OpenSimによるジオラマシステムの構築

井関文一* 金 武完**

あらまし オープンソースの３次元仮想空間用サーバであるOpenSimを改造し、実測された標高データとマップデータから、OpenSim上にリアルな地形（ジオラマ）を再現するシステムの構築を行った。本システムではOpenSimの制約上、マップデータのダウンロードに若干の時間がかかるものの、ほぼリアルタイムに地形を再現することが可能である。使用する標高データは予めデータ処理などを行う必要もなく、短時間に低コストでシステムを利用することができる。本システムは地理教育や地域学習、地域の広報用などの面での応用が期待される。

キーワード：OpenSim、ジオラマ、標高データ、数値地形モデル

Construction of Diorama System by OpenSim

Fumikazu Iseki and Moo Wan Kim

Abstract OpenSim is the open source server for 3D virtual space. And we constructed the system that reproduced real geographical features （diorama） on OpenSim from measured terrain data and map data by adding any functions to it. Geographical features can be reproduced almost in real time though it takes some time for this system to download the map data in the OpenSim restriction. The terrain data need not be processed beforehand, and the system can be used low-cost and easy. As for this system that is used for a geography education, regional study, and announcing to public the region, etc. are expected.

Keyword：OpenSim, Diorama, Terrain Data, Digital Terrain Model

*東京情報大学 総合情報学部 情報文化学科
Tokyo University of Information Sciences, Faculty of Informatics, Department of Media and Cultural Studies
**東京情報大学 総合情報学部 情報システム学科
Tokyo University of Information Sciences, Faculty of Informatics, Department of Information Systems

2010年12月7日受理

1．まえがき

OpenSimulator［1］（以後OpenSim）はオープンソースの3次元仮想空間構築用のサーバソフトウェアである。現在はまだαバージョンの段階ではあるが、商用の3次元仮想空間サービスであるセカンドライフと互換性を持ち、一部ではそれを超える機能も実装されている。

今回我々はOpenSimのソースコードを改造し、3次元仮想空間内にほぼリアルタイムで実際の地形を再現するジオラマシステムの構築を行った。

類似したサービスであるGoogle Earthなどと比較すると、描画速度や操作性の面などでは及ばないが、標高の強調や海水面の高さ、陰影などを細かく設定可能なことや、ジオラマ内をアバターとして自由に移動できるなど、Google Earthとは違う視点を与えることが可能である。

また、リアルタイムで地形を変形させることにより、時系列的な地形変動のシミュレーションを行うことができ、過去から現在までの地形変動の再現や、地球温暖化よる海水面の上昇に伴う水没地域のシミュレーションなどを行うことも可能である。

本論文ではこのOpenSim上のジオラマシステムについて解説を行う。

2．OpenSimでの地形の変形

2.1　標準的な地形の変形方法

OpenSimでは標高データを保存したデータファイルを読み込むことにより、リージョン（サーバの制御対象となる一区画）の地形を変形させることが可能である。しかしながらこの手法では、リージョンサーバのコマンドプロンプトからその都度コマンドを入力する必要がある。

OpenSimにログインした状態で地形を変形させる方法としては、llModifyLandという関数を使用したLSL（Linden Scripting Language）スクリプト［2］を作成する方法も考えられるが、この関数では座標ごとに標高を設定することが不可能なため、目的を達成するには不十分である。

一方、OpenSimにはOSSL（OpenSim Scripting Language）［3］と呼ばれる拡張関群が存在する。その中のosTerrainSetHeight関数は1点ごとに標高を設定することが可能である。ただし、256×256点のリージョン全体（通常ではリージョンの大きさは256x256となる）の標高を変更する際にも、1点ごとに関数を呼び出さなければならいないため、かなりの時間を要し実用的ではない。

2.2　OSSLの拡張

前節で説明したように、標準的な手法ではOpenSim内部からリージョン全体の標高をリアルタイムに変更することは不可能である。従って、我々は新たにOSSL関数を追加することにより、この機能の実現を図った。

