DataParticles: Block-based and Language-oriented Authoring of Animated Unit Visualizations

Yining Cao
University of California, San Diego
La Jolla, California, USA
yic069@ucsd.edu

Jane L. E
University of California, San Diego
La Jolla, California, USA
je@ucsd.edu

Zhutian Chen
Harvard University
Cambridge, Massachusetts, USA
ztchen@seas.harvard

Haijun Xia
University of California, San Diego
La Jolla, California, USA
haijunxia@ucsd.edu

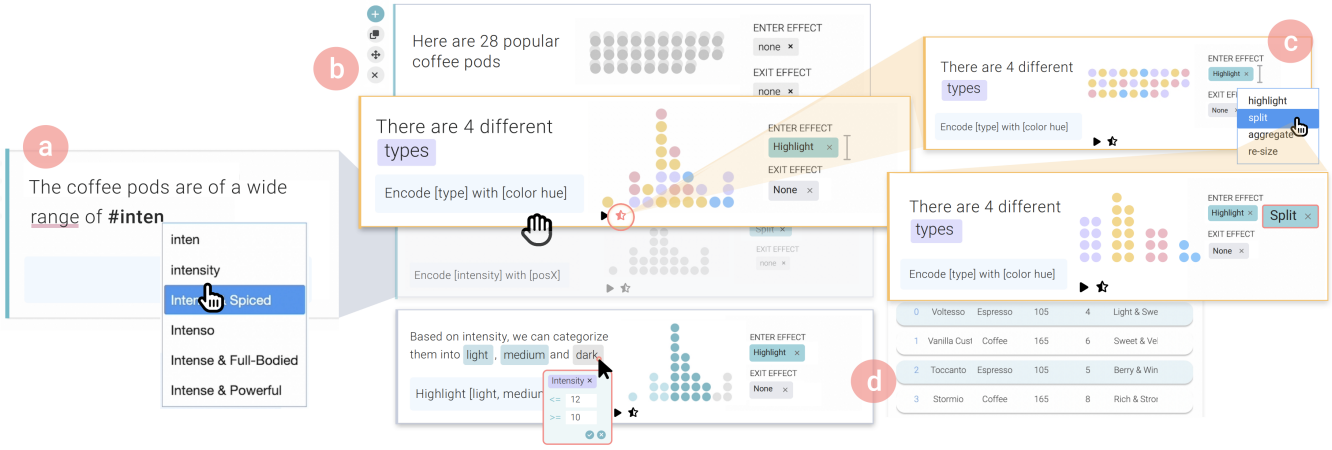

Figure 1: DataParticles is a natural language-oriented, block-based authoring tool for animated unit visualizations. Creators can author animated unit visualizations that are congruent with their story narratives by writing plain text in a (a) rich text editor. It supports flexible prototyping and quick iteration with (b) multiple block operations. Creators are able to easily (c) configure different animation effects and (d) maintain visual bindings across different views to assist their creation process.

ABSTRACT

Unit visualizations have been widely used in data storytelling within interactive articles and videos. However, authoring data stories that contain animated unit visualizations is challenging due to the tedious, time-consuming process of switching back and forth between writing a narrative and configuring the accompanying visualizations and animations. To streamline this process, we present DataParticles, a block-based story editor that leverages the latent connections between text, data, and visualizations to help creators flexibly prototype, explore, and iterate on a story narrative and its corresponding visualizations. To inform the design of DataParticles, we interviewed 6 domain experts and studied a dataset of 44 existing animated unit visualizations to identify the narrative patterns and congruence principles they employed. A user study with 9 experts showed that DataParticles can significantly simplify the process of authoring data stories with animated unit visualizations by encouraging exploration and supporting fast prototyping.

CCS CONCEPTS

• **Human-centered computing** → **Visualization toolkits**; **User interface design**; **Natural language interfaces**.

KEYWORDS

unit visualization, natural language, animation, storytelling

ACM Reference Format:
Yining Cao, Jane L. E, Zhutian Chen, and Haijun Xia. 2023. DataParticles: Block-based and Language-oriented Authoring of Animated Unit Visualizations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3544548.3581472

CHI '23, April 23–28, 2023, Hamburg, Germany

ACM ISBN 978-1-4503-9421-5/23/04.
https://doi.org/10.1145/3544548.3581472

1 INTRODUCTION

Unit visualizations [31] have become an increasingly popular form of data storytelling in both interactive articles and videos [15, 27].

The appeal of unit visualizations, in contrast to aggregate visualizations (e.g., bar charts), is that they harness one-to-one mappings between visual marks and data points to afford diverse, nuanced animations. These animations are especially suitable for communicating data insights as they deliver information in small chunks, smoothly leading viewers to a final takeaway message [14, 19, 54].

However, data-driven articles and videos are notoriously difficult to create due to the inherent complexity of combining a large number of heterogeneous elements (e.g., text, visualizations, animations, interactions, etc.) into a single piece of content [18]. To ensure that effective storytelling can occur, content must be semantically and temporally congruent [53], which leads creators to switch back and forth between a variety of tools that are dedicated to authoring different types of content and then manually resolve any discrepancies that arise when the content is composed into a single article or video. This process results in highly fragmented, repetitive, and tedious workflows [9, 26, 45]. These challenges can be further exacerbated by the complexity of animated unit visualizations (AUVs), which contain a large quantity of individual data points, as well as the diverse and complex animations that creators wish to author.

The goal of this research is thus to systematically explore the challenges that exist while creating data stories that employ complex and compelling AUVs, and then identify opportunities to streamline the creation process. To this end, we conducted a 2-part formative study that consisted of an interview-based study with 6 AUV creators to understand the challenges in their workflows, and a content analysis of 44 data-driven articles and videos to investigate the characteristics and patterns contained within such media.

The formative study found that authoring data stories with AUVs is extremely challenging because it is difficult to plan and prototype. Typical prototyping techniques (e.g., sketching) cannot accurately capture the number of visual units nor represent the dynamic animations that creators envision. As a result, creators resort to using coding languages to programmatically specify the parameters and behaviors of hundreds and thousands of data points. This process, however, is indirect, tedious, and time-consuming. The content analysis found that data stories use a section-based narrative structure, and within each section, the text in the story closely corresponds to the underlying data selections, operations, visual encodings, and animations needed for each AUV.

These formative studies highlighted the opportunity to leverage the desired connections and organization between text and visuals to supporting planning, exploring, and composing of data stories with rich AUVs. We propose DataParticles, a system that facilitates the authoring process of AUVs with two components, a *language-oriented animation authoring* technique that enables users to leverage natural language to create congruent visualizations and animations, and a *block-based story editor* that enables creators to flexibly iterate on their story narrative while maintaining congruence with the visuals. To assess the utility of DataParticles, we conducted an expert evaluation with 9 professional data story creators. The results showed that the language-oriented, block-based authoring environment enabled the authoring process to be story-focused and facilitated creation by enabling flexible prototyping.

This research thus contributes:

(1) A **formative study** with 6 expert creators that identifies the common workflows and pain points that exist when creating data-driven stories with AUVs.
(2) A **content analysis** of 44 stories with AUVs that identifies common patterns their narrative structures and the mappings that exist between the text and visuals.
(3) A **prototype system, DataParticles,** that leverages the text and visual mappings to create a language-driven, block-based authoring experience for data stories with AUVs.
(4) An **expert evaluation** that confirmes the utility of DataParticles, revealing insightful design implications for future design prototyping and language-oriented authoring tools.

While this research focuses on supporting creating data stories with AUVs, we envision the proposed user interface and interaction techniques can be applied to other types of data visualizations (e.g., aggregated charts).

2 RELATED WORK

This work draws on prior research on unit visualizations, data storytelling, natural language oriented interactions and block-based interfaces for content creation.

2.1 Unit Visualizations

Unit, or glyph-based, visualizations are visualizations that represent every data point using a unique visual mark, whose visual and spatial properties encode the attributes of the data point [5, 15, 31]. The rich visual representation of each individual data point has resulted in unit visualizations being an increasingly popular medium for communicating data in a range of scenarios, such as infographics [62], data-driven animations [3], AR visualizations [8], and data physicalizations [1]. Prior work has presented a series of tools, frameworks, and design spaces to facilitate the creation of unit visualizations. For example, Drucker and Fernandez characterized the design space of common unit visualizations and proposed a unifying framework [15]. Park et al. further explored the design space of unit visualizations and contributed an expressive grammar that formally specified unit visualizations [31]. This design space was later used during the development of SandDance, which supported the exploration, identification, and communication of insights about data using unit visualizations within a graphical user interface [31]. Besides using simple visual marks such as squares or circles to represent data points, some authoring tools have been developed to support the mapping of data points to glyphs with complex visual structures [5, 8, 62].

While these systems were shown to be effective at supporting the creation of *static* unit visualizations, creating *animated* unit visualizations remains a challenge. State-of-the-art data animation systems, such as data animator [52], created animated data visualizations by keyframing and interpolating between static visualizations. With such tools, significant manual effort is still required to create rich, complex AUVs. Recently, Lu et al. [27] presented a method to automatically generate AUVs by arranging data-facts However, the limited number of rule-based visual graphs and template-based text descriptions limited the expressiveness and controllability of the generated results. In contrast, DataParticles leverages the *context* or textual storytelling that AUVs are often embedded alongside to

reduce the manual effort required when creating AUVs by inferring and recommending animations that were congruent to the story.

2.2 Data-driven Storytelling

Data-driven storytelling studied how to leverage visualizations to effectively communicate data insights. Segel and Heer categorized seven genres of data-driven storytelling, including articles and videos [36]. Due to the storytelling power they offer and the challenging authoring processes required, these forms of narrative visualizations have been gaining traction in HCI and visualization communities over the past few years. Prior work in this domain has contributed to understanding the workflow of current authoring practices. Lee et al., for example, identified that data storytelling processes often involve three major steps: finding insights from data, turning insights into narratives, and communicating this narrative to the audience [26]. Similarly, Sultanum et al. showed that authors usually started with a dataset or a particular question to answer using the data and then organized the main findings into a linear structure to present [45]. During this process, synthesizing information across multiple resources, making decisions about rich visual effects, and contextualizing data facts into coherent narratives were found to be challenging [18, 45]. To address these challenges, significant research has been conducted to summarize design spaces to guide and simplify the authoring process [2, 7, 12, 45–47] and to develop new authoring tools [4, 13, 24, 40, 45, 48].

