2009 年 12 月 18 日

LAS LIDAR ポイントファイルの処理

TNTmips Pro を使うと、LAS 標準ファイル形式の LIDAR ポイントファイルを元の形式のまま直接表示、使用できます。TNTmips では、リンクされた LAS ファイルはシェイプオブジェクトとして扱われます。TNT の地理空間スクリプト言語 (SML) で書かれた処理スクリプトを使っても、LAS ファイルから LIDAR ポイントデータに直接アクセスしたり、ポイントを処理したり、処理結果を格納するための新規 LAS ファイルを作成できます。いかなる処理過程においても、LIDAR ポイントを TNT の地理空間形式にインポートする必要がありません。SML を使うと元の LAS ファイルのまま直接作業が行えるため処理時間を大幅に短縮でき、何百万ものポイントを含んだファイルをインポートするのにかかっていた時間を割くことができます。

SML 内の RVC_SHAPE クラスを用いてリンクした LAS ファイルを表します。このクラスには MakeLAS() メソッドが含まれており、処理結果を格納する新規 LAS ファイルを作成する際に使用されます。このメソッドを使って LAS Ver. 1.2 でサポートされている LAS ポイントデータレコード形式 (0、1、2、3) の LAS ファイルを作成します。入力 LAS ファイルをテンプレートとして使用し、出力ファイルの形式を設定できます。

LAS ファイルでは、各 LIDAR ポイントの地理座標と属性をデータベーステーブルの 1 つのレコードとして格納します。そのため、リンクした LAS ファイル内の各ポイントはシェイプオブジェクトのデータベーステーブルの 1 レコードとしてアクセスできます。また、通常のデータベースの方法 (データベース、テーブル、レコード、フィールド) を使って、データを読み込み、コピー、修正して、出力 LAS ファイルへ書き出すことが出来ます。

SML を使用した LAS ファイルの直接処理のデモのために、マイクロイメージ社はサンプルスクリプトをいくつか用意しました。2 ページ目にスクリプトの抜粋を掲載しました。スクリプトは以下のページから入手できます：https://www.microimages.com/sml/smlsamples-htm/Lidar.htm

LAS_GROUND スクリプトでは、SML スクリプトが LIDAR のポイント分類情報を使って、どのように制御処理をするかを示しています。このスクリプトでは、「地面 (Ground)」と分類されたポイントだけを新規の LAS ファイルにコピーします。次にこれらの地面を表わすポイントを入力として地表面の標高ラスタを作成します。LASextractByRegion スクリプトでは、リージョンを使ってリージョン内の全ての LIDAR ポイントを新規 LAS ファイルに抜き出しています。

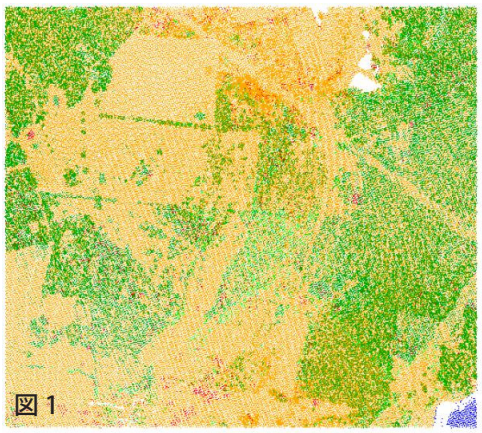
図 1

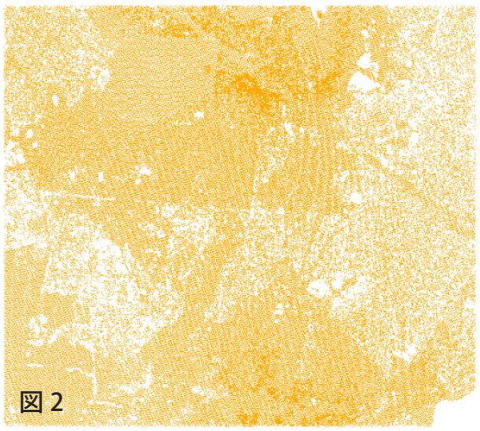
図 2

216,055 個のポイントが入った LAS LIDAR ポイントファイル (図 1)。ファイル内のポイントは左の凡例にあるカテゴリに分類されています。LAS_GROUND サンプルスクリプトを使って「地面 (Ground)」と分類されている全てのポイントを抜き出し、ポイント数 106,466 個の新規 LAS ファイルを作成しました (図 2)。