OpenSimでは表1の3つのファイルを編集することにより、自由に関数を追加することが可能である。

今回追加した関数を以下に示す。

void osTerrainSetByString（string str, double rate）

LSL/OSSLでは配列が扱えないため、この関数では標高データは文字列strで与えられる。一行ごと（X方向256個のデータごと）に¥nで区切って256行のデータが入力される。行、列それぞれ256個に達しない場合は、足りない部分は0.0で埋められる。また、256個を超える部分は切り捨てられる。

Region/ScriptEngine/Shared/Api/Implementation/OSSL_Api.cs
Region/ScriptEngine/Shared/Api/Interface/IOSSL_Api.cs
Region/ScriptEngine/Shared/Api/Runtime/OSSL_Stub.cs

表1．新しい関数を登録するための三つのファイル
Table 1　Three files to add new functions

rate は各標高データに乗算する係数である。与えられたデータをそのまま標高データとして使用する場合には1.0を指定する。

この関数はシステム内部の標高データの配列を直接書き換えてしまう。この関数に続いて、標高データの配列の内容が変化したことをシステムに通知する、osTerrainFlush関数（OSSL標準関数）を呼ぶことにより、リアルタイムにOpenSim内の標高を変更することが可能となる。

2.2 標高データ

本システムでは、標高データとして数値地形モデルのデータを使用する。具体的には、国土地理院の数値地図50mメッシュ（標高）［4］とSRTM３［5］が使用可能である。

これらのデータをWeb上に本来の形式のまま置き、それぞれのデータを文字列に変換するPHPのプログラムを通してOpenSimのLSL/OSSLからアクセスを行う（図１）。中間に変換用のPHPプログラムを置くことにより、本来のデータを予め加工しておく必要はなく、また他の形式の標高データであっても、それ専用のPHPプログラムを作成すれば、OpenSim側のLSL/OSSLプログラムを変更する必要はない。

なお、SRTM３では所々の標高データが欠損しており、ちょうどスパイクノイズのようになっている。この欠損データを補完するため、欠損データの存在する箇所については３×３のメディアンフィルタを適用している。

また、本システムの測地系はデフォルトで日本測地系が使用されており、SRTM３では簡易的な変換式により日本測地系による緯度経度に変換が行われている。

3．OpenSimでの地表テクスチャ

3.1 スカルプテッドプリム

OpenSim内の標高を実際の数値地形モデル

図1．Web上の標高データによるリージョン内の地形の変形

Figure 1 Transformation of land of region by terrain data on Web

に合わせて自由に変更できたとしても、それでリアルな地形を再現しているとは言い難い。実際の地形のジオラマを作成するためには、地表のテクスチャとして、衛星（航空）写真やマップ画像を貼り付ける必要がある。

OpenSimやセカンドライフでは、標高に合わせて4種類の地表テクスチャを設定することは可能であるが、地点ごとに自由にテクスチャを設定することはできない。

そこで我々は、地面にその凹凸に沿って変形可能なスカルプテッドプリム［6］を被せ、そのスカルプテッドプリムのテクスチャとして衛星（航空）写真やマップ画像を貼り付ける手法を考案した（図２）。

しかしながら、スカルプテッドプリムはシステム側の制約により、最大32×32の制御点しか持つことができない。制御点間の距離をリージョンの座標点間の距離（１m）に合わせれば、スカルプテッドプリムの大きさは最大で32m×32mとなり、（通常の）リージョンの全体を覆うには最低でも８×８=64枚必要となる。

またこの手法を実装する場合、地形の変形方法には依存しないように実装を行う。地形の変形方法に依存しなければ、後で変形方法が変更されたとしても、この地表テクスチャの貼り付け方法はそのまま使用可能となる。