While all of this prior work has made great strides to leverage visualization techniques for chart-based visualizations in general, little is known about how unit visualizations are used in data-driven articles and videos. Due to the flexibility and expressiveness of unit visualizations, their design and authoring involves specific challenges and opportunities. Our research seeks to further the understanding of current practices used to author AUVs and the usage of AUVs for storytelling, especially how rich animations can support narratives.

2.3 NLI for Data Visualization

Significant research has leveraged natural language interfaces (NLI) to create and manipulate data visualizations [39]. For example, Voder automatically generates data facts in natural language sentences with interactive widgets to facilitate data understanding and exploration [41]. DataTone enabled users to create aggregated visualizations with natural language queries and provided interactive widgets to resolve the ambiguities inherent in natural language expressions [16]. Similarly, Eviza leveraged the pragmatic structures in natural language and enabled users to create and revise data visualizations through continuous and interactive conversations [37]. Orko [44] and InChorus [42] supported multimodal interaction with visualizations by leveraging natural language.

To facilitate the development of NLIs for visual analytics and visualization design, Narechania et al. proposed a toolkit that used a tabular dataset and a natural language query as input and returned structured specifications of data visualizations [29]. Srinivasan et al. provided a benchmark of NLIs for visualization using a curated a dataset of visualization-oriented utterances [43]. ArkLang described an intermediate language that built upon a set of analytical concepts to infer intents in underspecified natural language utterances [38]. This prior research laid a foundation for us to further explore and leverage the links between natural language and data visualizations.

Instead of requiring users to make explicit natural language queries and specifications, recent work has leveraged the latent correspondences that exist between linguistic structures and content structures in more casual natural language expressions [9, 10, 23, 59, 60]. For example, CrossData provided rich interactions to help users author data documents by automatically establishing text-data connections based on users' natural writing in a document [9]. Kori linked visuals and natural language by suggesting references between text inputs and charts in an existing gallery to support the creation of an interactive visual story [23]. DataParticles draws from this line of research by employing desired mappings between visual and natural language throughout one's storytelling process. DataParticles leverages the flexibility of unit visualizations to create animated visual stories that are congruent with the narrations. It supports a range of interaction techniques that facilitate the synchronized creation and iteration of AUVs and the underlying narrative story.

2.4 Block-based Editing for Data-driven Content Creation

Computational notebooks (e.g., Jupyter [20], R Markdown [33], and Observable [30]), are modern embodiments of Knuth's notion of literate programming [22], which has been widely used in data science for data exploration and communication [34]. Many researchers have explored extending such notebooks with features including real-time collaboration [56], the synchronization of code and chat [57], and the synchronization of code and interaction results [58]. Recently, Lau et al. analyzed 60 computational notebooks and proposed a design space to characterize notebooks, revealing that most notebooks adopted a block-based editor style [25]. Compared to traditional document-based editing environments, block-based editors enable users to independently edit, execute, and rearrange blocks within a document, thereby enabling for the progressive authoring and immediate visual feedback.

The modulated planning technique is also commonly seen in the two-column scripts for planning visual stories [11, 61]. With a two-column format, film scriptwriters organize the narration in one column and the visual directions in the other (e.g., camera movement and background setup), where each row of the script specifies all the necessary information for a scene [32]. This technique has been broadly adopted by video creators to allow consistent organization and flexible reorganization of segments of videos.

DataParticles leverage the block-based editing paradigm to scaffold the authoring process for data-driven articles, in which authors frequently iterate, re-organize, and review the content they create. Considering these benefits, we explore the fusion of language-oriented authoring and block-based editing to offer intuitive, flexible interactions while creating AUVs.

3 FORMATIVE STUDY

This research did not begin with a specific focus on animated unit visualizations. Instead, we were initially interested in understanding how data-driven stories that employ complex animations were

created and the challenges that exist throughout creators' workflows. To achieve this, we conducted an interview-based study with experts who had significant experience employing visualizations and animations for storytelling. As animated data stories are often created by a group of creators, we recruited experts across different roles to have a comprehensive understanding of such workflows. Through these interviews, we found that among the various types of data stories the interviewees had created, the ones that consisted of AUVs were noted as being both compelling and challenging to create, as they exacerbated many of the pain points of complex animation authoring processes.

3.1 Interviewees and Procedure

Semi-structured interviews were conducted with six domain experts who had professional experience creating data stories with AUVs. Each of the interviewees had over 5 years of experience creating different types of animated data stories, where their roles acorss producer, animator, visualization engineer, or journalist.

The interviews began with background questions about interviewees' roles and professional histories. Interviewees were then asked about their workflows when creating stories with AUVs, including the tools that they used, and the pain points they encountered throughout the process. We summarized our findings in terms of the workflows and challenges in creating AUVs.

3.2 Workflows While Creating AUVs

The workflows employed while creating animated data stories consisted of three major steps. First, narratives about the data stories were developed to guide production. Second, visualizations and animations were created based on the narrative planning, starting from low-fidelity prototypes and moving to high-fidelity designs. Third, the narrative and visuals were composed to form a complete story. While these three high-level steps were commonly taken during the entire creation by all interviewees, we identified two different approaches during planning and prototyping phases: *data-driven* planning and *story-driven* planning.

In the *data-driven* planning approach, creators started with a general topic and an existing dataset. They then analyzed the data, gathered excerpts and organized the excerpts into a coherent story, which is consistent with the findings from prior research [26]. In this planning, creators often conducted a data-driven inquiry to learn more about the topic and created videos to communicate the insights they had discovered from the data analysis. This approach is often used to communicate complex ideas or data in a more accessible manner, making the information easier to understand and remember. The *story-driven* approach is a narrative-based approach, where creators begin with a written narrative and a pre-determined sequence for presenting their data. They then searched for data facts that aligned with their story and incorporating visualizations to provide supporting evidence. The focus is on creating an engaging narrative that captures the audience's attention and delivers information in a compelling and convincing way.

Despite the different processes while planning, both processes shared similar pain points. Herein, we report on the key challenges that creators encountered for each step and highlight how these challenges would be exacerbated during the creation of AUVs.

3.3 Challenges Encountered by Creators

3.3.1 Difficulties and Downstream Frustrations While Planning and Prototyping AUVs. All interviewees employed a two-column script to plan their data stories, where the first column broke down the text narrative into sections and the second column showed the corresponding visuals to be created. However, interviewees noted that specifying desired AUVs was extremely challenging compared to static graphics and aggregated visualizations because the animations of units were difficult to prototype using sketching or digital prototyping tools such as Keynote. It was thus challenging to speculate on the final look of an AUV in terms of whether it would coherently support their narrative. Due to this lack of concrete specifications during the planning stage, creators often made many design decisions during the creation process without knowing whether they might deviate from their overall storytelling goal, requiring more downstream iterations. On the other hand, the significant amount of effort needed to adjust complex AUVs made interviewees reluctant to make substantial changes at these later stages. As E1 noted, *"I feel like I'm randomly making 10 design decisions every second, the first time you finish this [AUV] is the first time you see the result, and you never want to go back*."

3.3.2 Tedious and Complex Workflows During AUV Creation. Creating high-fidelity AUVs required interviewees to use several tools to specify each different aspects of their design. Interviewees often first used spreadsheet applications (e.g., Excel) and programming toolkits (e.g., R, Python) to clean, prepare, and analyze their data. To animate visualizations, interviewees often resorted to using programming libraries (e.g., d3.js) instead of using direct manipulation tools, which would become very tedious with large datasets and limit the exploration of the visual encodings and animations. However, the nature of programming does not lend itself to rapid exploration, which demanded significant effort from interviewees and also implied a high barrier of entry to creating this type of visualization. As noted by E6, *"we have to have a decent background in both design and code to be able to build them, along with a considerable amount of time."* In addition to using the typical view transitions found with unit visualizations, data stories often employed 'per data point' animations (e.g., highlighting and pulsing) to guide users' attention through the flow of the story. Although these animations were not technically challenging to create, the sheer number of them required tedious manual effort. Especially in video-based stories, interviewees used dedicated animation tools (e.g., Adobe AfterEffects) for final touches or animations that were difficult to create using programming.

3.3.3 Repetitive Synchronization for Story-Visual Congruence. Interviewees stated that matching the story with the visuals was crucial for effective communication and storytelling. Although the story and visuals were initially planned to align, discrepancies in the content (e.g., data insights), format (e.g., movements of the units), and timing (e.g., duration of animation) between the visuals and the story often arose. In the final stage of creation, interviewees underwent multiple refinement iterations focused on eliminating the discrepancies and ensuring congruence. Changes involve tweaking animation timing, modifying unit visualization layout, or altering

the story narrative based on new insights gained during the creation of the AUVs.

Interviewees attributed the discrepancies to the separate development of the visualizations and story as well as the lack of immediate feedback on how the visuals corresponded with the underlying story. This resulted in the need to constantly switch between editing the story and the visuals. The late discovery of these discrepancies often caused more effort to be required in later stages, even for issues that could have been easily resolved earlier.

3.4 Summary

The findings revealed that while one-to-one mapping between data points and visualizations provides AUVs with the narrative power to express certain concepts, it also makes AUVs extremely challenging to plan, explore, and prototype. In addition, the authoring process is distributed across a wide range of tools and environments, resulting in difficulties in maintaining the linkage between the story, data and visuals. The findings suggest that creators of data stories can significantly benefit from flexible prototyping and structured scaffolding for the entire authoring process, especially earlier in the planning stage.

