SML を使ったインポートの自動化

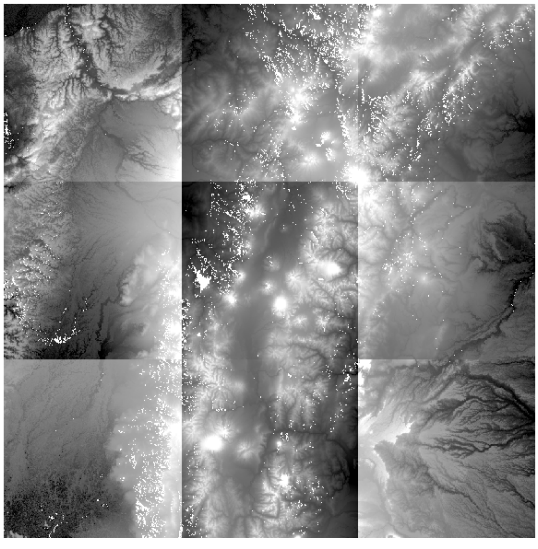
TNT 製品は、対話的なインポート / エクスポート処理でサポートされているデータ形式の種類の多さで広く知られています。しかし処理を行う特定の形式のファイルが多い場合、あるいはインポート処理が複数の処理ステップの最初の一段階にすぎない場合、対話的なダイアログはあまり効率的ではありません。空間操作言語の SML では、任意の形式の空間データのインポートおよびエクスポートを自動化するカスタムスクリプトを書くために必要なツールを提供しています。

SML のインポート / エクスポート関数と Mie クラスを使い、空間データをインポート / エクスポートする SML スクリプトを書きます。

インポート / エクスポートでサポートされているファイル形式には、それぞれに対して “Mie” (“MicroImages Import/Export” の頭文字) で始まる SML クラスがあり、各形式に固有のインポート / エクスポート設定をメンバーを使って指定します。適切な Mie... クラスに対してインスタンスを宣言し、そのクラスメンバーに対して希望する値を設定した後、インポート / エクスポートの関数グループの中からデータタイプや処理の目的に合った関数 (例えば ImportRaster() など) を呼び出します。スクリプトは Mie クラスの変数をこの関数のパラメータとして渡します。その関数はあなたが指定した設定を使って、指定されたデータ形式を処理します。

バッチによる一括インポートを自動化するスクリプトの例として、マイクロイメージ社は IMPORT_SRTM.SML を作成しました (このページの裏面に記載)。このスクリプトは NASA のスペースシャトル立体地形データ (SRTM) によって提供された標高グリッドデータをインポートするために作られました。この標高データは 16 ビット符号付き整数の標高値で、未処理のバイナリファイルとして配布されています。.hgt ファイルは 1201 行 x 1201 列の格子で、3 秒角間隔の緯度・経度でサンプリングされ、1 平方度の領域をカバーしています。これらのファイルをインポートするため、スクリプトは MieUSERDEFINEDRASTER クラスを使用しています。このクラスのメンバを使ってデータタイプやバイトオーダー、行数および列数、ラスタインポート用の他のパラメータを指定しています。

SML スクリプトは 1 つのディレクトリにある多数の標高ファイルをインポートするように作られています。SRTM 高度ファイル名には 1 度格子の左下 (南西) 隅の緯度経度の情報があるので、この情報を使ってインポートしたグリッドデータにジオリファレンスすることができます。ファイル毎にスクリプトはファイル名を解析して基準の緯度経度を求め、ファイルをインポートし、最後に、隅にコントロールポイントによるジオリファレンスサブオブジェクトを作成します。マイクロイメージ社ではこのスクリプトを使用して、南米のデータ (全 4.9 GB、1,813 ファイル) をインポートしました。



上図は、インポートした南米の SRTM 標高グリッドデータの内、3 x 3 個分を表示しています (エクアドルの一部)。それぞれの標高ラスタは緯度・経度の 1 度の範囲をカバーしており、個別の標高レンジを基にコントラストストレッチを行っているので、境界部でグレイトーンが連続しません。不規則に分布する白い領域は、レーダの影のヌル領域 (データ無し) です。

SRTM データについて

2000 年 2 月に打ち上げられたスペースシャトルエンデバーによって行われた SRTM ミッション。このミッションでは、地球の陸地面積の 80% 以上もの地形データを収集するために、2 個のレーダアンテナによる干渉画像レーダシステムを使用しました。予備的な生のデジタル標高データが研究者や一般に配布されました。データのない領域や水平でない水域、不明確な海岸線を多く含んでおり、まだ完全に処理された分けではありません。修正されたデータは後日配布される予定です。SRTM データの詳細は、http://www.jpl.nasa.gov/srtm/ をご覧下さい。ftp://edcsgs9.cr.usgs.gov/pub/data/srtm/ からダウンロードできます。

(翻訳) 株式会社 オープン GIS　東京都墨田区吾妻橋 1-19-14 紀伊国屋ビル 1F
Tel: (03)3623-2851　Fax: (03)3623-3025　E-mail: info@opengis.co.jp

TNTmips の空間操作言語 (SML) の使い方を紹介したサンプルスクリプトが用意されています。ここには可能な限り、スクリプト全部を掲載するようにしていますが、ページに収まらない場合は重要な部分のみ掲載しています。サンプルスクリプトは、www.microimages.com/freestuf/smlscripts.htm からダウンロード出来ます。

SRTM 標高グリッドデータを一括してインポートするスクリプト (import_srtm.sml)

