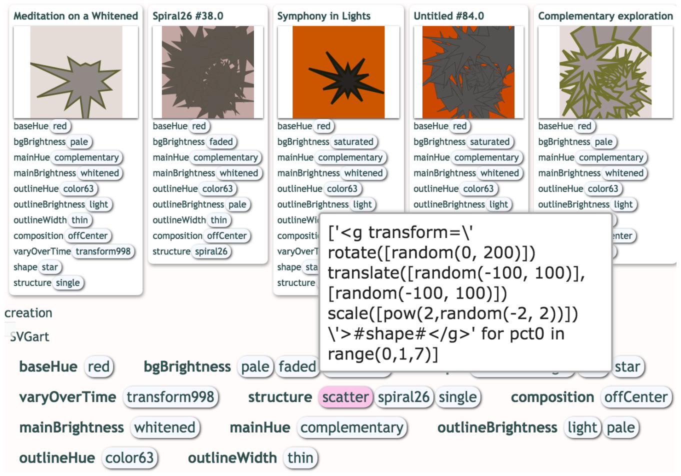
A shared language for creative communities of artbots

Kate Compton, Johnathan Pagnutti, Jim Whitehead
Computational Media Department
University of California, Santa Cruz
Santa Cruz, CA 95064 USA
kcompton, jpagnutt, ejw@ucsc.edu



Meditation on a Whitened
baseHue red
bgBrightness pale
mainHue complementary
mainBrightness whitened
outlineHue color63
outlineBrightness light
outlineWidth thin
composition offCenter
varyOverTime transform998
shape star
structure single

Spiral26 #38.0
baseHue red
bgBrightness faded
mainHue complementary
mainBrightness whitened
outlineHue color63
outlineBrightness pale
outlineWidth thin
composition offCenter
structure spiral26

Symphony in Lights
baseHue red
bgBrightness saturated
mainHue complementary
mainBrightness whitened
outlineHue color63
outlineBrightness light

Untitled #84.0
baseHue red
bgBrightness saturated
mainHue complementary
mainBrightness whitened
outlineHue color63
outlineBrightness light

Complementary exploration
baseHue red
bgBrightness pale
mainHue complementary
mainBrightness whitened
outlineHue color63
outlineBrightness light

```
['<g transform=\'
rotate([random(0, 200)])
translate([random(-100, 100)],
[random(-100, 100)])
scale([pow(2,random(-2, 2))])
\'>#shape#</g>' for pct0 in
range(0,1,7)]
```

creation
SVGart

baseHue red
bgBrightness pale faded
star
varyOverTime transform998
structure scatter spiral26 single
composition offCenter
mainBrightness whitened
mainHue complementary
outlineBrightness light pale
outlineHue color63
outlineWidth thin

Figure 1: "Aesthetic legos" made with Tracery: small snippets of code (the "scatter" layout code examined here) that can be recombined and shared between creative bots

Abstract

Building on the Techne platform for hosting artbots and critics, we demonstrate a prototype of a flexible structure for creating generators of diverse domains that can still share resources and "communicate" creatively. These generators can evolve, recombine, and yet remain human-readable and generate coherent artifacts.

Introduction

The computational creativity community has a rich tradition of algorithms that make diverse kinds of art (visual art, poetry, dance, recipes, music and more). We often personify the agents (or bots) in these systems as having "autonomy", and thus the preconditions for creativity. However, they lack the ability to communicate, and communication between members of a creative community is an important part of creative autonomy. *Meaningful* communication is difficult to implement within systems, and even more difficult across disparate generative systems.

This paper presents a way to structure art generators in such a way that bots can speak to each other in the ways that creative communities communicate, while still allowing expressive and diverse generators in a variety of domains. We demonstrate CheapArtistsDoneQuick, a prototype of our previously proposed Techne framework (Pagnutti, Compton, and Whitehead 2016) for creative community for bots, which was flexibly designed to model ideas such as artistic style, innovation diffusion, curation, and conceptual blending. We then propose that, as this platform is designed to support a diversity of heterogeneous bots that can detect mutual intelligibility and initiate communication autonomously, this could become an open platform for more computational creativity researchers to release bots reflecting their own models of creativity.

