track changes and maintain context. Lee et al. [27] showed that kinetic typography could convey emotional tone through text animation, while Hudson and Stasko [20] provided early toolkit support for integrating animation into interactive systems, establishing animation as a first-class primitive in UI construction.

The most directly relevant prior work is Kineticons [17], which defines a vocabulary of motion-based icons (bouncing, shaking, pulsing) that communicate states and affordances in graphical interfaces. A 200-participant study showed that users could reliably interpret many of these motions, but the mapping from motion to meaning must be learned: a "shake" means rejection only if the user knows the convention. Tellimation inverts this relationship. Rather than defining a fixed vocabulary of motions and asking users to learn their meanings, the system starts from a semantic objective and selects the animation most likely to make the corresponding narrative gap perceptible. This inversion is possible because our grammar draws on visual conventions that children already know from comics, cartoons, and video games: emanata (the symbolic particles cataloged by Walker [45] and analyzed by McCloud [32]), motion lines, speech bubbles, and the timing principles of classical animation [25, 44].

Recent advances in generative models have made it feasible to produce visual content on demand. Text-to-image models [38, 40] can generate scene illustrations from prompts, and vision-language models can analyze and describe visual content [30]. Tellimation uses these capabilities for both scene preparation (generating illustrations and extracting entity layers) and live interaction (analyzing the child's utterance against the scene). The animation itself, however, is procedural rather than generative: we define parameterized templates that transform pre-extracted entity layers, which ensures that the visual feedback is deterministic, fast, and controllable.

## 3 User Experience

A Tellimation session begins with a child looking at an illustrated scene and narrating what they see. There are no menus, no instructions, and no visible interface beyond the scene and a push-to-talk button. The child holds the button, speaks, releases it, and the scene reacts: a brief animation draws attention to something the child said wrong or left out. The animation loops until the child speaks again, establishing a turn-taking rhythm (speak, watch, speak) that structures the interaction without requiring the child to learn any rules.

Consider a child who says "the boy is sitting" while the scene shows the boy running. The system plays Motion Lines (A1), streaking dynamic lines around the character to make the movement visible. The child sees the contradiction and self-corrects. Later, a scene shows an old man sitting with his dog in a park. The child describes what they see but does not mention how the man feels. The system plays Emanation (P2), adding small heart particles around the man to make his affection for the dog visible, inviting the child to verbalize the emotion. In both cases, the system never tells the child what to say: it makes a gap visible and trusts the child to fill it. If the child ignores the feedback, the system replays the animation with exaggerated parameters (a brighter spotlight, more saturated colors, higher count of particles or range of motion...) on the next opportunity. Once the child addresses the element, the system moves on. Across multiple scenes, the system carries forward the child's profile (character names, narrative elements already practiced) so that feedback builds on what came before. The overall experience is designed to feel closer to a responsive picture book than to a tutoring system: the child controls the pace, and the system's role remains invisible.

## 4 Tellimation

A Tellimation is an animation that responds to a gap between what a child says and what the scene contains. When a child describes "the cat" but the scene shows a fluffy orange cat perched on a fence, the system must make the missing detail perceptible without telling the child what to say. This section presents the design constraints that shape both the visual style and the animations themselves, the design space that organizes the types of narrative gaps the system can address, and the resulting animation grammar that populates it.

### 4.1 Design Constraints

Generating real-time visual feedback for children's storytelling imposes six constraints. The visual style must be expressive (rich enough for narrative immersion), fast to produce (compatible with just-in-time generation), programmatically addressable (individual entities can be isolated and manipulated at runtime), and aesthetically coherent (elements from heterogeneous sources must look like they belong together). Two additional constraints shape the animations themselves: implicit scaffolding (the animation draws attention to a narrative gap without telling the child what to say) and legibility without instruction (animations must be interpretable on first encounter by children aged 7 to 11, which rules out arbitrary mappings and favors visual conventions from picture books, cartoons, and comics).

Under these constraints, we adopt a high-definition children's storybook illustration style. To satisfy aesthetic coherence, we use reference-guided generation: key entities are first generated individually to establish a consistent visual identity, then each scene is generated as a complete illustration that includes all entities, so that they share the same lighting and perspective by construction. Entities are then extracted as independent transparent layers (Section 5.1), retaining correct scale and color relationships when composited back onto the background. We considered and rejected pixel art (insufficient expressiveness), vector illustration (too slow), flat design (insufficient richness), and paper cutout style (incompatible with procedural animation).

### 4.2 Animation Design Space

We derived the design space through a bottom-up process that began with language errors, not with animations. Children's oral language is traditionally analyzed across several levels of linguistic structure, from phonology and morphology through syntax and semantics to pragmatics [34], and errors at each level have been classified in different ways and at different granularities. At a high level of abstraction, the surface strategy taxonomy of Dulay, Burt, and