4 CONTENT ANALYSIS OF DATA STORIES CONTAINING AUVS

To better understand the characteristics and design patterns of AUVs within data stories, especially how AUVs correspond with the text, we conducted a content analysis of data stories that employed AUVs. We surveyed data stories that were part of interactive data articles and data videos and curated 44 data stories from well-known platforms known for producing data stories, including Vox [55], the Pudding [51], the New York Times [50], the Guardian [49], etc. We divided the stories based on the section structure into 403 sections, which contain the textual narrative and the corresponding AUV. We focused on two aspects of the data stories: (1) the overall narrative and organizational structures of the stories, and (2) the correspondences between the stories and their AUVs.

4.1 Narrative and Organization Structures

Despite their different forms, data-driven articles and videos were found to employ similar narratives and content organizational structures. We summarized some story patterns as here (Figure 2a).

4.1.1 Inverted Pyramid Narrative Structure. A popular narrative structure surfaced through the analysis was the inverted pyramid narrative structure (35/44 stories). With this structure, stories first introduced the high-level context of the story: the scope, background, and key messages. The stories then gradually narrowed to specific arguments and evidence with animations.

4.1.2 Section-based Organization. Most of the data articles that we analyzed employed a section-based structure to organize the content of the story, where each section contained a few short paragraphs and the corresponding visualizations. Sections of text and animated visualizations were organized spatially in a variety of layouts, such as embedding a visualization in between the text or locating them side-by-side. Navigation of the text or the visuals is synchronized with the other. For example, scrolling on the text can trigger the view transitions of the visualization or vice versa. For data videos, the section structure was less explicit, and was often signified by a transition between different visualizations.

4.1.3 Incremental Transitions and Parallel Structures. To make the content easier to digest, the data insights delivered in the story were often broken down into small, manageable bites and communicated incrementally. As a result, most of the sections in a story only introduced one aspect of the data (253/403 sections). We also observed that parallel structures were commonly used within a story (32/44 stories), resulting in consistent visual content across different sections. For example, when describing basketball players from different teams, the same highlight animation was applied to the relevant groups per section. By breaking down the data into smaller, more manageable pieces and using parallel structures, authors could effectively communicate complex information in a way that was easy to follow and understand.

4.2 Mappings Between the Text and Visuals

In our interview-based study, we found that creators needed to constantly ensure that their visualizations and animations were congruent with the story. To better understand how the text in a section corresponds with the visual transitions of AUVs, we analyzed our collected corpus and identified four categories of words and phrases that could be used to inform the linkage between text and visuals: *data property*, *data value*, *data operation*, and *visual operation* phrases. By examining these phrases, we were able to understand and identify patterns that employed to create more engaging and effective visual stories.

4.2.1 Data Property and Value Phrases. In our study, we found that the data property and value phrases were often mentioned to refer to the data being discussed. These phrases are used differently depending on the data types. When referring to categorical data, only the data value was typically mentioned. For example, in the sentence *"most of the popular drinks have pudding as topping,"* only the word *"pudding"* (the value of the *"topping"* property) is mentioned. When referring to numerical data, both data property and value phrases were often used. For example, in the sentence *"the basketball players ranked between 50 to 100,"* both the data property (i.e., *"rank"*) and values (i.e., *"50"* and *"100")* were mentioned. Phrases such as *"larger than"* and *"over"* were often used to indicate a range of values. In some cases (143/403 sections), a section might not include any data property or value phrases if it simply referred to the data set from the previous section.

4.2.2 Data Operation Phrases. The data operation phrases were used to indicate the analytical functions performed on the data to support the delivery of data insights. Within the 403 sections, we identified four types of analytical functions informed by text: calculating derived value (e.g., *"total," "average'*), finding extrema (e.g., *"largest," "minimum"*), sorting (e.g., *"rank," "from ... to ..."*), and finding an alternative (e.g., *"rest"* and *"remaining"*).

4.2.3 Visual Operation Phrases. The visual operation phrases were used to indicate preferred visual effects in AUVs. They specifically referred to the changes in visual encodings, such as color, size, and

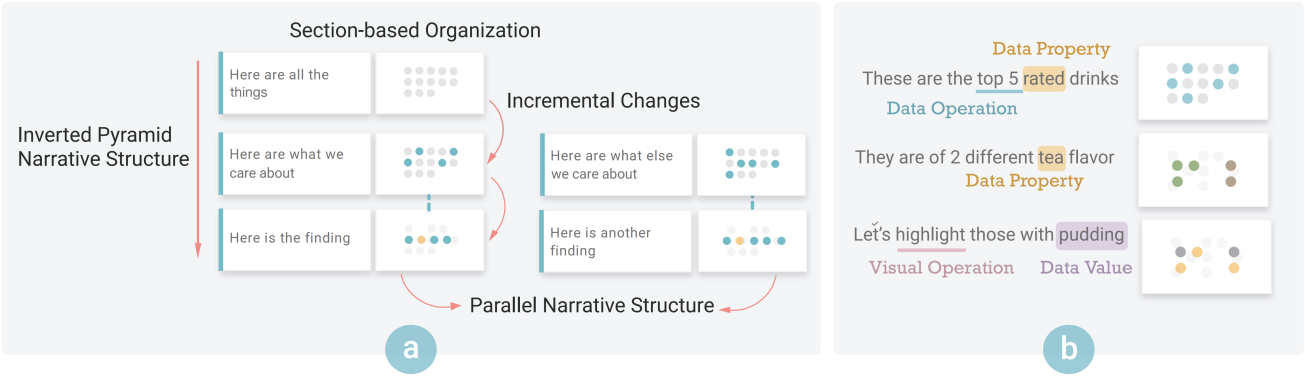

Figure 2: Patterns in existing data stories containing AUVs. (a) The structural patterns including the inverted pyramid structure, section-based organization, incremental changes between sections, and the commonly used parallel structure with consistent visual effects. (b) Examples of 4 key phrases we identified in AUVs that link language to visual effects in AUVs.

position. For example, phrases such as *"highlight"* and *"coloring"* indicated a change in color encoding; *"distribute," "range,"* and *"across"* indicated a change in position encoding; *"width"* and *"height"* indicated a change in size of the visual units. These phrases were used to help create engaging and effective visual stories by highlighting key information and guiding the viewer's attention.

4.3 Summary

Our content analysis showed that data stories with AUVs often used a section-based structure to organize the narration and visualizations. These stories deliver data insights through incremental visual changes and specific narrative structures, such as inverted pyramid and parallel structures. We also observed a strong correspondence between the segments of an informative story and their desired visualizations. In particular, we identified four types of phrases in the narration that could be used to inform the visual states and transitions in these stories. These findings can be useful for creating more effective and engaging visual stories.

5 DESIGN DECISIONS

The formative study and content analysis revealed strong correspondences between the text and visuals in data stories with AUVs. While the correspondences are typically achieved at the planning stage (e.g. using a two-column script) and in the final content (e.g. using a section-based structure), they are hard to maintain during the prototyping and creation stages, resulting in repetitive actions to keep the story and visuals synchronized. This presents an opportunity to leverage the text in a narrative to ensure the desired correspondences throughout the entire design process. We hence propose the following design decisions:

- **D1.** We will use **language-oriented editing**, where the necessary data selection, operations, encodings, and animations will be inferred from the story text to **automatically generate** the desired visualizations and animations, enabling fast prototyping and content creation.
- **D2.** We will use **block-based editing**, where a section of a narrative and its corresponding visuals will be **organized within one block throughout the entire design process** to enable flexible prototyping of the story, visualizations, and animations while ensuring their correspondence.

6 DATAPARTICLES

DataParticles is a prototype system that uses *language-oriented editing* and *block-based editing* to address the challenges of prototyping and iterating on data stories with AUVs. DataParticles consists of five UI components. The **rich text editor** (Figure 3a) allows the user to input text with natural language to tell a story. The **view panel** (Figure 3b) animates the corresponding visualization while the user is typing. The **visual effects configuration panel** (Figure 3c) enables the user to customize animation effects. The **block operation toolbox** (Figure 3d) provides four block operations, allowing the user to flexibly manipulate a block sequence. The **dataset view panel** (Figure 3e) enables the user to view the link between the visualization and the original dataset. These components work together to enable users to quickly create and iterate on data stories with AUVs.

Next, we introduce how AUVs can be generated from natural language (Section 6.1), how iteration and exploration are supported via the block-based editor (Section 6.2). We then provide an example walk-through to showcase the user experience of using the DataParticles to create an AUV data story (Section 6.3).

6.1 Natural Language to AUVs

To support the authoring of congruent visual stories [D1], we designed a four-step pipeline that uses the text in a block to generate and update AUVs: (1) selecting the relevant data points, (2) performing data operations on the selected data points, (3) specifying the visual encoding, and (4) configuring the animations between blocks. While the user is writing in a block, DataParticles detects the four key types of phrases and uses constituency parsing [21] to understand the pragmatic structure of their writing. Based on the parsed sentence, DataParticles automatically determines the intended data selection and encoding. This information defines a visual state of the block, which the system uses to generate an

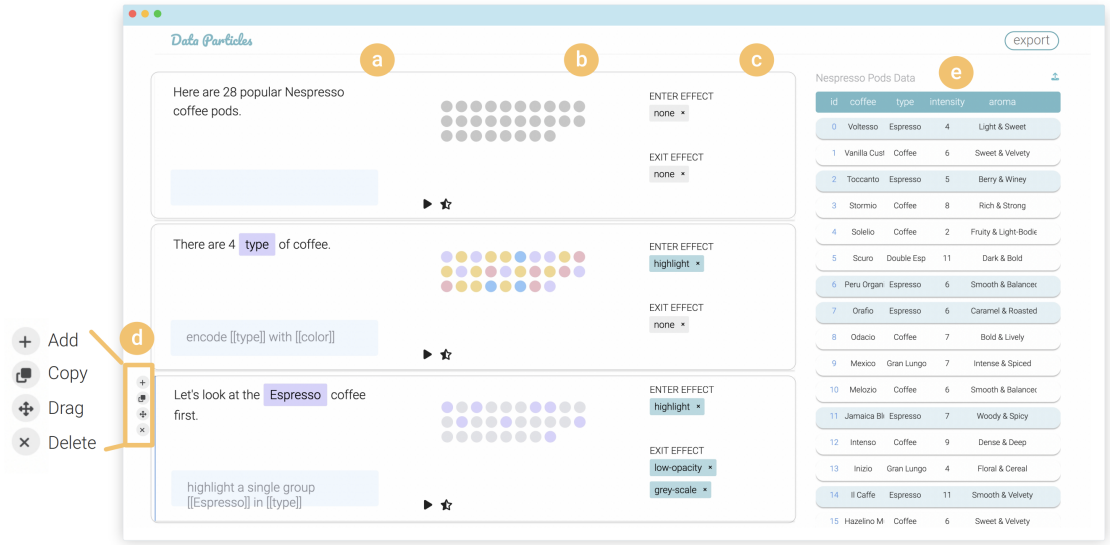

Figure 3: The DataParticles user interface includes (a) a rich text editor for natural language inputs, (b) a view panel that shows AUVs in real time, (c) a visual effects configuration panel to allow for the customization of animation effects, (d) a block operation toolbox that provides four block operations including add, copy, drag and delete, and (e) a dataset view panel that connects the data points to the original dataset.

animation that illustrates the transition of visual states between the current and previous blocks. This four-step pipeline is illustrated in Figure 4 and described step-by-step below.