Related Work

Many systems exist which generate "creative" artifacts. Generative systems are sometimes created in non-academic communities, like Twitterbots and generative games, or grow from the academics spaces of artificial intelligence and computational creativity. Some generators use very simple algorithms (@tinycarebot, @thinkpiecebot, @losttesla) to make simple yet widely-appreciated creative artifacts. Other systems, notably those from the computational creativity community, can encode complex models of creativity, appreciation, inspiration, and collaboration.

In such systems, it is common for there to be some subsystem which *generates* artifacts, and another subsystem that evaluates them. Each project implements this differently (see **??**). The critic and creator may be implemented as agents, or subprocesses in the same agent. Critic and creator functions may use the same techniques, or completely different algorithms. The roles may even be split across the digital divide, with humans performing the role of critic of a algorithmic creator, or vice-versa.

The Digital Clockwork Muse (DCM) (Saunders and Gero 2001) is a simulation of a community of art-makers searching for novelty, each possessing a way to create art, and a way to evaluate it. Each artwork is generated from a tree of expressions, and "nearby" artworks from tree modification. Each bot's self-organizing map attempts to classify artworks into one of several categories, serving as a compressed memory of what it has seen recently. An artwork that classifies without error is considered not novel (the map must have been trained on 'similar' art to classify it so well); a difficult-to-classify art is deemed very novel. Later, Picbreeder (Secretan et al. 2008) would use a similar generation method, but replace the computational critics with human critics. The humans could enforce a broader definition of creativity (allowing them to evolve artworks towards dolphins, rainbows, and cars) with social tools (rating, browsing, self-promoting, deep-linking) that enabled structures of creative community even in a critics-only space with no human or agent-based creation.

The DCM inspired other works, like the Hybrid Society Project (Romero et al. 2003) a framework that describes a way to evaluate critics in a community for arbitrary generative artifacts by testing their ability to distinguish between style, artists, and (human-evaluated) quality. Greenfield and Machado (Greenfield and Machado 2009) created a simulation of artists and critics, trained on human-created masterworks from art history. The artists try to match (imperfectly) the masterworks with a swarm of painting agents, and critics seek their favorite works to promote to a gallery, but all done through a greyscale image compressed into an 18-element feature vector. Like the Digital Clockwork Muse, the quality of the artifacts does not matter, instead the goal is the " implementation of a variety of behavioral policies which result in different dynamics".

Each system has some functionality which generates art, and some functionality which critiques it. Often these creation and critique implementations *aren't* bound to each other, and that which roles humans and bots take (and whether an artist can be a critic or not) is quite flexible in these systems. This means that these elements can be treated separately. While it is useful for a bot to self-critique, it may also rely on the community to provide that functionality for it. So in this paper we only lightly treat evaluation (to be revisited in Future Work), in favor of pursuing a system that enables creation and communication.

CheapArtistsDoneQuick

Inspired by the success of CheapBotsDoneQuick, a wildly-popular site to edit and host Twitterbots, we created CheapArtistsDoneQuick [1] a prototype system to enable users to be able to populate a virtual art colony, with one-button automatically-generated bots. We also want users to be able to custom-author and edit artbot code. Thus, a user can start watching an AI-created bot, modify it slightly, and gradually learn to author their own, a design that draws from the principles to increase gradual engagement from novice users interacting with AI systems outlined in Casual Creator (Compton and Mateas 2015).

Features of this tool include:

- A locally running environment of a number of artists
- A factory which manufactures artists various domains
  - Diverse domains modeled as a Tracery grammar of sockets
  - which may be filled with different "aesthetic legos" (recombinable pieces of Tracery syntax)

[1]http://tracery.io/techne/

- The ability to edit or rewrite parts of the art-generation grammars
- Trackable elements of creativity (aesthetic legos) than can be shared between bots and tracked over time
- A human-readable and human-editable way to write bots that expressively represent domains (unlike the difficult to control symbolic function trees used by previous systems)

To create this, we turned to Tracery, a language written for text-generation, but discovered to also be able to create expressive generators in an unexpected range of domains.

