

# CGWORLD

CG・映像クリエイター総合誌 [月刊シーザー・ワールド] 1,470 (incl. tax)

+digital

求人  
企画

## スマートフォン ゲーム開発を知る

[特別企画]

映画『長ぐつをはいたネコ』

東京モーターショー2011『TOYOTA 86 Full Throttle』

第10回VESアワード授賞式

# 2012春のゲーム グラフィックス

バイナリー ドメイン / アスラズ ラース / フノ国 白き聖巫の女王





# 長ぐつをはいたネコ

TEXT\_倉地紀子

昨年の全米公開で大ヒットを記録したドリームワークスの新作3DCGアニメーション映画『長ぐつをはいたネコ』。ここでは、本作品のVFXスーパーバイザーを務めたケン・ビーレンバーグ氏と、本作品においてポリューメトリックな表現の新境地を切り開いたR&Dのケン・ミュセス氏とのインタビューを通して、本作ならではの映像的魅力を作り出した技術的チャレンジの数々を紹介する。



VFXスーパーバイザー  
(ドリームワークス)  
ケン・ビーレンバーグ/  
Ken Bielenberg氏

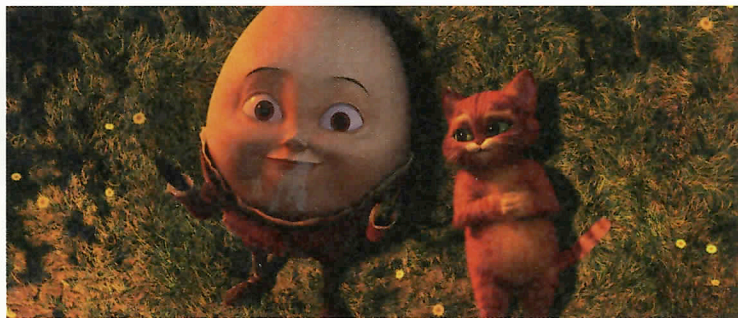


R&D FXスーパーバイザー/  
プリンシパルエンジニア  
(ドリームワークス)  
ケン・ミュセス/  
Ken Museth氏



TOPIC  
1

## 『シュレック』シリーズとの差別化

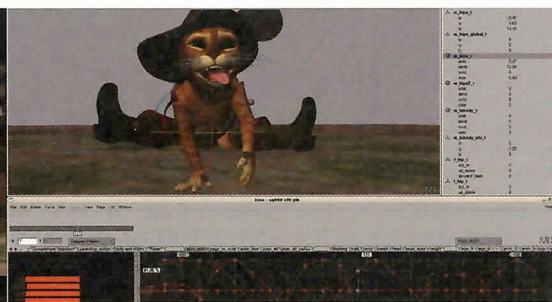


『長ぐつをはいたネコ』は、題名通り長靴がトレードマークのネコ、プスが主人公。『シュレック』シリーズの人気キャラクターでもあったプスだが、今回はその生い立ちの秘話や新キャラクターの登場も伴って、『シュレック』シリーズとは一味違う魅力を持ったストーリー展開となっている。映像的には、キャラクターに関しても環境に関しても、おしなべて形や色は『シュレック』シリーズ以上に大胆に様式化する一方で、『シュレック』シリーズ以上にオーガニックな雰囲気を持たせるように心掛けられたという

TOPIC  
2

## 主人公プスのキャラクター表現

今回の作品で真正正銘の主人公となるプスに対しては、『シュレック』シリーズから受け継がれたものは基本的なデザインのみだったとさえ言える。よりバラエティ豊かな演技が可能となるように、リグの設定などもすべて新たなものに作り変えられ、毛のシミュレーションなども大幅に改善された。表情の作成においては、特に“目”の動きに細心の注意が払われたという



## 『シュレック』シリーズから受け継いだものと新たに生み出されたものの融合

『シュレック』シリーズの人気キャラクターを主人公にしながらも新たなストーリーと世界観を携えた本作品には、『シュレック』シリーズとはまた違った奥深い味わいがある。キャラクター表現をはじめとして環境の生成やライティングなどの技術も大幅に向上しているが、S3Dアニメーション映画という路線の創始者にあたるドリームワークスだけに、S3Dでの上映を大きく意識した技術革命も忘れていない。中でも、まだこれから論文発表されるという新技術を導入したボリューメトリックな雲の世界の表現は圧巻だ。

