Google EarthとRの連携(2)
―rcomによる地図の操作―

データキューブ㈱医療情報システム部主任
牧山　文彦（Makiyama Fumihiko）

■琉球大学医学部大学院保健学研究科修了。女子栄養大学大学院保健学研究科修了、保健学博士（学術）。琉球大学医学部保健管理学助手、秀和綜合病院企画・電算室学術情報主任、ちばなクリニック健康管理センター事務チーフを経て、2007年4月より現職。E-mail：fumihiko.maki@gmail.com

1. RからGoogle Earthを操作する

Google EarthはCOM API[1]が公開されているため、COMインターフェースを持つ外部のアプリケーションから操作することが可能である。

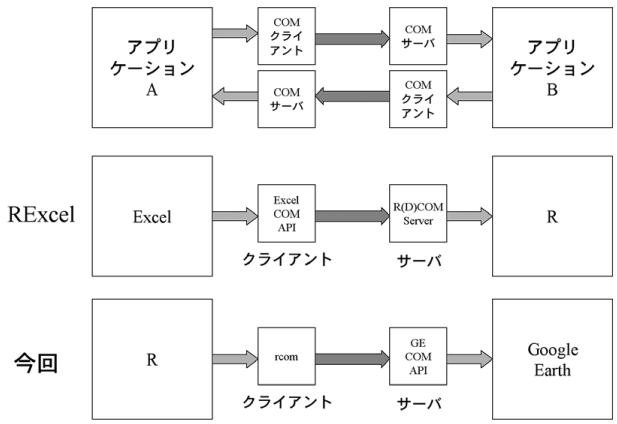
また、Rには、R(D)COM ServerおよびrcomというCOMインターフェースをRに実装するパッケージが存在している（詳しくは、RjpWikiのRDCOM/RCOM解説を参照[2]）。

R(D)COM Serverは、RにCOMサーバの機能を実装するパッケージであるが、CRANには登録されていないため、サイトからダウンロードしてインストールする必要がある。一方、rcomはCRANに登録されているため、通常のパッケージのインストールで実装できる。

R(D)COM ServerはRをCOMサーバ化し、rcomはRをCOMクライアント化するという違いがあり、R(D)COM ServerにはRexcelというExcelからRを利用するツールが付随している。

今回は、RからGoogle Earthを操作することが目的であるため、rcomパッケージを用い、RからGoogle Earthを直接操作することを試みる（図1）。

図1　COMの仕組み



2. rcomパッケージ

まず、rcomパッケージを通常の方法でRにインストールしておいてほしい（Google Earthはすでにインストール済みであること。また、

一連のコードはRjpWiki「空間的なデータの分析」[3]の地図作成練習用データrcomdemo.txtにあるので、ダウンロードしてC:/GISdataフォルダに入れておいてほしい)。

パッケージがインストールされ、準備が整ったなら早速起動する。

```
>library(rcom)
```

これで、COMクライアントが利用できるようになった。

次に、Google Earthを呼び出してみよう。

```
>x<-comCreateObject("GoogleEarth.ApplicationGE")
```

Google Earthが起動して、おなじみの地球が表示されたと思う。

ここでは、Google Earth COM APIのApplicationGEクラスを呼び出してオブジェクトを生成している。Google Earth COM APIには多くのクラスが実装されており、そのクラスを呼び出してオブジェクトを生成することで、種々の操作が可能となる。

xの中身を見てみると、

```
>x
<pointer: 0x000aa584>
attr(,"class")
[1]"COMObject"
```

と表示される(「0x000aa584」の部分は任意に設定されるので、同じではない)。

また、クラス構造を見てみると、

```
>str(x)
Class 'COMObject'<externalptr>
```

となっているので、COMObjectというクラスであることがわかる。

rcomはApplicationGEクラスに接続する際に、ApplicationGEと同じ構造のCOMObjectというクラスを生成してRで利用できるようにする。そのため、理論的にはCOM APIを持つどんなアプリケーションとでもCOM接続することができる仕組みになっている。

表1には、Google Earth COM APIのクラスを示した。実際の細かい操作はクラス内のプロパティやファンクションを呼び出すことで実行される。クラス内のプロパティやメソッドについては、Google Earth COM APIのサイトを参照されたい[1]。

表1　Google Earth COM APIクラス

IAnimationControllerGE
IApplicationGE
ICameraInfoGE
IFeatureCollectionGE
IFeatureGE
IPointOnTerrainGE
ISearchControllerGE
ITimeGE
ITimeIntervalGE
ITourControllerGE
IViewExtentsGE

