プロシージャル技術とデザイン

Procedural Technique and Design

宮田 一乘
北陸先端科学技術大学院大学

Kazunori Miyata
Japan Advanced Institute of Science and Technology

１．はじめに

プロシージャル技術とデザインというタイトルから真っ先に想起されたのが，ブルーノムナーリの「木をかこう」という絵本であった．子供向けのワークショップでもよく取り上げられる素材であり，図のように2本に枝分かれする Y の字のパターンを組み合わせて，シンプルなルールから複雑な絵を描くことができる．プロシージャルな手法の可能性や発展性をわかりやすく伝えることができる良書である．

一方，画面の高精細化にともない，映像コンテンツの制作作業は大規模化している．特にゲーム業界では，高品質な動画を多用することもあり，コンテンツ制作に映画制作並みの人員を数えることも珍しいことではない．現状では，コンテンツ制作の多くは手作業で行われているが，人的リソースの制約から一層の作業の効率化が求められている．

本稿では，このような要求への解決法の一つになりうるプロシージャル技術について解説するとともに，デザイン分野への適用の可能性を探る．

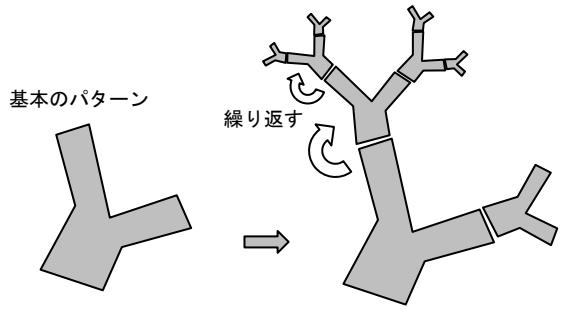


図１「木をかこう」による木のデザイン

２．プロシージャル技術について

プロシージャル(procedural)には“手続きの”という意味があるように，CG におけるプロシージャル技術とは，何らかのアルゴリズムで形状や模様などをコンピュータで自動生成する手法を指す．

2.1 プロシージャル技術とは

プロシージャル技術は，以下に示すようなコンテンツの制作支援的な特徴を持ち合わせている．

(1) コンテンツの圧縮:コンテンツをプログラムで表現するため,固定的で膨大なコンテンツデータを制作し蓄積しておかなくても,必要な時に求められた大きさのデータを生成できる．すなわち，コンテンツの圧縮に相当する．

(2) 多くのバリエーション:パラメータの変更で，手間をかけずに多様なコンテンツを得ることができる.アルゴリズム次第では，ほぼ無限のコンテンツを生成できる．

(3) 手間の削減:パラメータを調整するだけで，手間のかかる制作作業をコンピュータに任せることができる．

(4) 詳細度の制御:必要な画面の大きさや解像度に合わせて出力するコンテンツの詳細度を自在に制御でき,適切な粒度での制作が可能である．

(5) コンテンツの再利用:別な応用場面でのコンテンツの再利用時に，柔軟な対応が可能である．すなわち，再利用の場面に合わせてパラメータを調整し,新たなコンテンツを制作することができる．

(6) 超絶的な描画能力:人の手では描けないような微細な形状や膨大な量のコンテンツを,コンピュータで自動的に生成することが可能である．また，描画スキルなどに左右されず，均質かつ良質なコンテンツを得ることができる．

これらはプロシージャル技術の利点であるが，以下に示すような課題が残されている．

(1) 設計の困難さ:アーティストやデザイナが直接デザインするコンテンツとは異なり，開発者が使用場面を理解し，そのコンテンツの特徴をアルゴリズムとして設計･実装する必要がある．

(2) 試行の煩雑さ:実装したアルゴリズムが希望のコンテンツを生成可能かどうかの検証には,その都度プログラムを実行し確認する必要がある．

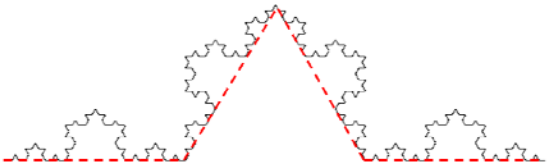
(3) 時間がかかる：コンテンツの生成には計算時間を要する．生成に時間がかかるようだと,プロシージャル技術を用いるメリットも小さくなる．

2.2 フラクタルとプロシージャル技術

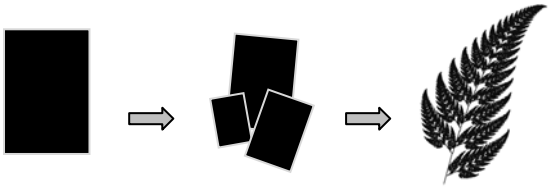
プロシージャル技術の解説に不可欠なものとしてフラクタルがある．フラクタル(fractal)とは，マンデルブロ(Benoit Mandelbrot)が創出した幾何学の概念[4]である．

フラクタル図形には,図1に示すような全体の形状が細部にも繰り返し現れるという自己相似性と呼ぶ特徴がある．図1(a)に示すコッホ曲線の例では，三等分した線分の中央に対し正三角形を成すように変形する操作を繰り返すことで図形を描画して