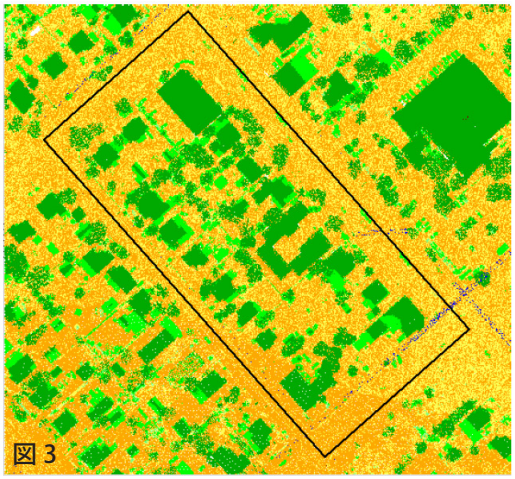
図 3

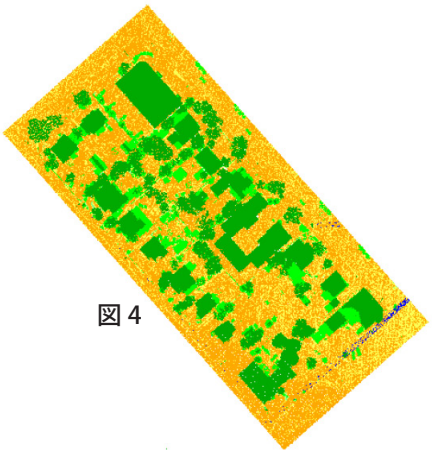
図 4

市街地の空間密度が高い大きな LAS LIDAR ポイントファイル (図 3)。ファイルには 9,863,071 個のポイントが含まれており、その多くは (自動処理によって) 地面や植生として分類されています。植生のカテゴリには建物も含まれています。黒の四角はリージョンオブジェクトを示すもので、いくつかのブロックの建物が含まれています。LASextractByRegion.sml サンプルスクリプトを使ってこのリージョン内の全てのポイントを抜き出し、ポイント数 276,420 個の新規 LAS ファイルを作成しました (図 4)。

www.microimages.com/downloads/scripts.htm にはダウンロード可能な多くのサンプルスクリプトがあります。スクリプトやクエリーで TNT 製品のスクリプト言語をどのように利用したらよいか解説しています。

LAS_GROUND.sml スクリプト抜粋

```
class RVC_SHAPE lasIn;
class RVC_OBJITEM objItemIn;
DlgGetObject("Select input LAS shape object:", "Shape", objItemIn,
"ExistingOnly");
lasIn.Open(objItemIn, "Read");

class RVC_GEOREFERENCE georef;
lasIn.GetDefaultGeoref(georef);

class RVC_DBASE_SHAPE dbIn;
dbIn.OpenAsSubobject(lasIn, "Read");

class RVC_DBTABLE tableIn;
tableIn.Open(dbIn, 0, "Read");

class FILEPATH path = GetOutputFileName("output.las",
        "Select LAS file to make:", "las");

class RVC_SHAPE lasOut;
lasOut.MakeLAS(path, georef.GetCoordRefSys(), tableIn);

class RVC_DBASE_SHAPE dbOut;
dbOut.OpenAsSubobject(lasOut, "Write");
class RVC_DBTABLE tableOut;
tableOut.Open(dbOut, 0, "Write");

class RVC_DBTABLE_RECORD recordIn(tableIn);
class RVC_DBTABLE_RECORD recordOut(tableOut);
class RVC_RECORDNUM recordNum;

for i = 1 to tableIn.GetNumRecords()
   {
   recordNum.Number = i;
   tableIn.Read(recordNum, recordIn);
   if (recordIn.GetValue("Classification") == 2)
      {
      recordIn.CopyTo(recordOut);
      tableOut.AddRecord(recordOut);
      }
   }
```

入力 LAS シェイプオブジェクトを入手

入力 LAS ファイルからデフォルトのジオリファレンスを入手

読み込み用に入力シェイプデータベースとメインテーブル (テーブルナンバー：0) を開く

出力 LAS ファイル用のファイルパスを入手

地面を表わすポイント用の出力 LAS ファイルを作成。既存の LAS ファイルに対して RVC_DBTABLE クラスインスタンスを入手するメソッドを使い、新規 LAS ファイルに対して同じポイントデータのレコードタイプを設定します。