3. Google Earthのバージョンの取得

まず手始めに、Google Earthのバージョンの取得を行ってみる。

```
>comGetProperty(x,"VersionMajor") #バージョン（メジャー番号）
[1] 4
>comGetProperty(x,"VersionMinor") #バージョン（マイナー番号）
[1] 2
>comGetProperty(x,"VersionBuild") #バージョン（ビルド番号）
[1] 198
```

ここで利用しているのは、ApplicationGEのプロパティで、VersionMajor、VersionMinor、VersionBuildの3つである。

Google Earthメニューバーの[ヘルプ][Google Earthについて]を選択して表示されるウィンドウの一番上の行、Google Earthの文字の横を確認すると「4.2.0198.****(beta)」と表示されていると思う（数字自体は、インストールされているGoogle Earthのバージョンによって異なる）。

このように、RからGoogle Earthのバージョン情報の取得ができた。

rcomでは、プロパティの中の情報を取り出すには、**comGetProperty**( )関数を用いる。引数は、[ApplicationGEObject名, "プロパティ名"]となる。VisualBasic等で一般的な[オブジェクト名. プロパティ名]ではないので注意が必要である。

4. 視点（カメラ）の移動

さて、次にGoogle Earthの視点（カメラ）を移動させてみよう。

```
>cam<-comCreateObject
("GoogleEarth.CameraInfoGE")
>comSetProperty(cam,"Focus
PointLatitude","35.651015")
#フォーカス－緯度
NULL
>comSetProperty(cam,"Focus
PointLongitude","139.727556")
#フォーカス－経度
NULL
>comSetProperty(cam,"Range",
"100") #フォーカスからの距離
（上空：m）
NULL
>comSetProperty(cam,"Tilt",
"50") #傾き（度）
NULL
>comSetProperty(cam,
"Azimuth","111") #方位（度）
NULL
>comInvoke(x,"SetCamera",
cam,"0.1") #実行
[1] NA
```

まず、CameraInfoGEクラスからcamオブジェクトを生成する。

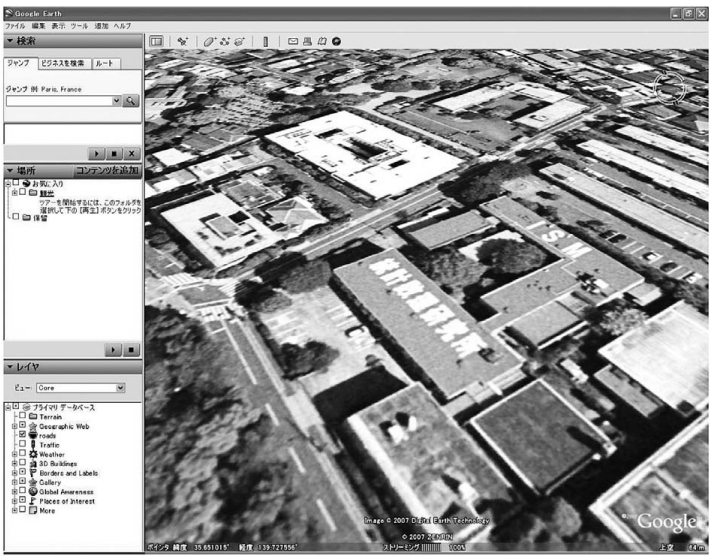
次に、プロパティに視点のデータをセットする。セットするデータは、フォーカス（緯度、経度）、フォーカスからの距離（上空何mか）、カメラの傾き（度）、カメラの方位（度）である。

データをプロパティにセットするためには、**comSetProperty**( )関数を用いる。この関数の引数は、[CameraInfoGEObject名, "プロパティ名", "データ"]となる。

データがセットできたところで、実際に視点を移動させる。始点の移動を実行するには、**comInvoke**( )関数を用いる。ここでの引数は、[ApplicationGEObject名, "SetCamera"ファンクション, CameraInfoGEObject名, "移動スピード"]となる。