6.1.1 Data Selection. Given the text narrative and the dataset as input, DataParticles first selects the relevant data points for further data operations, visual encodings, and animations. The data selection is performed by matching the words and phrases in the text with data properties and values in the dataset. For categorical data, the system matches the data value phrases directly with the corresponding data points. For example, if the text mentions *"men"*, the system will match the text to the value of the *"gender"* property and select the corresponding data points. For numerical data, the system matches both data property and value phrases. For example, if the text mentions *"basketball players ranked between 50 and 100,"* the system will identify the *"rank"* property and select all data points with a rank value within the mentioned range. The system supports several phrases that can indicate a range of data values, such as *"larger than," "over," "range,"* and *"between."* If no new selection is inferred from the sentence, the system will automatically inherit the selection from the previous block.

When writing the text with DataParticles, the user can use a "#" symbol to bring up a dropdown list of all the data properties in the dataset. This allows the user to map a numerical property to a categorical property by defining a new data value tag to specify a range over a numerical property. This feature gives the user more control over the data selection process and allows them to use the text in their narrative to select more complex sets of data points. These interactions are described in Section 6.3.3.

6.1.2 Data Operations. DataParticles identifies the *Data Operation Phrases* in a sentence and applies the corresponding operations to the selected data points. For example, in the sentence *"The average pocket size for men."* DataParticles will identify the operation *"average"* and calculate the average pocket size for men. DataParticles supports three types of data operations, including calculating derived values, finding the extreme , and finding an alternative.

Occasionally, a data operation is intended to be applied to subgroups rather than the entire selection. In these cases, the system first determines whether there is a *"group by"* operation based on the sentence structure, and then applies the operation to each group. Taking the sentence *"The average intensities of different types of coffee are similar"* as an example. There are two properties mentioned in the sentence (i.e., *"intensities"* and *"types"*). Based on the pragmatic structure, *"type"* is a dependent component of *"intensity"* and *"average"* is an adjective of the noun *"intensities."* The sentence parsing indicates that the data operation *"average"* should be performed on the *"intensity"* within each *"type."* Thus, DataParticles first groups the selected data points by *"type"* and calculates the average *"intensity"* of data points in each group.

6.1.3 Visual Encoding. This step specifies how data attributes, indicated by *Data Property Phrases*, are mapped to visual channels. DataParticles currently supports three visual encoding channels: size, position, and color. The encoding assignment follows a greedy algorithm and always assigns the best available visual channel to the new data property. Each data property will be assigned a default encoding priority list based on the property name, e.g., a data property named *"width"* or *"height"* indicates a size encoding with the corresponding visual channels, and the data types adapted from Mackinlay's ranking rules [28]. The priority of the encoding can change dynamically based on the encoded properties from the previous block and the visual operation phrases used in the text. The system strives to maintain the same encoding of a data property

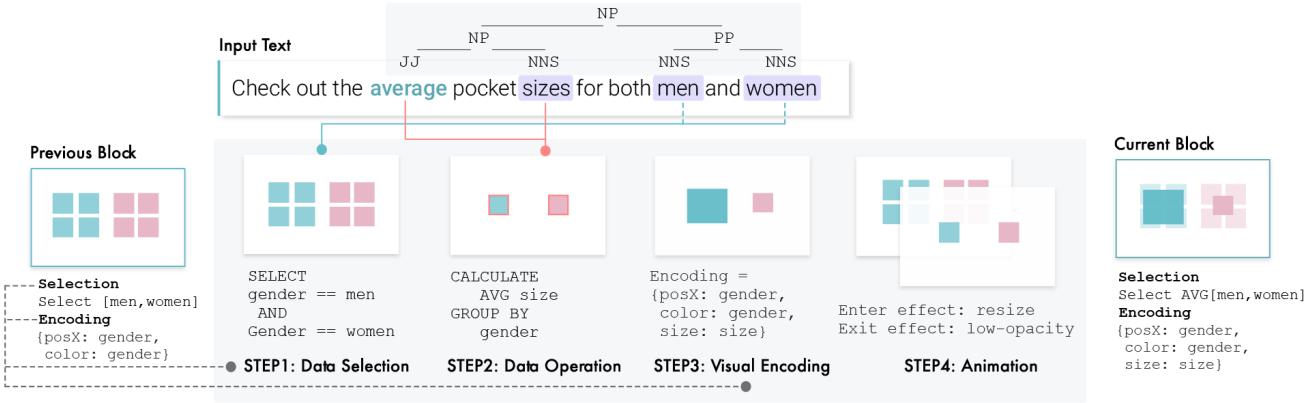

Figure 4: DataParticles uses a pipeline to generate AUVs in a block by taking the visual states from the previous block and the text narration in the current block as input. In Step 1 data selection, DataParticles selects corresponding data points based on the previous state and the mentioned data value phrases. In Step 2 data operation, by mapping keywords and parsing the sentence structure, DataParticles calculates new data points representing the average pocket size. In Step 3 visual encoding, DataParticles determines the appropriate encoding using the new data property. In Step 4 animation, DataParticles determines visual effects and transitions the previous visualization to the current visualization.

across blocks, while prioritizing the desired encoding indicated by certain visual operation phrases. The mapping between encoding and visual operation phrases is described in Section 4.2.3.

6.1.4 Animations. In DataParticles, animations are staggered transitions between adjacent visual states that are applied to selected and non-selected data points. These animations use enter and exit effects to show how the data points change over time, allowing the user to see how the data evolves as the story progresses.

DataParticles supports four enter and three exit effects. The enter effects specify the animation of the selected data points, while the exit effects are applied to non-selected visual units to set a contrast and highlight the data points that are changing. Among the enter effects, three are directly mapped to the encoding: *resize* for size encoding, *split* for positioning encoding, and *highlight* for color encoding. The fourth enter effect, *aggregate*, is designed to move data points from the same group closer to each other. These enter effects will default to the encoding of the new data property being mentioned in the corresponding text. The three exit effects include *grayscale*, *low-opacity*, and *remove*, which are commonly used in existing AUVs. The *low-opacity* effect is applied by default. These effects can be customized by the user using the visual effects configuration panel in DataParticles.

DataParticles allows the user to customize the animation effects by changing the enter and exit effect tags. The enter effect tags allow the user to manually specify the desired encoding for the data property mentioned in the corresponding text. Multiple enter effects will map the data property to multiple graphic attributes, while the exit effects are exclusive to each other.

To create more effective and engaging visual transitions, DataParticles employs a set of staggering rules to determine the proper sequence of animations. These rules help direct the viewer's attention and support more effective communication by ensuring that the animations are well-coordinated from one to the next:

(1) *Exit animations occur before enter animations.* By applying the exit animation first, viewers' attention will be directed first to the selected data points and then to the enter animation to understand how the data points are being visually encoded and animated.
(2) *Enter sequence of the data points is determined by narrative sequence.* For example, if the text mentions *"men and women"* in sequence, the corresponding visual units will be highlighted in the same order. This staggering helps align the timing of the narrative sequence with the graphic appearance of the data. By carefully coordinating the sequence of the enter animations with the narrative sequence, the system can help ensure more effective visual comprehension.
(3) *In-situ changes first.* If multiple enter effects are applied, the *highlight* and *resize* effects are staggered to appear before *split* or *aggregate*, which introduce positioning changes. By presenting in this sequence, the correspondences between them can be more easily observed, enabling a clearer understanding of the changes between visual states

6.2 Block-based Editing

To help a user flexibly prototype and maintain visual correspondence with text narrations, DataParticles employs block-based editing to manage the sections of the story narrative and their corresponding visualizations [D2]. The blocks are organized linearly following the story narrative and can be flexibly reorganized to prototype the story. The data selection, visual encoding, and animation states of each block can be accessed by the next block to allow for the reuse and smooth transition of the selection, encoding, and animation. DataParticles uses four block operations and two view operations to support the prototyping process.

