情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2011-EC-19 No.22
2011/3/27

文脈自由文法を利用した再帰的画像生成への時間軸の導入

添 田 洋 貴[†1]　丸 山 一 貴[†2]　寺 田 実[†1]

文脈自由文法を用いて再帰的な絵を作成するシステムとして Context Free Art が存在する．Context Free Art では，生成規則の右辺の記号それぞれに座標系や色の変換 (Shape Adjustments) を指定することにより，平面上の再帰的パターンをつくり出す．

本研究では，この変換に「遅れ」や「動き」といった時間的な指定 (Temporal Adjustments) を追加することで，動きのある再帰的な絵 (アニメーション) を生成することを可能とした．本稿では，拡張した変換の詳細，生成したパターン例などについて報告する．

Adding Motion to Context Free Art

Hiroki Soeda,[†1] Kazutaka Maruyama[†2]
and Minoru Terada[†1]

Context Free Art is a system which generates Computer Graphics using context-free grammar. For each of symbols in right hand side of productions, user append adjustments which specify changes in coordinates and color. The system renders recursive patterns on a 2D plane.

We add a new class of adjustments (Temporal Adjustments) which indicates motions and gradual color change for shapes. Our system generates various changing recursive patterns. We report the datails and implementations of temporal adjustments.

†1 電気通信大学 情報通信工学科
Department of Information and Communication Engineering, The University of Electro-Communications
†2 東京大学 情報基盤センター
Infomation Technology Center, The University of Tokyo

1. はじめに

文脈自由文法に基づき，平面上に再帰的な図形を描くシステムとして Context Free Art[1] (以下 CFA) が存在する．CFA では，終端記号として正三角形などの図形といくつかの曲線を用い，生成規則の右辺に通常の文脈自由文法と同様に非終端記号と終端記号の列の指定を行うことで描画を行う．

このシステムにおける特徴的な点は，右辺の記号のそれぞれに Shape Adjustments と呼ばれる指定を付加することにより，座標系の変換 (拡大，縮小，移動，アフィン変換) と色彩の変換 (色相，彩度，明度，透明度) のパラメータを変更することができるという点である．これによって，右辺の記号から生成される図形の位置，形や色が自由に指定が可能であり，二次元の図形を描画することができる．

本研究では CFA の Shape Adjustments の一種として新たに Temporal Adjustments を追加した．これにより表示図形に時間的な変化を指定できるようにし，動きのある再帰的な図形の作成が可能となった．なお，文脈自由文法としての生成規則そのものは時間による変化はなく，表示される終端記号のレンダリングにのみ動きが関連する．

2. 関連システム

2.1 Context Free Art

CFA を用いることによって，図 1 のような幾何学アートを描画することが出来る．

図 1 Context Free Art を用いて作成した画像の例
Fig. 1 Computer Graphics generated from Context Free Art

2.1.1 文脈自由文法[2](Context-Free Grammar)

文脈自由文法は CFA の基礎となる理論である．文脈自由文法は以下に与えられる生成規則を持った文法で定義される．

文脈自由文法：$G = (N, \Sigma, P, S) =$
(N：非終端アルファベット, Σ：終端アルファベット,
$P : A \to \alpha (A \in N, \alpha \in (\Sigma \cup N)*)$ の形をした生成規則の有限集合, S：開始記号 $(S \in N)$)

一例として $N = \{S\}$, $\Sigma = \{a, b\}$, $P = \{S \to aSb, S \to ab\}$ を与えた場合, $G = ab, aabb, aaabbb \ldots$ という文字群, すなわち $G = a^n b^n$ という言語を得る．

2.1.2 L-system[3]

L-system とは, フラクタル図形の一種であり, A. Lindenmayer によって導入された，生物の細胞分化の様子をモデルするものである．L-system は，開始記号に相当する axiom と枝の置き換えを表す生成規則からなり, 単純なルールで下図 2 のような図形が表現できる．本研究の基盤となる CFA では文脈自由文法を基に, L-system に則った動作を行う．

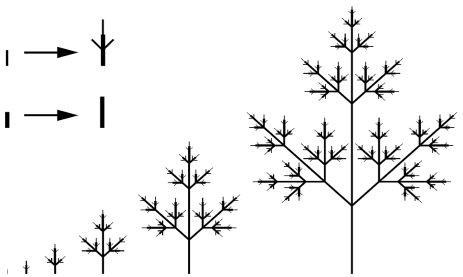

図 2 L-system の例:左上の正則規則を再帰的に適用することでフラクタル図形を得る[3]
Fig. 2 Developmental model of a compound leaf, modeled as a configuration of apices and internodes.