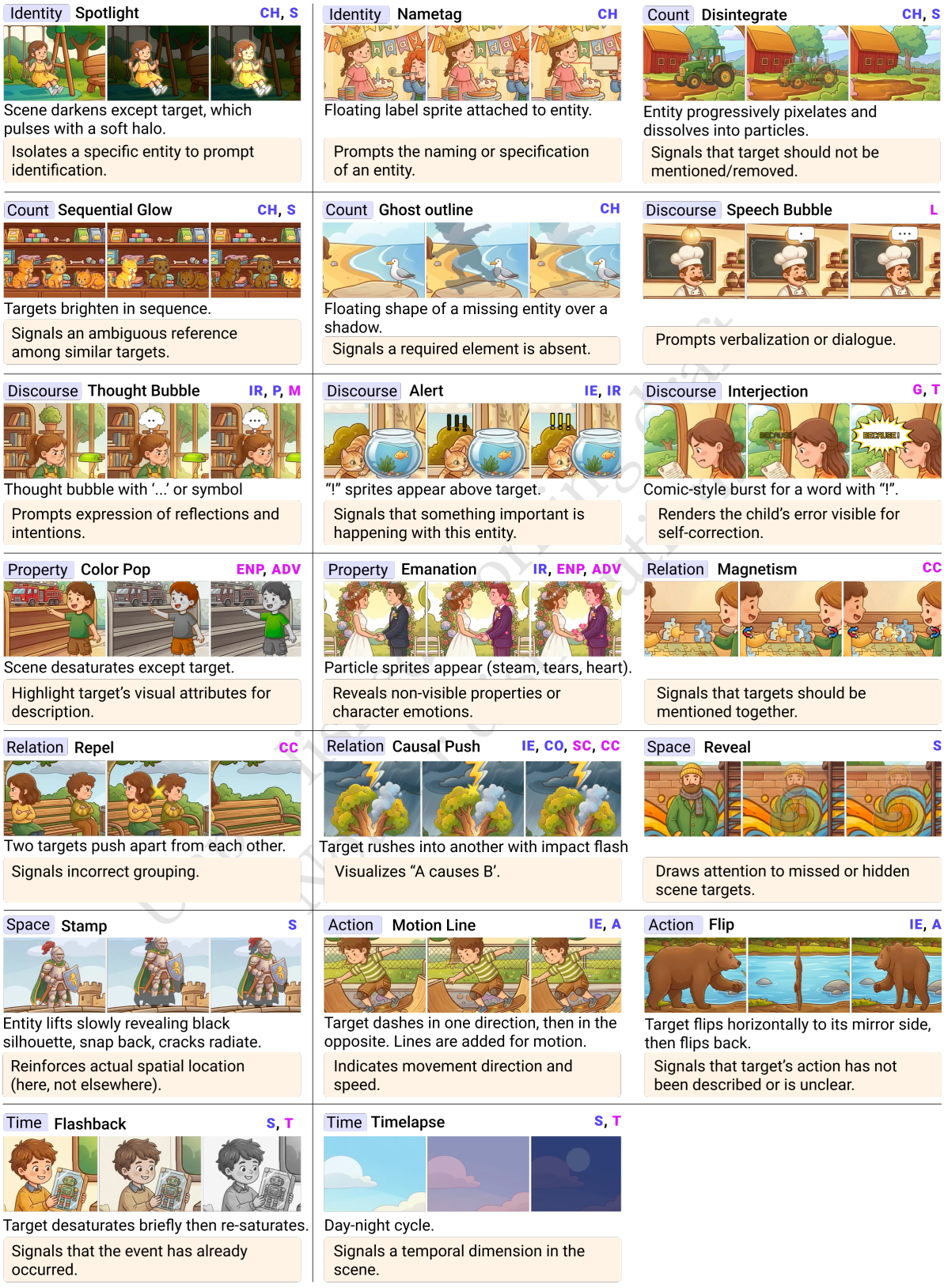


**Figure 2: The 20 animations of the Tellimation grammar. Each cell shows three keyframes, the animation's category (top left), the MISL elements it scaffolds (top right), its description and its scaffolding rationale. Categories span a progression from concrete (Identity, Count) to abstract (Relation, Discourse). The many-to-many mapping between animations and MISL elements is visible in the tags: a single animation may scaffold multiple elements depending on context, and a single element may be addressed by different animations.**

Krashen [9] groups all production errors into four operations (omission, addition, misformation, and misordering), while finer-grained catalogs used in clinical and educational practice enumerate dozens of specific phenomena such as verb tense inconsistencies, pronoun reference ambiguities, or determiner-noun mismatches [36]. Neither level of description was directly usable for our purposes. The high-level taxonomy is too abstract to suggest a visual response: an "omission" animation would be meaningless without knowing what was omitted. Fine-grained error lists, conversely, are too numerous and too close to surface form to cluster into a manageable set of visual strategies.

We therefore compiled errors from both levels and asked, for each one, what aspect of the illustrated scene it relates to. This question served as the clustering principle: errors that require the system to draw attention to the same type of scene element belong together, regardless of their linguistic classification. Pronoun clarity errors, vague references, and pronoun case errors, for instance, all concern which entity the child is talking about and can be resolved by directing the child's attention to that entity. Verb tense errors and temporal ordering mistakes both concern when an event takes place and call for a visual cue that foregrounds the scene's temporal dimension. This clustering naturally excluded phonological errors, because Tellimation operates on illustrated scenes and has no visual means to address errors of pronunciation or sound production. The process was iterative: as we later mapped candidate animations to error groups, we refined the groups themselves. We initially maintained separate categories for "entity reference" and "entity absence," for example, but found that the same set of visual strategies (sequential highlighting, dissolution, ghost outlines) could address both, which led us to merge them. The process converged on eight categories (Figure 2), each corresponding to a distinct type of mismatch between utterance and scene that requires a qualitatively different perceptual strategy to resolve. These categories span a progression from concrete to abstract scaffolding: the first categories concern basic identification and description of entities, while the later ones address the causal and temporal connectives that discourse theory identifies as the critical differentiator between emerging and proficient narrators [22].

With these categories established, we turned to the question of which animations could make each type of gap perceptible. The legibility constraint (Section 4.1) requires that children interpret the animations on first encounter, without training, which means we could not invent arbitrary visual conventions and expect children to learn them. We therefore searched for conventions that children already know. The first two authors surveyed animated media, watching cartoons ranging from Disney productions of the 1940s to more recent series such as Oggy and the Cockroaches, and cataloging the visual devices used to convey meaning without dialogue: the streaking lines that signal a character's speed, the dust clouds that mark an impact, the spirals that indicate dizziness. They also surveyed the visual language of comics, drawing on McCloud's taxonomy of iconic conventions [32] and Walker's catalog of emanata, the symbolic particles that signal emotions or sensations [45]. A third source proved particularly relevant: the animated sprites commonly used in video games. Because sprites are procedurally generated overlays that can be produced at runtime without prior asset creation, they are compatible with the generation speed constraint. This survey produced a list of 37 candidate animations.

We selected from this list by retaining only animations that met three requirements: the animation should suggest a single interpretation in context, it should work when applied to an object, a character, or the scene as a whole, and it should not require access to internal geometry (compatibility with indivisible entity layers). The principles of classical animation [44], in particular anticipation, staging, and squash-and-stretch, informed the timing and dynamics of the retained animations, while the visual language of comics provided the symbolic vocabulary. This led us to favor two primary mechanisms: whole-entity transformations (pulsing, translating, scaling, palette shifts) and the addition of temporary overlays (bubbles, nametags, particles). The selection process yielded 20 animations that collectively cover the eight error categories, forming a grammar of visual feedback for error correction: when the child says something that contradicts the scene, the system selects the animation whose category matches the type of discrepancy and makes it visible.

But we also want to use these same animations when no factual error has occurred but the narration could be richer. This is where the MISL rubric 2.1 enters the design space. We performed a second mapping that associates each MISL element with the subset of animations capable of scaffolding it. Because a single narrative concept can be made perceptible in multiple ways, this mapping is many-to-many (Figure 2). Emanation (P2), for instance, scaffolds Elaborated Noun Phrases when it renders a physical property visible (steam rising from a hot soup invites "the hot soup"), but scaffolds Internal Response when it renders an emotion visible (tears on a character invite "the fox was sad"). Conversely, Internal Response can be addressed by Emanation when the emotion has a clear visual particle, or by Thought Bubble (D2) when the internal state is more abstract. This dual mapping over the same set of animations (one for error correction, one for narrative enrichment) is what allows Tellimation to use a single visual vocabulary for two complementary purposes, and the system resolves the resulting ambiguity at runtime by consulting the child's interaction history to select the most effective candidate (Section 5).