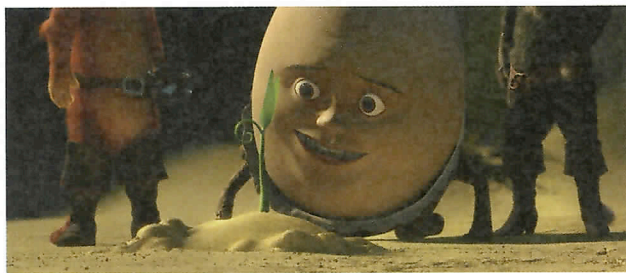
『長ぐつをはいたネコ』という作品の映像的な豊かさや深みは、『シュレック』シリーズから受け継いだものと、『シュレック』シリーズを越えようとするチャレンジ精神が生み出したものが上手く手を取り合って生み出されたと言

える。ピーレンバーグ氏いわく、厳密な意味で『シュレック』シリーズからそのまま受け継がれているのは、長靴がトレードマークの主人公のネコ、プスの基本的なデザインだけだったという。目指されたのは『シュレック』シリーズとは独立した本作ならではの雰囲気を持った映像美で、色や形に関してはより大胆な様式化を施しつつも、『シュレック』シリーズと比較してよりオーガニックな見え方を作り出すことが目標となっていた。後述するようにこの目標を達成するためには、より高度なシミュレーション技術やレンダリング技術が必要となったのだ。

## 主人公として刷新されたネコのプス

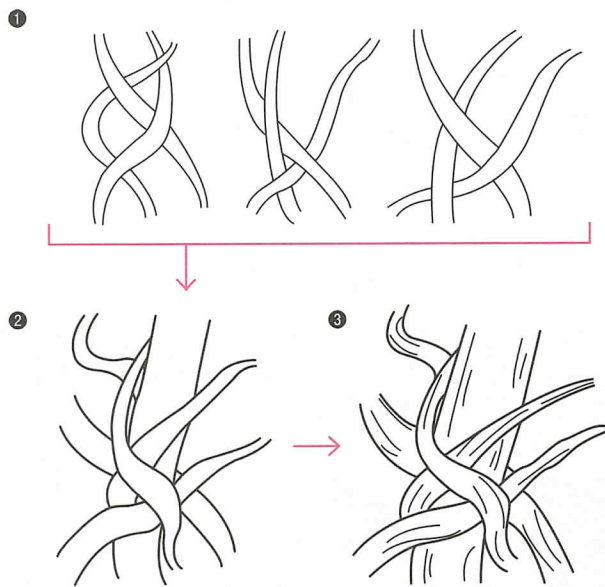
プスに関して言えば、今回の作品では真正正銘の主人公となるので、よりバラエティ豊かな演技が可能となるように、リグの設定なども全て新たなものに作り変えられた。特に重要視されたのは“目”の動きであったという。また

“毛”に関して、特に毛の動きのシミュレーションには、『シュレック』シリーズの頃のシステムを大幅に向上させたものが用いられた。質点同士をバネで繋いだシミュレーション・モデルであることには変わらないのだが、毛のシミュレーションでは、キャラクターの身体の細かい動き、その動きによって引き起こされる毛同士の干渉、風や外部の物体との干渉などのような環境から受ける影響といった様々な要因を考慮する必要がある。従来のシステムでは、これらの要因を全て同時に作用させてシミュレーションが行われていた。これに対して、今回は自社製ツールとHoudiniのシミュレーションツールを上手く組み合わせることによって、全ての要因を一度に作用させるのではなく、個々のまたは部分的な要因だけを作用させてシミュレーションを行っていき、その結果を上手く融合させるということも行われた。全ての要因を一度に作用させる方法が物理的には最も正確で、今回も精緻なリアリズムが必要とされるシーンではこの方法が用いられたのだが、個々の