いる．また，図1(b)の例では，IFS(Iterated Function System)[5]という考え方で，画像の縮小写像を繰り返す変換を適用することで図形を生成する．



(a) コッホ曲線



(b) シダ

図 1 フラクタル図形

このように人手で描くのは躊躇するような複雑な図形も，プロシージャルな手法で反復計算を行うことで生成することが可能である．反復計算によるコンテンツ制作はコンピュータの特性を生かしたものであり，山岳形状の自動生成や大理石などのテクスチャ生成を発端に，CG の分野でフラクタル手法は注目を浴びるようになる．

2.3 スクリプト言語とプロシージャル技術

プロシージャル技術による生成アルゴリズムのデザインには，スクリプト言語を用いることが多い．1989年に Pixar 社が開発した RenderMan Shading Language[1]では，テクスチャを手続き的に記述可能なスクリプト言語が提供された．

一方，代表的な3DCG 制作ソフトでは，マルチウィンドウによるシーンのデザインを対話的に行うことが多い．これに対し，例えば CG 制作のフリーソフトウェアである Pov-Ray では，図2に示すようなスクリプトを記述することでシーンをデザインし，CG 画像を生成することができる．また，Generative Modeling Language(GML)[3]では，3D モデルの基本形状を関数で操作・組み合わせする過程をスクリプトで記述することで，多様かつ複雑な形状を構築できる．

このように，対話的な CG 制作手法とスクリプトによる制作手法と対比させると，プロシージャル技術の特徴が理解しやすいだろう．スクリプトによる制作法では，データ量の少ないテキストでシーンを記述でき，スクリプト内のパラメータ変更でバリエーションが増やせ，かつ，スクリプトを工夫することで手作業では困難な複雑な形状を表現することができる．また，スクリプトの再利用も可能である．すなわち，CG デザイナの位置付けが，対話的な手法とスクリプトによるプロシージャルな手法とでは大きく異なり，後者の手法は，例えるならば，「絵を描くソフトウェアロボット」を設計するような手法であると考えられる．

```
#include “colors.inc”
#include “textures.inc”

camera { location < -4.0, 2, -3.0 >
          look_at   < 0.0, 0.0, 0.0 >
          angle 60
        }

light_source { < 20, 30, -25 > color White }

sphere { < 0, 0, 0 >, 1
         pigment { Green }
         finish { Shiny }
       }

box { < -2, -1.2, -2 >, < 2, -1, 2 >
      texture { Red_Marble }
      finish { ambient 0.2
               diffuse 0.8
               phong 1
             }
    }

background { color Gray50 }
```

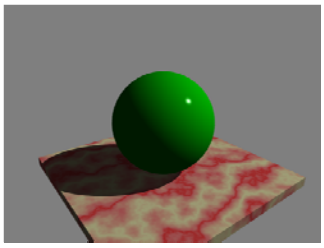


図 2 Pov-Ray でのシーン記述と生成例

3．プロシージャル技術による CG 表現

自然界にはフラクタル的な特性を持つものが多く，山や雲，植物の形状や，河川の分岐，銀河の星の分布などは，フラクタル手法で近似表現が可能である．本章では，フラクタル手法を中心に CG 表現におけるプロシージャル技術を紹介する．

3.1 山岳の表現

フラクタルが CG 界で脚光を浴びたきっかけは，マンデルブロらが作った山岳形状である[6]．これは，山の起伏をフラクタル関数で表現するものであり，2次元の各格子における山の高さ情報を，fBm(fractional Brownian motion)関数で求めている．すなわち，ブラウン運動のような複雑な軌道を山の垂直断面として与え，山岳形状を表現する．この手法により，起伏の激しい山はゴツゴツした岩肌で覆われ，なだらかな山は滑らかな山肌を持つような自然な山岳形状を，プロシージャルに生成することが可能になった．

一方，図3に示すように，三角形の三辺の各中点を，ある高さだけ持ち上げて折り曲げる操作を繰り返すことでも，山肌に似た複雑な形状を自動生成することが可能である[7]．これにより生成した山岳形状の例を図4に示す．

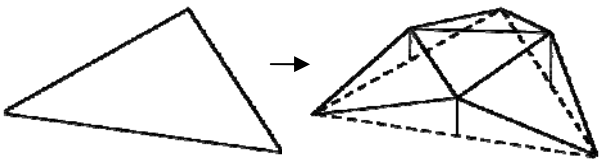


図 3 三角形の再帰分割による山肌の生成

図 4 フラクタル手法による山岳形状の表現例

このようなプロシージャルな地形データ生成の代表的なソフトウェアとして，Planetside Software 社の「Terragen」[8]がある．GUI ベースで地形をオーサリング可能なこのソフトウェアでは，ノイズや山の起伏などのパラメータを調整することで多様な地形を生成できる．同ソフトでは，水面や大気の効果などの付加も可能であり，統合的な景観 CG ソフトとなっている．図5にその生成例を挙げる．