## 4.3 Animation Grammar

The resulting grammar contains 20 animations organized into the 8 categories of the design space (Figure 2). Section 3 illustrated how these animations manifest in the child's experience; here we describe their structure.

These examples illustrate a fundamental distinction in how the grammar operates on the scene. Some animations work by transforming entities that are already visible: Color Pop (P1) desaturates the entire scene except the target entity, making its visual attributes (color, size, texture) impossible to ignore; Reveal (S1) makes an occluding entity translucent to expose a hidden spatial detail; Flashback (T1) briefly desaturates and re-saturates an entity to signal that the event has already occurred. These transformations do not add anything to the scene; they redistribute perceptual salience so that what was already there becomes harder to overlook. Other animations, by contrast, add information that is not visually present

in the illustration. Emanation (P2) generates particle sprites (steam, frost, hearts) to externalize a property or emotion that the static image cannot show on its own. Speech Bubble (D1) and Thought Bubble (D2) display bubbles with ellipses or keywords to prompt dialogue and internal states that no illustration can depict without text. Alert (D3) places "!!!" above an entity to signal narrative importance, and Interjection (D4) renders the child's own problematic word in a comic-style burst to make a speech error visible for self-correction. Ghost Outline (C3) goes further still: it borrows the silhouette of an entity from another scene and displays it as a translucent outline to signal that something is missing from the current scene entirely. These overlays are generated procedurally as pixel-art sprites, and their stylized appearance is a deliberate design choice. The contrast between the high-definition storybook illustration and the flat, pixelated overlay serves as a perceptual cue: it signals immediately that the overlay is system feedback rather than part of the original artwork. The child does not need to be told that the floating label or the particle effect is a prompt; the stylistic break makes it self-evident, satisfying the legibility constraint without any instruction.

Each animation is parameterized. Color Pop can have various saturation levels, Motion Lines can have various amplitudes, and the system adjusts these parameters based on the scene context and the child's profile. When the same type of discrepancy has been raised before and went unresolved, the system shifts parameters toward their maximum value to make the cue more noticeable, embodying the implicit scaffolding constraint: a gentle visual hint first, then a progressively amplified one if the child does not respond (Section 5). Each animation is defined by an independent JavaScript template for its rendering code and a JSON file that specifies its metadata, parameters, and scaffolding intents. This separation makes the grammar straightforward to extend: adding a new animation requires only a new template and a new JSON file, without modifying any other part of the codebase. At present, the 20 animations expose a total of 47 adjustable parameters, ranging from scalar values (saturation, amplitude, speed) to discrete choices (particle type, bubble content). The grammar thus functions not as a closed set, but as an extensible vocabulary that any system can adopt and build upon.

Where Kineticons [17] require users to learn what each motion means, the grammar described above lets the system choose the motion: it starts from the narrative gap and selects the transformation most likely to make it perceptible, given the child's interaction history and the scene's affordances.

# 5 System Architecture

The previous section defined what the system communicates (the animation grammar) and why (the design space); this section describes how. Tellimation operates in two phases. An offline phase generates illustrated scenes, extracts entity layers, and annotates each scene with its narrative potential. A live interaction phase processes the child's speech in real time, detects discrepancies between the utterance and the scene, selects an animation from the grammar, and sends it to the client for execution. We describe each phase in turn.

## 5.1 Scene Preparation

Before a storytelling session begins, each scene goes through an offline preparation pipeline that produces the visual assets and narrative metadata required by the live interaction loop. This pipeline involves image generation, entity extraction, and narrative annotation, with human verification at each stage (Figure 4 in Appendix). We used this pipeline to prepare 20 scenes for the children's study (4 stories of 5 scenes each) (section 6.3) and 100 additional scenes for the pipeline validation and interpretability studies (Sections 6.1 and 6.2).

*Visual asset generation.* For each story, we first write prompts describing the key characters and objects and generate them individually on a white background using Gemini 3.1 Flash Image Preview. These isolated references establish a consistent visual identity for each entity across scenes. For each scene, we then write two separate prompts: one for the background alone, and one for the complete scene including all entities. Both are generated by the same model, which uses the entity references to maintain visual consistency across scenes. This scene-first strategy (generating all entities within a single coherent illustration) ensures that characters share the same lighting, palette, and perspective by construction, as discussed in Section 4.1.

*Entity extraction.* We then extract each entity from the complete scene as an independent layer. A multimodal model (Gemini 3 Flash, used here as a vision-language model) receives the complete scene and isolates each character and object onto a white background, preserving the entity's position and the global image dimensions. A first human verification step ensures that the background is indeed uniformly white, with manual correction in an image editor when needed. A background removal model (BRIA) then converts the white background to transparency, after which a second human verification step confirms that the result is a clean transparent layer. At the end of this process, we have the background as a standalone image and each entity as an independent transparent layer already positioned at its correct location in the scene. Each entity receives a stable identifier (e.g., `boy`, `grandmother`, `balloon`) that persists throughout the interaction. These layers enable the animation runner to apply transformations to individual entities at runtime without altering the rest of the scene.

*Narrative annotation.* The complete scene image serves a second purpose beyond visual asset generation: it is the input for narrative analysis. The same model, again acting as a vision-language model, analyzes the scene and extracts both physical and narrative details. This produces two outputs. First, a rich scene description (a detailed natural-language text capturing everything visible in the scene) that serves as the ground truth for the correction module during the live interaction (Section 5.2.3). Second, a structured narrative potential annotation that identifies, for each of the 15 MISL elements, whether the scene can support it, together with concrete examples. For instance, a park scene might list Setting → ["pretty park", "sunny day"], Internal Response → ["the boy looks lonely", "the seller seems cheerful"], and Elaborated Noun Phrases → ["the round glasses", "a shy little boy", "huge colorful balloons"]. These annotations constrain the enrichment path (Section 5.2.3) to only

