A Discourse-first Approach to Visual Narrative Generation through Operationalizing Comic Panel Transitions
Paper type: System Description

Chris Martens
Expressive Intelligence Studio
Computational Media Department
University of California, Santa Cruz
Santa Cruz, CA, USA
crmarten@ucsc.edu

Rogelio E. Cardona-Rivera
Liquid Narrative Group
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
recardon@ncsu.edu

Abstract

This abstract is so awesome.

Introduction

- What are we trying to do?
- What is our approach?
- Talk about how creative the discipline is
- Why are comics a great domain for computational creativity?

System Description

Our approach to generating visual narratives begins as a linear process that selects next comic panels based on the contents of previous panels, choosing randomly among indistinguishably-valid choices. The concepts we represent formally are *transitions*, *frames*, and *visual elements*, which we define below.

A **visual element (VE)** is a unique identifier from an infinite set, each of which is possible to map to a distinct visual representation. We do not explicitly tag visual elements as specifically characters, props, or scenery, making the representation agnostic to which of these narrative interpretations will apply. In the visual rendering of our comics, we represent VEs as random combinations of shape, color, and size, supplying additional inputs to the human cognitive processes that may interpret these elements' narrative role.

A **frame** is a panel template; at the abstract generation level, it includes an identifier or set of tags and a minimum number of required visual elements. The reason a frame specifies a *minimum* number of VEs is to allow for augmentation of the frame with pre-existing elements: for example, the *monologue* frame requires at least one visual element, indicating a single, central focal point, but other visual elements may be included as bystanding characters or scenery elements. At the rendering level, a frame includes instructions for where in the panel to place supplied visual elements. A **panel** is a frame instantiated by specific visual elements.

Finally, a **transition** is a specification for how a panel should be formed as the next panel in a sequence, which we describe formally below.

Transition types were first described by McCloud (McCloud 1993) as a means of analyzing comics. He gave an account of transitions including *moment-to-moment*, *subject-to-subject*, and *aspect-to-aspect*, referring to changes in temporal state, focal subjects, and spatial point-of-view. As Cohn (XXX cite ch 4 of visual lang of comics) points out, these transition types are highly contextual; they presume the reader has a semantic model of the "story world" in which the comic takes place. For the sake of computational generation, we derive a more *syntactic* notion of transition defined purely in terms of frames and (abstract) visual elements. So, for example, while McCloud could refer to an action-to-action transition as one where a character is depicted carrying out two distinct actions, we have no notion of *character* and *action*, so instead must refer to which visual elements appear and in which frame. The rendering of a frame itself may position VEs in such a way that a reader would read certain actions or meaning into it; however, this kind of reader interpretation is not modeled to inform generation.

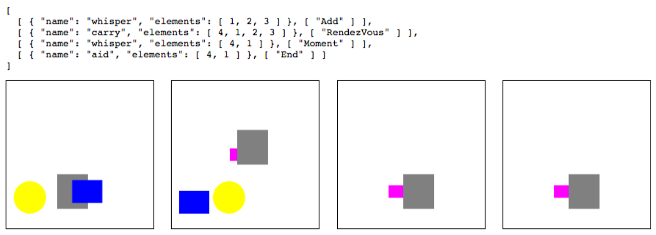
Formal Transition Types

We introduce six formal transition types: **moment**, **add**, **subtract**, **meanwhile**, and **rendez-vous**, each of which specifies how a next panel should be constructed given the prior sequence.

- **Moment** transitions retain the same set of VEs as the previous panel, changing only the frame.
- **Add** transitions introduce a VE that didn't appear in the previous panel, but might have appeared earlier (or might be completely new). A new frame may be selected.
- **Subtract** transitions remove a VE from the previous panel and potentially choose a new frame.
- **Meanwhile** transitions select a new frame and show *only* VEs that did not appear in the previous panel, potentially generating new VEs.
- **Rendez-vous** transitions select a random subset of previously-appearing VEs (from anywhere in the sequence) and selects a new frame to accommodate them.