6.2.1 Block Operations. DataParticles enables a user to add, copy, drag, and delete blocks. Specifically,

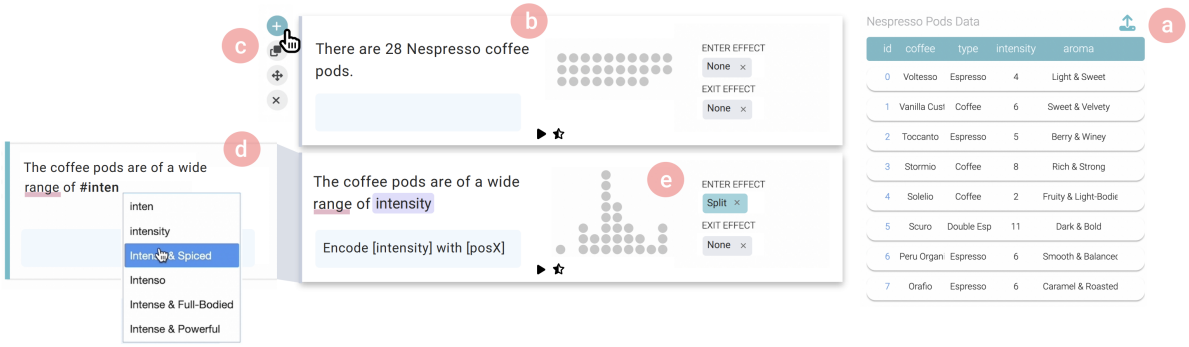

Figure 5: A user setting up and adding blocks to develop a story with DataParticles. When the user (a) imports data to set up the story, DataParticles (b) automatically generates the first block enabling the user to (c) add a block and (d) write a story using natural language. The system (e) shows the corresponding visualization while the user is writing.

- The [Add] operation enables a user to add a block with a blank text narrative to reuse visual states from the previous block as its initial state, including the data selection and visual encoding.
- The [Copy] operation creates a duplicated block with the text narrative and visual states of the original. This enables a user to modify phrases in the text to quickly switch the groups of data points that are selected, which supports a parallel narrative structure.
- The [Drag] operation is used to change the sequence of blocks, which can affect the link between them. After a block is dragged, the visual state (i.e., selection and encoding) of all the blocks will remain the same, but the resulting animation will update to reflect the new adjacent states. This operation enables a user to rearrange a sequence of blocks while maintaining the link between them.
- The [Delete] operation is used to remove a block from the sequence. This operation affects the link between the blocks by directly linking the two blocks that were adjacent to the deleted block. Like the [Drag] operation, the visual states of the blocks will remain the same after deletion, but the resulting animation will be updated to reflect the changes in the block sequence and link. This operation enables a user to remove blocks from the sequence while maintaining the links between the remaining blocks.

6.2.2 View Operations. DataParticles supports two operations, i.e., play and propagate, within each view. Specifically,

- The [Play] operation animates the transition from the previous visual state to the current visual state. This operation enables a user to see how an animation will change based on the current state of the blocks and their sequence. A user can use it for testing and previewing purposes, to understand how an animation will be affected by their changes to the block sequence and links.
- The [Propagate] operation manually creates links between adjacent blocks after the flow has changed. This operation is typically used after a block is *dragged* or *deleted*, and it forces the current block to inherit the previous visual state in the current flow. This operation enables a user to control the links between blocks in a sequence and ensure that the resulting animation reflects the desired flow.

6.3 System Walkthrough

We demonstrate the interactions supported by DataParticles via an example of creating a data story about coffee pods.

6.3.1 Initial Setup. Fiona is a journalist who wants to create a story introducing basic concepts about coffee to the public. She collected a dataset of 28 coffee pods from the with 4 attributes: coffee name, coffee type, intensity, and aroma. Fiona first opens DataParticles in a web browser. To begin, she imports her dataset using the upload button in the top right of the data panel (Figure 5a). DataParticles prepares the first block for her with the sentence: *"Here are 28 Nespresso coffee pods."* to initiate the story (Figure 5b).

6.3.2 Developing a Story by Adding Blocks. She begins the sentence by adding a block using the [Add] button (Figure 5c). In the new block, she starts explaining the "intensity" concept and writes *"These coffee pods are of a wide range of #intensity"* (Figure 5d). DataParticles recognizes *"intensity"* as one of the columns in the dataset and her intention to show the *"range"* of the intensity. Thus, the visual units animate to show the distribution of the coffee pods over different intensity levels across the horizontal axis (Figure 5e).

6.3.3 Defining a New Data Value. Fiona then plans to introduce the concept of light, medium, and dark coffee, as defined by intensity. She adds a block and writes *"Based on the intensity, we can categorize them in to #light, #medium and #dark roasted coffee."* However, this time DataParticles does not recognize '*light,*" *"medium"* or *"dark"*, as they were not initially defined in the dataset. Fiona clicks on the tag *"light"*, which triggers a specification box asking her to define *"light"* (Figure 6a). Since she has talked about "intensity" in the previous block, DataParticles suggests that she should define a new data value based on *"intensity"*. She inputs an intensity range that defines the light coffee and clicks the "tick" button. Fiona follows the same process to define *"medium"* and *"dark"*. While she is creating new data values, DataParticles generates and displays a sequential highlighting effect of the corresponding visual units with different shades (Figure 6b).

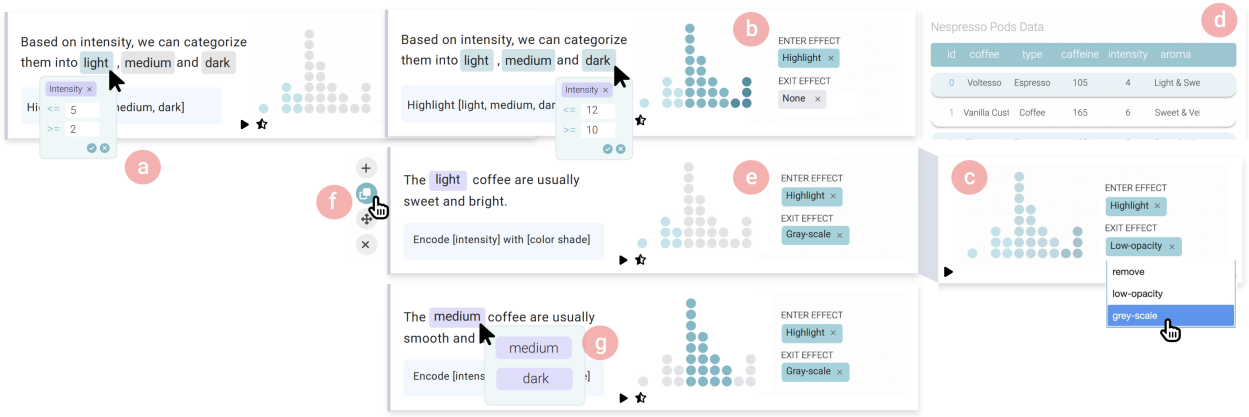

Figure 6: A user defining a new data value and copying blocks for parallel narrations using DataParticles. The user (a) map numerical data values to new categorical data values. (b) The system animates selected visual units in a sequence corresponding to the user's narrative. The user can (c) customize the effects and (d) see the corresponding data row in the data view. (e) The newly defined data value can be used throughout the story. When authoring parallel narratives, (f) the user can use the copy function and (g) click on defined value tags to select alternative values.

6.3.4 Using Parallel Structures by Copying Blocks. After introducing the concept, Fiona moves on to write about each type of roast. She adds a block and starts writing *"#light coffees are"*. DataParticles recognizes that she wants to focus on the light coffee and highlights the units belonging to light coffees with the same color as the previous block, while applying a default *low-opacity* exit effect to other units (Figure 6c). The data rows in the data panel are highlighted (Figure 6d). She checks the *"aroma"* column of the highlighted data rows and finds that light coffees share a similar *"sweet and bright"* aroma, so she completes the sentence with *"#light coffee are usually sweet and bright."* To make the selection clearer, she applies a *grey-scale* effect by changing the exit effect tag and the visual units transform accordingly (Figure 6e). As she is satisfied with the current visual transition, she uses the [Copy] button to duplicate the current block for medium coffee (Figure 6f). This time, she clicks on the *"light"* tag, and a drop-down with *"medium"* and *"dark"* pop up (Figure 6g). As she selects *"medium,"* the same visual effects are applied to highlight the medium coffee units. She repeats the process to author the dark coffee view.

6.3.5 Changing and Maintaining Visual Flow. Fiona starts to describe different types of coffee. However, rather than having this as the end, she wants to try a different flow by talking about coffee type before intensity. She inserts a block directly under the first block and writes *"These coffee are of different #types"* (Figure 7a). After parsing the sentence, DataParticles highlights the visual units into different colors. As she sees the visualization, she wants the same type of coffee to appear adjacently, so she adds an "aggregate" tag to the enter effects and the units move accordingly (Figure 7b). She wants to propagate this color encoding to the next stage, so she clicks the [Propagate] button at the bottom of the view. The visualization (Figure 7c) shows type and intensity in the same view. As she is authoring, all the encodings are translated into plain text (Figure 7d blue tinted box). To help readers understand the visualization, Fiona adds this caption to the third block.

6.3.6 Review and Export. Once finished, Fiona clicks the [Play] button and reviews the animations in each block. When satisfied with the story flow, she clicks on one of the visual units, and a file browser pops up, enabling her to import any .svg as a customized unit shape. She imports a coffee pod graphic and all the visual units are replaced with the coffee pod shape. Finally, she clicks the "export" button at the top right of the page. DataParticles provides three output formats: a webpage as a scrollable interactive article, a video with a chunked timeline, and a side-by-side comic book format. Fiona chooses to export her article as an interactive article so the DataParticles renders the story into a webpage that she can easily share with others.

7 EXPERT EVALUATION

To evaluate DataParticles and its two core interaction techniques (i.e., language-oriented and block-based editing), we conducted an expert evaluation to understand how the tool could address creators' pain points and its potential to be integrated into their workflows, as well as to identify any limitations of and ideas for extending DataParticles.

7.1 Participants

We recruited nine experts in the domain of data storytelling (i.e., 5 females, 4 males, aged 27-37) through emails. All participants had more than 3 years of experience in data storytelling and had professional experience authoring AUVs. The studies were conducted remotely via Zoom. Participants accessed DataParticles through a web browser. They received a $40 dollar gift card for the 75-90 minute session.