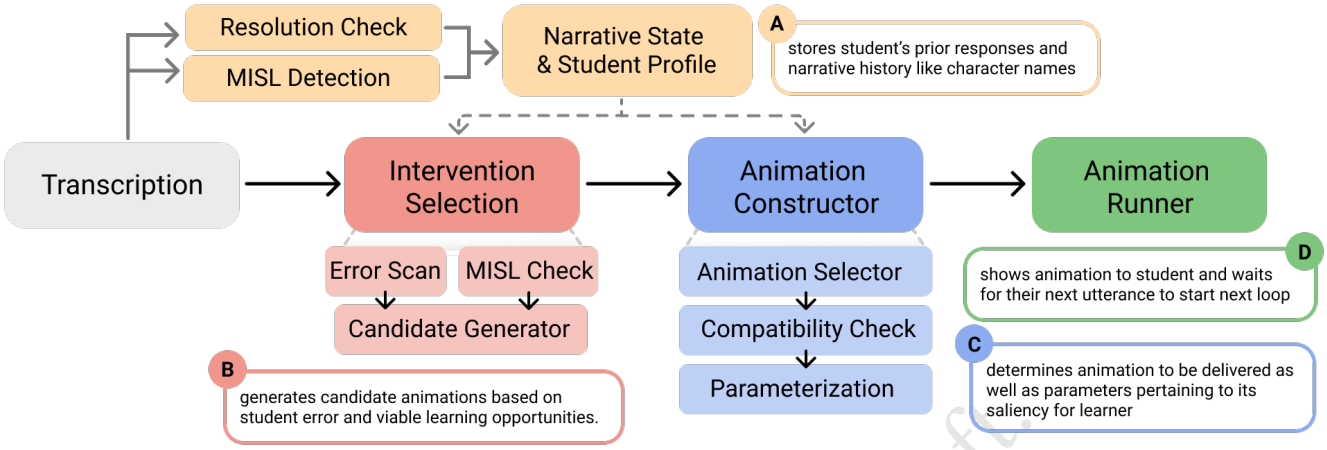


**Figure 3: Overview of the live interaction loop. Each cycle begins when the child finishes speaking. The narrative state and student profile (A) are updated with the child's resolution status and MISL elements. The intervention selection stage (B) pursues error scanning and MISL checking concurrently, generates candidate discrepancies, and selects at most one intervention per cycle (corrections take priority over suggestions). The animation constructor (C) validates the selected animation against the scene's entity registry and modulates its parameters based on resolution history. The animation runner (D) executes the animation and waits for the child's next utterance.**

scaffold elements that the scene actually affords. A final human verification step checks the accuracy of the generated descriptions and MISL annotations, correcting any hallucinated or missing details.

## 5.2 Live Interaction Loop

Each interaction cycle is triggered when the child finishes speaking (Figure 3). The child uses a push-to-talk interface; audio is streamed to the server over a WebSocket connection and transcribed by a speech-to-text model (Google Cloud Speech-to-Text, Chirp 3). The transcribed utterance then enters a pipeline with four stages: (1) parallel utterance analysis, (2) dual-path discrepancy assessment, (3) animation selection, and (4) client-side execution. A full cycle completes in a median of 5.2 seconds (mean 6.5s, 95th percentile 15.5s). The two parallel Gemini calls (resolution check and MISL detection) account for approximately 3 to 4.5 seconds; the deterministic selection and validation steps are negligible (under 300ms); and animation construction and dispatch take approximately 500ms.

*5.2.1 Student Profile.* The system maintains a persistent student profile throughout the storytelling session that accumulates interaction history across scenes. The profile stores three structures that are consulted at different stages of the pipeline. A mention counter tracks, for each of the 15 MISL elements, how many times the child has addressed it over the course of the story; this counter accumulates across scenes and drives the enrichment selection strategy described below (Section 5.2.3). A character name registry records the names the child assigns to entities (e.g., mapping boy → "Pete"), which persist across scenes and are injected into subsequent prompts so that the system can refer to characters by the child's chosen names. Finally, a resolution history logs, for each discrepancy the system has raised, whether the child subsequently addressed it. This history informs both the enrichment selector (which avoids re-suggesting resolved elements) and the parameter escalation mechanism (Section 5.2.4).

*5.2.2 Parallel Utterance Analysis.* Upon receiving a transcription, the system launches two lightweight analyses in parallel using Gemini 3 Flash, the same model used throughout the live pipeline.

*Resolution check.* If the previous cycle produced an animation (whether a correction or a suggestion), the system evaluates whether the child's current utterance addresses that prior feedback. The model receives the child's utterance, the rationale of the previous discrepancy, and the scene description, and returns one of three outcomes: resolved (the child made a reasonable attempt), unresolved (the child did not address the feedback), or not applicable (no prior feedback exists). We designed this check to be intentionally generous: any reasonable attempt counts as resolved, so as to avoid discouraging the child.

*MISL element detection.* Simultaneously, a second call identifies which MISL elements the child naturally produced in the current utterance (e.g., Character, Action, Elaborated Noun Phrases). The detected elements increment the corresponding entries in the student profile's mention counter. Together with the resolution check, these two analyzes provide the pipeline with an up-to-date picture of the child's progress before deciding how to respond.

*5.2.3 Dual-Path Discrepancy Assessment.* After the parallel analysis completes, the system pursues two assessment paths concurrently: error correction and narrative enrichment. Each path may produce at most one discrepancy, a structured object specifying the type of narrative gap, the target entities, a rationale, and the animation to trigger. This discrepancy object constitutes the system's planned intervention.

*Path A: Error correction.* The correction path detects factual and grammatical errors in the child's utterance relative to the scene. The prompt includes the correction intents of every animation in

the grammar, allowing the model to jointly identify the error and select the appropriate visual response in a single call. The model returns zero or more corrections, each specifying an animation identifier and target entities, whose category is derived from the grammar definition rather than from the model's free-text output. The prompt also includes explicit tolerance rules: the system does not flag creative elaborations, minor grammatical deviations typical of the target age range, or reasonable tense choices. Only clear contradictions with visible scene content produce corrections. When multiple errors are detected, a deterministic selection step retains only the highest-priority one according to a fixed category ordering (Identity > Count > Space > Action > Property > Relation > Time > Discourse).

*Path B: Narrative enrichment.* While the correction path looks for what the child said wrong, the enrichment path looks for what the child has not yet said. The system first determines which MISL element to scaffold next by consulting the scene's narrative annotation (Section 5.1) and the student profile's mention counter, retaining only elements mentioned fewer than three times and prioritizing those whose last occurrence went unresolved. Macrostructure elements are considered before microstructure elements, following a fixed ordering (Character > Setting > Initiating Event > Action > Consequence > Internal Response > Plan). When Character is selected and unnamed entities remain in the scene, the system directly triggers a Nametag (I2) animation without calling the model. For all other cases, the selected element is passed to Gemini 3 Flash along with the suggestion intents of the relevant animations, and the model produces one suggestion: an animation, its target entities, and a rationale.

*5.2.4 Compatibility Check and Parameterization.* At this point, the pipeline may hold up to two discrepancies: one error from Path A and one suggestion from Path B. The animation selector retains at most one per cycle, and corrections take strict priority over suggestions: a factual error always overrides an enrichment opportunity.

Before the selected animation can be dispatched, a compatibility check performs two validation steps. First, it checks that the number of target entities matches the animation's declared target type (`entity`, `duo`, `group`, or `scene`); if a mismatch is detected, the system searches for an alternative animation within the same category. Second, entity identifiers produced by the model are matched against the scene's entity registry through substring matching, and identifiers that cannot be mapped are discarded.