このスクリプトでは ' ヘッダのない ' *.hgt SRTM-3 ファイルをインポートし、ジオリファレンスを設定します。

```
clear();
class FILEPATH filepath;
class STRINGLIST filenames;
class MieUSERDEFINEDRASTER srtm;
```

*.hgt ファイル名から緯度の値を取得します

```
func getLatitude( hgtfile$ ) {
    local numeric latvalue = StrToNum( GetToken( hgtfile$, "NSEW.hgt", 1, 1 ) );
    string hemisphere = GetToken( hgtfile$, "EW.hgt0123456789", 1, 1 );

    if ( hemisphere == "N" )  {
        return latvalue;
    }
    else if ( hemisphere == "S" ) {
        return -latvalue;
    }
    else    {
        print( "Error parsing filename for latitude" );
        print( "Bad file was: " + hgtfile$ );
        return -9999;
    }
}
```

北半球の場合

南半球の場合

*.hgt ファイル名から経度の値を取得します

```
func getLongitude( hgtfile$ ) {
    local numeric longvalue = StrToNum(GetToken(hgtfile$, "NSEW.hgt",2,1) );
    string hemisphere = GetToken( hgtfile$, "NS.hgt0123456789", 1, 1 );

    if ( hemisphere == "E" ) {
        return longvalue;
    }
    else if ( hemisphere == "W" )    {
        return -longvalue;
    }
    else    {
        print( "Error parsing filename for longitude" );
        print( "Bad file was: " + hgtfile$ );
        return -9999;
    }
}
```

東半球の場合

西半球の場合

USERDEFINEDRASTER によるインポート用のパラメータを設定します

```
proc setImportParameters(path$)    {
    srtm.NumLins = 1201;
    srtm.NumCols = 1201;
    srtm.DataType = "16-bit signed";
    srtm.ByteOrder = "High-Low";
    srtm.UseFile = 0;
    srtm.DoPyramid = 1;
    srtm.DoCompress = 1;
}
```

SRTM データのあるディレクトリを取得します

```
string defaultpath$ = _context.ScriptDir;
filepath.SetName( GetDirectory( defaultpath$,
        "Please select the directory containing the SRTM files for import" ) );
setImportParameters( filepath.GetPath() );
print( filepath.GetPath() );
filenames = filepath.GetFileList( "*.hgt" );

numeric filecount = filenames.GetNumItems();
print( filecount, "files found" );
```

出力ファイル名を取得します

```
string outputfile$ = GetOutputFileName(filepath.GetName(),
        "Enter a file name for the imported srtm raster rvc", ".rvc" );

numeric lat, long;
numeric i;
```

```
for i = 0 to filecount-1    {
    print( "Now importing: " + filenames.GetString( i ) + " raster number: " +
        NumToStr( i + 1 ) );
    lat = getLatitude( filenames.GetString( i ) );
    long = getLongitude( filenames.GetString( i ) );

    if ( lat != -9999 && long != -9999 )    {
        string inputfile$ = filepath.GetPath() + "\\" + filenames.GetString( i );
        string objname$ = FileNameGetName( filenames.GetString( i ) );

        ImportRaster( srtm, inputfile$, outputfile$, objname$,
            "file imported from: " + filenames.GetString( i ) );
```

ラスタのインポート

ジオリファレンスのためラスタを開きます

```
        Raster importedRaster;
        OpenRaster( importedRaster, outputfile$, objname$ );

        SetNull( importedRaster, -32768 );

        numeric xmin, ymin, xmax, ymax;
        xmin = 0.5;         # left edge
        xmax = 1200.5;      # right edge
        ymin = 0.5;         # top edge
        ymax = 1200.5;      # bottom edge
```

隅のセルの中心を基に、(オブジェクト座標で) ラスタ範囲を設定します

ラスタの四隅にコントロールポイントを作成するため、オブジェクトの範囲を 'source'、緯度・経度値を 'dest' 配列に保存します：
[1] = 左上隅
[2] = 左下隅
[3] = 右上隅

```
        array numeric sourceX[3];
        array numeric sourceY[3];
        array numeric destX[3];
        array numeric destY[3];

        sourceX[1] = xmin;
        sourceY[1] = ymin;
        sourceX[2] = xmin;
        sourceY[2] = ymax;
        sourceX[3] = xmax;
        sourceY[3] = ymin;

        destX[1] = long;
        destY[1] = lat + 1;
        destX[2] = long;
        destY[2] = lat;
        destX[3] = long + 1;
        destY[3] = lat + 1;
```

投影パラメータを設定します

```
        class MAPPROJ mapproj;
        mapproj.SetSystemLatLon();
        mapproj.Datum = "World Geodetic System 1984";
```

'source' と 'dest' 配列を使用し、オブジェクトに対してコントロールポイントによるジオリファレンスを作成します。

```
        CreateControlPointGeorefDefaultAccuracy( importedRaster, mapproj, 3,
            sourceX, sourceY, destX, destY );

        CloseRaster( importedRaster );
    }
    else    {
        print( "Check file name and structure, lat/long not computed correctly" );
    }
}
```

必要であれば、最後に表示するステートメントを設定します

```
string s$ = "s";

if ( filecount == 1 )    {
    s$ = "";
}
```

ユーザに終了を知らせます

```
print( "Done. ", filecount, "raster" + s$ + " imported." );
```

(翻訳) 株式会社 オープン GIS　東京都墨田区吾妻橋 1-19-14 紀伊国屋ビル 1F
Tel: (03)3623-2851　Fax: (03)3623-3025　E-mail: info@opengis.co.jp