7.2 Study Protocol

Participants first completed a consent form and were then instructed to complete three tasks with DataParticles via screen

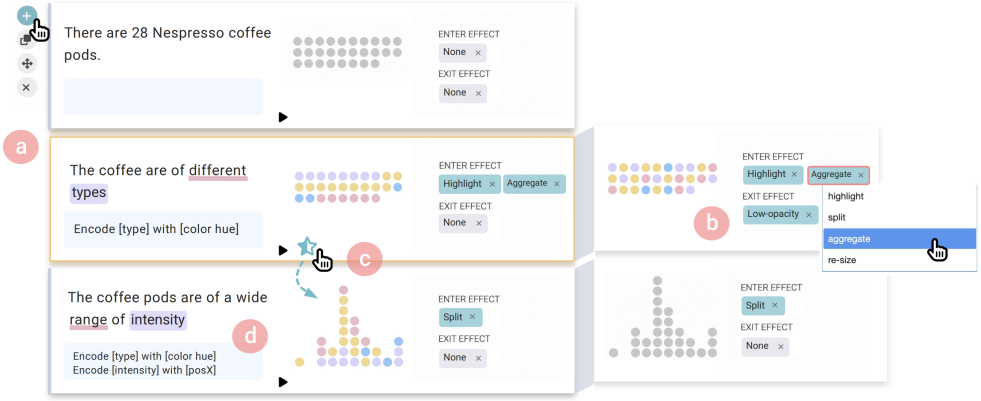

Figure 7: A user changing and maintaining visual flow using DataParticles. The system enables the user to (a) insert a block, (b) customize effects, (c) propagate visual states, and (d) see the encodings in plain text.

sharing during the study. The study concluded with a feedback questionnaire and post-study interview.

Introduction and System Walk-through (~20 minutes). Participants were first given a general introduction to the motivation behind DataParticles. The experimenter then walked the participants through DataParticles's features by guiding them through creating a tutorial story using a dataset containing ten data items with four data properties. During the walkthrough, the experimenter described the interaction verbally and asked participants to perform specific actions. After the system walkthrough, participants were given time (~5 minutes) to practice using DataParticles.

Reproduction Task (~15 minutes). Participants were asked to reproduce the coffee pod story described in section6.3. The dataset contained 28 data items with 5 properties. The text narratives and corresponding visualizations were provided in a Google Doc.

Creation Task (~20 minutes). Participants were given a dataset of 32 jean pockets, adapted from an existing AUVs story [14]. Participants were instructed to create their own story with 3-5 blocks We encouraged participants to try different story narratives and asked them to think aloud during the creation process.

Questionnaire and Semi-structured Interview (~20 minutes). After completing all the tasks, participants filled out a questionnaire about the usefulness and usability of DataParticles using a 7-point Likert scale, followed by a semi-structured interview for feedback on the system utility, effectiveness, and future improvements.

7.3 Results

All participants successfully completed the three tasks. Most participants found the interface easy to learn (3 strongly agree, 6 agree) and easy to use (2 strongly agree, 7 agree). They also provided that they would likely use DataParticles to create data stories (1 strongly agree, 7 agree, 1 somewhat agree). We first report the participants' responses to the two interaction techniques and our observations of how they use DataParticles and then discuss the system limitations we learned from the participants.

7.3.1 Feedback on Language-oriented Authoring. Participants found that being able to immediately see the visualizations helped them focus on the story they were telling, and they generally agreed that the visualizations generated by the system matched their narrations (2 strongly agree, 6 agree, 1 somewhat agree). This supports the first design goal (D1) of DataParticles, which was to maintain the connections between narrative stories and the visualizations.

Participants found that the system's ability to automatically maintain congruence between narrations and visualizations allowed them to *"get directly to the story"* (P1). This was made possible by the integrated environment offered by the system, which was a marked improvement over participants' previous workflows, which employed spreadsheets to explore data but struggled to focus on the story, e.g., *"like a spreadsheet, you just feel you are not pointing to the right direction of telling the story"* (P1) and *"I have to kind of mold the story after the processing the data"* (P5). Additionally, using natural language to express thoughts was found to be a more direct and natural way of communicating, especially in the early stages of story creation where the story was usually guided by high-level intentions rather than specific configurations. P2 described the process as *"Really cool! Because that's the natural way we think about a story, so you don't let all the tedious configurations and editing get in the way of your thoughts."* Overall, participants thought that the system was *"a lot faster"* (P3) to prototype with and more effective than their previous workflow.

7.3.2 Feedback on Block-based Editing. The block-based authoring environment allowed participants to easily explore the data (2 strongly agree, 5 agree, 2 somewhat agree) and try out different narratives (2 strongly agree, 5 agree, 2 somewhat agree), supporting the need for flexible prototyping. As P1 noted, *"it allows you to be more creative without worrying too much if it is making sense."* P3 said *"I really like the idea of using blocks to author the flow. You can basically do all the things that you are interested in."* In fact, the flexibility of prototyping was not only offered by the operations of blocks, but also by the fact that using blocks ensured that visualizations remained synchronized with corresponding parts of the narrative (6 strongly agree, 2 agree, 1 somewhat agree). P5 also

mentioned that the initial auto-generated block, which displayed all of the data points, was helpful for getting started and encouraged their exploration. However, they also suggested that it would be beneficial to offer more options to configure the initial layout based on the story context and the size of the dataset.

The ability to propagate visual states was preferred by participants, who reported that it enabled them to easily prototype and explore different story flows (3 strongly agree, 6 agree). Participants found this function useful when the sequence changed during prototyping. They tended to click the [`propagate`] button for multiple blocks that came after a changed block to update the visualizations. In contrast, the [`play`] button was mostly used when participants had finished their stories and wanted to preview the final result.

7.3.3 Workflows with DataParticles. During the construction of the story, the authoring process followed a pattern of exploration, construction, and polishing. In the exploration stage, the authors tended to incrementally reveal data facts by inputting sentences that involved a single segment of the dataset. For example, instead of mentioning the pocket size for men and women in one sentence, most participants (8/9) explored the gender and size properties separately. Moreover, it was common to start by adding several blocks and then delete them. In the construction stage, participants adjusted the sequence more often by inserting and adding blocks that they had explored before than dragging blocks as dragging could lead to significant changes in the entire sequence, which made them feel less in control (P9). Participants may thus benefit from block operations that further support less disruptive design exploration, as discussed in the limitations section below.

Multiple participants mentioned that they valued being able to quickly see whether their story made sense or not. DataParticles offered a unique way to plan by using visual reasoning. As P6 noted, *"Sometimes I am not sure whether to use size, position, opacity, or color to represent the data... it can be hard to think of, so why not just write down what makes the most sense and see how it looks?"* The potential for visual reasoning is further discussed in Section 8.2.

7.3.4 System Limitations. Our participants had mixed opinions on the creative freedom provided by the system (1 strongly agree, 2 agree, 3 somewhat agree, 2 neutral, 1 somewhat disagree). We identified limitations of the current system that may explain this in terms of the expressiveness of the visual content and flexibility in supporting prototyping.

While DataParticles was able to support most data selections and operations during the authoring process, participants felt limited by the visual effects available in the system. This was due to a lack of comprehensive effects and mappings between natural language expressions and these effects. For example, P9 wanted to encode data points *"using different shades of blue"* which is not currently supported by DataParticles. Participants also reported wanting to create animations that simulated camera movements, such as zooming in and out of scenes, and that corresponded to more complex narrative structures, such as using a side-by-side view to present the comparisons of groups of data points.

Participants also found that the linear structure of blocks limited the flexibility of the prototyping. We observed that participants constantly added and deleted blocks to explore different flows, before finalizing the story. P5 said *"it would be great if I could review my editing history and choose from them."* P8 and P9 also desire to see multiple designs at the same time. These findings suggest that DataParticles could benefit from additional block organization and operations, such as branching, hiding blocks, and grouping and folding sequences of blocks, to support more flexible and less disruptive parallel prototyping.

We recognized that the current task design encouraged a "data-driven" planning process, where participants first explore data without polished narratives. Within the process, participants felt limited by the data exploration DataParticles offers. P7 explicitly mentioned *"I see it fits into the middle part of my workflow, when I have the dataset cleaned up and want to figure out what to tell."* The current blocks only maintained the link between the text and the visuals, which could benefit from setting linkage to the dataset. As P4 noted, *"It would be powerful to show different data view for different blocks."* Interestingly, participants also mentioned they imagined using it for the *story-driven* planning process. As mentioned by both P2 and P8, it would be useful if users could paste their script into the system and generate visualizations using simple markups. This may be challenging to the current prototype system, which is designed to support well-chunked and descriptive text input. The design decision was made with the unique language characteristics of data stories, where the visualizations unfold incrementally with small changes. While it was able to properly understand the participants' intentions during the study, this may be insufficient for a *story-driven* planning. In order to support this, the system should be able to handle more flexible text input with improved natural language processing capabilities.

7.4 Summary

All nine expert participants were excited about the potential of using DataParticles to improve their workflows when creating data stories with rich AUVs. They appreciated that DataParticles enabled them to quickly visualize their ideas and experiment with different story ideas like a "digital sketch book for data." Specifically, language-oriented authoring enabled participants to be story-driven and block-based editing enables quick and flexible prototyping. The study also noted that the limited set of visual effects and block operations offered by the system may have restricted users' creative freedom and could prevent them from fully expressing their design ideas with the tool.

8 DISCUSSION AND FUTURE WORK

Based on our observations throughout the study, we believe that there are several opportunities to improve the system and provide more support for the creation process. In this discussion, we summarize them in terms of streamlining the creation process, supporting visual reasoning, and extending the concepts to broader visualizations. We hope that with future research in these areas, we can enhance the capabilities of DataParticles and make it a more powerful and versatile tool for AUV creation.