Animation parameters (such as dim strength for Spotlight, saturation boost for Color Pop, or amplitude for Motion Lines) are loaded from the grammar definition with default values. The system then consults the student profile's resolution history to modulate these parameters: if the same type of discrepancy was raised in a previous cycle and went unresolved, parameters are shifted 40% toward their maximum value, with a ±10% random perturbation to avoid mechanical repetition. This escalation strategy embodies the implicit scaffolding constraint introduced in Section 4.1: the system first attempts a gentle visual cue, and progressively amplifies it if the child does not respond.

*5.2.5 Client-Side Animation Execution.* The selected animation is transmitted to the client as a WebSocket message specifying the animation template, its parameters, target entities, and duration. The client's animation runner operates on a high-definition buffer containing the composited scene. Entity bounds are precomputed from the extracted masks during scene preparation, enabling the runner to apply transformations to individual entities without affecting the rest of the scene. For animations targeting multiple entities (`duo` or `group`), the runner applies the transformation to each entity independently, preserving correct layering. At every frame, the runner snapshots and restores the buffer state, ensuring that animations are non-destructive: the scene returns to its original appearance once playback ends. Animations that produce temporary overlays (Nametag, Emanation, Speech Bubble, Thought Bubble, Alert, Interjection, Ghost Outline) generate sprite elements procedurally, using the stylistic contrast between overlay and illustrated scene described in Section 4.3 to signal that the overlay is system feedback. The animation loops continuously until the child presses the push-to-talk button to speak again, at which point the next interaction cycle begins.

## 6 Evaluation

We evaluate Tellimation in three stages that move from system to human. We first assess whether the pipeline selects appropriate animations for a given discrepancy (Section 6.1). We then assess whether observers can interpret those animations without prior instruction (Section 6.2). We finally observe how children use Tellimation in a real storytelling session (Section 6.3). The first two evaluations were conducted in a single online study with adult participants on Prolific. In that study, the interpretability task (Block 1) always preceded the pipeline validation task (Block 2) to prevent Block 2 from priming participants with explicit descriptions of the pipeline's reasoning. We present them here in the opposite order because the logical progression runs from the system's behavior to the user's perception to the child's experience.

### 6.1 Pipeline Validation

To construct stimuli for this evaluation, we generated 200 synthetic child utterances using Claude Opus 4.6, each paired with a scene description and a narrative context. The utterances were designed to span all eight categories of the design space and to include both factual errors and enrichment opportunities. The first two authors reviewed every utterance and its scene context to confirm that each one presented an unambiguous ground-truth discrepancy. We then ran each utterance through the live pipeline and recorded the animation it selected together with the rationale it produced.

All 200 pipeline decisions were used as stimuli in Block 2 of the Prolific study. Each stimulus showed participants the scene, the child's utterance, the animation the system chose, and the rationale it produced. Participants rated on a 5-point Likert scale whether the system's response was appropriate for the situation. Each participant saw 10 stimuli (stratified across categories and balancing corrections and suggestions); each stimulus was rated by approximately 4 participants. 80 participants completed Block 2, yielding 800 ratings.

*Results.* The overall mean rating was 3.79 out of 5 (median: 4), and two thirds of all ratings fell at 4 or above, indicating that participants generally found the pipeline's choices appropriate. At the animation level, Sequential Glow (C1, 4.45), Alert (D3, 4.42),

Stamp (S2, 4.36), and Emanation-Hearts (P2e, 4.22) received the highest ratings. Ghost Outline (C3, 2.12) and Disintegration (C2, 2.84) received the lowest, consistent with the interpretability results reported below. Suggestions (mean 3.95) were rated higher than corrections (mean 3.62), suggesting that the pipeline's enrichment decisions were perceived as more natural than its error-correction decisions. This pattern reverses in the interpretability study (Section 6.2), where corrections are easier to interpret than suggestions: the pipeline appears to make better choices for enrichment, but corrections are easier for the viewer to act on once they understand them.

*False positive analysis.* The Likert ratings above evaluate the quality of the pipeline's decisions when it does intervene, but they do not capture cases where the pipeline incorrectly flags a valid utterance as an error. To assess this, we randomly sampled two utterances per animation from the 200 stimuli (50 utterances total), manually corrected each one so that it accurately described the scene, and ran these corrected utterances through the pipeline. Of the 50 corrected utterances, 6 produced a spurious correction. Two were caused by model hallucinations (the model flagged an error that did not exist in the utterance), and four by a misunderstanding of the narrative context (the utterance was consistent with the scene, but the model interpreted it as contradicting the scene description). We did not observe any false negatives during the review (errors present in the utterance that the pipeline failed to detect).

## 6.2 Animation Interpretability

The pipeline validation confirms that the system selects the right animation, but the animation must also communicate its intended meaning to the viewer. If adults cannot interpret an animation without instruction, children will not be able to either. We created 100 stimuli: 25 animations (the 20 grammar entries, with Emanation tested separately for each of its 6 particle variants) paired with 4 scene variants each. Each stimulus presented a picture of a scene, a narration prompt describing what a child has just said, followed by a video of the system's animated response. The first two authors reviewed all stimuli to confirm that the target entity and intended narrative gap were unambiguous.

We recruited 80 participants through Prolific (age $\geq$ 18, fluent in English, normal or corrected-to-normal vision), not informed of the study's connection to children's storytelling. The 100 stimuli were organized into 8 counterbalanced lists of 25, each assigned to 10 participants. Each trial presented 5 options: the target interpretation, a within-category distracter, a between-category distracter, a vague option, and a non-comprehension option.

*Results.* Across all trials, overall accuracy was 49.5%, well above the 20% chance level. We tested whether each animation exceeded chance using a one-sided binomial test with Bonferroni correction ($\alpha = .002$); 22 of the 25 animations significantly exceeded chance. Accuracy was highest for Emanation-Tears (P2f, 75.6%) and lowest for Nametag (I2, 10.8%), which performed below chance: participants interpreted the floating label as flagging a mistake rather than inviting naming. Ghost Outline (C3, 24.4%) and Reveal (S1, 32.5%) also failed to reach significance. At the category level, Property (56.1%) and Action (57.3%) were the best-interpreted categories, while Identity (33.6%) and Time (38.7%) were the weakest. Corrections (53.1%) were interpreted more accurately than suggestions (45.8%), consistent with corrections highlighting a visible contradiction while suggestions draw attention to an absence.