実際の動きでは、comInvoke( )関数が実行されたところで、ゆっくり視点が移動して統計数理研究所の上空に停止する（図２）。

図２　視点（カメラ）の移動



5. カメラの位置のデータの取得

次に、移動したカメラの位置のデータを取得してみる。

```
>cam0<-comInvoke(x,"GetCamera","1")
>comGetProperty(cam0,"FocusPointLatitude")
#フォーカス－緯度
[1] 35.65101
>comGetProperty(cam0,"FocusPointLongitude") #フォーカス－経度
[1] 139.7276
>comGetProperty(cam0,"FocusPointAltitude")
#フォーカス－高度（m）
[1] 0
>comGetProperty(cam0,"FocusPointAltitudeMode") #フォーカス－高度の基準
[1] 1
>comGetProperty(cam0,"Range") #フォーカスからの距離（上空：m）
[1] 100
>comGetProperty(cam0,"Tilt") #傾き（度）
[1] 50
>comGetProperty(cam0,"Azimuth") #方位（度）
[1] 111
```

まず、comInvoke( )関数で"GetCamera"ファンクションを用いて現在のカメラのオブジェクトを生成する。引数の中の"1"はTerrainの指定となる。後は、生成されたカメラオブジェクトから種々のプロパティの中のデータを取り出すだけである。

さて、取り出されたデータは、前節で設定したデータとほぼ同じであるが、フォーカスの緯度・経度のデータについては、小数点以下の桁数が、緯度については5桁、経度については4桁に丸められている。

6. 検索の実行

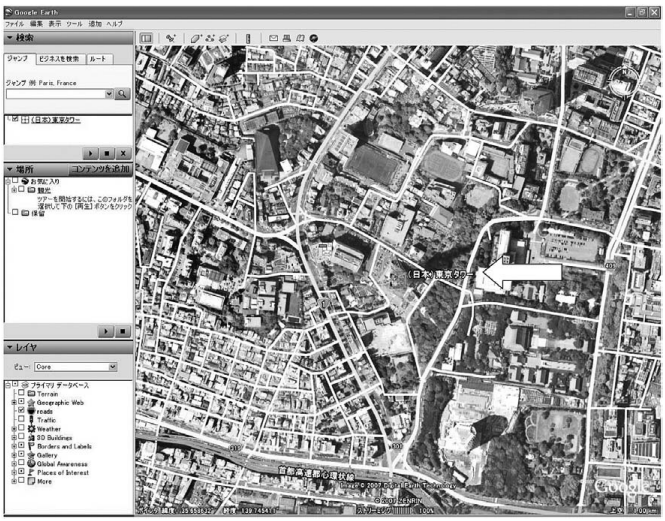
Google Earthは検索のジャンプのタブ内に「住所」ではなく「名前」を指定するだけで、その場所にジャンプさせることが可能となっている。

```
>search<-comCreateObject("GoogleEarth.SearchControllerGE")
>comInvoke(search,"Search","東京タワー")
[1] NA
```

Google Earthで検索させるためには、サーチオブジェクトを生成させ、次に、引数に検索する名称を指定するだけである。

上記のコードを実行すると、東京タワーの真上に移動して停止する（図3）。

図3　東京タワーの検索



7. KMLを直接表示させる

「Google EarthとRの連携(1) ― 3D地図の作成 ―」の回（2007年12月号）では、KMLを一旦ファイルとして出力し、その後Google Earthに読み込ませることで表示させたが、今回はR内部で生成したKMLを直接Google Earthに渡して表示させてみる。