図 5 Terragen による地形の生成例

3.2 雲の表現

一番簡単な雲の表現法としては，ノイズ関数の値を雲の厚みと仮定して，濃淡表現する手法である．すなわち，図6に示すように，山の高さを雲の厚みに見立てる考え方である．ただし，この例のように平面的な雲しか表現できない.図4の背景の雲も同様の手法で表現したものである．

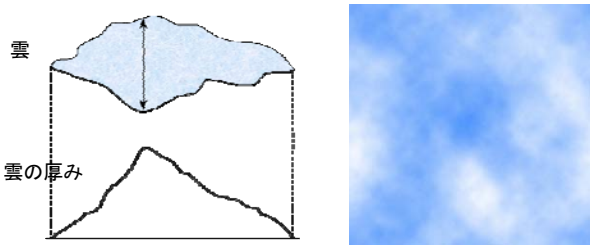


図 6 ノイズ関数による雲の表現

Grumman 社の Gardner は，二次曲面からなる3D オブジェクトに雲テクスチャを適用し,より立体的な雲の高速な表現を試み，同社のフライトシミュレータに適用した[9]．さらに，Ebert はフラクタルノイズ関数に陰関数を組み合わせた手法で，雲のボリュームレンダリング[1]を実装し，量感のある雲の表現を行った[10]．以降，雲に限らず煙や炎のようなファジーな物体の表現が多数試みられてきた．これらの表現に対しては，乱流方程式を近似的に解くような物理シミュレーションの方向に進化してきており，プロシージャル技術とは異なったアプローチとなっている．

3.3 植物の表現

植物のプロシージャルな表現は，植物学者 Lindenmayer が提唱した形式文法 L-system を緒とする．L-system は，植物の成長プロセスを記号で表現することが可能であり，再帰的な手続きで生成される記号列を幾何形状に翻訳することで，植物の複雑な形状を自動生成する[14,15]．

L-system では，初期値と変換ルールを記号で表現する．例えば，初期値の記号列を AB，変換ルールとして，A→xB(A を xB に置換)，B→yA の2つを定義する．この場合，記号列は，AB → xByA → xyAyxB → xyxByxyA と順次変換される．

これに，記号[(状況をスタックに積む)，](状況をスタックから取る)の2つの記号を加え，[と]で囲まれたところが分岐した小枝に相当する枝分岐を考慮すると，植物のような形状を生成することができる．例として，初期値 F，変換ルールとして，F → FF-[-F+F+F]+[+F-F-F]を定義し，+は時計回りにある角度だけ枝を旋回，-は反時計回りの旋回とする．そして，このルールで生成された文字列のＦを線分として翻訳すると，図7(a)に示すような形状が生成される．

このままでは，結果は必ず同じ形状となり，植物の個体差を表現することができない.同一種としての類似性を保持しつつ，多様性を表現するために，Stochastic(確率論的) L-system が考案された．これは，変換ルールを複数定義し，適用するルールをその都度ランダム選択することで，生成形状に揺らぎを与えるものである．例えば，P1,P2，P3の3つの変換ルールがあった場合，それぞれを1/3の確率でランダムに選択する．これにより，同じような枝ぶりの多様な植物を生成することが可能である．

以上の手法を3次元に拡張すれば,立体的な植物モデルを生成できる.さらに,植物の成長ホルモンを考慮した花序の表現[16]や，隣接する植物間の勢力分布をシミュレーションした植生の表現[17]なども可能である．

植物のプロシージャル生成に関する商用ミドルウェアとして，IDV 社の「SpeedTree」[18]がある．SpeedTree ではゲームへの適用を考慮し，多ポリゴンから低ポリゴン，さらには2D ベースのビルボードテクスチャへの出力というように,LOD(Level of Detail，詳細度)制御に対応している．

1 量で定義された物体の表現法

(a) (b) (c)

図 7 L-system による植物の表現例

3.4 テクスチャの表現

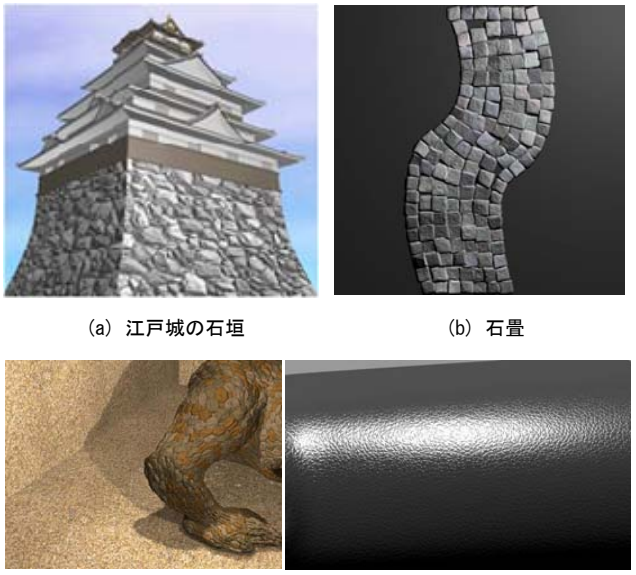
ゲームコンテンツをはじめとする3次元 CG の制作における作業負荷の高いものの一つとして，テクスチャ素材の作成が挙げられる．テクスチャ素材としては，写真などの素材集やデザイナが描画したもの以外に，プロシージャルに生成したものを用いている．