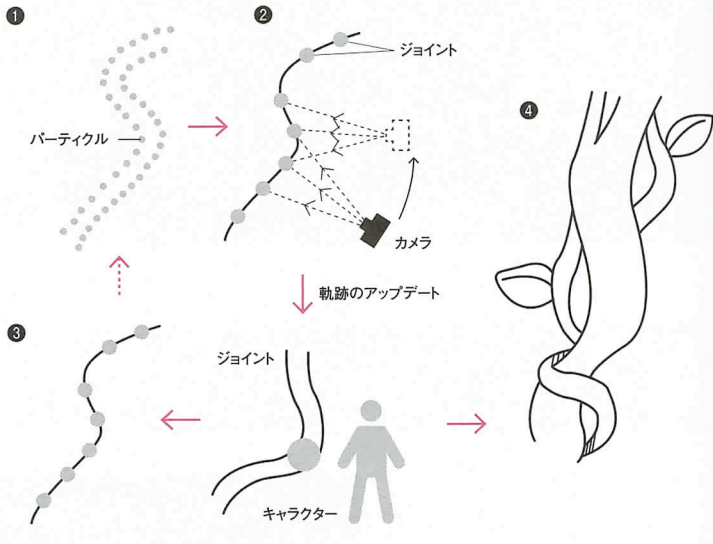
本作では、環境の作成に関して、これまでにない画期的なチャレンジがなされた。小さな豆から巨大な木へと瞬間に成長する“豆の木”はその代表例の1つだ。この豆の木に対しては、カメラが非常に近くまで寄ってそのディテールを捉えようとするこもあれば、カメラが少し遠のいて成長の過程を捉えようとするこもある。このような多彩な表情を効率的に作り出すために、そのモデリングやアニメーションにおいて、数多くのプロシージャルなツールが開発された

■豆の木のモデリング



豆の木はうねるようにして成長していくのだが、モデリングの段階ではこのような“うねり”は加えていないニュートラル (Neutral) な形状やポーズが下記の手順で作成された。こうして作成された豆の木のローカルな形状やポーズは、アニメーションの工程で作成される“軌跡”に沿ってリマップ (Remap) される。  
 ① “つる”の絡み具合のパターンをプロシージャルな手法で作成する (パターンを記述する簡易言語のようなものが開発され、この言語を用いて様々なパターンが作成されたようだ)。  
 ② ①のパターンを組み合わせて豆の木のローカルな構造を作成する。  
 ③ ②に対して成長の段階に合った“年齢”の見え方を与える (Vine Aging)

■豆の木のアニメーション



豆の木が急速に成長するアニメーションは、エフェクトチーム、レイアウトチーム、アニメーションチームの連携によって下記の手順で作成された。豆の木のアニメーションモデルにはジョイント (Joint) というものが設定されており、カメラ位置の決定や豆の木とキャラクターとの干渉を拘束する作業は、このジョイントを用いて行われた。  
 ① エフェクトチームがパーティクルシステムによって豆の木が成長していく“軌跡”の原型を作成する。  
 ② レイアウトチームは①を基にしてカメラ位置を決定する。  
 ③ アニメーションチームは①②の情報及びキャラクターのキーフレーム・アニメーションを基にして、豆の木とキャラクターとの干渉が整合性の取れたものになるように、キャラクターのキーフレーム・アニメーションや豆の木の軌跡をアップデートする。アップデートが困難な場合には①に戻り、アップデートが成功した場合には④に進む。  
 ④ ③のアップデート後の軌跡に、モデリングの工程で作成された豆の木のニュートラルな形状とポーズ (図「豆の木のモデリング」の③) をリマップ (Remap) する

要因を個別に作用させてシミュレーションを行なっておく方法にはコントロールの自由度が高いという利点がある。今回は動きに関して『シュレック』シリーズ以上に様式化された部分が多かったため、この利点が功を奏したようだ。

3チームの連携による  
うねるような豆の木の動き

キャラクター表現もさることながら、本作では環境の表現にもこれまでにない高度な技術が導入された。その代表例が、ストーリーの要ともなっている“豆の木”と“雲”だった。『ジャックと豆の木』の逸話にも絡んでくる今回の豆の木は、急激に成長し竜巻まで引き起こすというマジカルな植物で、そのモデリングやアニメーションのためには数々のツールが新たに開発された。手順としては、まず豆の木の静的なポーズが作成された。このためには、“つる”

の絡み具合の原型を自動生成するプロシージャルなツール、この原型を調整して上手く繋ぎ合わせるツール (このツールにはアーティストがインタラクティブに指示を与えることのできるユーザーインターフェイスも付いていたそうだ)、さらに豆の木の成長の各時点における“年齢”に対応した見え方を自動生成するプロシージャルなツールなどが開発され用いられた。ただし、“つる”の絡み具合にしても“年齢”に対応した見え方にしても、ここで用いられたプロシージャルなツールは決して植物学的な知識やデータに裏付けされたものではなく、フラクタル関数やノイズ関数などをベースにしたアルゴリズムであったようだ。こうして作成された豆の木のポーズはあくまで一方向に向かって伸びていくものだったが、実際にシーンで必要とされたのは、激しいカメラの動きを伴って豆の木がうねるように伸びていく光景だった。