表2に、コード（rcomdemo.txt内、練習5）を示した。

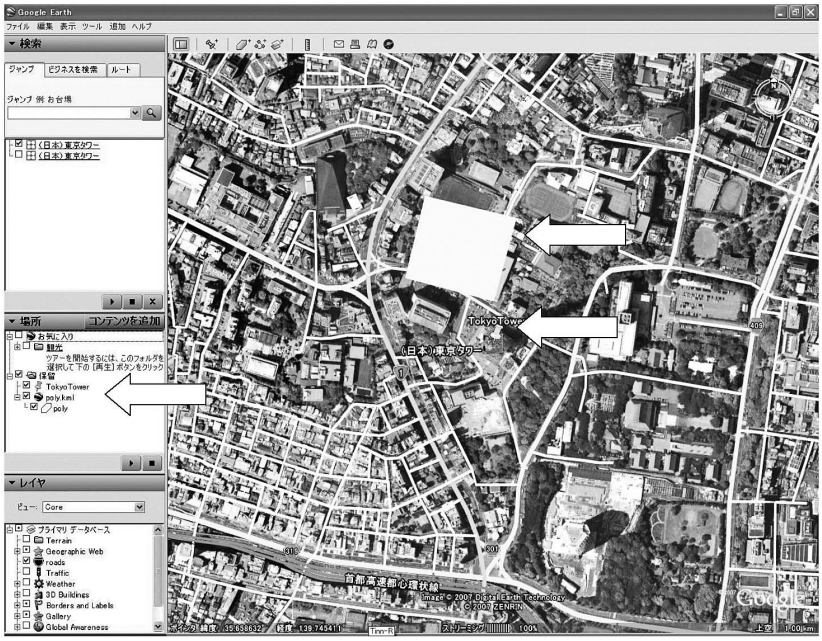
まず、2種類のKMLデータ（KMLData、KMLData2）を作成する。

次に、ApplicationGEオブジェクトを生成した後、comInvoke()関数で"LoadKmlData"ファンクションを実行させる。引数にはKMLData

表2　rcomdemo.txt（練習5）

```
#練習5
KMLData<-paste('<?xml version="1.0" encoding="UTF-8"?> ',
'<kml xmlns="http://earth.google.com/kml/2.2">',
'<Document>',
'<name>poly.kml</name>',
'<Placemark>',
'<name>poly</name>',
'<styleUrl>#msn_ylw-pushpin</styleUrl>',
'<Polygon>',
'<tessellate>1</tessellate>',
'<altitudeMode>relativeToGround</altitudeMode>',
'<outerBoundaryIs>',
'<LinearRing>',
'<coordinates>',
'139.7441431187752,35.66085751628702,50 139.7437770251541,35.65962355418283,50
139.7454177799272,35.65927100666627,50 139.7458381440458,35.66052699303938,50
139.7441431187752,35.66085751628702,50',
'</coordinates>',
'</LinearRing>',
'</outerBoundaryIs>',
'</Polygon>',
'</Placemark>',
'</Document>',
'</kml>',sep='')
KMLData2 <-paste("<?xml version=¥"1.0¥" encoding=¥"UTF-8¥"?>",
<kml xmlns=¥http://earth.google.com/kml/2.1¥">",
<Placemark>,
<name>TokyoTower</name>,
<LookAt>,
<longitude>139.4285</longitude>,
<latitude>35.3874</latitude>,
<altitude>0</altitude>,
<range>316.3662914479763</range>,
<tilt>0</tilt>,
<heading>6.199453434125936</heading>,
</LookAt>,
<Point>,
<coordinates>139.745408,35.658617,0</coordinates>,
</Point>,
</Placemark>,
</kml>,sep="")
GEI<-comCreateObject("GoogleEarth.ApplicationGE")
comInvoke(GEI,"LoadKmlData",KMLData)
comInvoke(GEI,"LoadKmlData",KMLData2)
```

図４　KMLデータの直接書き込み



およびKMLData2を直接渡す。

正常に実行されると、東京タワーの左上に四角い白いポリゴンが上空50mの地点に表示され、更に東京タワーの地点にTokyoTowerの黄色いピンが立つ。また、渡されたKMLデータは「場所」タブ内の「保留」フォルダ内に保存される（図４）。

ただし、現在のところLoadKmlDataファンクションにはいくつかの制限が見つかっている。「東京タワー」のようなマルチバイト文字は画面表示で文字化けを起こすこと、大きな容量のKMLデータを渡すとGoogle Earth自体がフリーズしてしまうことである。

また、LoadKmlDataファンクション経由だと描画スピードが遅くなるため、大量のKMLデータを描画させることには向かない。

従来のように、一旦ファイルに落としてからGoogle Earthに読み込ませる方法が安全で高速である。

さて、このように、rcomを用いるとRからGoogle Earthを操作することができた。

Google Earth COM APIはGoogle Earthのほとんどの機能を提供していることから、rcomとRコマンダー（Rcmdr）を組み合わせることで、Google Earth操作用のGUIを作成することも可能となり、RとGoogle Earthのみで3DGISを作ることも夢ではない。

＊参考文献・URL

[1] Google Earth：COM API（http://earth.google.com/comapi/）.
[2] RjpWiki：RDCOM/RCOM解説（http://www.okada.jp.org/RWiki/?RDCOM%2FRCOM%B2%F2%C0%E2）.
[3] RjpWiki：空間的なデータの分析（http://www.okada.jp.org/RWiki/?%B6%F5%B4%D6%C5%AA%A4%CA%A5%C7%A1%BC%A5%BF%A4%CE%CA%AC%C0%CF）.