以上の点を考慮した上で、この手法を実現す

るために我々は新たにOSSLのosTerrainGetSculpt関数と、BitmapStringRenderモジュールの作成を行った。

システムの概略を図3に示す。

osTerrainGetSculpt関数は、地面に被せるスカルプテッドプリムを作成するために、地面の標高データをスカルプテッドプリム用のデータに変換して返す関数であり、次のようなインターフェイスを持つ。

string osTerrainGetSculpt（double x, double y, double z, double size, int mhsz）

ここで、（x, y, z）はスカルプテッドプリムの中心と重なる地表の座標で、sizeが正方形のスカルプテッドプリムの一辺のサイズを表す。この関数によって、（x,y）座標を中心に±size/2の範囲の標高データが文字列化される（図3の②～③）。また、mhszはスカルプテッドプリムのメッシュサイズで、システム側の制約により通常は32または64が指定される。

osTerrainGetSculpt関数により生成された文字列の標高データは、OSSLの既存のosSetDynamicTextureData関数を経由して、新しく作成したBitmapStringRenderモジュールに渡される。

BitmapStringRenderモジュールは渡された標高データの文字列から、スカルプテッドプリム用の画像データを生成する（図3の④～⑤）。画像データはそのままosSetDynamicTextureData関数を経由してメインプログラムに渡され、そこでスカルプテッドプリムの変形が行われる（図3の⑥～⑦）。

3.2　マップテクスチャの貼り付け

一方、それぞれのスカルプテッドプリムごとに、その位置から、テクスチャとして貼り付けるべき画像のURLが自動的に計算され（図3の①、⑧）、OSSLの既存のosSetDynamicTectureURL関数により Yahoo マップまたはGoogleマップより地表の画像データ（マップテクスチャ）がダウンロードされる（図3の⑨～⑫）。

ただし、Googleのマップサービスでは、ダウンロード枚数が1日で1,000枚以内という制限がある。本システムでは、1回の稼動で最低でも64枚の画像を必要とするため、1日で16回以上は地形の変更ができない計算になる（実際にはこれよりも少ない回数でダウンロードが禁止されるようである）。

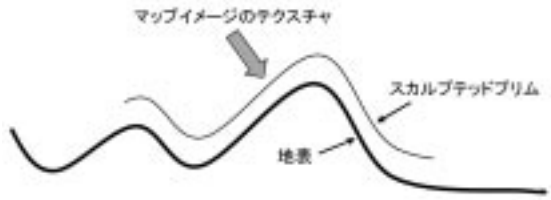

図2．地表の凹凸に従って変形可能なスカルプテッドプリム

Figure 2　Sculpted Prim that can be transformed according to ruggedness of surface of the ground

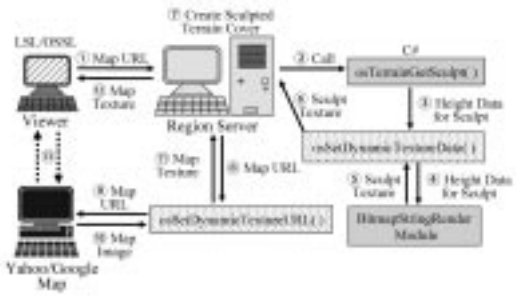

図3．スカルプテッドプリムによる地表テクスチャの設定

Figure 3　Setting of surface texture of the ground by Sculpted Prim

4．実行例

図 4 ～ 6 に本システムによる実行例を示す。図4は標高データとして国土地理院の数値地図50mメッシュ（標高）を使い、地表テクスチャとしてGoogleの衛星データを使用している。

図5は標高データとしてSRTM 3 を使用し、地表テクスチャとしてYahooの衛星データを使用している。

また図 6 は標高データとして国土地理院の数値地図50mメッシュ（標高）を使い、地表テクスチャとしてYahooのマップ画像を使用している。

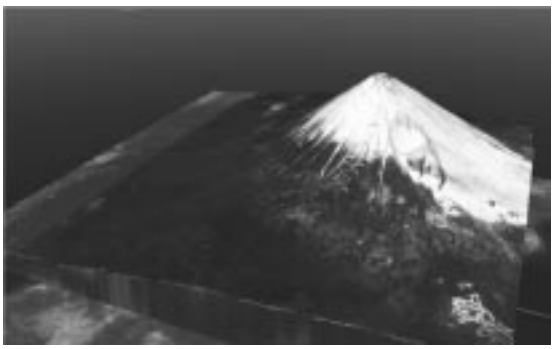

図4．国土地理院の標高データと Google マップの衛星画像から再現した富士山

Figure 4　Mt.Fuji reproduced from terrain data of Geospatial Information Authority of Japan and satellite image of Google Map