2.2 システム概要

CFA では, 文脈自由文法 G=(N, , P, S) において, N として具体的な終端図形 (円, 正方形, 正三角形など) が与えられる．さらに, 生成規則の右辺の記号の各々に Shape Adjustments を指定することができる．Shape Adjustments には, 座標系を変換する Geometry Adjustments と, 色彩を変換する Color Adjustments があり, 終端図形のレンダリングに影響を与える．Shape Adjustments については, 2.2.4 節で詳述する．

2.2.1 Startshape: 開始宣言

開始記号 S は, CFA において表 1 のように記述する．

表 1 startshape の記述方法
Table 1 How to write Startshape

記述方法	効果
startshape [ルール名]	開始する Rule を指定

2.2.2 Rule: 生成規則

非終端アルファベット N を左辺とする生成規則 P は CFA において表 2 のように記述する．

表 2 Rule の記述方法
Table 2 How to write Rule

記述方法	効果
rule [ルール名] [重み] { 記号と Shape Adjustments の列 }	Shape Adjustments を適用した 右辺の記号列を生成する

ここで, 同じ名前のルールを複数指定することで非決定的な生成を行なうことができる．実際には乱数によりいずれか一つが選ばれるが, その確率分布を「重み」により指定できる．

2.2.3 Shape: 図形指定

CFA において終端記号 Σ は描画できる図形を表し, 表 3 の 3 つの図形が利用できる．

表 3 Shape の記述方法
Table 3 How to write Shape

記述方法	効果
CIRCLE{ ShapeAdjustments }	円
SQUARE{ ShapeAdjustments }	正方形
TRIANGLE{ ShapeAdjustments }	正三角形

2.2.4 Shape Adjustments: 位置・形状・色彩指定

生成規則の右辺の記号に付加する Shape Adjustments には座標系の変換を指定する Geometry Adjustments と, 色彩の変換を指定する Color Adjustments がある． Geometry Adjustments を表 4, Color Adjustments を表 5 に示す．なお, ここでの z 軸は図形の重なりの上下を表わすものであり立体を意味するものではない．

表 4 Geometry Adjustments 一覧
Table 4 list of Geometry Adjustments

記述方法		効果
位置指定	x num1 y num2 z num3	x, y, z 軸における移動
	flip num	描画方向の num 度の転換
形状指定	size num1 num2	x,y 軸方向への拡大・縮小
	rotate num	図形を num 度回転
	skew num1 num2	x 軸, y 軸にそれぞれの歪みを加える

表 5 Color Adjustments 一覧
Table 5 list of Color Adjustments

記述方法		効果
色彩指定	hue num	色相を num 度変化
	saturation num	彩度を変化 ($-1 \leq num \leq 1$)
	brightness num	明度を変化 ($-1 \leq num \leq 1$)
	alpha num	透明度を変化 ($0 \leq num \leq 1$)

2.3 画像の描画例

これらをルールをもと CFA に沿ったコードを記述することにより, CFA では文脈自由文法を用いた画像を描画する. 一例として図 3 のようにコードを書いた場合, 非決定性により, 図 4 のように同一のコードで描画が行われるごとに異なる画像が生成される.

この例において, 1〜4 の 4 つの S というルールを設定した.

1. 色相 0, 彩度 1, 明度 1 の円を描いた後, 再度 S を描く（2 行目）
2. 色相 0, 彩度 1, 明度 1 の正三角形を描いた後, 再度 S を描く（4 行目）
3. 色相 0, 彩度 1, 明度 1 の正方形を描いた後, 再度 S を描く（6 行目）
4. 何も描かない（8 行目）

再帰的に呼び出される S には「角度が 30 度ずれ, x 軸に 1 移動した位置に, 大きさを 0.9 倍し色相を 30 度加えた状態で描く」という Adjustment が加えられている. また, それぞれに重みが付けられており, 1〜4 が描画される確率は 3/7, 2/7, 1/7, 1/7 となる.