書き込み用にシェイプデータベースとメインテーブルを開く。

レコードの読み込みやコピー、書き込み用のレコードクラスインスタンス

レコード番号の変数

LIDAR ポイントレコードをループして、地面と分類されたポイントを検索

入力 LAS からレコードを読み込む

Classification フィールド内の値をチェックし、地面を表わすポイントのみをコピーします。

フィールド値を入力のレコードから出力用の新規レコードにコピーします。

新規レコードを出力 LAS ファイルに書き込みます。

LASextractByRegion.sml スクリプト抜粋

```
class RVC_SHAPE lasIn;
class RVC_OBJITEM objItemIn;

class SR_COORDREFSYS crs;

class RECT3D lasExtents;

class REGION Reg;

proc selectRegion () {
  GetInputRegion(Reg);
  Reg.ConvertTo(crs);

  if (lasExtents.Overlaps(Reg.Extents) == 0) {
     PopupMessage("Region selected does not overlap LAS file extents;
        please select another ");
     selectRegion();
     }
  }
```

シェイプオブジェクトとしてリンクされた入力 LAS ファイル

入力 LAS ファイルの座標参照系

座標参照系における入力 LAS ファイルの範囲

選択したリージョンオブジェクト

リージョンオブジェクトを選択し、その範囲が LAS ファイルと重なっているかチェック。

```
DlgGetObject("Select input LAS shape object:",
      "Shape", objItemIn, "ExistingOnly");
lasIn.Open(objItemIn, "Read");

class RVC_GEOREFERENCE georef;
lasIn.GetDefaultGeoref(georef);
crs = georef.GetCoordRefSys();

lasIn.GetExtents(lasExtents);

class RVC_DBASE_SHAPE dbIn;
dbIn.OpenAsSubobject(lasIn, "Read");

class RVC_DBTABLE tableIn;
tableIn.Open(dbIn, 0, "Read");

selectRegion();

class FILEPATH path = GetOutputFileName("output.las",
      "Select LAS file to make:", "las");

class RVC_SHAPE lasOut;
lasOut.MakeLAS(path, crs, tableIn);

class RVC_GEOREFERENCE georefOut;
lasOut.GetDefaultGeoref(georefOut);
printf("Input CRS: %s\n", georefOut.GetCoordRefSys().Name);

class RVC_DBASE_SHAPE dbOut;
dbOut.OpenAsSubobject(lasOut, "Write");

class RVC_DBTABLE tableOut;
tableOut.Open(dbOut, 0, "Write");

class RVC_DBTABLE_RECORD recordIn(tableIn);
class RVC_DBTABLE_RECORD recordOut(tableOut);
class RVC_RECORDNUM recordNum;

class STATUSCONTEXT status;
class STATUSDIALOG statusDLG;
statusDLG.Create();
status = statusDLG.CreateContext();
status.BarInit(tableIn.GetNumRecords(), 0);
status.Message = "Processing LIDAR points...";

class POINT2D pt;

for i = 1 to tableIn.GetNumRecords()
   {
   status.BarUpdate(i, tableIn.GetNumRecords(), 0);
   recordNum.Number = i;
   tableIn.Read(recordNum, recordIn);

   pt.x = recordIn.GetValue("X");
   pt.y = recordIn.GetValue("Y");

   if (Reg.IsPointInside(pt) )
      {
      recordIn.CopyTo(recordOut);

      tableOut.AddRecord(recordOut);
      }
   }
statusDLG.Destroy();
```

入力 LAS シェイプオブジェクトを入手

入力 LAS ファイルからデフォルトのジオリファレンスを入手

リージョンと比較するため、LAS シェイプオブジェクトの範囲を入手

読み込み用に入力シェイプデータベースとメインテーブル (テーブルナンバー：0) を開く。

ユーザ定義の手順を呼び出し、抜き出しリージョンを選択・チェックします。

出力 LAS ファイル用のファイルパスを入手

抜き出したポイント用の出力 LAS ファイルを作成。既存の LAS ファイルに対して RVC_DBTABLE クラスインスタンスを入手するメソッドを使い、新規 LAS ファイルに対して同じポイントデータのレコードタイプを設定します。

書き込み用にシェイプデータベースとメインテーブルを開く。

レコードの読み込みやコピー、書き込み用のレコードクラスインスタンス

レコード番号の変数

LIDAR ポイントレコードをループして、リージョン内のポイントを検索

入力 LAS からレコード変数にレコードを読み込みます。

カレントポイントの地図上の位置を入手

これらの地図座標が抜き出したリージョンの内側にあるかチェックします。

フィールド値を入力のレコードから出力用の新規レコードにコピーします。

新規レコードを出力 LAS ファイルに書き込みます。