KML LIBRARY FOR MATLAB

1. INTRODUCTION

The kml library allows to represent routes in Google Earth using Matlab very easily. How to use these libraries is described next.

KML LIBRARY FUNCTIONS

This library contains a set of functions that produces kml files that can be viewed in Google Earth. There are three main types of figures that can be drawn with the following functions:

- `function kmlwrite_polyline(wp, filename,attr);`

  It draws a polyline defined by a number of points.

- `function kmlwrite_polygon(wp, filename,attr);`

  It draws a polygon defined by a number of points.

- `function kmlwrite_volume(wpu, wpl, filename,attr);`

  It draws a volume defined by two surfaces (polygons) with the same number of edges.

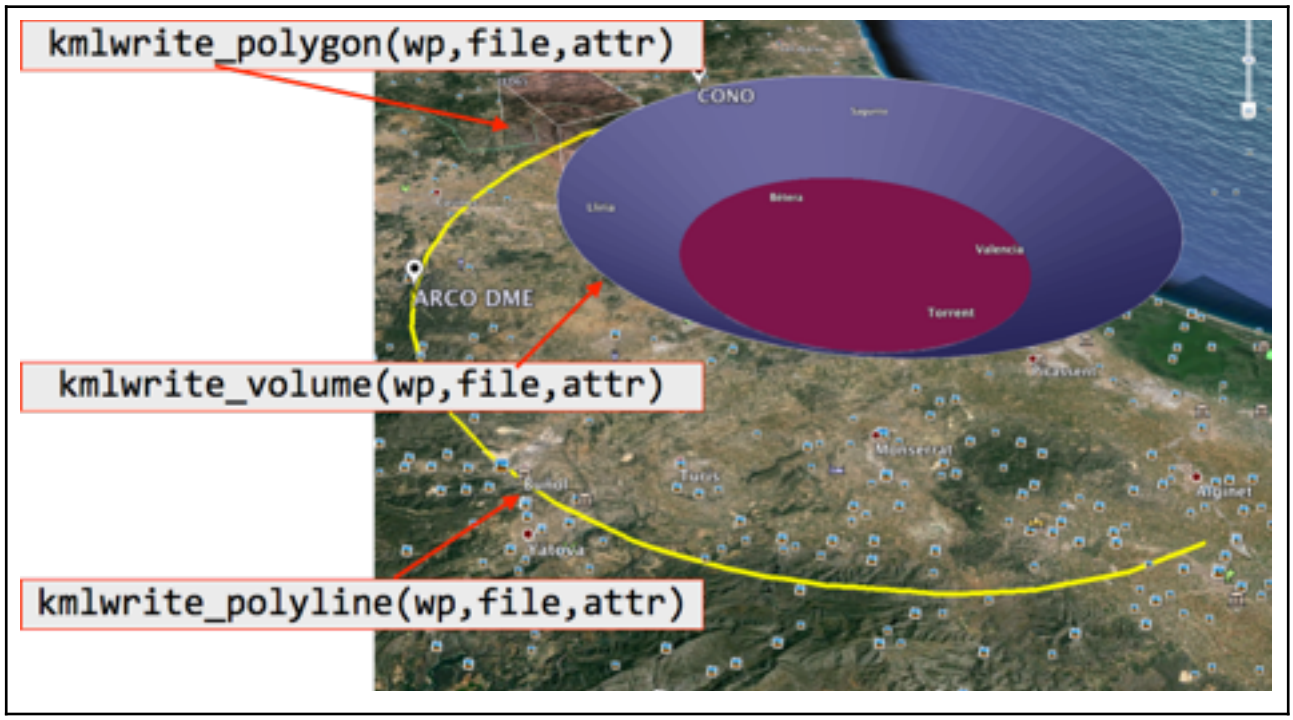

Figure 1

There is also a function for defining the "look" or visual attributes of these figures:

- `function create_attr();`

There is a function for generating circles:

- `function [xp,yp]=circle(x,y,r,phi1,phi2,n);`

And there is a file with examples of how to use all previous functions:

- `function examples_kmlwrite;`

This function produces the graphic of Figure 1.

2. THE KMLWRITE_POLYLINE FUNCTION IN DETAIL

All kmlwrite functions work similarly. The following example, describes how to generate the route with four waypoints of figure 2 for Google Earth using **`kmlwrite_polyline`**.