なお, この例では 4 番目のように再帰を停止させる規則をいれたが, 停止条件を設定していない場合, CFA は描画すべき図形が十分小さくなったところで自動的に記号の生成を停止する.

```
1: startshape S
2: rule S 3 { CIRCLE{hue 0 sat 1 b 1}
3:            S{x 1 r 30 s 0.9 h 30} }
4: rule S 2 { TRIANGLE{hue 0 sat 1 b 1}
5:            S{x 1 r 30 s 0.9 h 30} }
6: rule S   { SQUARE{hue 0 sat 1 b 1}
7:            S{x 1 r 30 s 0.9 h 30} }
8: rule S   {}
```

図 3 図 4 を生成するコード
Fig. 3 Code to generate Fig.4

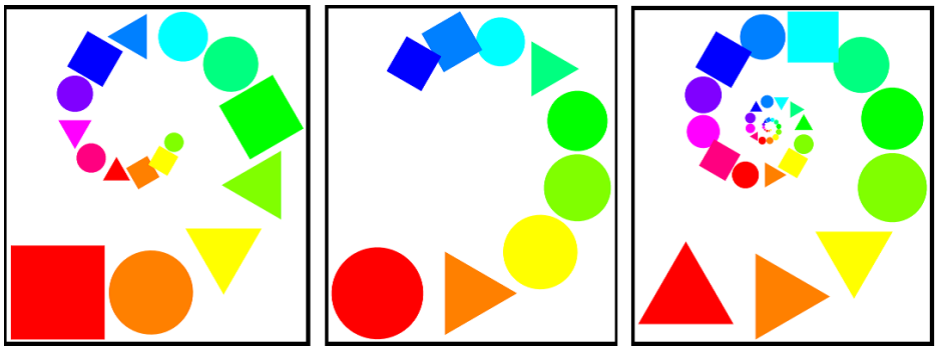

図 4 図 3 を基に得られる画像の例
Fig. 4 Computer Graphics generated from same coding

2.4 着 眼 点

CFA の利点として, 図 1 のような画像を数行の簡単なコードで CG の描画を行うことができる点があげられ, 作品には独自の面白さが存在する. 本研究ではその面白さを発展させる手法の一つとして時間軸の設定に着目し, 生成される図形に動きや色の時間的変化を与えることで, より興味深いパターンを生成することが可能ではないかと考えた.

Vol.2011-EC-19 No.22
2011/3/27

3. 提案システム

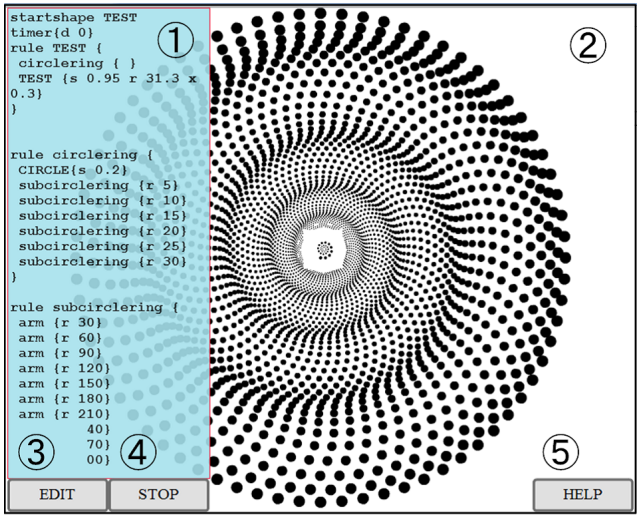

図 5 提案システムの画面構成
Fig. 5 Overview of proposal system

本提案システムは, CFA の Shape Adjustments の一つとして, 位置や色の時間的な変化を指定する Temporal Adjustments を導入した. 現状の Temporal Adjustments は生成規則の右辺にある終端記号にだけ付加できる. これによって, その終端記号は「動くパターン」としてレンダリングされることになる. また, 生成規則の適用に一定の遅延をはさむことによって, 生成の過程も動きとして捉えることができるようにした.

実装における言語として画像処理に強い ActionScript を用い, Flash Player[⋆1]上で動作するよう制作を行った.

提案システムでは次のような指定が可能である.

3.1 生成規則の適用間隔の制御

CFA では開始記号を出発点とし, 生成規則の再帰的な適用によって表れる終端記号, すなわち図形を順次描いていく. 時間軸の設定により描画間隔を 0.5 秒と設定, そのペースを表 6 のように記述することで描画間隔を制御し, 規則が適用されていく様子を見ることが可能となった.

表 6 timer の記述方法
Table 6 How to write timer

記述方法	効果
timer{ d num }	生成規則の適用の間隔を 初期設定時間の num 倍にする

3.2 Temporal Adjustments

Temporal Adjustments で終端記号に指定できる時間変化には, 一定時間をかけて一度だけ変化するものと, 周期的な変化を続けるものがある.