プロシージャル手法でテクスチャを生成する際に重要な役割を果たすのがノイズ関数である．ノイズ関数の代表例としては，ホワイトノイズ，ピンクノイズ，ブラウニアン(fBm)ノイズの3つが挙げられる．ホワイトノイズは全ての周波数で強度が均等となるノイズで，非常に不規則な印象を与える．ピンクノイズは1/f 揺らぎとも呼ばれ，周波数と強度が反比例する特性を持つ．ブラウニアンノイズはブラウン運動の軌跡のようなノイズであり，強度が周波数の2乗に反比例する．ピンクノイズとブラウニアンノイズは高周波成分と比較して低周波成分が強く，大きな揺らぎの中に細かい揺らぎが乗るようなノイズであり，人に自然な印象を与える．これらのノイズ関数を用いて色づけを行うと，図8に示すような木肌や迷彩模様が得られる[19]．

(a) 樹皮 (b) 迷彩模様

図 8 ノイズ関数によるテクスチャの例

ノイズ関数として CG コンテンツ制作で多用されているのが，パーリン(Perlin)ノイズである．パーリンノイズは，2次元で考えた場合，ある点における値をその周囲の値に対して重み付け補完する操作を繰り返して求め，自然な印象のノイズを生成する手法である[2][20]．また，パーリンノイズを用いて形状とテクスチャを定義する空間充填関数により，火の玉や岩肌のアーチ，毛の塊などの複雑な3次元モデルを，プロシージャルに生成する手法[21]も提案されている．ノイズ関数による雲や雪などの自然な背景の模様付けはゲームのシーンにも多用されている．ノイズ関数を用いて表面の凹凸付けを行い，さまざまなテクスチャ[22,23,24,25]を表現した例を図9に挙げる．



(a) 江戸城の石垣 (b) 石畳

(c) 恐竜のうろこ (d) 革シボ

図 9 凹凸付けされたさまざまなテクスチャの例

一方，シミュレーション的なアプローチで模様付けを行う手法として，Reaction-Diffusion(反応拡散)モデルが提案されている[26,27]．代表的なモデルとしては，アラン・チューリング(Alan Turing)による「チューリング反応拡散方程式」が挙げられ，2つの化学物質が互いに拡散と反応を繰り返す過程をシミュレーションすることで，有機的かつ複雑な模様を生成することが可能である．このモデルを用いることで，キリンやシマウマなどの動物の毛並みや興味深い模様を効果的に生成できる．その反面，パラメータの制御が難しく，結果は実行し終わるまでわからないという問題点があるが，一種の発見的な要素も含まれており，予想もつかない模様を生成できるという大きなメリットはある．反応拡散モデルによる興味深い模様生成は，参考文献に挙げたサイト[28,29]を参照されたい．

また，「Cellular Texture Generation」というパーティクルシミュレーションを応用したプロシージャルなテクスチャ生成法[30]もある．これは，対象の3D モデルの表面に“振る舞い”を定義したパーティクルを多数配置し，それらの相互作用をシミュレーションすることで模様を生成する．パーティクルには，”ある方向に並ぶ”，”近くのものと結合する”などの振る舞いが割り当てられており，最終的には各パーティクルを3次元の幾何形状に変換し，鱗や棘状の立体的なテクスチャを表面に発生する．

3.5 都市景観の表現

[2] パーリンノイズの考案者である Ken Perlin は，この業績により 1997 年のアカデミー技術賞を受賞している．

ゲーム開発や映像制作では，主役となるキャラクタのデザインばかりでなく,背景となる都市景観のデザインも必須である．都市景観は作品の雰囲気作りを左右する重要な要素であるが，作業量は膨大になる．こだわりが必要な場合は手作業による作りこみは不可欠であるが，ゲームの世界では，背景物として大局的な街並みが表現できれば十分であるケースが多い.そこで，道路網や林立するビル群を，L-system をベースにプロシージャルに生成する研究が，この数年間精力的に行われている．例えば，ハイトマップによる陸地と，水域ならびに人口密度の分布をマップデータとして入力することで，地形と人(車)の移動を考慮した道路網の自動生成が可能である[31]．また，建物の窓や屋根の付き方などをルール化し，膨大な量の建築物を自動生成する手法も提案されている[32,33]．

これらの手法は「CityEngine」[34]というプロシージャルな都市景観生成のミドルウェアとして統合化されており，同成果による古代ローマの街並みを再現する[35]までに至った．