8.1 Supporting the Entire Creation Process

While the participants appreciated the prototyping power of DataParticles, they pointed out that more *"low-level control"* is needed

to produce high-fidelity content and create unique stories and visuals. P5 said, *"the upside of that (using code) is that each project is distinctive and we don't have to settle for pre-set styles and patterns."* Moreover, intelligent tools can lead to *downstream frustration* because creators may not know the limits of the system. P6 said, *"I am willing to put more time and effort into my final product... so I don't mind learning a tool that gives me every bit of control over it."*

Participants suggested that the system should be able to export designs in a format that can be improved with other dedicated tools. As we found in the formative study, developing content in separate environments may result in breaking the congruence between components, which have required significant effort to establish at earlier stages. As our research has shown, by uniting the story and the corresponding AUVs within the blocks, creators can freely explore different story narratives without breaking the connections between the narrative stories and AUVs. Therefore, we believe that to support a fluid workflow for the entire design process, the intrinsic relationships amongst various content components should be preserved throughout the entire creation process.

One idea to explore is to extend our tool from a two-column to an N-column format to accommodate the various elements and tools involved in the creation process. With N-columns, iterations can incrementally occur between columns with a side-by-side view, allowing creators to gradually mold a low-fidelity prototype into a higher-fidelity product using more advanced tools and programming toolkits. To achieve this vision of streamlining the creation process, it is critical to find proper intermediate representations that can preserve enough information from lower-fidelity stages and be flexible to use in higher-fidelity stages.

8.2 Support Visual Reasoning

In our formative research, we identified two approaches to data-driven storytelling: a data-driven approach, where the story was planned based on the data facts that creators wanted to convey, and a story-driven approach, where creators used the data and visualizations that best fit their story. Our user study also revealed an alternative approach, where participants focused on visual storytelling. For example, P8 said, *"I found it interesting that when I constructed the story with this system, I was reasoning on whether the visualizations and transitions between them made sense to me rather than the text itself. After I was satisfied with the animations, I went back to the text and completed the story."* This aligned with what we observed in many of the data-driven stories that employed rich AUVs, from which the key data insights were often self-explanatory using just the visual content.

Moreover, to facilitate visual comprehension and engagement, AUVs are also blended with other types of visual content (e.g, images, simulations, and character animations). These blendings could create more immersive and engaging narratives that better situate the visualizations within the context of the story. For example, when the story was about basketball players, the units were animated to bounce like basketballs on the ground [35]. When discussing constellations, the visual units were animated to rotate in a circular layout like actual constellations and situated within a collection of satellite images [6]. These visualizations are compelling yet more challenging to create due to the high level of creativity it requires.

We are interested in exploring the potential of incorporating the state-of-the-art generative models into our system to facilitate the creation of these visualizations. With the advent of powerful large language models (e.g., GPT-3), we envision an intelligent system that could suggest possible blending strategies given the narratives to help better situate visualizations within the story.

8.3 Beyond Unit Visualizations

The type of unit visualization supported by DataParticles falls between aggregated visualizations (e.g., bar charts, line charts) and more complex particle animations (e.g., waffle charts, physical simulations). Although these visualizations are not within the scope of our system, they are commonly used in data storytelling [17] and could potentially benefit from a more flexible prototyping experience. Herein, we discuss the opportunities and challenges when extending our system to support these types of visualizations.

Aggregated visualizations contain simpler graphic elements and more structured layouts, which make them a promising candidate to establish a link between text and visuals throughout a story, especially with the help of existing work in this domain [9, 38]. However, unlike unit visualizations, whose visual marks almost always remain unchanged during animations, transitions of aggregated visualizations often employ complex graphical transformations, such as transforming a line chart into a bar chart by transforming lines and dots into rectangles. On the other hand, physical simulations, while using consistent visual marks, often employ the complex movements of particles to represent semantic meanings. In contrast to the 1-1 mapping between data points and visual marks in unit visualizations, other types of visualizations often utilize more flexible mappings between the data points and the visuals.

In both aggregated visualizations and simulations, a visual unit could represent multiple data points or only selected data points, where the number of visual units could be determined by the narration. For example, a waffle chart that consists of a grid of small cells often animates according to the approximated percentages mentioned in the text, rather than being truly data-driven. The complex animations and flexible data-visual mappings do not conflict with language-oriented and blocked-based editing that DataParticles proposes. We are interested in extending DataParticles with a more advanced animation engine and intelligent data selection techniques, to support a broader coverage of visualizations.

9 CONCLUSION

This work presents a systematic exploration of a specific domain of data storytelling that uses animated unit visualizations (AUVs). The research prototype, DataParticles, leveraged language-oriented authoring and block-based editing to address the pain points that exist when creating stories containing AUVs. With DataParticles, users could leverage the latent connections among text, data, and visualizations to quickly and flexibly prototype, explore, and iterate on both a story narrative and its corresponding visualizations. Feedback from creative experts confirmed the potential of this approach and pointed to future directions for improvement. We are excited to further extend the concepts in DataParticles to support the creation process of a broad range of visual content.

REFERENCES

[1] Parastoo Abtahi, Victoria Ding, Anna C. Yang, Tommy Bruzzese, Alyssa Romanos, Elizabeth L. Murnane, Sean Follmer, and James A. Landay. 2020. Understanding Physical Practices and the Role of Technology in Manual Self-Tracking. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4 (2020), 115:1–115:24. https://doi.org/10.1145/3432236

[2] Fereshteh Amini, Nathalie Henry Riche, Bongshin Lee, Christophe Hurter, and Pourang Irani. 2015. Understanding Data Videos: Looking at Narrative Visualization through the Cinematography Lens. In *Proc. of CHI*. ACM, 1459–1468.

[3] Fereshteh Amini, Nathalie Henry Riche, Bongshin Lee, Andrés Monroy-Hernández, and Pourang Irani. 2017. Authoring Data-Driven Videos with DataClips. *IEEE Trans. Vis. Comput. Graph.* 23, 1 (2017), 501–510. https://doi.org/10.1109/TVCG.2016.2598647

[4] Fereshteh Amini, Nathalie Henry Riche, Bongshin Lee, Andres Monroy-Hernandez, and Pourang Irani. 2017. Authoring Data-Driven Videos with DataClips. *IEEE TVCG* 23, 1 (2017), 501–510.

[5] Rita Borgo, Johannes Kehrer, David H. S. Chung, Eamonn Maguire, Robert S. Laramee, Helwig Hauser, Matthew Ward, and Min Chen. 2013. Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications. In *34th Annual Conference of the European Association for Computer Graphics, Eurographics 2013 - State of the Art Reports, Girona, Spain, May 6-10, 2013*. Eurographics Association, 39–63. https://doi.org/10.2312/conf/EG2013/stars/039-063

[6] Nadieh Bremer and Jenna Mukuno. 2018. *CONSTELLATIONS*. Retrieved December 11, 2022 from https://nbremer.github.io/planet-constellations/

[7] Ruochen Cao, Subrata Dey, Andrew Cunningham, James A. Walsh, Ross T. Smith, Joanne E. Zucco, and Bruce H. Thomas. 2020. Examining the Use of Narrative Constructs in Data Videos. *Vis. Informatics* 4, 1 (2020), 8–22.

[8] Zhutian Chen, Yijia Su, Yifang Wang, Qianwen Wang, Huamin Qu, and Yingcai Wu. 2020. MARVisT: Authoring Glyph-Based Visualization in Mobile Augmented Reality. *IEEE Trans. Vis. Comput. Graph.* 26, 8 (2020), 2645–2658. https://doi.org/10.1109/TVCG.2019.2892415

[9] Zhutian Chen and Haijun Xia. 2022. CrossData: Leveraging Text-Data Connections for Authoring Data Documents. In *CHI '22: CHI Conference on Human Factors in Computing Systems, New Orleans, LA, USA, 29 April 2022 - 5 May 2022*. ACM, 95:1–95:15. https://doi.org/10.1145/3491102.3517485

[10] Zhutian Chen, Qisen Yang, Xiao Xie, Johanna Beyer, Haijun Xia, Yingcai Wu, and Hanspeter Pfister. 2022. Sporthesia: Augmenting Sports Videos Using Natural Language. *To appear in IEEE Transactions on Visualization and Computer Graphics* (2022).

[11] Peggy Chi, Tao Dong, Christian Frueh, Brian Colonna, Vivek Kwatra, and Irfan Essa. 2022. Synthesis-Assisted Video Prototyping From a Document. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–10.

[12] Neil Cohn. 2013. Visual Narrative Structure. *Cogn. Sci.* 37, 3 (2013), 413–452.

[13] Matthew Conlen and Jeffrey Heer. 2018. Idyll: A Markup Language for Authoring and Publishing Interactive Articles on the Web. In *Proc. of UIST*. ACM, 29–67. https://doi.org/10.1017/9781108657846.003

[14] Jan Diehm and Amber Thomas. 2018. *Women's pockets are inferior*. Retrieved September 15, 2022 from https://pudding.cool/2018/08/pockets/

[15] Steven Drucker and Roland Fernandez. 2015. A unifying framework for animated and interactive unit visualizations. *Microsoft Research* (2015).

[16] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie Karahalios. 2015. DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization. In *Proc. of UIST*. ACM, 489–500. https://doi.org/10.1145/2807442.2807478

[17] Pascal Goffin, Jeremy Boy, Wesley Willett, and Petra Isenberg. 2016. An exploratory study of word-scale graphics in data-rich text documents. *IEEE transactions on visualization and computer graphics* 23, 10 (2016), 2275–2287.

[18] Fred Hohman, Matthew Conlen, Jeffrey Heer, and Duen Horng Polo Chau. 2020. Communicating with interactive articles. *Distill* 5, 9 (2020), e28.

[19] Julia Janicki. 2022. *Visualizing Shark Numbers*. Retrieved September 15, 2022 from http://the-tardigrade.com/projects/sharks/

[20] Project Jupyter. 2021. Jupyter. https://jupyter.org/

[21] Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. *arXiv preprint arXiv:1805.01052* (2018).

[22] Donald E. Knuth. 1984. Literate Programming. *Comput. J.* 27, 2 (1984), 97–111. https://doi.org/10.1093/comjnl/27.2.97

[23] Shahid Latif, Zheng Zhou, Yoon Kim, Fabian Beck, and Nam Wook Kim. 2021. Kori: Interactive Synthesis of Text and Charts in Data Documents. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 184–194.