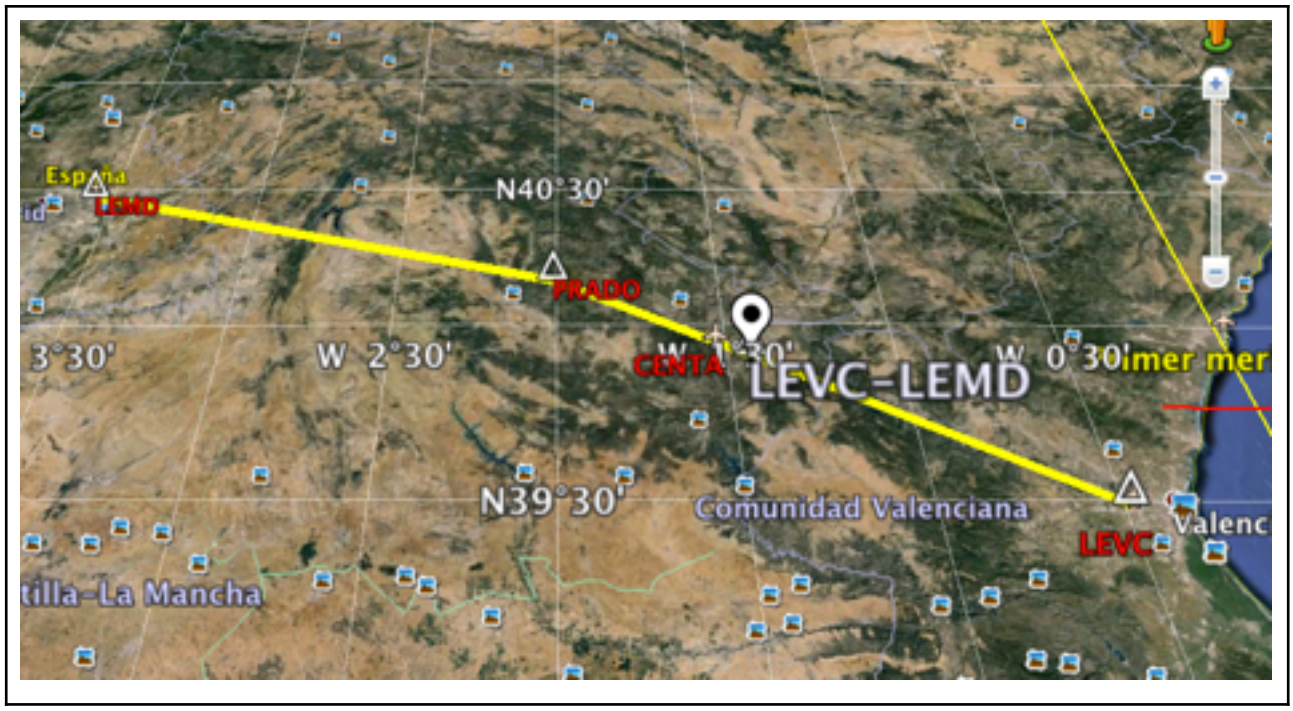

Figure 2

Step 1: Add the path of the libraries to your program.

```
addpath('kml','geo');
```

Step 2: Define a line (or a polygon) through a number of waypoints `wp` in Matlab.

The `wp` data structure is an array of structs which must contain the following fields for each waypoint: `name, desc, lat, long`, and `alt`.

These are the four waypoints that define the route:

```
wp(1).name= 'LEVC';
wp(1).desc= 'This is wp1';
wp(1).lat = sex2dec('N039°29''22.00"');
wp(1).lon= sex2dec('W000°28''54.00"');
wp(1).alt=0;

wp(2).name= 'CENTA';
wp(2).desc= 'This is wp2';
wp(2).lat = sex2dec('N039°54''02.22"');
wp(2).lon= sex2dec('W001°25''55.21"');
wp(2).alt=30000;

wp(3).name= 'PRADO';
wp(3).desc= 'This is wp3';
wp(3).lat = sex2dec('N040°08''50.96"');
wp(3).lon= sex2dec('W002°00''37.23"');
wp(3).alt=40000;

wp(4).name= 'LEMD';
wp(4).desc= 'This is wp4';
wp(4).lat = sex2dec('N040°28''20.00"');
wp(4).lon= sex2dec('W003°33''39.00"');
wp(4).alt=0;
```

Step 3: Create a structure of visual attributes of these route using function create_attr.