## 6.3 Storytelling with Children

We conducted a within-subjects study with 8 children (two aged 7, three aged 8, two aged 9, and one aged 11; mean = 8.4, SD = 1.3), recruited via a flyer circulated by email. All used English at school; six had been formally diagnosed with a speech or language difficulty, and the two remaining children were identified by their parents as presenting language difficulties. We deliberately used the broad term "language difficulties" in our recruitment materials rather than a clinical label, so as to include children whose difficulties have not been formally assessed. Each child told 4 stories (2 with animations, 2 in a control condition where the pipeline ran identically but suppressed all visual output). Stories were assigned via a Latin square and sessions were conducted over Zoom, lasting 42 minutes on average (SD = 6.7). Each session began with an onboarding phase in which the child practiced with a dedicated scene to familiarize themselves with the push-to-talk interface and the animations. For both conditions, the pipeline logged every discrepancy it identified and whether the child's subsequent utterance addressed it (a resolution); in the control condition, resolutions reflect spontaneous self-corrections.

*Results.* In the animation condition, the pipeline ran 30.2 cycles per story, of which 22.9 produced a displayed animation and 3.1 were filtered out by the compatibility check (Section 5.2.4). The remaining 4.2 cycles produced no animation, either because the pipeline detected no discrepancy or because no suggestion was relevant to the current scene. In the control condition, the pipeline ran 22.4 cycles per story but all visual output was suppressed. The number of distinct MISL elements per story was higher in the animation condition (12.1/15) than in control (7.4/15). The strongest difference appeared in resolutions: the pipeline identified 22.9 gaps per story in the animation condition, of which 13.4 were addressed by the child; in the control condition, only 2.5 of 14.4 identified gaps were addressed spontaneously.

The interaction logs reveal how scaffolding unfolds over successive cycles. In one episode, a child (P1, age 7) said "The boy came back to the man with the balloon." The system played Interjection (D4) to flag the tense error; the child corrected to "A boy is coming back." The system then played Motion Lines (A1) to highlight that the boy is running, not simply coming back; the child responded "The boy is running to the man with the balloon." Over two cycles and two animation types, the description moved from an inaccurate past-tense summary to a correct present-tense action. In a second episode, a child (P7, age 8) said "The grandma is giving the list to the boy" while the scene showed the mother. The system played Spotlight (I1); the child self-corrected to "the mom," described the setting, and noticed the monkey for the first time. A single correction triggered a cascade of observations well beyond the targeted element. Across all participants, corrections were resolved more readily than suggestions, consistent with the interpretability results.

# 7 Discussion

## 7.1 What Makes an Animation Readable

The interpretability results reveal a gradient across the grammar. Animations that transform entities already visible in the scene (Color Pop, Motion Lines, Stamp, Flashback) were consistently well-interpreted: they redistribute perceptual salience without introducing new symbols. Animations that add iconic overlays grounded in familiar conventions (Emanation with tears or steam, Speech Bubble, Alert) also performed well, because the symbols they introduce are ones that children already know from cartoons and comics. The animations that failed (Nametag at 10.8%, Ghost Outline at 24.4%, Reveal at 32.5%) share a common property: they introduce a visual element whose meaning is not self-evident from the scene context. Nametag displays an empty label, which participants interpreted as flagging an error rather than inviting naming. Ghost Outline displays a translucent silhouette from another scene, an unfamiliar convention. Reveal makes an entity translucent, which participants confused with disappearing. This gradient (transformation > iconic overlay > arbitrary overlay) is a design principle for any system that uses animation as a communication channel. But the children's study suggests the picture is more nuanced. Adults decoded each animation as a message, selecting among five interpretations in a forced-choice task. Children likely did this too, but the cascades we observed (a single Spotlight triggering a self-correction, a description of the setting, and the discovery of a new entity in the same turn) suggest that semantic interpretation and attentional reorientation coexist: the child decodes the animation's intent but also notices adjacent elements that the animation was not targeting. These single-cycle spillovers, where one animation triggers observations beyond its target, are distinct from the cross-cycle sequences discussed below, where successive animations of different types address the same unresolved gap. Our study does not allow us to dissociate these two mechanisms, but the distinction matters for evaluation, as we discuss below. In practice, the child both decodes the animation's intent and attends to adjacent elements that the animation was not targeting, which means the design challenge is not only selecting the right semantic message but also staging it so that it opens the child's gaze to the surrounding scene. This distinction between interpretability (can the viewer decode the intended meaning?) and attentional affordance (does the animation redirect attention effectively?) has implications for how we evaluate visual feedback systems.

## 7.2 Reformulate Rather Than Repeat

The escalation mechanism described in Section 5.2.4 amplifies animation parameters when a discrepancy goes unresolved: a brighter spotlight, a wider pulse, more saturated colors. But the interaction logs suggest that the most productive escalation sequences were not those where the system replayed the same animation louder, but those where it switched to a different animation type altogether. In one episode, an Interjection (D4) targeting a tense error was followed by Motion Lines (A1) targeting the underlying action discrepancy; the child resolved both across two cycles, producing a richer utterance than either animation alone would have elicited. In another, a Spotlight (I1) that went unaddressed was followed by an Interjection (D4) targeting a pronoun error in the child's response, which the child resolved immediately together with the original identity error. These inter-animation escalation sequences were more effective than intra-animation ones, where the same animation simply played with stronger parameters.

This pattern is enabled by a structural property of the grammar: the many-to-many mapping between animations and MISL elements means that multiple animations can address the same underlying narrative gap from different perceptual angles. When a Nametag (Identity) fails to make the child name an entity, a Speech Bubble (Discourse) targeting the same entity may succeed, not because it is louder but because it reframes the task from naming to verbalization. This is analogous to a well-known principle in language scaffolding: when a child does not respond to a prompt, effective practitioners recast rather than repeat, offering the same target in a different linguistic frame. Our current implementation does not exploit this deliberately; cross-category switches arise from the pipeline's natural cycle rather than from an explicit reformulation strategy. These episodes point toward a hypothesis worth testing: when feedback fails, changing the framing may be more effective than increasing the intensity. A future version could implement explicit cross-category escalation by consulting the grammar's many-to-many mapping to select an alternative animation that addresses the same MISL element through a different perceptual channel.

## 7.3 Limitations

Our evaluation has several limitations. The children's study involved 8 participants observed in a single session each over Zoom, which limits both the generalizability of the findings and our ability to observe learning effects over time. A longitudinal deployment in a classroom setting would provide stronger evidence for the system's impact on narrative development. The scene preparation pipeline requires offline generation and human verification at each stage, which limits scalability to new stories and domains; automating entity extraction and narrative annotation is a necessary step before deployment at scale. The grammar's current scope is also constrained by the use of indivisible entity layers, which prevents animations that manipulate internal geometry such as facial expressions, posture, or gaze. As generation models evolve toward articulated entities, the design space would expand considerably.