[24] Shahid Latif, Zheng Zhou, Yoon Kim, Fabian Beck, and Nam Wook Kim. 2022. Kori: Interactive Synthesis of Text and Charts in Data Documents. *IEEE Trans. Vis. Comput. Graph.* 28, 1 (2022), 184–194. https://doi.org/10.1109/TVCG.2021.3114802

[25] Sam Lau, Ian Drosos, Julia M. Markel, and Philip J. Guo. 2020. The Design Space of Computational Notebooks: An Analysis of 60 Systems in Academia and Industry. In *IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2020, Dunedin, New Zealand, August 10-14, 2020*. IEEE, 1–11. https://doi.org/10.1109/VL/HCC50065.2020.9127201

[26] Bongshin Lee, Nathalie Henry Riche, Petra Isenberg, and Sheelagh Carpendale. 2015. More than telling a story: Transforming data into visually shared stories. *IEEE computer graphics and applications* 35, 5 (2015), 84–90.

[27] Junhua Lu, Wei Chen, Hui Ye, Jie Wang, Honghui Mei, Yuhui Gu, Yingcai Wu, Xiaolong Luke Zhang, and Kwan-Liu Ma. 2021. Automatic Generation of Unit Visualization-based Scrollytelling for Impromptu Data Facts Delivery. In *14th IEEE Pacific Visualization Symposium, PacificVis 2021, Tianjin, China, April 19-21, 2021*. IEEE, 21–30. https://doi.org/10.1109/PacificVis52677.2021.00011

[28] Jock Mackinlay. 1986. Automating the design of graphical presentations of relational information. *Acm Transactions On Graphics (Tog)* 5, 2 (1986), 110–141.

[29] Arpit Narechania, Arjun Srinivasan, and John Stasko. 2021. NL4DV: A Toolkit for Generating Analytic Specifications for Data Visualization from Natural Language Queries. *IEEE TVCG* 27, 2 (2021), 369–379. https://doi.org/10.1109/TVCG.2020.3030378 arXiv:2008.10723

[30] Inc. Observable. 2021. Observable. https://observablehq.com/

[31] Deokgun Park, Steven Mark Drucker, Roland Fernandez, and Niklas Elmqvist. 2018. Atom: A Grammar for Unit Visualizations. *IEEE Trans. Vis. Comput. Graph.* 24, 12 (2018), 3032–3043. https://doi.org/10.1109/TVCG.2017.2785807

[32] Norton S Parker. 1968. Audiovisual Script Writing. (1968).

[33] Studio R. 2021. R Markdown. https://rmarkdown.rstudio.com/index.html

[34] Adam Rule, Aurélien Tabard, and James D. Hollan. 2018. Exploration and Explanation in Computational Notebooks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*. ACM, 32. https://doi.org/10.1145/3173574.3173606

[35] Amber Samora, Russell & Thomas. 2019. *How many high school stars make it in the NBA?* Retrieved December 10, 2022 from https://pudding.cool/2019/03/hype/

[36] Edward Segel and Jeffrey Heer. 2010. Narrative Visualization: Telling Stories with Data. *IEEE TVCG* 16, 6 (2010), 1139–1148.

[37] Vidya Setlur, Sarah E. Battersby, Melanie Tory, Rich Gossweiler, and Angel X. Chang. 2016. Eviza: A Natural Language Interface for Visual Analysis. In *Proc. of UIST*. ACM, 365–377. https://doi.org/10.1145/2984511.2984588

[38] Vidya Setlur, Melanie Tory, and Alex Djalali. 2019. Inferencing underspecified natural language utterances in visual analysis. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. 40–51.

[39] Leixian Shen, Enya Shen, Yuyu Luo, Xiaocong Yang, Xuming Hu, Xiongshuai Zhang, Zhiwei Tai, and Jianmin Wang. 2021. Towards natural language interfaces for data visualization: A survey. *arXiv preprint arXiv:2109.03506* (2021).

[40] D. Shi, F. Sun, X. Xu, Xingyu Lan, David Gotz, and Nan Cao. 2021. AutoClips: An Automatic Approach to Video Generation from Data Facts. *Comput. Graph. Forum* 40, 3 (2021), 495–505. https://doi.org/10.1111/cgf.14324

[41] Arjun Srinivasan, Steven M Drucker, Alex Endert, and John Stasko. 2018. Augmenting visualizations with interactive data facts to facilitate interpretation and communication. *IEEE transactions on visualization and computer graphics* 25, 1 (2018), 672–681.

[42] Arjun Srinivasan, Bongshin Lee, Nathalie Henry Riche, Steven M. Drucker, and Ken Hinckley. 2020. InChorus: Designing Consistent Multimodal Interactions for Data Visualization on Tablet Devices. In *Proc. of CHI*. ACM, 1–13. https://doi.org/10.1145/3313831.3376782 arXiv:2001.06423

[43] Arjun Srinivasan, Nikhila Nyapathy, and Bongshin Lee. 2021. Collecting and Characterizing Natural Language Uterances for Specifying Data Visualizations. In *Proc. of CHI*. ACM. https://doi.org/10.1145/3411764.3445400

[44] Arjun Srinivasan and John T. Stasko. 2018. Orko: Facilitating Multimodal Interaction for Visual Exploration and Analysis of Networks. *IEEE Trans. Vis. Comput. Graph.* 24, 1 (2018), 511–521. https://doi.org/10.1109/TVCG.2017.2745219

[45] Nicole Sultanum, Fanny Chevalier, Zoya Bylinskii, and Zhicheng Liu. 2021. Leveraging text-chart links to support authoring of data-driven articles with vizflow. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–17.

[46] Junxiu Tang, Lingyun Yu, Tan Tang, Xinhuan Shu, Lu Ying, Yuhua Zhou, Peiran Ren, and Yingcai Wu. 2020. Narrative Transitions in Data Videos. In *31st IEEE Visualization Conference, IEEE VIS 2020 - Short Papers, Virtual Event, USA, October 25-30, 2020*. IEEE, 151–155. https://doi.org/10.1109/VIS47514.2020.00037

[47] Tan Tang, Junxiu Tang, Jiayi Hong, Lingyun Yu, Peiran Ren, and Yingcai Wu. 2020. Design guidelines for augmenting short-form videos using animated data visualizations. *J. Vis.* 23, 4 (2020), 707–720. https://doi.org/10.1007/s12650-020-00644-z

[48] Tan Tang, Junxiu Tang, Jiewen Lai, Lu Ying, Peiran Ren, Lingyun Yu, and Yingcai Wu. 2020. SmartShots: Enabling Automatic Generation of Videos with Data Visualizations Embedded. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*. ACM, 4509–4511. https://doi.org/10.1145/3394171.3414356

[49] the Guardian. 2022. *the Guardian*. Retrieved September 15, 2022 from https://www.theguardian.com/us

[50] the New York Times. 2022. *the New York Times*. Retrieved September 15, 2022 from https://www.nytimes.com/

[51] the Pudding. 2022. *the Pudding*. Retrieved September 15, 2022 from https://pudding.cool/

[52] John R. Thompson, Zhicheng Liu, and John T. Stasko. 2021. Data Animator: Authoring Expressive Animated Data Graphics. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*. ACM, 15:1–15:18. https://doi.org/10.1145/3411764.3445747

[53] Barbara Tversky, Julie Bauer Morrison, and Mireille Betrancourt. 2002. Animation: can it facilitate? *International journal of human-computer studies* 57, 4 (2002), 247–262.

[54] Vox. 2020. The Electoral College, explained. Retrieved September 15, 2022 from https://www.youtube.com/watch?v=ajavsMbCapY

[55] Vox. 2022. *Vox*. Retrieved September 15, 2022 from https://www.vox.com/

[56] April Yi Wang, Anant Mittal, Christopher Brooks, and Steve Oney. 2019. How Data Scientists Use Computational Notebooks for Real-Time Collaboration. *Proc. ACM Hum. Comput. Interact.* 3, CSCW (2019), 39:1–39:30. https://doi.org/10.1145/3359141

[57] April Yi Wang, Zihan Wu, Christopher Brooks, and Steve Oney. 2020. Callisto: Capturing the "Why" by Connecting Conversations with Computational Narratives. In *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*. ACM, 1–13. https://doi.org/10.1145/3313831.3376740

[58] Yifan Wu, Joseph M. Hellerstein, and Arvind Satyanarayan. 2020. B2: Bridging Code and Interactive Visualization in Computational Notebooks. In *UIST '20: The 33rd Annual ACM Symposium on User Interface Software and Technology, Virtual Event, USA, October 20-23, 2020*. ACM, 152–165. https://doi.org/10.1145/3379337.3415851

[59] Haijun Xia. 2020. Crosspower: Bridging Graphics and Linguistics. In *UIST '20: The 33rd Annual ACM Symposium on User Interface Software and Technology, Virtual Event, USA, October 20-23, 2020*. ACM, 722–734. https://doi.org/10.1145/3379337.3415845

[60] Haijun Xia, Jennifer Jacobs, and Maneesh Agrawala. 2020. Crosscast: Adding Visuals to Audio Travel Podcasts. In *UIST '20: The 33rd Annual ACM Symposium on User Interface Software and Technology, Virtual Event, USA, October 20-23, 2020*. ACM, 735–746. https://doi.org/10.1145/3379337.3415882

[61] Haijun Xia, Hui Xin Ng, Zhutian Chen, and James Hollan. 2022. Millions and Billions of Views: Understanding Popular Science and Knowledge Communication on Video-Sharing Platforms. In *Proceedings of the Ninth ACM Conference on Learning@ Scale*. 163–174.

[62] Haijun Xia, Nathalie Henry Riche, Fanny Chevalier, Bruno Rodrigues De Araújo, and Daniel Wigdor. 2018. DataInk: Direct and Creative Data-Oriented Drawing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*. ACM, 223. https://doi.org/10.1145/3173574.3173797