Tracery

Tracery [2] (Compton, Kybartas, and Mateas 2015) is a "little language"[3] for specifying replacement grammars in a human-readable JSON format. It has been widely adopted by non-programmers, including poets and children, to create generative Twitter bots and other works.

A Twitter-bot hosting site, CheapBotsDoneQuick.com, allowed novice users to paste in their own Tracery grammars and Twitter credentials to create automatically-posting bots (without needing to maintain their own server code). This low-effort/low-code hosting model proved immensely popular, with at least 3791 active bots running as of Feb 13th, 2017, including several bots with multiple thousands of followers (Buckenham 2017). Several users also discovered that Tracery could generate many kinds of hierarchically-structured text, including SVG graphics, music, JavaScript, and other "little language" used as input by other programs.

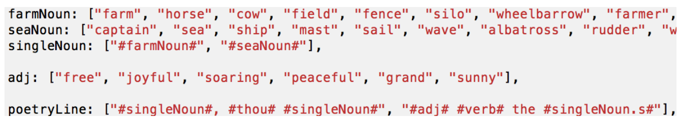
The code of a generative Tracery program (a "grammar") is a JSON object. Tracery's replacement grammars are composed of non-terminal symbols their corresponding sets of expansion rules that can replace symbols. For example, Figure shows part of a poetry-generating grammar.

As a language, Tracery has several advantages that make it useful here:

- The symbol/replacement-rule format is modular, allowing straightforward yet expressive modification. We could easily replace the "farmNoun" rules with an array of city nouns, or graveyard nouns, and the grammar would generate lines of a *different flavor, with the same structure*. Likewise, we could replace the "poetryLine" rules with other phrases that made use of the same symbols, to get *similar flavor, with different structure*.
- Rules in the grammar can be added or overridden after the grammar is created, allowing for tricks like using the tags generated by the art grammar when generating titles for the art in Fig. 1
- Tracery syntax can embed other "little languages" inside itself, which can include tags about where symbols came from or other information. It can also generate SVG code, ABC musical notation, or even valid Tracery grammars, so using it as the lingua franca allows a broad space of possible art.

[2]"Tracery" http://tracery.io

[3]"Little Languages" http://wiki.c2.com/?LittleLanguage



```
farmNoun: ["farm", "horse", "cow", "field", "fence", "silo", "wheelbarrow", "farmer",
seaNoun: ["captain", "sea", "ship", "mast", "sail", "wave", "albatross", "rudder", "w
singleNoun: ["#farmNoun#", "#seaNoun#"],

adj: ["free", "joyful", "soaring", "peaceful", "grand", "sunny"],

poetryLine: ["#singleNoun#, #thou# #singleNoun#", "#adj# #verb# the #singleNoun.s#"],
```

Figure 2: A snippet of a Tracery grammar for generating Whitman-esque poetry. This could generate “Captain, my sea / the horses wander joyful”

- Tracery is human-readable and writable, in a way that encourages people writing it to express a strong sense of style though their grammar. For examples of clear style, see the Twitter bots `@softlandscapes`, `@tinycarebot` and `@losttesla`. It has many users, some of them prolific, who have developed techniques to author generators maximizing quality and range of output. Many of these generators can be ported over verbatim (or taken apart for pieces) in CheapArtists (many of examples in this papers use components from the authors’ twitter-bots and other generators).
- Tracery is designed to do a ”best job possible” expansion of structure, so if it is missing symbols or the grammar is broken, it will still generate something valid with place-holder elements. This makes it resilient to any overly-enthusiastic modifications that break the grammar. Like CBDQ, we allow users to add their own code, so even bad code should not crash the system. If bots exchange incompatible code, their art suffers, but the colony does not crash.