3.6 さまざまな現象の表現

プロシージャル技術は，以上述べた手法以外にも，さまざまな現象の表現にも適用されている．

例えば，巻貝をある軸周りに螺旋状に成長させ，さらに貝表面に反応拡散テクスチャで模様付けを行い，リアルな貝を表現する手法が提案されている[36]．このようなプロシージャルな成長モデルによる有機的なモデリング手法は，河口も古くから提案し[37]，多数の作品を発表している．

また,経年変化(エージング)をプロシージャルに行うことで，錆[38]や汚れ[39]，大理石の黄色化や雨による浸食作用[40]などの表現も可能である．例えば汚れの表現では，汚れの粒子が風雨や重力により流れ出して移動し，その軌跡が汚れとなって付着する過程をシミュレーションしている．このようなエージングは，製造したての新品の質感になりがちな CG 表現において，現実感を増すのに有効な手段であり，通常は，デザイナが手作業で物体表面に汚しのテクスチャを付加している．プロシージャルなエージング手法は，より繊細で忠実な表現が可能であり，制作支援の一助になると考える．

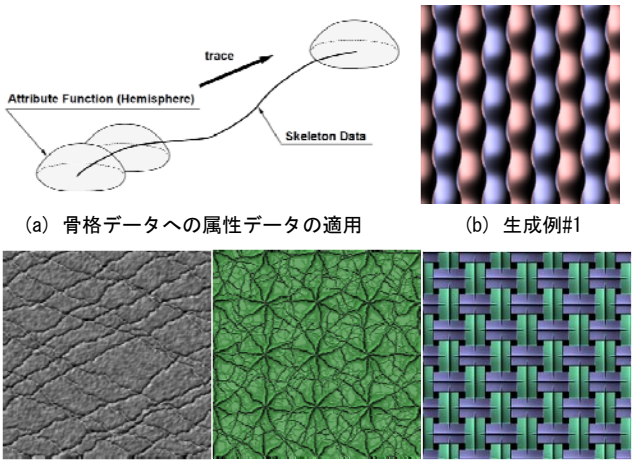
4．プロシージャル技術とデザイン

デザインやアートの分野でもプロシージャル技術は幅広く適用されている．プロシージャル技術のデザイン分野との親和性の良さは,試作物のバリエーションを容易に拡張できることや，デザインのプロセスを記録・記述し試行錯誤の検証が可能である点などが挙げられる．本章では，プロシージャル技術によるいくつかのデザインの例を紹介する

4.1 テクスチャデザイン

物体の素材感を表現するためのテクスチャデザインの代表的な手法としては，スクリプト記述によるデザインや各種関数モジュールの組み合わせによるものなどが挙げられる．例えば DarkTree では，図 X に示すように，各種関数モジュール間のロジックプログラミングをインタラクティブに行うことで，テクスチャ生成のプロシージャをデザインできる．

一方で，テクスチャパターンの特徴的な骨格部分をデザインし，それに沿って凹凸や色などの属性データを発生させる手法が提案されている．例えば，図 X(a)に示すように，与えられた骨格データに，何らかの属性関数（この例では半球を定義する関数)で凹凸付けや色付けなどを施してテクスチャを生成する．この手法は，ユーザのマニュアル的なデザインの柔軟性と，プロシージャル技術による自動的なデザインの両社のメリットを併せ持つ．図 X(b)-(e)に生成例を示す．



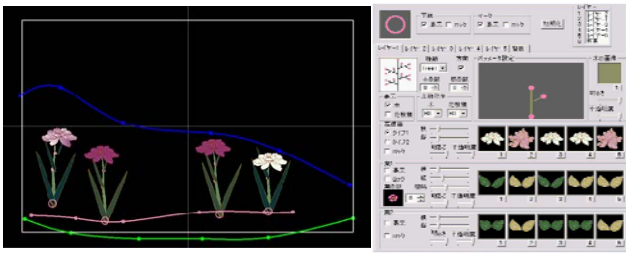
(a) 骨格データへの属性データの適用　(b) 生成例#1

(c) 生成例#2　(d) 生成例#3　(e) 生成例#4

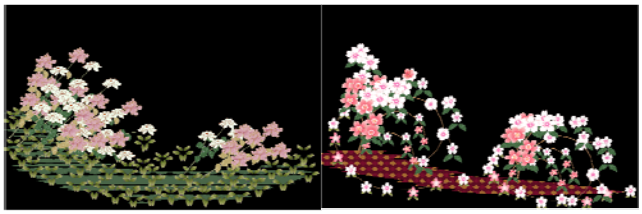
図 11 テクスチャのデザイン例

4.2 着物パターンのデザイン

加賀友禅の着物のパターンデザインに，植物の生成手法で述べたＬシステムを適用した研究例が報告されている．この研究では，ユーザは着物柄のおおまかなデザインと，用いる草木や花，雲などの種類や配置ルールを指定することで，描画スキルや手間が必要とされる細かなパターン生成はコンピュータが行う．