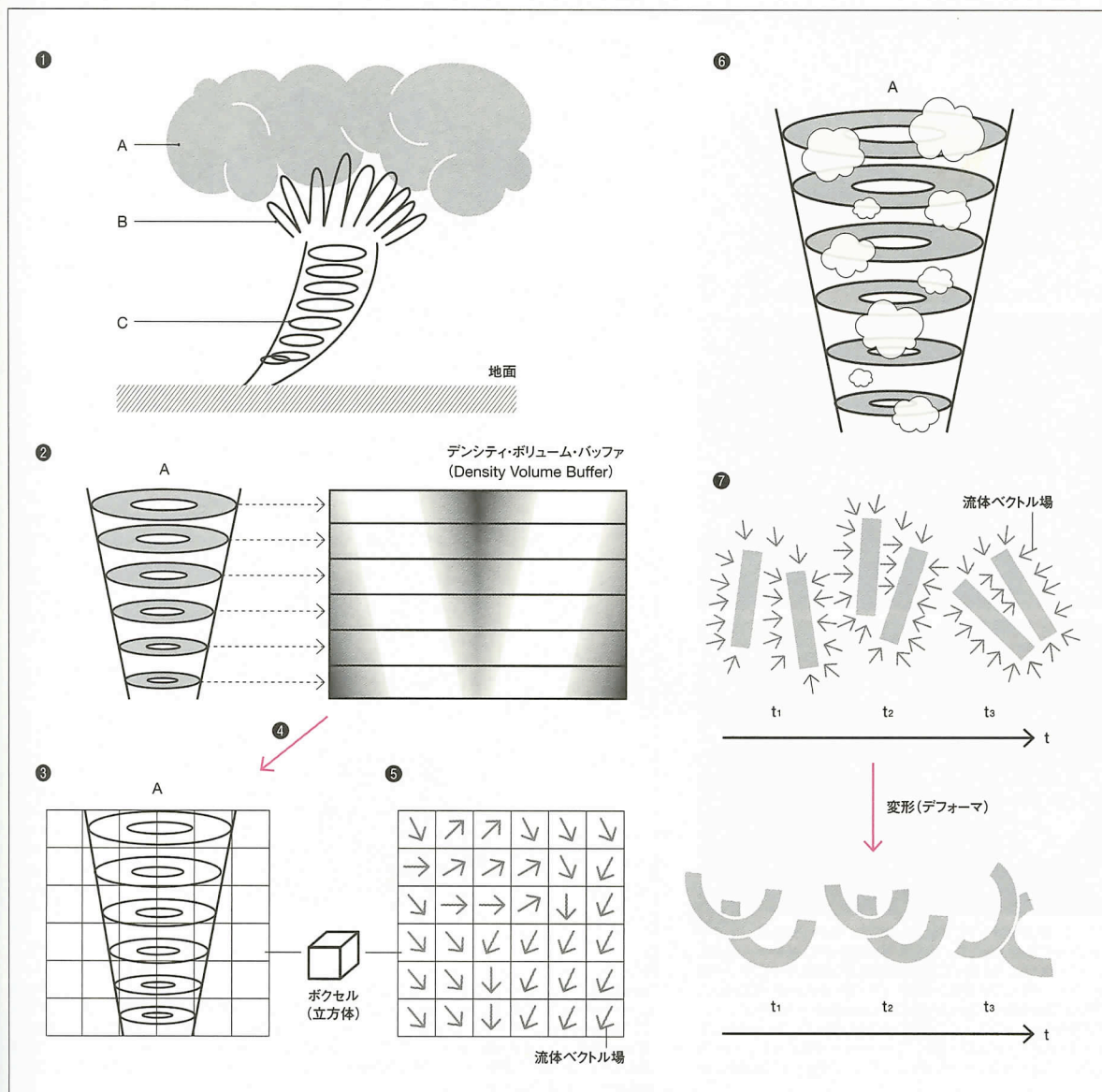
そこで、豆の木がうねるように伸びていく“軌跡”を作成し、この軌跡に沿って上記のようにモデリングされた豆の

木の形状やポーズを変形させるという方法が採られた。軌跡の原型はエフェクトチームがパーティクルシステムを用いて作り出し、これを参考にしてレイアウトチームがカメラの動き、さらにそのデータを参照してアニメーションチームが最終的な軌跡を作成した。困難を窮めたのは豆の木とキャラクターが複雑に干渉する場合で、エフェクトチームはカメラの動きやキャラクターの動きといったキーフレーム・アニメーションの情報を持たずにプロシージャルに軌跡の原型を作成するため、このプロシージャルな作業工程をやり直さない限り豆の木とキャラクターとのインタラクションを整合性の取れたものにできないケースもあったという (ここでいうプロシージャルな作業工程とは、軌跡の原型の作成だけに留まらず、場合によってはつるの絡み具合の原型を作成するモデリングの工程も含んでいる)。