# 8 Conclusion

We presented Tellimation, a system that animates scene elements in response to children's speech to scaffold their storytelling. A grammar of 20 animations organized into 8 semantic categories provides visual feedback on both factual errors and narrative gaps, driven by a hybrid pipeline that combines model-based detection with deterministic selection and structural validation. An interpretability study (N=80) confirmed that 22 of the 25 animations are correctly interpreted above chance, and a within-subjects study with 8 children showed that children addressed 13.4 of 22.9 pipeline-identified gaps per story in the animation condition, compared to 2.5 of 14.4 in control. The grammar is designed as an extensible vocabulary, and we believe the approach generalizes to any task where a system must provide implicit visual feedback on verbal production.

## References


[1] Ronald Baecker and Ian Small. 1990. Animation at the Interface. In *The Art of Human-Computer Interface Design*, Brenda Laurel (Ed.). Addison-Wesley.

[2] Ruth A. Berman. 2009. Trends in Research on Narrative Development. In *Narrative Development in Adolescence*. Springer, 1–16.

[3] Nicola Botting. 2002. Narrative as a Tool for the Assessment of Linguistic and Pragmatic Impairments. *Child Language Teaching and Therapy* 18, 1 (2002), 1–21.

[4] Hugh W. Catts, Marc E. Fey, J. Bruce Tomblin, and Xuyang Zhang. 2002. A Longitudinal Investigation of Reading Outcomes in Children With Language Impairments. *Journal of Speech, Language, and Hearing Research* 45, 6 (2002), 1142–1157.

[5] Bay-Wei Chang and David Ungar. 1993. Animation: From Cartoons to the User Interface. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*. 45–55. doi:10.1145/168642.168647

[6] Jiaju Chen, Minglong Tang, Yuxuan Lu, Bingsheng Yao, Elissa Fan, Xiaojuan Ma, Ying Xu, Dakuo Wang, Yuling Sun, and Liang He. 2025. Characterizing LLM-Empowered Personalized Story Reading and Interaction for Children: Insights From Multi-Stakeholder Perspectives. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1002, 24 pages. doi:10.1145/3706598.3713275

[7] Fanny Chevalier, Pierre Dragicevic, and Steven Franconeri. 2014. The Not-so-Staggering Effect of Staggered Animated Transitions on Visual Tracking. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 2241–2250.

[8] John Joon Young Chung, Melissa Roemmele, and Max Kreminski. 2025. Toyteller: AI-powered Visual Storytelling Through Toy-Playing with Character Symbols. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 331, 23 pages. doi:10.1145/3706598.3713435

[9] Heidi C. Dulay, Marina K. Burt, and Stephen D. Krashen. 1982. *Language Two*. Oxford University Press.

[10] Min Fan, Xinyue Cui, Wanqing Ma, Haiyan Li, Xin Tong, Lin Yang, and Yonghui Wang. 2025. From Words to Wonder: Designing and Evaluating an AI-Empowered Creative Storytelling System for Elementary Children. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. doi:10.1145/3706598.3713478

[11] Ronald B. Gillam and Nils A. Pearson. 2004. *Test of Narrative Language*. Pro-Ed, Austin, TX.

[12] Sandra Laing Gillam and Ronald B. Gillam. 2016. Narrative Discourse Intervention for School-Aged Children With Language Impairment: Supporting Knowledge in Language and Literacy. *Topics in Language Disorders* 36, 1 (2016), 20–40.

[13] Sandra Laing Gillam, Ronald B. Gillam, Jamison D. Fargo, Abbie Olszewski, and Hailey Segura. 2017. Monitoring Indicators of Scholarly Language: A Progress-Monitoring Instrument for Measuring Narrative Discourse Skills. *Communication Disorders Quarterly* 38, 2 (2017), 96–106.

[14] Arthur C. Graesser, Keith K. Millis, and Rolf A. Zwaan. 1997. Discourse Comprehension. *Annual Review of Psychology* 48 (1997), 163–189.

[15] Karen S. Greenhalgh and Catherine J. Strong. 2001. Literate Language Features in Spoken Narratives of Children With Typical Language and Children With Language Impairments. *Language, Speech, and Hearing Services in Schools* 32, 2 (2001), 114–125.

[16] Todd M. Griffin, Lowry Hemphill, Lynne Camp, and Dennis Palmer Wolf. 2004. Oral Discourse in the Preschool Years and Later Literacy Skills. *First Language* 24, 2 (2004), 123–147.

[17] Chris Harrison, Gary Hsieh, Karl D.D. Willis, Jodi Forlizzi, and Scott E. Hudson. 2011. Kineticons: using iconographic motion in graphical user interface design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1999–2008. doi:10.1145/1978942.1979232

[18] Nancy L. Hedberg and Carol E. Westby. 1993. *Analyzing Storytelling Skills: Theory to Practice*. Communication Skill Builders, Tucson, AZ.

[19] Jeffrey Heer and George Robertson. 2007. Animated Transitions in Statistical Data Graphics. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (2007), 1240–1247.

[20] Scott E. Hudson and John T. Stasko. 1993. Animation Support in a User Interface Toolkit: Flexible, Robust, and Reusable Abstractions. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*. 57–67.

[21] Jaewoong Im and Byungdoo Chang. 2026. Human–AI Co-creativity in Storytelling: A Scoping Review of Literature, Education, Media, and Interactive Systems. In *Interactive Storytelling (ICIDS 2025)*. Lecture Notes in Computer Science, Vol. 16375. Springer. doi:10.1007/978-3-032-12405-0_24

[22] Megan Israelsen-Augenstein, Carly Fox, Sandra L. Gillam, Sarai Holbrook, and Ronald Gillam. 2022. Monitoring Indicators of Scholarly Language: A Progress Monitoring Tool for Documenting Changes in Narrative Complexity Over Time. *Frontiers in Education* 7 (2022). doi:10.3389/feduc.2022.918127

[23] Walter Kintsch. 1998. *Comprehension: A Paradigm for Cognition*. Cambridge University Press.

[24] Jacqueline M. Kory Westlund, Sooyeon Jeong, Hae Won Park, Samuel Ronfard, Aradhana Adhikari, Paul L. Harris, David DeSteno, and Cynthia L. Breazeal. 2017. Flat vs. Expressive Storytelling: Young Children's Learning and Retention of a Social Robot's Narrative. *Frontiers in Human Neuroscience* 11 (2017). doi:10.3389/fnhum.2017.00295

[25] John Lasseter. 1987. Principles of Traditional Animation Applied to 3D Computer Animation. *ACM SIGGRAPH Computer Graphics* 21, 4 (1987), 35–44.