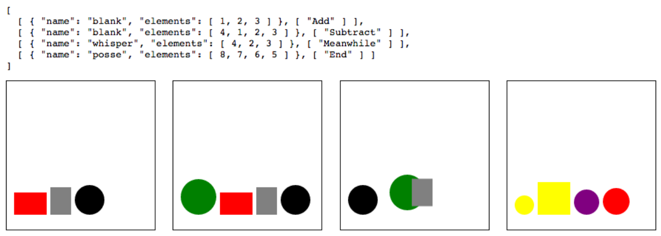
Implementation

Our generator accepts as inputs length constraints (minimum and maximum) and a number of VEs to start with in the



```
[
  [ { "name": "whisper", "elements": [ 1, 2, 3 ] }, [ "Add" ] ],
  [ { "name": "carry", "elements": [ 4, 1, 2, 3 ] }, [ "RendezVous" ] ],
  [ { "name": "whisper", "elements": [ 4, 1 ] }, [ "Moment" ] ],
  [ { "name": "aid", "elements": [ 4, 1 ] }, [ "End" ] ]
]
```

Figure 1: Example of output.



```
[
  [ { "name": "blank", "elements": [ 1, 2, 3 ] }, [ "Add" ] ],
  [ { "name": "blank", "elements": [ 4, 1, 2, 3 ] }, [ "Subtract" ] ],
  [ { "name": "whisper", "elements": [ 4, 2, 3 ] }, [ "Meanwhile" ] ],
  [ { "name": "posse", "elements": [ 8, 7, 6, 5 ] }, [ "End" ] ]
]
```

Figure 2: Example of underconstrained output.

first panel. Its output is a sequence of panels (frame names and VE sets) together with a record of the transitions that connect them.

The generation algorithm is:

- Generate transition sequence by choosing transitions uniformly at random, constrained by supplied minimum and maximum length
- Generate unique identifiers matching the number of specified starting VEs
- Feed transition sequence and starting VEs to panel sequencer, which selects a next frame and VE set for each new panel based on each transition type's definition (described above). Generate new VEs when necessary.

In addition to our OCaml implementation, which can be found on GitHub [1], we have implemented a web front-end in JavaScript [2]. This front-end assigns each frame type to a set of coordinates given by percentage of the vertical and horizontal panel size, then renders panels by placing visual elements at those coordinates. Visual elements are represented by randomly generated combinations of size, shape (circle or rectangle), and color. An example of the generator's output can be seen in Figure 1.

Constraining generation with Cohn grammars

Generating random transition sequences may result in nonsensical output, such as ending a comic with a "meanwhile" frame in which completely new visual elements are introduced at the end of the comic, but not connected back to previous elements; see Figure 2 for an example.

In an attempt to understand the global structure of comic panel sequences, Cohn et al. (XXX cite) investigate the *linguistic structure* of visual narratives. They claim that understandable comics follow a grammar that organizes its global structure. Instead of transition types, Cohn's grammar of comics consists of *roles* that each panel plays in the narrative. These roles are **establisher**, **initial**, **prolongation**, **peak**, and **release**, which allow the formation of standard narrative patterns including the Western "initial, peak, release" dramatic arc. Formally, Cohn gives the following grammar as a general template for sequences of panel roles:

(Establisher) – (Initial (Prolongation)) – Peak – (Release)

Symbols in parentheses are optional. In our expression of this grammar (and in several of Cohn's examples), we also assume that prolongations may occur arbitrarily many times in sequence.

Grossman built a generator based purely on Cohn's arc grammar,[3] picking hand-annotated panels for each of an *initial*, *peak*, and *release* slot in the comic. However, this generation scheme does not manipulate the internal structure of the comic panels, allowing for less variability in the output than our scheme with visual elements and frames. Additionally, codifying the syntactic structure of individual panels allows us to characterize *relatedness* between panels in the manner of Saraceni (XXX cite). In our second iteration of the generator, we combine two approaches to discourse, using *global* Cohn grammars to guide the *local* selection of syntactically-defined transitions.

In particular, we enumerate every possible role bigram in Cohn's grammar, such as *initial to prolongation*, *prolongation to peak*, and so on, and describe sets of transition types that could plausibly model the relationship. This mapping is determined by the code below:

```
let valid_transitions roles =
  match roles with
    (Establisher, Initial) -> [Moment; Subtract; Add; RendezVous]
  | (Establisher, Prolongation) -> [Moment; Subtract; Add]
  | (Establisher, Peak) -> [Add; Meanwhile]
  | (Initial, Prolongation) -> [Moment; Subtract; Add]
  | (Prolongation, Prolongation) -> [Moment; Subtract; Add]
  | (Prolongation, Peak) -> [Subtract; Add; RendezVous]
  | (Initial, Peak) -> [Subtract; Add; Meanwhile; RendezVous]
  | (Peak, Release) -> [Subtract; Add; RendezVous]
  | _ -> [End] (* error! *)
```

XXX describe randomly generating an instance of the grammar and feeding it to the otherwise-same generator

Examples of the constrained generator's output can be found in Figure 3.