そして、エフェクトチームによって与えられたこのような変更を受けて、アニメーションチームもキャラクターと豆の木との干渉がパーフェクトになるように再びキーフ

TOPIC  
4

## 竜巻の作成



豆の木の急激な成長が引き起こす竜巻の表現では、(A) 漏斗状の竜巻本体、(B) 竜巻本体から上空に向かって“指”を広げたように伸びている部分、(C) 上空に渦巻く雲の3種類のアニメーションが、プロシージャルな手法と流体シミュレーションとを併用して作成された。

①時間軸に沿った3Dノイズ関数などのプロシージャルな手法を用いて、(A)及び(B)の形状(この形状はサーフェスだけではなく内部も詰まったボリュームメトリックなもの)とアニメーションを作成する。

②①で作成された(A)の形状を各時間においてラスタライズして、Density Volume Bufferに記録。

③①で作成された(A)の動きに流体シミュレーションを駆使した動きを加えるため、(A)を含む空間全体をボクセルに分割。

④②を参照して③の各ボクセルの各時間における密度(Density)を算出。

⑤④を参照して、ガスソルバによる高解像度の流体シミュレーションを実行。

⑥①で作成した動きに、⑤で作成した動きを加えたものが、最終的な(A)の動きとなる。

⑦(C)の動きも、ガスソルバを用いた流体シミュレーションによって作り出す。ただし、ここではガスソルバが算出した流体ベクトル場を“渦巻く”雲の形状に直接作用させるのではなく、まず(a)まっすぐに伸びた短冊状の帯としてモデル化された雲の構成要素に対してガスソルバが算出した流体ベクトル場を作用させ、(b)その結果をデフォーマによって変形させることによって、最終的な渦巻く雲の動きが作り出された。

レーム・アニメーションのディテールに手を加える必要があった。このように作業プロセスが必ずしも一方ではなく堂々巡りになりがちであったことは、かなりの痛手であったようだ。

### プロシージャルとシミュレーションを 組み合わせたダイナミックな竜巻

豆の木の成長は、ダイナミックな“竜巻”のエフェクトも伴う。デザイン的にはかなり大胆に様式化されているながらも真に迫った動きが印象的なこのエフェクトは、プロシージャルな手法と流体シミュレーション技術を融合して作り出された。手順としてはまず、漏斗(funnel)のような竜巻本体と(竜巻本体を“手”の甲と考えると)そこから外に向かって“指”のように伸びている突起の形状変化をプロシージャルな手法で作成する。これには時間軸に沿った3Dノイズ関数が用いられたようだ。次に竜巻の中心部分に対

して、自社製のガスソルバ(流体の運動方程式を解いて煙・霧・雲のような気体の流体の動きを算出するためのツール)を用いて高解像度(300×300×1,000)の流体シミュレーションを実行する。このシミュレーションはプロシージャルな手法で作成された竜巻本体の動きを引き継いでいる必要がある。そこで、竜巻本体の漏斗のような形状を輪切りにしたリングをボリュームバッファにラスタライズする。ガスソルバは、このボリュームバッファの情報を読み取って、計算の単位ユニットであるボクセルの密度を算出し、これを入力情報として流体シミュレーションの計算を行う。このシミュレーションの結果をプロシージャルな手法で作成されたボリュームにブレンドすることによって、最終的な竜巻のエフェクトが完成する。

上記のガスソルバは、空に渦巻く雲の動きを作り出すためにも活用された。空に渦巻く雲は、スパイラル状になった長い帯の集合としてモデル化されており、本来はこのスパイラル状の帯がガスソルバの算出した流体ベクトル場か

ら受ける影響を計算する必要がある。しかし実際には計算を効率化するため、まっすぐに伸びた長い短冊状の帯がガスソルバの算出した流体ベクトル場から受ける影響を考慮してその各フレームにおける位置や形状をあらかじめ計算しておき、デフォーマ(変形ツール)を用いてこれを変換することによってスパイラル状の帯の各フレームにおける位置や形状が算出されたのだそうだ。

