

Shared Mental Models in Improvisational Digital Characters

Brian Magerko, Peter Dohogne, and Daniel Fuller

School of Literature, Communication and Culture

Georgia Institute of Technology

{magerko, pdohogne3}@gatech.edu, gentristar@gmail.com

Abstract

Improvisational theatre is a unique art form that requires actors to co-construct stories on stage in real-time without the benefits of any explicit communication. All negotiation about the content of the scene, including characters, setting, plot, and relationships, must be done within the context of the performance. This negotiation process is a special form of constructing *shared mental models* between the performers as well as with the audience. This article explores the process of building shared mental models in improvisation and describes computational improv agents that employ this process in an interactive implementation of the improv game called *Party Quirks*.

Introduction

Improvisational theatre (improv) has been the subject of a handful of interactive narrative systems over the past two decades (Hayes-Roth et al. 1994; Bruce et al. 1999; Perlin and Goldberg 1996; Swartjes, Kruizinga, and Theune 2008; Harger 2008). The computational systems that have been developed typically focus on a specific aspect of improvisation teachings or practice. For example, Harger's (2008) system explores how to represent character status (i.e. how powerful or confident a character is) with virtual actors who walk out onto a stage. In an earlier system, Hayes-Roth et. al explored how status can effect the interactions between two virtual characters (1994).

The above computational approaches to improvisation have primarily based themselves on single phenomena described in seminal improv texts or concepts generally known by practitioners in improvisational theatre. This approach has yielded relatively shallow agents that can exhibit one particular aspect of improvisation; it has not produced larger, more complex agents capable of performing as an improvisational actor would. We aim to build computational representations of the formal understandings gained from studying human actors. Of particular note in our findings have been two broad categories of data: narrative development (how improvisers reason about and co-create stories on stage in real-time) and the construction of shared mental models (how improvisers reach a shared understanding of where the scene is going, what is true in the story world of the scene, etc.). For example, two actors

on stage during an experimental session established early on that a) they were in a national forest, b) they were both plumbers and friends, and c) one of them was unhappy with his life as a plumber. The progression of establishing these facts in the scene can be dissected into two parts. First, there is the story content, also called the *frame*. Second, there is the process through which the actors presented their characters, mutually agreed on who and where those characters were, offered any hints when the other was unclear what was being established, etc.

Sawyer refers to this process of establishing the frame as the process of *creative convergence* – the act of co-creation in a creative act (2003). Through the lens of studying problem solving in organizational psychology, this process can be thought of as the process of actors building *shared mental models* with each other and with the audience as they perform¹. Shared mental models involve a) individuals having their own model of the world, b) individuals having their own model of what is publicly known, and c) a process for reconciling unknowns or conflicts in their models (e.g. actor A thinks that they are in a movie theatre but actor B then states that they are in a baseball stadium). This article presents our work on studying shared mental models in human improvisers and the computational representation of the results of that study in agents that play an improv game called *Party Quirks* with human interactors.

Shared Mental Models in Improvisational Theatre

Misunderstandings and miscommunications are common in improv because coordination between improvisers is not an explicit act (i.e. improvisers do not directly communicate their intentions in a scene outside of what occurs in the performance on stage). The free-flowing, unscripted nature of improv makes all the more transparent the process of recognizing and resolving divergences in mental

¹ This is closely related to Clark and Schaefer's contribution model (1989) and Traum's grounding acts model (1999). The key differences between these works and ours is the performative nature of the domain of improvisational theatre as opposed to a narrow focus on the utterance level of human discourse.

models in order to achieve *cognitive consensus* (a state of agreement about some aspect of the scene) and create shared mental models among the improvisers. Some improv “games” (scenes that have specific rules for the improvisers to follow), such as “Party Quirks,” even have this mechanic (which we call “knowledge disparity”) built into the structure of their performance. In Party Quirks, one improviser plays the part of a party host to three other improvisers, all of whom are given specific character quirks known to everyone except the host. It is the goal of the host to infer the quirks of all three other improvisers from their behavior and interactions on stage. In other words, the host must deliberately seek out cognitive consensus with his fellow improvisers and vice versa.

Other improv games which do not deliberately disrupt cognitive consensus still often involve divergences between improvisers. Improvisers constantly have to communicate their internal understanding of the scene’s frame via their performance as a character as opposed to explicitly saying what their understanding of the frame is. We have constructed a model of these communicative acts (Fuller and Magerko 2010). This paper presents the improvisational agents we have built based on this model.