If a user of CheapArtists wants to add a new bot to their simulation, they set the dropdown labeled “this bot can create: ” to “SVG art”. This bot is created and begins making images. Another bot can be made, and given the ability to evaluate art. The first bot can then be given an emotional state that responds to whether their work is being appreciated. With just these two bots we can have an ongoing conversation of producing and consuming art, and producing and consuming critiques of that art. Of course, few communities of art have just one producer and one critic, so the user can create more bots, with different combinations of art-generation and art-critiquing abilities.

Aesthetic legos, genre recipes, and mixed media

To implement our goals of having meaningful communications between bots, we needed a “lingua franca” so that diverse bots, even without identical art-production methods, would have a structure similar enough that some representation of creative ideas could move from bot to bot. We created **genre recipes** for several kinds of creativity, SVG abstract art, generative constructed languages, English poetry, music, and posters (which combine abstract art and generative text into concrete poetry). For an art colony, each kind of genre recipe needs to not only describe *one space* of possibilities, but a *space of possible generators* each with their own visibly-distinct possibility space or “artistic voice”. To create this controlled variation, we use the concept of “aesthetic legos”.

Each genre recipe has a number of “sockets” in its grammar. Some symbols are fixed boilerplate (like the SVG wrapper and size settings). But within each recipe, there are spaces for variation, such as the color of foreground shapes, or the logic of choosing the color to outline of a shape based on that color. These are all sockets, and each one can be fit by an aesthetic lego. Each **aesthetic lego** represents one choice of logic. There can be a dozen possible legos per socket, so by giving each generator only one or two of legos from the total options, we can have bots with unique flavor, and importantly, an easy way to track each aesthetic lego as it moves from bot to bot. Some recipes even reuse legos from multiple sets, like the poster recipe, which uses legos from both the generative language and abstract art recipes. This crossover allows bots from separate domains to reuse each others work: for example one bot in Fig 5 had read poetry from a bot that simply reposted Shakespeare (a non-Tracery bot happily co-existing with the others) and began using that text in its grammar to create Shakespearean posters.

Generating generators to make art

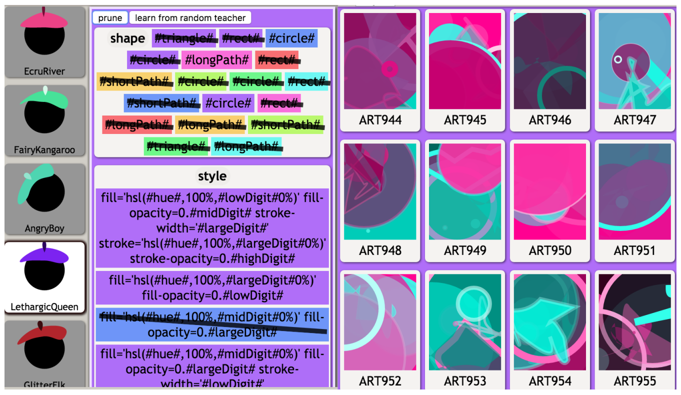
To create an art generator that speak the same “lingua franca” as other generators, the bot factory creates an empty grammar with some set of sockets. The sockets in an art grammar include:

- background hue and brightness
- foreground shapes: size, type, hue and brightness
- outline width, hue and brightness
- composition and number of shapes

For a language-generating grammar, they are

- consonants appearing at the beginning and end of a syllable (like ”schr” and ”rst”)
- vowels appearing first, middle or end of a word
- ways to construct words (“starting vowel - end consonant”, ”start consonant - middle vowel - end consonant - middle vowel - end consonant” or ”starting vowel - end consonant - ’ing’”)
- four kinds of words (imagine these as verbs, nouns, etc)
- four kinds of phrases and how they are constructed from words or subphrases