3.2.1 生成時点から一定時間変化を行うもの

表 7 は終端記号が描画された時点から一定時間アニメーションを行う引数である.

表 7 一定時間変化を行う Temporal Adjustments の記述方法
Table 7 How to write Temporal Adjustments changes during a period of time

記述方法		効果
色彩変化	ch num	色相を 10 秒かけて num 度追加
	ca num	透明度を 10 秒かけて num 倍 ($0 \leq num \leq 1$)
形状変化	sdw num	10 秒かけて num 度の位置に影を付加
位置変化	mvx num	10 秒かけて x 軸方向に num 移動
	mvy num	10 秒かけて y 軸方向に num 移動
時間変更	d num	上記の引数の変化時間を num 倍

3.2.2 生成時点から周期的な変化を行うもの

表 8 は終端記号が描画された時点から周期的にアニメーションを行う引数である.

表 8 周期的な変化を行う Shape Adjustments の記述方法
Table 8 How to write Temporal Adjustments changes cyclically

記述方法		効果
アニメーション	act num	num 秒かけて前後に動く
	spn num1 num2	num2 秒かけて num1 度の回転

⋆1 http://www.adobe.com/jp/products/flashplayer/ .

Vol.2011-EC-19 No.22
2011/3/27

4. 画 面 構 成

図 5 は提案システムの動作画面であり, ユーザインターフェースとして以下を用意した.

1 エディタ ··· CFA のコード記述を行う.
2 キャンバス ··· エディタの記述に基づいた描画結果を表示する.
3 EDIT ボタン ··· エディタの表示・非表示を行う
4 DRAW/STOP ボタン ··· 描画の途中停止と再描画を行う
5 HELP ボタン ··· 記述方法の一覧を表示する

5. 評　　価

著者を含む電気通信大学の学生 10 名が提案システムを使用し, アニメーション CG 制作を通してシステムに関する評価を行った.

5.1 システムの使用

被験者に提案システムを使用してもらい, 図 6,7,8 といったアニメーションを行う CG を得た他, 機能面での問題点や使用感について様々な意見を得た.

5.2 システムの評価

被験者から得られた代表的な意見は以下の通りである.

良い点

- コードを少し変えるだけで動作に大きな変化が得られて面白い
- 描画の様子が見ることが出来るだけでも面白い
- 簡単に CG が制作できるのは良い
- 少し時間変化を加えるだけで違った CG が得られる
- 複数の機能の組み合わせで面白い変化が得られる

悪い点

- 各引数に対し対応するパラメータの範囲がわかりにくい
- 多くの図形を描こうとすると処理が遅くなる
- カンマなどで引数に区切りをつけたい

良い点として, 追加した Temporal Adjustments は CFA に則り, 数文字の記述でアニメーションを行うことが出来るため, CFA 本来の面白さを損なうことなく画像を生成することが出来たという点が挙げられ, 時間変化によるアニメーションに関しても高い評価を得ることが出来た. また, システムの使いにくさの面からいくつか今後の課題点が挙げられた.