図5．SRTM3 と Yahoo マップの衛星画像から再現した富士山

Figure 5　Mt.Fuji reproduced from terrain data of SRTM3 and satellite image of Yahoo Map

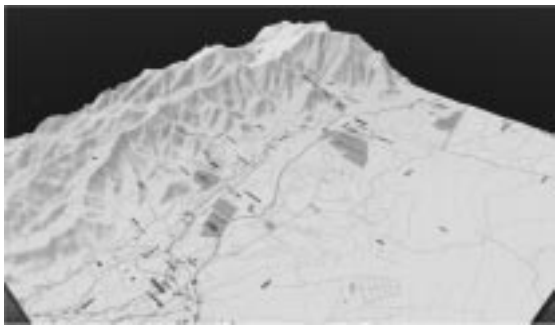

図6．国土地理院の標高データと Yahoo マップの地図画像から再現した富士山西部白糸方面

Figure 6　Mt.Fuji west Shiraito district reproduced from terrain data of Geospatial Information Authority of Japan and map image of Yahoo Map

利用ユーザはアバターとしてこれらの 3 次元仮想空間内を自在に移動することができ、自由な視点から地形を観測することが可能である。

5．問題点と対応策

5.1　テクスチャのダウンロードスピードの問題

本システムにおいて、最も問題となる点はマップテクスチャのダウンロードスピードである。OpenSimやセカンドライフでは、図 3 の⑫の部分は通常はUDPにより通信が行われ、テクスチャなどの画像データは複数のパケットに分解されて転送される。テクスチャデータは他のUDPデータと比べるとパケットのサイズおよびパケット数が大きくなるため、ダウンロードスピードはどうしても低下してしまう。

図 3 の⑬の経路でテクスチャをダウンロードできればスピードは上昇するはずであるが、現在のビューアでそのような機能を持つものは存在しない。

一方、実験的ではあるが、⑫のダウンロード部分をUDPではなく、TCP（HTTP）で行う

という機能を持つビューアが存在する。その最も代表的ものはSnowglobe［7］と呼ばれるビューアであり、HTTPによるテクスチャデータのダウンロード機能はHTTP Get Texture機能［7］と呼ばれている。

従って、Snowglobeを用いてHTTP Get Texture機能を使用すればマップテクスチャのダウンロードスピードの問題をクリアできると思われるかもしれないが、実際には解決には至らない。

なぜならば、Snowglobeなどの一連のビューアはテクスチャデータをダウンロードする場合に、アバターの近くにあるテクスチャやアバターが注目しているテクスチャを優先的にダウンロードし、そうでないテクスチャデータについてはヘッダ情報のみをダウンロードするからである。一般にこのアルゴリズムはUDP通信の場合にも使用されるが、HTTP Get Texture機能を使用している場合は、より厳密に適用されている。

通常の状態ではこのようなアルゴリズムは通信の一時的な集中を回避する上で有効であると思われるが、本システムのようにリージョン全体のテクスチャデータを一度に表示したい場合などには不都合である。

本システムにおいて HTTP Get Texture機能を使用した場合、アバターの周りのマップテクスチャのみが表示され（この部分は確かに高速である）、他の部分のマップは表示されないという状況に陥ってしまう。

5.2 プロキシシステムを用いた高速ダウンロード

上記の問題を解決するために、sl_proxy［8］と呼ばれる、セカンドライフ/OpenSim用プロキシシステムを利用する。sl_proxyは本来は大学や企業などの組織内からファイアウォールを越えて外部のセカンドライフやOpenSimに接続するためのアプリケーションゲートウェイであり、最新版は HTTP Get Texture機能にも対応している。