### 論文発表前の新技術DVBが 活用された雲の表現

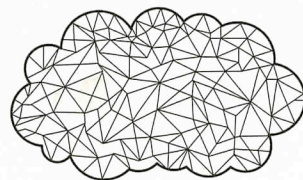
そして、『長くつをはいたネコ』のプロジェクトにおける最大の技術的チャレンジであったとピーレンバーグ氏が指摘するのが、“雲”の世界の表現だった。一言に雲と言っても空にぶかぶか浮いているものではなく、ストーリーの要となる大きな城に通じる環境全体を空気のように満たしているものだ。なおかつキャラクターはこの雲の環境と



本作における最大の技術的チャレンジは、城に通じる環境全体を覆う“雲”の世界の表現だ。過去の映画プロジェクトではこのように広域にわたる雲の表現には2Dのアプローチが用いられてきたが、今回はS3Dでの上映を伴い、なおかつこの雲はキャラクターと密接にインタラクションするというのもあって、極端に遠景の雲以外はすべて3Dで作成された。ポリュメトリックなアニメーションやレンダリングにはボクセルデータが用いられたが、今回はカメラが非常に近づくショットも多かったため、非常に高解像度でボクセルに分割する必要があり、従来のポリュメトリックデータのフォーマットを用いるとそのデータ量は膨大なものとなる。そこで、符号付きのインデックスや独自の階層構造を与えられたDVBという全く新しいコンセプトのポリュメトリックデータのフォーマットが考案された。DVBはデータを大幅に圧縮できるだけでなく、流体シミュレーションやポリュメトリックなレンダリングとの相性が良いところも大きな特徴となっているようだ

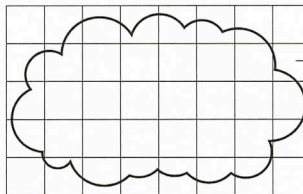
### ■雲の構造とアニメーション

①

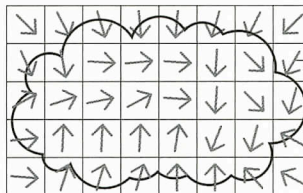


ポリゴンサーフェス

②

ボクセル  
(立方体)

③



流体ベクトル場



雲の構造としては、まずポリゴンサーフェスを用いてその形状をモデリングし①、次にこのサーフェスの内部をボクセルグリッドで分割してボクセルデータに変換したものが用いられた②。これらのボクセルデータに対して、プロシージャルなアルゴリズムや流体シミュレーションを適用することによって、ふわふわとした雲らしい表情や動きが作り出された③。ボクセルデータをDVBという新たなポリュメトリックデータのフォーマットで記述することによって、特定の表現に必要なとされる最小限のボクセルデータに絞ってこれをコンパクトに圧縮した形で扱うことが可能となった

複雑にインタラクションする。従来はこのように広範囲にわたる雲を描き出すためにはマップペインティングなどの2D手法が用いられてきたのだが、今回はキャラクターがこの雲と複雑にインタラクションする上、S3Dでの上映を伴っている。従って2Dの技法を用いたごまかしが効かず、一部のごく遠景の雲以外はすべて3Dで作成されることになった。とはいえ、映画終盤にはこの雲の環境が4分以上に渡って登場する。これだけの長尺の間アニメーションする3Dのポリュメトリックデータは膨大なものとなり、これを従来のデータフォーマットで記述しようとすると、プロダクションでの使用には無理があるほど大きなファイルを山ほど扱わなくてはならない。そこで、今回はこのようなポリュメトリックデータを効率的に扱うことのできる新たなフォーマット(DVB)が生み出された。

DVBを考案したミュセス氏によると、DVBのDはダイナミック(Dynamic)、Vはポリュメトリック(Volumetric)、RはR+Treeという階層構造を意味しているという。商業

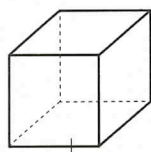
の場合と同様に、雲の動きもノイズ関数を用いたプロシージャルな手法と高解像度の流体シミュレーションを併用して作成されており、ポリゴンサーフェスによってモデリングされた雲はこのようなアニメーションの手法で採用されているボクセルデータに変換される。DVBは、3D空間をボクセルグリッドで分割したボクセルの集合を時間軸に沿って展開したボクセルの集合に上限や下限のない符号付きのインデックスを振って、これらに独自の階層構造を持たせている。今回の広域に渡る雲の表現では、カメラが極端に近くに寄ってクローズアップになる場合も多く、非常に高解像度なボクセルへの分割が必要とされた。その結果として膨大な量のボクセルデータを扱うことになるのだが(カメラからの視線に沿ってボクセル数が15,000を超えることもあったという)、DVBを用いるとこれを非常にコンパクトなデータに圧縮できる。階層構造を持っているゆえにカメラからの距離に応じた解像度の変化に対応しやすく、データアクセスもインデックスを用いた1Dのアクセ