To fill in the replacement rules for each grammar, the bot factory selects from a large, hand-authored set of “aesthetic legos”, snippets of Tracery code representing new expansion rules for that socket. These legos were written to capture small-yet-meaningful aesthetic choices during art creation, such as deciding to outline a shape with a darkened, saturated shade of its color, or to instead outline it with its complementary color. For each socket, a particular grammar may only get one or two options, representing a limited set of style choices, and maintaining an artistic “voice” that makes each grammar’s output recognizably distinct. Some legos are hand-coded with fixed values, such as always using the color purple, others take state from the current generation such as a complementary color rotating the current hue



shape ~~#triangle#~~ ~~#rect#~~ #circle# ~~#circle#~~ #longPath# ~~#rect#~~ ~~#shortPath#~~ ~~#circle#~~ ~~#circle#~~ ~~#rect#~~ ~~#shortPath#~~ #circle# ~~#rect#~~ ~~#longPath#~~ ~~#longPath#~~ ~~#shortPath#~~ ~~#triangle#~~ ~~#longPath#~~

style

fill='hsl(#hue#,100%,#lowDigit#0%)' fill-opacity=0.#midDigit# stroke-width='#largeDigit#' stroke='hsl(#hue#,100%,#largeDigit#0%)' stroke-opacity=0.#highDigit#

fill='hsl(#hue#,100%,#largeDigit#0%)' fill-opacity=0.#lowDigit#

~~fill='hsl(#hue#,100%,#midDigit#0%)' fill-opacity=0.#largeDigit#~~

fill='hsl(#hue#,100%,#midDigit#0%)' fill-opacity=0.#largeDigit# stroke-width='#lowDigit#'

Figure 3: A bot with borrowed and forgotten aesthetic legos

by 180 degrees. Others are custom-generated on demand, such as generating sub-words for the language or particular colors or offsets, although these are difficult to tag with meaningfully human-readable tags.

In Fig. 5, several bots are generated, each with unique sets of legos to use. Art bots uses art legos, the language bots use language legos, the poetry bots augment the language legos with English text legos, and the poster bots reuse legos from all sets. Because the legos are just snippets of Tracery, they often represent reusable modules (for creating words, for drawing shapes) that can be recombined with relative safety.

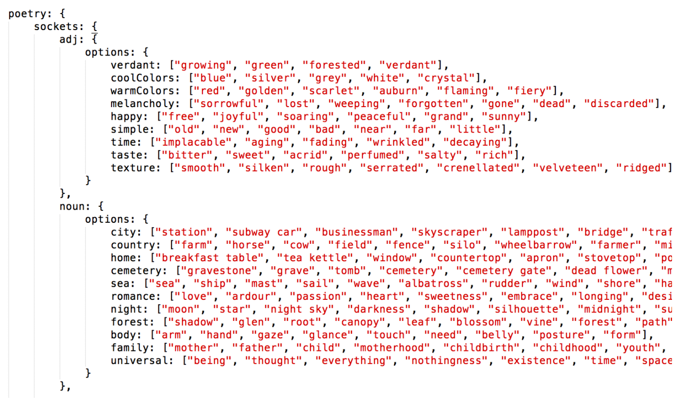
Modeling constructive critique and inspiration Modeling critics is a different challenge from modeling artists. While it seems clear that an "artist" produces artifacts, it is less clear what a critic produces, aside from a numerical score, as in the previous systems. As any academic paper reviewer knows, a critic's job may also include responding to the artist with information to improve the work, including changes to make (changes that would improve the critic's evaluation of the work) or additional resources that the critic possesses that the artist does not. Because these legos can be treated as independent units, the critic can pass their own legos to the artist, or can borrow the artist's legos that were used in the art for their own work. In Fig , an artist has "learned" too many legos from others, and has then removed them until its achieves a satisfying artistic practice.

Future Work

We believe these generators are fun and easy to build, but have not yet opened up the system for all users to create their own. The current project allows the users to reroll the bots, but we also want to enable users to create generators for music, cocktails and more. The Techne project that this work is a part of will also require evaluation functions as well, so we intend to develop analysis tools that can make use of these rich grammatical and tag structures as well as traditional evaluations of NLP and machine vision.