(a) 着物パターンのデザイン過程



(b) デザイン例#1　　(c) デザイン例#2
図 11 着物パターンのデザイン例

4.3 漆工芸品のデザイン

漆工芸に用いられる金箔および漆素材の計測データを用いた質感の高い素材表現と，素材の組み合わせをプロシージャルに行い，漆工芸のデザインを試みた研究が報告されている．この手法では，素材の反射特性を分光測定器で計測し，素材の組み合わせによる加飾時のデザインにプロシージャルな手法を適用している．漆工芸品の加飾プロセスに対しては，プロシージャルにテクスチャパターンを発生させ，そのパターンにしたがって金箔や漆の塗布をシミュレーションしている．工芸素材は工程に長時間を要し，かつ素材そのものが高価であるために，このようなビジュアルシミュレーションは有効であると考える．図 X に漆工芸のデザイン例を示す．

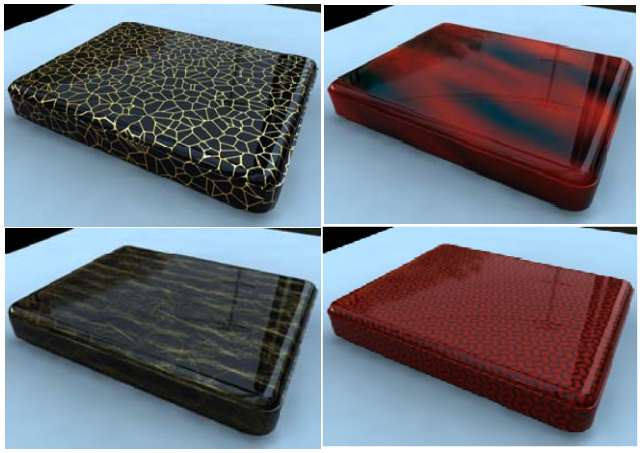
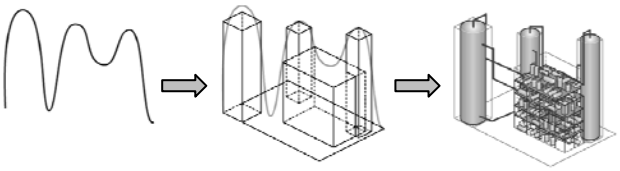


図 11 漆工芸品のデザイン例

4.4プロセスプラントのデザイン

化学工場などのプロセスプラントのプロシージャルな生成法[]を紹介する．プロセスプラントでは大量の部品が空間を充填するように配置されており，それらを接続するパイプが複雑に張り巡らされている．これらのモデリングを手作業で行うのは非常に困難であり作業効率が極めて悪い．図？が処理の概要であり，プロセスプラントの2次元シルエットから3次元の幾何モデルを自動的に生成することが可能である．すなわち，得たいプロセスプラントの概略スケッチをデザインすることで，細部に至る大容量の形状データを生成できる．この手法で得られるシーンの例を図？に示す．



(a) 2 次元スケッチ　(b) 3 次元外接形状の生成　(c) モデルの生成
図 11 プロセスプラントの生成手順

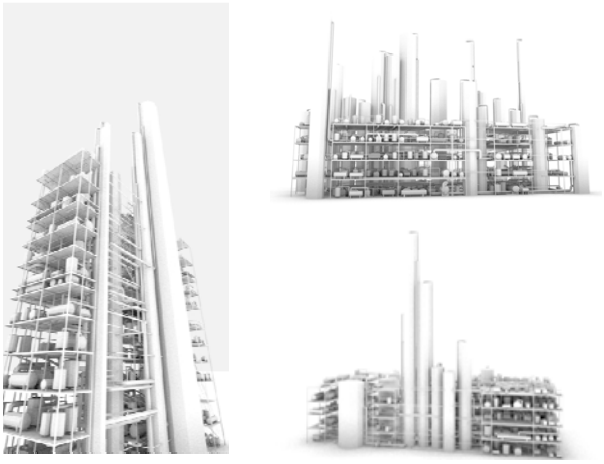


図 11 プロセスプラントのデザイン例

4.5 対称性を持つ室内景観のデザイン

対称性を持つ室内景観のプロシージャルなデザイン手法として，ルールベースの文法を用いた図書館シーンのモデリング手法が提案されている[]．この手法では，シーンの生成ルールとして， 本棚，本，本棚への本の配列，ならびに本棚のフロアへの配置ルールを階層的に追加していくことで，様々なバリエーションのモデルの生成を可能にした．図？に 7 段の本棚モデルの生成ルールと生成例，ならびに本を配架した例を示す．このような階層毎の生成ルールを定義することで，図？に示すような図書館シーンのデザインが可能である．

```
rule bookshelf {
   shelf_frame
   7*{y height}shelf_board
}

rule shelf_frame {…}
rule shelf_board {…}
```