Ambiguity in Knowledge

The main reason cognitive divergences in improvisation occur is because actors’ communication of intention, knowledge, and goals on stage is imperfect and ambiguous; they do not coordinate entire scenes backstage nor do they perfectly know and communicate everything on stage. Improvisers often execute actions on stage that can be interpreted in a variety of fashions (e.g. starting a scene doing a raking action on the ground may lead to another actor coming on stage and commenting on how they are sweeping the floor, mopping, or even dancing – depending on their interpretation of the raking motion).

The communication and representation of ambiguous knowledge is a main feature of our current implementation of agents that can play as guests in *Party Quirks*. Guests in the game typically execute actions on stage that give hints to the party host about their quirk. A common strategy is to give hints that are very ambiguous and then to give more obvious hints over time, a strategy we call *reverse scaffolding*. Therefore, the agents must be able to a) reason about what kinds of actions their character may execute, and b) how ambiguous (or, conversely, iconic) those actions are in terms of communicating their character’s quirk.

Within this particular improv game, we view quirks as prototypical characters, such as “ninja” or “alien,” for simplicity (e.g. we do not handle quirks that attempt to blend characters or concepts together). Each prototype has a degree-of-membership (DOM) value for each of a list of *attributes*⁴. Attributes are characteristics of each persona,

such as “strength,” “attractiveness,” or “cleverness.” DOM values can run anywhere from 0 (no membership) to 1 (full membership).

Actions in the Virtual Stage

Attributes themselves describe character prototypes, but lend no information about how those descriptors are portrayed on stage. *Actions*, which are observable gestures, animations, and/or dialog that can be executed on our virtual stage, are associated with one or more attributes for a range of DOM values. For example, the action “hides behind things” is a member of the attributes *stealth*, *fearless*, and *immunity to projectiles* with DOM ranges 0.7-1.0, 0.0-0.4, and 0.0-0.3 respectively. This means any prototype with a DOM from 0.7 to 1.0 for *stealth* can hide behind things, as can any prototype with DOM from 0.0 to 0.4 for *fearless*.

If an agent wants to do something on stage to portray something about its prototype / quirk, it knows which actions it can execute by reasoning about a) their DOM values for attributes (i.e. “what is my prototype’s membership in each attribute set?”) and b) what actions map to those <DOM, attribute> pairs (i.e., “Given my attribute values, what actions are associated with those attribute values?”) However, some DOM values for attributes are very generic; the attribute “eats,” for example, has many prototypes with DOM around 0.5, which represents “eats an average amount.” This means that if an agent that has a common value for this attribute chooses to it, nothing will really be learned about it (i.e., knowing that a character eats normall provides little information about its prototype). In order to determine which actions are more unique to a given prototype, we introduced the concept of *DOM ambiguity*. In terms of attributes, the ambiguity of a given DOM value is a factor of the number of other prototypes with a similar DOM value (unique values are very characteristic) and how distant the value is from the “normal” value for that attribute (e.g. a value of 0.2 for “facial hair,” is fairly unique in our dataset, but is not very distant from the normal value, which is 0. However, only zombies have a high value for “eats_brains,” which means it is very unique / not ambiguous for portrayal). In terms of actions, ambiguity is a factor of the number of attributes the action can represent and how many prototypes can naturally execute that action.

Portraying Prototypes

The selection of which attribute to portray depends on which *portrayal technique* the character is assigned (either randomly or predetermined by setting internal variables) to use for this scene. There are several techniques we have observed human actors employing while playing Party Quirks and other similar games that involve knowledge disparity. Many actors reference the idea of “pacing” in a

⁴ This non-Boolean description of categorical knowledge is informed by contemporary views in cognition and category

theory, such as work by Lakoff (1987) and Rosch et. al (1999).

scene, which relates to making the scene “interesting to watch”. As such, actors often purposefully do not give obvious hints at the beginning of a scene, as that would cause the game to end too soon and not be interesting. Our agents represent this approach (called *reverse scaffolding*) by executing progressively less ambiguous (more iconic) actions over the course of the game. Another technique is *caricaturization*, which is when an actor is very obvious about their prototype. In our system, agents that use this technique present only the least ambiguous actions for the attributes characteristic to their prototype (i.e. being an obvious caricature of their prototype). Finally, actors sometimes use a technique called *opposing*. When actors oppose, they choose an attribute characteristic to their prototype and invert its value (e.g., someone who can fly but cannot control their movement). This aids in comedic effect and makes the scene more interesting as it conflicts with the normal model of the prototype.