スとなるので本来のデータのジオメトリを意識することなく高速に行うことができる。また、雲のようにサーフェスを持ったポリュメトリックの場合には、流体シミュレーションの計算を行う上で着目するボクセルがサーフェスの内側にあるか外側にあるかということが重要になってくるケースも多いのだが、符号付きのインデックスをレベルセットの手法と結び付けると、このような判断も効率的に行うことができる。さらに、このデータ構造には“時間軸”という概念も取り込まれているため、時間軸に沿ったデータの急激な変化にも上手く対応できる。このようにDVBは、高解像度で複雑なアニメーションを伴いがちな映画プロジェクトにおけるポリュメトリック表現に非常に適したデータフォーマットと言えるのだ。DVBの詳細は、まだこれから論文発表されるそうで、論文に先立つ新技術が今回の映画プロジェクトで使用されたことの意義も大きかったと言えよう。

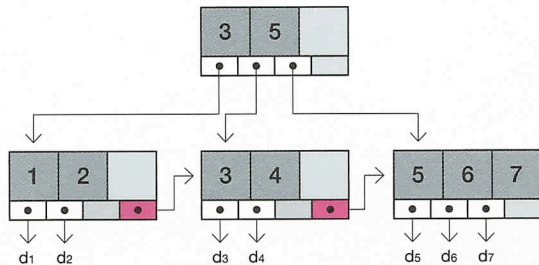
雲の表現ではレンダリングに関しても、雲の内部で散乱する光の効果をとことん物理的に正確に計算するという画

## ■ DVB

①

ボクセル  
3Dインデックス  
(±i, ±j, ±k)

②



雲のレンダリングでは、まず環境光のオクルージョンを計算した上で、雲の内部で散乱する光の効果が物理的に正確に計算された。雲の内部で太陽からの光が直接当たっていない部分のリアリズムを高める上で、これは非常に重要だと考えられたからだ。この計算は、2種類のシェーダ(①②)を用いて下記のような手順で行われた。

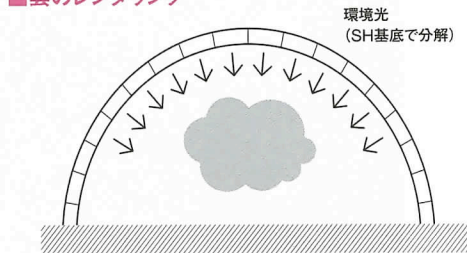
①1種類目のシェーダは、環境からの光が雲の内部の散乱成分と最初に干渉した際に、干渉した環境光のうち(a)どれだけカメラ方向に散乱し、(b)どれだけ干渉した地点の周りに散乱するかを計算する。(a)はシングルスキャタリングの効果に相当し、(b)は②のマルチスキャタリングの効果の計算に引き渡される。(a)のシングルスキャタリングの効果を表す画像は、この段階でレンダリングしておく。

②2種類目のシェーダは、マルチスキャタリングの効果の計算するためのもので、①の(b)で算出された光を“キーライト(Key Light)”と考え、キーライトの最も近くにあるボクセルからその周りのボクセルへの光の伝達を波動方程式(Eikonal Equation)によって記述する(波の進むスピードはHeney-Greensteinのフェーズ関数によって決定する)。そして、この波動方程式を解いてキーライトからの光がその周りの各ボクセルに到達する時間を算出する。直感的には光が到達した時間の遅いボクセルほど受け取る光は少なくなる。そこで、キーライトを発生した光が各ボクセルに到達するまでに掛かる時間の比を計算してグレースケールのテクスチャとして記録する。

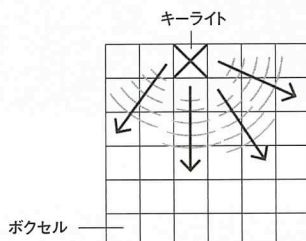
③ピーレンバーク氏の説明によると、②のテクスチャはMultiplierとして役割を果たしていたということなので、例えばどのボクセルもキーライトから均一な光を受け取ったと仮定してこれらがカメラ方向に作り出す光を計算したレンダリング画像を作成しておくなど、前計算によってあらかじめ作成された画像に②でスケールを掛けてマルチスキャタリングの効果を表す画像が生成されたようだ。