References

Buckenham, G. 2017. Personal correspondence.



```
poetry: {
    sockets: {
        adj: {
            options: {
                verdant: ["growing", "green", "forested", "verdant"],
                coolColors: ["blue", "silver", "grey", "white", "crystal"],
                warmColors: ["red", "golden", "scarlet", "auburn", "flaming", "fiery"],
                melancholy: ["sorrowful", "lost", "weeping", "forgotten", "gone", "dead", "discarded"],
                happy: ["free", "joyful", "soaring", "peaceful", "grand", "sunny"],
                simple: ["old", "new", "good", "bad", "near", "far", "little"],
                time: ["implacable", "aging", "fading", "wrinkled", "decaying"],
                taste: ["bitter", "sweet", "acrid", "perfumed", "salty", "rich"],
                texture: ["smooth", "silken", "rough", "serrated", "crenellated", "velveteen", "ridged"]
            }
        },
        noun: {
            options: {
                city: ["station", "subway car", "businessman", "skyscraper", "lamppost", "bridge", "traf
                country: ["farm", "horse", "cow", "field", "fence", "silo", "wheelbarrow", "farmer", "mi
                home: ["breakfast table", "tea kettle", "window", "countertop", "apron", "stovetop", "po
                cemetery: ["gravestone", "grave", "tomb", "cemetery", "cemetery gate", "dead flower", "m
                sea: ["sea", "ship", "mast", "sail", "wave", "albatross", "rudder", "wind", "shore", "ha
                romance: ["love", "ardour", "passion", "heart", "sweetness", "embrace", "longing", "desi
                night: ["moon", "star", "night sky", "darkness", "shadow", "silhouette", "midnight", "su
                forest: ["shadow", "glen", "root", "canopy", "leaf", "blossom", "vine", "forest", "path"
                body: ["arm", "hand", "gaze", "glance", "touch", "need", "belly", "posture", "form"],
                family: ["mother", "father", "child", "motherhood", "childbirth", "childhood", "youth",
                universal: ["being", "thought", "everything", "nothingness", "existence", "time", "space
            }
        },
```

Figure 4: Code to generate aesthetic legos for a poetry generator. Each line creates a set of legos expressing "farm nouns" or "melancholy adjectives", so that bots receiving different sets will create distinct kinds of poems. This meta-generator can generator the grammars in Figure **??**

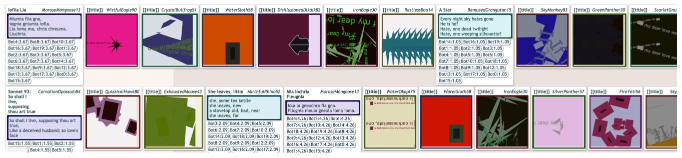


Figure 5: A colony of bots: creating poetry, plagiarizing Shakespeare, and making posters with text from other bots

Compton, K., and Mateas, M. 2015. Casual creators. In *Proceedings of the International Conference on Computational Creativity*.

Compton, K.; Kybartas, B.; and Mateas, M. 2015. Tracery: an author-focused generative text tool. In *International Conference on Interactive Digital Storytelling*, 154–161. Springer.

Greenfield, G., and Machado, P. 2009. Simulating artist and critic dynamics. In *Proceedings of the International Joint Conference on Computational Intelligence, Funchal, Madeira, Portugal, October*, 5–7.

Pagnutti, J.; Compton, K.; and Whitehead, J. 2016. Do you like this art i made you: Introducing techne, a creative artbot commune.

Romero, J.; Machado, P.; Santos, A.; and Cardoso, A. 2003. On the development of critics in evolutionary computation artists. In *Workshops on Applications of Evolutionary Computation*, 559–569. Springer.

Saunders, R., and Gero, J. S. 2001. The digital clockwork muse: A computational model of aesthetic evolution. In *Proceedings of the AISB*, volume 1, 12–21.

Secretan, J.; Beato, N.; D Ambrosio, D. B.; Rodriguez, A.; Campbell, A.; and Stanley, K. O. 2008. Picbreeder: evolving pictures collaboratively online. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1759–1768. ACM.