Create an `attr` variable and initialize it by making this call:

```
attr=create_attr();
```

This function creates and gives default values to the visual attributes of a **polyline**, a **polygon**, or a **volume.** These attributes can be grouped into General, Upper surface, Lower surface, Lateral Surfaces, and Waypoints as shown in figure 3.

The function source code is self-explicative. So see the comments of the function for further explanation.

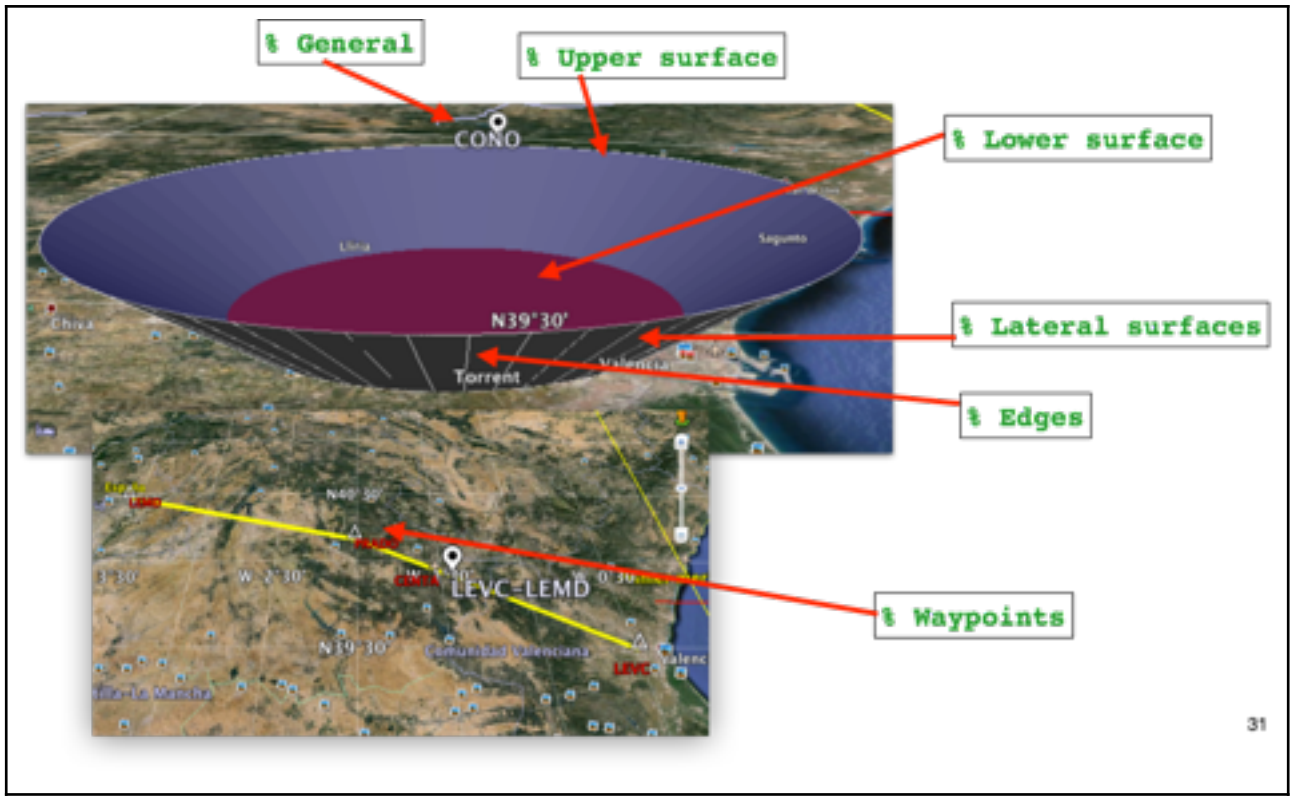

Figure 3

```
function attr=create_attr()
%-------------------------------------------------------------------------
% GENERAL
%-------------------------------------------------------------------------
% Applicable to kmlwrite_polyline, kmlwrite_polygon,and kmlwrite_volume
% Attributes that apply to the label and icon of the figure
%-------------------------------------------------------------------------
% attr.label:Label for the whole figure
% attr.labelscale: Label scale of figure label
% attr.labelcolor: Label color of figure label
% attr.iconurl: File or URL where the graphic for the label of the icon is defined
% attr.iconscale: Icon scale;
% attr.extrude: Specifies whether to connect the figure to the ground
% (see KML reference).
%-------------------------------------------------------------------------
attr.label='label';
attr.labelscale=1.0;
attr.labelcolor='ffffeeee';
attr.iconurl='http://maps.google.com/mapfiles/kml/paddle/wht-circle.png';
attr.iconscale=1.0;
attr.extrude=false;

%-------------------------------------------------------------------------
% UPPER SURFACE
%-------------------------------------------------------------------------
% Applicable to kmlwrite_polyline, kmlwrite_polygon ,and kmlwrite_volume
% Attributes that apply to the upper surface of a volume,
% the main surface o a polygon
% and also to the open surface defined by a polyline
%-------------------------------------------------------------------------
% attr.color: Color in RGB.
%             Example: FF0055AA is Transparency level=FF, B=00, G=55, R=AA
% attr.fill: Fill surface with color: true OR false.
```

```
% attr.altmode: altitude mode =absolute OR clampToGround OR relativeToGround
%-------------------------------------------------------------------------
attr.color='55555555';
attr.fill=true;
attr.altmode='absolute'; %'clampToGround', 'relativeToGround'

%-------------------------------------------------------------------------
% LOWER SURFACE
%-------------------------------------------------------------------------
% Applicable to kmlwrite_volume only
% Same attributes that for upper surface
%-------------------------------------------------------------------------
attr.color2='551111ff';