④①のシングルスキャタリングの効果を表す画像と、③のマルチスキャタリングの効果を表す画像を合成したものが、ファイナル画像となる

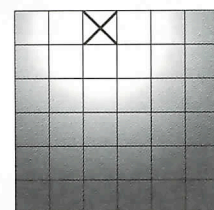
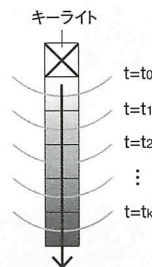
## ■ 雲のレンダリング



②



ボクセル



テクスチャ

これから論文発表されるというDVBの詳細はまだ明らかにされていないが、代表的な特徴としては、下記のようなものが挙げられる。

①各ボクセルに対しては、上限や下限を持たない符号付きの3Dインデックスが与えられている。

②ボクセルの集合全体に対してB+Treeと呼ばれる階層構造が設定されている。B+Treeでは実質的なデータは最下層のノードのみに存在し、その間のノード間の検索は全て“キー”と呼ばれるインデックスを用いて行われる(例えば図②のB+Treeでは、1から7までのキーが与えられている)。①と②を併用することによって、インデックス・スペースを用いてボクセルデータが表すジオメトリを意識することなく高速にデータにランダムアクセスすることが可能となる。

③ボクセルデータが持つ情報が時間軸に沿って急激に変化する場合にも、これを時間軸に沿って上手く平均化し、個々の時間で安定した状態でアクセスできるようになっている。今回の雲の作成では、特定の表現において必要とされる最小限のボクセルに絞り込んで計算が行われたが、演出的もしくはアートの見地の要請から部分的に新たにボクセルを加えたり逆に部分的にボクセルを取り除いたりする必要が頻りに起こったという。①及び②はこの作業を効率化するために大きく貢献した。①の“符合付き”という特徴は、雲のようなサーフェスを持った流体の動きを効率的に扱うために活用されるレベルセット(Level-Set)の考え方も相性が良かったようだ。また、今回の雲の表現では、カメラが雲に非常に近づくショットもあればカメラが少し離れたところから雲の環境全体を捉えるシーンもあり、それに伴ってボクセルの解像度を変える必要があり、②の階層構造はこのような解像度の変化に効率的に対応するために適していた。そして言うまでもなく、複雑なアニメーションを伴った今回の雲では、③の利点が大きな意味を持っていた

期的なチャレンジが行われた。雲の内部で一度だけ散乱する光の効果をシミュレーションするという試みはすでに過去の映画プロジェクトでも行われたことがあったが、複数回の散乱までシミュレーションした例は映画史上今回が初めてであったと言える。方法論としても光の伝達を波の伝播と考えて物理方程式を解くという本格的なものだ。ただし、波の伝播の計算はボクセルベースで行い、その結果をテクスチャマップとして出力するなど、効率化でアーティストに馴染みやすい形に調整するという工夫も凝らされている。DVBは光の伝達の計算をボクセルベースで行う上でも、その効率化に大きく貢献したようだ。ライティングには環境光をSH基底で分解したものが用いられ、環境光のオクルージョンを前計算してSH基底の係数として保存しておくという方法が適用された。リアリティを高めるため、環境光を表す環境マップのユニットとして実写素材のHDR画像が用いられたこともあったようだ。

『シュレック』シリーズの最終作品でインタビューを

行なった際に、ドリームワークスのスーパーバイザーは『シュレック』シリーズが終焉を迎えても、そのチャレンジ精神は受け継がれていく」と語っていた。そして、その代表例として挙げていたのが『長ぐつをはいたネコ』だった。『シュレック』シリーズは、3DCGアニメーションとしては異例なほど表現のディテールやリアリズムに重きを置いた作品であったと言え、それゆえに最新のCG技術を初めて映画に導入することを可能にしたという実績を数多く残してきた。今回の『長ぐつをはいたネコ』でも、上述した“雲”をはじめとして、S3Dならではのボリューム表現の新境地を切り開くために数多くの果敢なチャレンジがなされたと言える。楽しいストーリーとアート性豊かな映像の美しさが印象的な作品の裏で、よもやこのような高度な技術革命が行われたとはまったく予測がつかないところも感慨深い。『シュレック』シリーズから受け継がれ、それを大きく進化させた歩みも、今後どう展開していくのかが今から楽しみだ。

## info.

『長ぐつをはいたネコ』

3月17日(土)新宿ピカデリー他

全国春休みロードショー

[3D/2D同時公開]

配給:パラマウント ピクチャーズ ジャパン

www.naganecko.jp