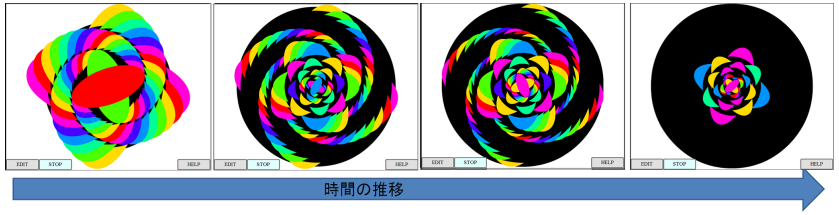

図 6 被験者 1 が生成した CG の例
Fig. 6 Computer Graphics generated by Subject 1

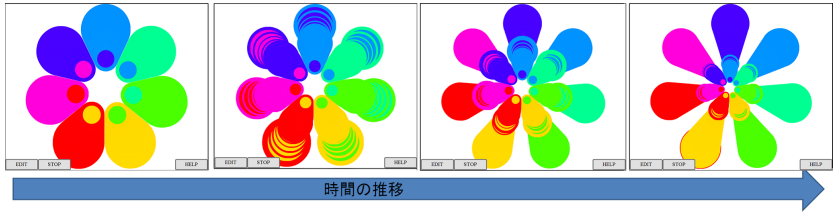

図 7 被験者 2 が生成した CG の例
Fig. 7 Computer Graphics generated by Subject 2

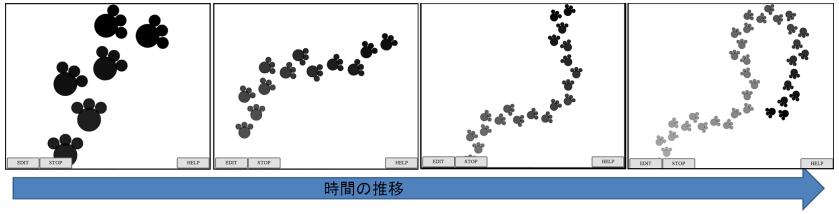

図 8 被験者 3 が生成した CG の例
Fig. 8 Computer Graphics generated by Subject 3

6. 関 連 研 究

本研究と同様に画像に時間軸を設定する研究やアニメーションを生成する研究は複数行われており, その中で本研究の関連として時間軸の設定という観点から櫻井による Sequential Graphics, 簡単なコーディングでのアニメーションとして原田による Viscuit を提示する.

6.1 Sequential Graphics[4)]

Sequential Graphics[4)] は画家が絵画を描く際の心理的な働きに着目し, 線を描く際の手の動きの情報を保持することによって, 図 9 のような描画時の臨場感を再現するペイントソフトであり, 描画ストロークの時間変化を持つことから絵を描く様子をそのまま再現できる.

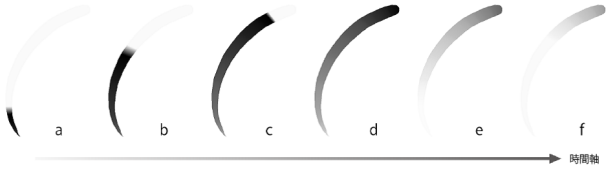

図 9 Sequential Graphics における一筆の時間変化[4)]
Fig. 9 Temporal variation in Sequential Graphics

6.2 Viscuit[5)]

Viscuit[5)] とは, 原田により制作されたビジュアルプログラミング言語であり, 柔らかい書き換えという原理に基づいて図形の配置を変更することで, アニメーションを生成する. プログラミングと実行モードが無いなどなるべく使用者への敷居を低くすることで子供でも簡単にプログラミングに触れることが出来ることを目的として開発された.

図 10 Viscuit を用いて実際にアニメーションを生成した例[6)]
Fig. 10 Animation generated from Viscuit

7. 終わりに

本研究では CFA に時間軸の導入を行い, Shape Adjustments に新しく Temporal Adjustments を設定することでアニメーションを実現した. 評価の結果, 時間変化を行うアニメーションに関して高い評価を得ることができた一方, 処理が遅くなる「引数のパラメータの範囲が分かりにくい」といった意見のように, 実際にシステムを使用するにあたっての使用感の悪さについての意見が目立った.

これらの意見を基として, システムにおける処理の高速化とユーザを意識したシステムの設計により, システムの使用感を改良することが今後の課題として挙げられる.

また今回の提案システムでは Temporal Adjustments を終端記号である図形に適用することでアニメーションを行ったが, 時間軸を活用した他の手法として

- 非終端記号に Temporal Adjustments を適用することで再帰的にアニメーションが適用されるような機能を実装する
- 関連研究の Sequential Graphics のような描画手法を用いることで, 終端記号となる図形そのものが時間経過で描画される

など, 本研究とは別の手法が考えられ, 別の観点からの時間軸の活用とアニメーション効果の発展を望むことができる.

これらの課題と方向性を踏まえ, 今後新たな手法での発展を目指していきたい.

参 考 文 献

1) Context Free Art: http://www.contextfreeart.org/ .
2) R. モル, 他.:『情報科学 形式言語理論入門』, 東京電機大学出版局, pp10-31, 1995.
3) P.Prusinkiewicz, et al.: L-systems:from the theory to visual models of plants, Proc. of the 2nd CSIRO Symposium on Computational Challanges in Life Sciences. CSIRO Publishing, 1996.
4) 櫻井 稔, 江渡 浩一: Sequential Graphics: 描画時の臨場感を再現するペイントソフト, pp29-34, WISS 2008.
5) 原田康徳, 加藤美由紀, Richard Potter: Viscuit:柔軟な動作をするビジュアル言語, pp41-48, WISS 2003.
6) Viscuit ホームページ: http://www.viscuit.com/ .