(a) 生成ルール　(b) 本棚の生成例　(c) 本を配架した例
図 11 本棚の生成例

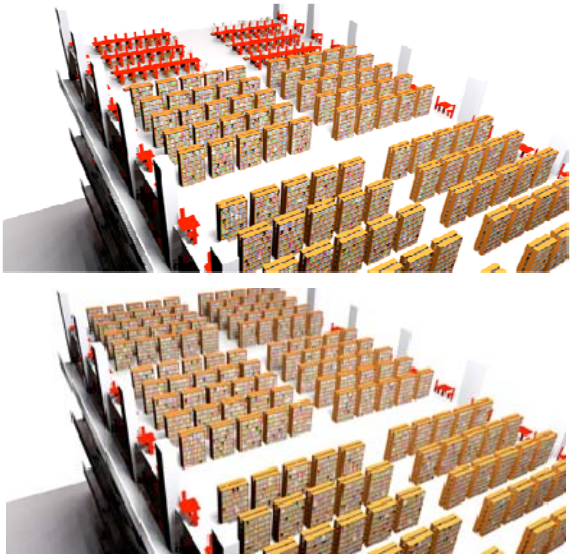


図 11 図書館シーンのデザイン例

4.6 アートへの展開

プロシージャル技術のアートへの展開も多数試みられている．すなわち，アルゴリズムの筆で独創的な絵を描こうというものである．

例えば，フラクタルベースのフリーソフトウェアである「Apophysis」[41]では，図10に示すようなパターンの生成ルールをグラフィカルにかつ対話的に編集することができ，ほぼ無限の複雑なパターンを得ることができる．また，Processing[42]を用いたプロシージャルな平面アートを公開しているサイト[43]もあり，図11に示すようなどこか見覚えのある複雑な作品を web 上で作成することが可能である．また，自身の芸術活動の過程と付随する事柄を明確化し，知識の外部化を行うことで，コンピュータ画家「Aaron」を実装した例[44]や，遺伝的アルゴリズム(GA)により複数の基本形状の遺伝子を組み合わせたり突然変異させることで，形状を生成していく作例[45]もある．

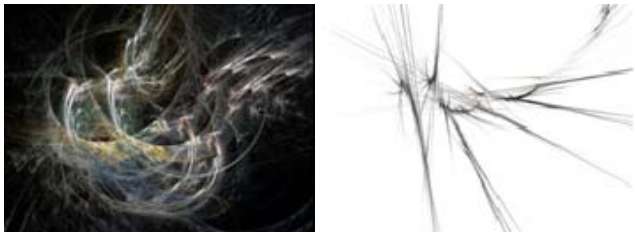
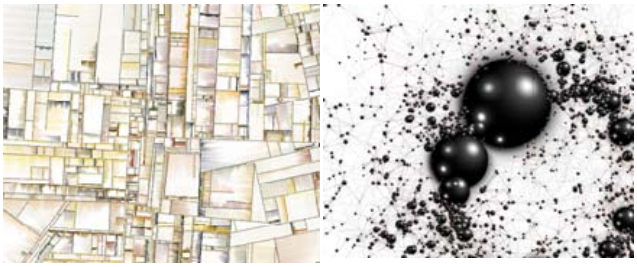


図 10 Apophysis による作例



(a) substrate　(b) node garden
図 11 プロシージャルアートの例

5．おわりに

絵筆やペンで描く行為は直観的な作業であり，体に染みついた知識，すなわち“身体知”により幻影に肉付けしていく作業といえる．一方，コンピュータで CG 映像を制作する行為は，PC を操作して間接的に描き，かつ，論理的な知識と操作といった“形式知”で設計していく作業といえる．今回述べたプロシージャル技術は，さらにもう一段階間接的な操作で絵を描く技術といえる．すなわち，絵を描くソフトウェアロボットを設計するようなものである．

近い将来，デザイナやアーティストと呼ばれる人の立ち位置が，「絵が描ける人」から「絵を描くプログラムをデザインできる人」へと，変化する可能性があると考える．しかしながら，デザインの本質は別なところにあり，プロシージャル技術はあくまでもデザインを支援するツールとしての位置づけにある．プロシージャル技術には，デザイン支援の大きな可能性が秘められており，今後のさらなる発展を期待する．

【参考文献】

1) Steve Upstill, “The RenderMan Companion: A Programmer's Guide to Realistic Computer Graphics,” Addison-Wesley (1990)
2) http://www.renderman.org/RMR/Shaders/BMRTShaders/index.html
3) http://www.generative-modeling.org/
4) Benoit B. Mandelbrot, “Fractals: Form, chance and dimension” W.H.Freeman (1977)
5) Michael Barnsley, “Fractals Everywhere”, Academic Press (1988)
6) B.B.Mandelbrot, “The Fractal Geometry of Nature,” W. H. Freeman, (1982)
7) 安居院，宮田，中嶋，”三次元山岳形状の等高線からの自動生成法”，電子通信学会論文誌，Vol.J69D, No.12, pp.1905-1912 (1986)
8) http://www.planetside.co.uk/
9) Geoffrey Gardner, “Visual Simulation of Clouds,” Computer Graphics, 19, No. 3, pp. 297-304, (1985)
10) D. Ebert, “Volumetric Procedural Implicit Functions:

A Cloud is Born," ACM SIGGRAPH 97 Visual Proceedings (Technical Sketch), (1997)
11) http://www.microsoft.com/japan/games/fsx/default.mspx
12) N.Wang, "Realistic and Fast Cloud Rendering" http://ofb.net/~eggplant/clouds/
13) http://game.watch.impress.co.jp/docs/20051207/3dwa.htm
14) P. Prusinkiewicz and A. Lindenmayer, "The Algorithmic Beauty of Plants," Springer-Verlag (1990)
15) Algorithmic Botany, http://algorithmicbotany.org/papers/
16) P. Prusinkiewicz, A. Lindenmayer, and J. Hanan, "Developmental Models of Herbaceous Plants for Computer Imagery Purposes," Computer Graphics 22(4), pp. 141-150, (1988)
17) O. Deussen, P.Hanrahan, B.Lintermann, R.Mech, M.Pharr, and P. Prusinkiewicz, "Realistic modeling and rendering of plant ecosystems," in Proc. of SIGGRAPH 98, pp. 275-286 (1998)
18) http://www.speedtree.com/
19) R.P.C. J. Rajapakse and K. Miyata, "Generating procedural texture effects by categorizing noise colors," Proc. of IEVC2007, 2A-5, pp.1-6 (2007)
20) Ken Perlin, http://www.noisemachine.com/talk1/
21) Ken Perlin, "Hypertexture," Computer Graphics, pp.253-262 (1989)
22) K.Miyata, "A Method of Generating Stone Wall Patterns," in Proc. of SIGGRAPH'90, Vol.24, No.4, pp.387-394 (1990)
23) K.Miyata, "A Method of Generating Textures by Using Skeleton Lines," Vol.35, No.7, pp.1332-1341 IPSJ Journal (1994)
24) T.Itoh, K.Miyata, K.Shimada, "Generation of Organic Textures with Controlled Anisotropy and Directionality via Packing Rectangular and Elliptical Cells," Vol.23, No.3, pp.38-45 IEEE CG & A (2003)
25) 宮田,坂口,今尾,須崎," パーティクルとメタボールを用いた皮革テクスチャの生成法", 情処学会グラ CAD 研報 Vol.2006, No.119, pp.13-18 (2006)
26) A Witkin, M Kass, "Reaction-Diffusion Textures," Computer Graphics, Vol. 25, No. 4, pp. 299-308 (1991)
27) Greg Turk, "Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion," Computer Graphics, Vol. 25, No. 4, pp. 289-298 (1991)
28) Softology, http://softology.com.au/gallery/galleryrd.htm
29) Aric Hagberg, "REACTION- DIFFUSION PATTERNS," http://cnls.lanl.gov/~aric/Simulations/Simulations.html
30) K.W. Fleischer, D. H. Laidlaw, B. L. Currin, A. H. Barr," Cellular texture generation," in Proc. of SIGGRAPH'95, pp. 239-248 (1995)
31) Y. Parish and Pascal Müller, "Procedural Modeling of Cities," in Proc.of SIGGRAPH'01, pp.301-308 (2001)
32) P. Wonka, M. Wimmer, F. Sillion and W. Ribarsky," Instant Architecture," in Proc.of SIGGRAPH'03, pp.669-677 (2003)
33) P.Müller, P. Wonka, S. Haegler, A. Ulmer and L. V. Gool," Procedural Modeling of Buildings," in Proc.of SIGGRAPH'06, pp.614-623 (2006)
34) Procedural Inc., "CityEngine", http://www.procedural.com/
35) Rome Reborn, http://www.romereborn.virginia.edu/
36) D. R. Fowler, H. Meinhardt and P. Prusinkiewicz," Modeling seashells, in Proc. of SIGGRAPH '92, pp. 379-387 (1992)
37) Yoichiro Kawaguchi, "A morphological study of the form of nature," In Proc of SIGGRAPH'82, pp. 223 - 232 (1982)
38) J. Dorsey and P. Hanrahan, "Modeling and Rendering of Metallic Patinas," in Proc. of SIGGRAPH '96, pp.387-396 (1996)
39) J. Dorsey, H.K. Pedersen and P. Hanrahan, "Flow and changes in appearance," in Proc. of SIGGRAPH '96, pp.411-420 (1996)
40) J. Dorsey, A. Edelman, H. W. Jensen, J. Legakis and H.K. Pedersen, "Modeling and rendering of weathered stone," in Proc. of SIGGRAPH '99, pp.225-234 (1999)
41) Apophysis, http://www.apophysis.org/index.html
42) http://www.processing.org/
43) http://www.complexification.net/gallery/
44) P. McCorduck (著),下野 隆生 (訳)," コンピュータ画家アーロンの誕生—芸術創造のプログラミング", 紀伊國屋書店 (1998)
45) Willian Latham, http://www.doc.gold.ac.uk/~mas01whl/
46) GPGPU, http://gpgpu.org/
47) Allegorithmic, "MaPZone," http://www.mapzoneeditor.com/
48) Dice, "FrostByte Engine,"
49) Electrinic Arts, "Spore," http://www.japan.ea.com/spore/