[26] Jungeun Lee, Suwon Yoon, Kyoosik Lee, Eunae Jeong, Jae-Eun Cho, Wonjeong Park, Dongsun Yim, and Inseok Hwang. 2024. Open Sesame? Open Salami! Personalizing Vocabulary Assessment-Intervention for Children via Pervasive Profiling and Bespoke Storybook Generation. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 120, 32 pages. doi:10.1145/3613904.3642580

[27] Johnny C. Lee, Jodi Forlizzi, and Scott E. Hudson. 2002. The Kinetic Typography Engine: An Extensible System for Animating Expressive Text. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*. 81–90. doi:10.1145/571985.571997

[28] Laurence B. Leonard. 2014. *Children with Specific Language Impairment* (2nd ed.). MIT Press.

[29] Betty Z. Liles, Robert J. Duffy, Donna D. Merritt, and Sherry L. Purcell. 1995. Measurement of Narrative Discourse Ability in Children With Language Disorders. *Journal of Speech and Hearing Research* 38, 2 (1995), 415–425.

[30] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual Instruction Tuning. *Advances in Neural Information Processing Systems* 36 (2024).

[31] Allyssa McCabe and Carole Peterson. 1991. Getting the Story: A Longitudinal Study of Parental Styles in Eliciting Narratives and Developing Narrative Skill. *Discourse Processes* 14 (1991), 339–381.

[32] Scott McCloud. 1993. *Understanding Comics: The Invisible Art*. William Morrow.

[33] Donna D. Merritt and Betty Z. Liles. 1989. Narrative Analysis: Clinical Applications of Story Generation and Story Retelling. *Journal of Speech and Hearing Disorders* 54 (1989), 438–447.

[34] Louisa C. Moats. 2020. *Speech to Print: Language Essentials for Teachers* (3rd ed.). Paul H. Brookes Publishing.

[35] Deval Panchal, Christopher Collins, and Mariana Shimabukuro. 2024. LingoComics: Co-Authoring Comic Style AI-Empowered Stories for Language Learning Immersion with Story Designer. In *Adjunct Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) *(UIST Adjunct '24)*. Association for Computing Machinery, New York, NY, USA, Article 100, 3 pages. doi:10.1145/3672539.3686352

[36] Rhea Paul and Courtenay Norbury. 2012. *Language Disorders from Infancy through Adolescence: Listening, Speaking, Reading, Writing, and Communicating* (4th ed.). Elsevier.

[37] Carole Peterson and Allyssa McCabe. 1983. *Developmental Psycholinguistics: Three Ways of Looking at a Child's Narrative*. Plenum Press, New York.

[38] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125* (2022).

[39] Catherine E. Renfrew. 1997. *Bus Story Test: A Test of Narrative Speech* (4th ed.). Speechmark Publishing, Milton Keynes, UK.

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684–10695.

[41] Elisa Rubegni and Monica Landoni. 2014. Fiabot! Design and Evaluation of a Mobile Storytelling Application for Schools. In *Proceedings of the 2014 Conference on Interaction Design and Children*. doi:10.1145/2593968.2593979

[42] Trina D. Spencer and Douglas B. Petersen. 2015. *Story Champs: A Multi-Tiered Language Intervention Program*. Language Dynamics Group. https://languagedynamicsgroup.com/story-champs-2/

[43] Nancy L. Stein and Christine G. Glenn. 1979. An Analysis of Story Comprehension in Elementary School Children. *New Directions in Discourse Processing* 2 (1979), 53–120.

[44] Frank Thomas and Ollie Johnston. 1981. *The Illusion of Life: Disney Animation*. Disney Editions.

[45] Mort Walker. 2000. *The Lexicon of Comicana*. iUniverse. Originally published 1980.

[46] Lyumanshan Ye, Jiandong Jiang, Yuhan Liu, Yihan Ran, and Danni Chang. 2025. Colin: A Multimodal Human-AI Co-Creation Storytelling System to Support children's Multi-Level Narrative Skills. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, Article 139, 11 pages. doi:10.1145/3706599.3719837

[47] Zheng Zhang, Ying Xu, Yanhao Wang, Bingsheng Yao, Daniel Ritchie, Tongshuang Wu, Mo Yu, Dakuo Wang, and Toby Jia-Jun Li. 2022. StoryBuddy: A Human-AI Collaborative Chatbot for Parent-Child Interactive Storytelling with Flexible Parental Involvement. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. doi:10.1145/3491102.3517479

# A Appendix

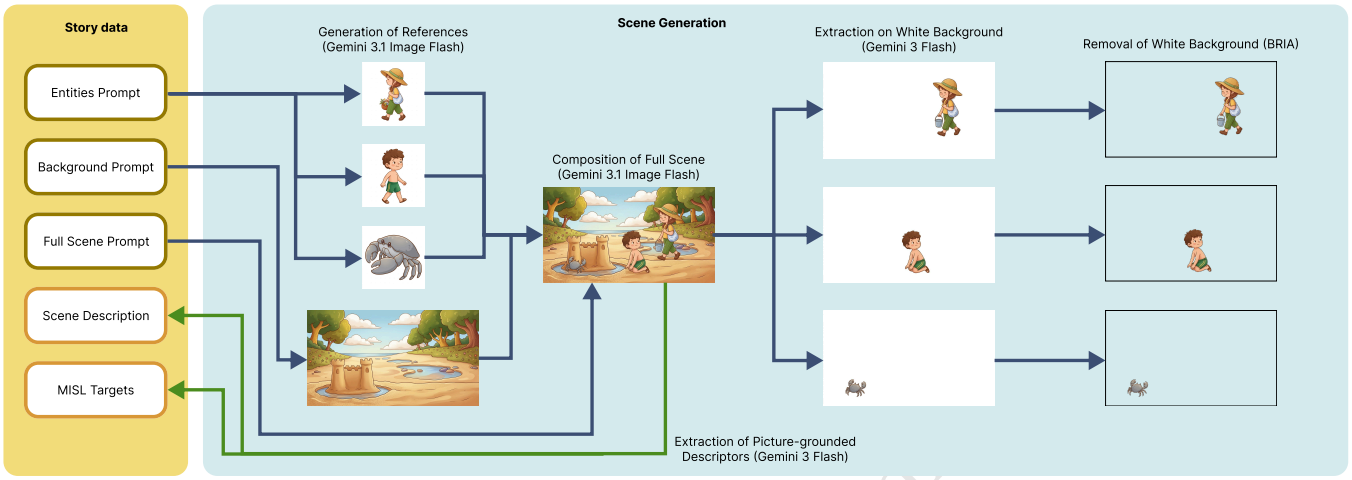


**Figure 4: Scene preparation pipeline. Story data (left) provides prompts for entities, background, and the full scene. Entity references are generated individually, then used to compose a coherent full scene. The complete scene is processed in two directions: forward through entity extraction (Gemini 3 Flash) and background removal (BRIA) to produce independent transparent layers; and backward into the story data, where Gemini 3 Flash analyzes it to produce the scene description and MISL targets that will ground the live interaction loop.**