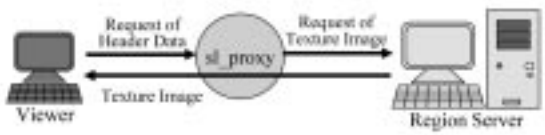

図7．sl_proxyによるリクエストデータの書き換え
Figure 7　Rewriting of request data by sl_proxy

sl_proxyではパケットの中継を行う際に、パケットの内容を書き換えることが可能である。つまり、ビューアからの、HTTP Get Textureのヘッダリクエストに対して、ヘッダ部分だけでなく、データ本体も転送するようにリクエストを書き換えてサーバへ転送することができる（図７）。

この場合、リージョンサーバから一度にマップテクスチャのデータがダウンロードされるので、HTTP Get Texture機能を使用している場合でも、高速にリージョン全体のマップテクスチャを表示することが可能となる。ただし、やはり通信が短時間の間に集中してしまうため、低速のPCを使用している場合には、ビューアの処理が一時的に遅くなってしまう場合がある。

6．今後の課題

国土地理院の数値地図データは、本システムのような使用方法を想定しておらず、その使用規約通りに従えば、本研究の成果を一般に公開することは難しい（ただし研究のための論文での使用は認められる）。

また、マップデータの使用に関しても、マップの使用枚数の制限や今後のサービスの継続性などの問題がある。例えば、Yahooマップでは現在のところ、１日のダウンロード枚数に制限は存在しないが、今後も制限がかからないとの保障は何処にもない。

以上の点を考慮すれば、本システムのようなシステムでは、全てのデータに関してフリーのデータを使用すべきであるように思われる。例えば、標高データは全てSRTM 3 とし、衛星写真としてはLandsat7［9］のデータを、マップ画像としてはOpenStreetMap［10］のデータを使用することなどが考えられる。

これらのマップテクスチャを予めローカルなマシンにダウンロードしておけば、マップテクスチャのダウンロード速度の問題も解決できる可能性がある。

本システムは現時点ではまだ日本国内の地形の再現しかサポートしていないが、SRTM 3 とLantsat 7 , OpenStreetMapのデータを組み合わせれば、世界中の任意の箇所の地形を再現することが可能となる。

また、OpenSimにはメガリージョンと呼ばれる、複数のリージョンをまとめて一つのリージョンと見なすモードが存在する。そのメガリージョンにおいて、巨大なジオラマを作成することが可能かどうかの検証も行いたい。

7. むすび

OpenSimのソースコードに改良を加えることにより、OpenSim内にリアルなジオラマを作成するシステムの構築を行った。

まだ速度やデータの使用規約などの問題が残るものの、一旦システムを組み上げれば、短時間に低コストで任意の場所のリアルなジオラマを作成することが可能となる

本システムとOpenSimのLSL/OSSLを組み合わせれば、地球温暖化による海水面の上昇に伴う水没地域の確認や地形変動のシミュレーションなどの興味深い現象をリアルタイムに再現することも可能である。

データさえ揃えれば、任意の場所を再現できるので地理教育や地域学習等にも使用可能である。また、 3 次元仮想空間を利用して 3 D動画を作成するマシニマと組み合わせれば、地域の広報用などにも使用できる可能性がある。

本システムは、類似したサービスであるGoogle Earthなどと比較すれば、速度や操作性などの点で見劣りする部分も存在するが、Google Earthには無い視点を与えることも可能であり、上記に述べたような様々なサービスへの応用の可能性も秘めていると言える。

なお、本システムはオープンソースとして、OpenSimの公式プロジェクトサイト［11］で公開を行っている。

【文献・参照】

［1］http://opensimulator.org/
［2］http://wiki.secondlife.com/wiki/LSL_Portal
［3］http://opensimulator.org/wiki/OSSL_Implemented
［4］http://www.gsi.go.jp/geoinfo/dmap/dem50m-index.html
［5］http://www2.jpl.nasa.gov/srtm/
［6］http://wiki.secondlife.com/wiki/Sculpted_Prim
［7］http://snowglobeproject.org/
［8］井関 文一、“Second Life用プロキシサーバの作成”，東京情報大学研究紀要、vol.12, No.1, pp.1-11, Chiba, Sep. 2008
［9］http://landsat.usgs.gov/
［10］http://www.openstreetmap.org/
［11］http://forge.opensimulator.org/gf/