Related Work

- Talk about Understanding Comics (McCloud 1993)
- Talk about Visual Language of Comics (Cohn 2013)

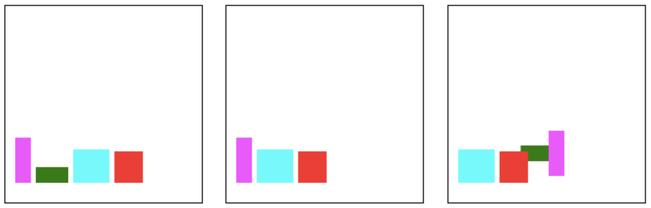
[1] http://www.github.com/chrisamaphone/comicgen

[2] http://www.cs.cmu.edu/~cmartens/comicgen

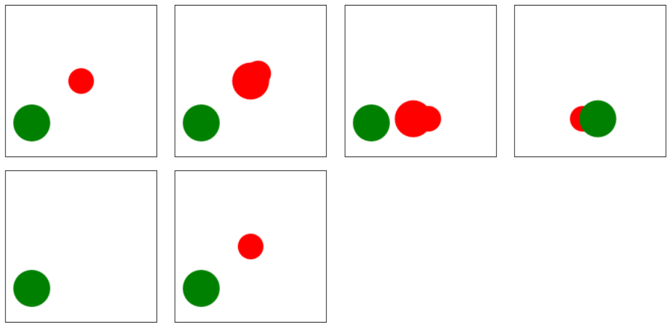
[3] http://www.suzigrossman.com/fineart/conceptual/Sunday_Comics_Scrambler

Figure 3: Example of grammatically-constrained output.

```
{
  "sequence": [ [ "Establisher" ], [ "Initial" ], [ "Peak" ] ],
  "comic": [
    [ { "name": "dialog", "elements": [ 1, 2, 3, 4 ] }, [ "Subtract" ] ],
    [ { "name": "dialog", "elements": [ 1, 3, 4 ] }, [ "Add" ] ],
    [ { "name": "aid", "elements": [ 2, 1, 3, 4 ] }, [ "End" ] ]
  ]
}
```



Another example:



- Talk about the MEXICA System (Pérez y Pérez and Sharples 2001) and how we're different
- Talk about the departure from traditional narrative generation work
  - Talk about the pipeline model of narrative generation (primarily simulation focused)
  - We're exploring an alternative account - focus on the telling of the story, let story consumers "fill in the gaps"

Historically, the computational generation of narrative has followed what Ronfard and Szilas (2014) term the *pipeline model*: a narrative artefact is computationally generated by first simulating the story world as a collection of events, and then piping the story world information to a discourse generator, which generates a selective presentation of story world events in a particular medium. Current work in the computational creativity community has primarily pursued this pipeline model for narrative generation. Work by **[RCR:** *There are several papers to cite here, but the gist is: pipeline model is pervasive.***]**

As Ronfard and Szilas argue, the pipeline model is neither *necessary* nor *sufficient* for the successful generation of narrative structure, but rather represents one paradigm of narrative generation. **[RCR:** *Present one example of why it's not necessary, and one example of why it's not sufficient.***]**

Our work presents a departure from the pipeline model, opting instead for a *discourse-first approach to narrative generation*. In this model, the story world is simulated inasmuch as is necessary to support the telling of story events in the discourse. The work we present here is a first step in this discourse-driven model, focused on understanding how the discourse of visual language narratives enforces constraints on the underlying story worlds they represent, and how these can further guide subsequent choices for discursive presentation.

Narrative authors design their narratives to affect audiences in specific ways (Bordwell), and authors care to structure discourse to achieve specific narratological effects. This is achievable in the pipeline model, but reasoning about discourse first helps you catch things that are not realizable in a specific target medium (e.g. identity for murderers in thrillers.)

Acknowledgments

These acknowledgments are tubular.

References

Cohn, N. 2013. *The Visual Language of Comics: Introduction to the Structure and Cognition of Sequential Images*. London, England, UK: Bloomsbury.

McCloud, S. 1993. *Understanding Comics: The Invisible Art*. New York, NY, USA: Harper Collins.

Pérez y Pérez, R., and Sharples, M. 2001. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence* 13(2):119–139.

Ronfard, R., and Szilas, N. 2014. Where story and media meet: computer generation of narrative discourse. In *Proceedings of the 5th Workshop on Computational Models of Narrative*, 164–176.