Reaching Cognitive Convergence

Our implementation of Party Quirks guests has relied heavily on the computational representation of the process of reaching cognitive convergence. The guests’ goal in Party Quirks is to help host’s mental model match the prototype; therefore, the guests react to the host’s actions. Based on our observations of how improvisers communicate to build cognitive convergence, the first and most common host action is to *defer* (i.e. wait to see what happens next before) and let the guest present naturally. In response to a deferment, an agent picks which action to present based on the selected technique (reverse scaffolding, caricature, or opposing) and how far they are into the scene.

The next distinctive action hosts execute is to *guess* which quirk (i.e., prototype) the guest is representing. This is a type of verification, as detailed above. In response to a correct guess, the agent acknowledges the host’s success and leaves the game. In response to an incorrect guess, the agent indicates the host was wrong and refutes the guess by presenting an action from an attribute with a significantly different DOM value than the corresponding value from the prototype the host guessed. This demonstrates to the host a reason why their guess was incorrect while providing guidance in the right direction.

A more proactive technique a host can use to get information from a guest is to make a *blind offer*. In our representation, an offer is a prompt for an attribute, essentially asking “What is your value for this attribute?” In response to an offer, the agent responds with a presentation representing their value for that attribute.

Another type of blind offer involves the physical environment. Just as guests can assert something about the state of the environment, so can the host (e.g. “I’m turning off the lights”). Guests respond to this with an action that uses the new environmental state as a precondition if possible (“Don’t turn off the lights, I’m afraid of the dark!”).

While an offer allows the host to attempt to gather new information, a host can also try to *verify* their assumptions about the guest, which often happens when the host has a

specific guess about a guest’s quirk but still has uncertainties they desire to resolve before committing to the guess. Verifying involves stating assumptions about the guest’s value for an attribute. For example, if the host thinks the guest is a ninja, before they make a direct guess, they might say “I think you are very good with a sword.” In response to this statement, the agent responds with either a confirmation or a denial of the host’s assumption. Next, the guest makes a presentation for the attribute in question and then continues the scene as normal.

In some cases, the host may be unsure exactly what the agent was trying to demonstrate with a presentation. For example, if the action is “strikes a pose,” it might represent multiple attributes, such as fame or strength. In this case, the host asks, “Did you mean this attribute?” This is another type of verification in which the host is trying to clarify what the guest just presented. The agent will respond with either a confirmation or denial of the host’s guess as well as a different action for the same attribute, basically to say “Yes, that is what I meant, see?”

Finally, when the host is completely lost, they can make a generic *clarification request* and ask the guest to give more obvious clues. In response to such a request, the agent becomes less ambiguous with its presentations by narrowing the list of possible attributes for presentation selection.

In summation, by better understanding how human improvisers construct shared mental models, we have taken steps towards building computational actors that can employ similar processes. This is one major step towards creating improvisational actors that can interact with each other and with human users within an improv theatre framework.

References

- Clark, Herbert H, and Edward F Schaefer. 1989. Contributing to Discourse. *Cognitive Science* 13: 259--294.
- Fuller, D. and Magerko, B. 2010. Shared Mental Models in Improvisational Performance. *3rd Workshop on Intelligent Narrative Technologies at the Foundations of Digital Games*, Monterey, CA.
- Harger, Brenda. 2010. Project Improv. *Project Improv*. <http://www.etc.cmu.edu/projects/improv/>.
- Hayes-Roth, B., E. Sincoff, L. Brownston, R. Huard, and B. Lent. 1994. Directed Improvisation. In *Technical Report KSL-94-61*. Palo Alto, CA: Stanford.
- Lakoff, G. 1987. Cognitive models and prototype theory. In *Concepts and conceptual development: Ecological and intellectual factors in categorization*, ed. Eric Margolis and Stephen Laurence, 63–100.
- Rosch, E. 1999. Principles of categorization. In *Concepts: core readings*, ed. Eric Margolis and Stephen Laurence, 189–206.
- Traum, David. 1999. Computational Models of Grounding in Collaborative Systems. In *Working Papers of the AAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, 124-131. AAAI.