attr.fill2=true;
attr.altmode2='relativeToGround';

%-------------------------------------------------------------------------
% LATERAL SURFACE
%-------------------------------------------------------------------------
% Applicable to kmlwrite_volume only
% Same attributes that for upper surface except the altitude
%-------------------------------------------------------------------------
attr.latcolor='ff999999';
attr.latfill=true;

%-------------------------------------------------------------------------
% EDGES
%-------------------------------------------------------------------------
% Applicable to kmlwrite_polyline, kmlwrite_polygon ,and kmlwrite_volume
% Attributes of all edges of a polyline, polygon, or volume.
%-------------------------------------------------------------------------
attr.edgewidth=1;
attr.edgecolor='88333333';

%-------------------------------------------------------------------------
% WAYPOINTS
%-------------------------------------------------------------------------
% Applicable to kmlwrite_polyline, kmlwrite_polygon ,and kmlwrite_volume
% Attributes of the waypoint that define the upper surface of a figure
% Mainly used with kmlwrite_polyline
%-------------------------------------------------------------------------
% attr.wplabelscale: Scale of wp(i).name
% attr.wplabelcolor: Color of wp(i).name
% attr.wpiconurl: File or URL where the graphic for the label of the wp(i).name is
defined
%-------------------------------------------------------------------------
attr.wplabelscale=.8;
attr.wplabelcolor='ffffffff';
attr.wpiconurl='http://maps.google.com/mapfiles/kml/shapes/triangle.png';
attr.wpiconscale=0.5;
end
```

Step 4: Modify the default visual attributes as you like.

Just modify the attr fields that you want to be different from the default ones. Note that some attributes are not applicable or make no sense for **`kmlwrite_polygon,`** as for example all tose which refer to the lower and lateral surfaces. In this case we want to modify:

```
attr.label='LEVC-LEMD';
attr.labelscale=1.8;
```

```
attr.labelcolor='ffffeeee';
attr.iconurl='http://maps.google.com/mapfiles/kml/paddle/wht-circle.png';
attr.iconscale=1.5;
attr.extrude=false;

% Upper surface
attr.altmode='clampToGround';

% Edges
attr.edgewidth=6;
attr.edgecolor='ff00ffff';

% Waypoints
attr.wplabelscale=1.0;
attr.wplabelcolor='ff0000ff';
attr.wpiconurl='http://maps.google.com/mapfiles/kml/shapes/triangle.png';
attr.wpiconscale=1.0;
```

Step 5: Call the kmlwrite function.

This function creates the kml file for Google Earth:

```
kmlwrite_polyline(wp,'DME-LEVC.kml',attr);
```

Figure 4

The generated KML file is:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<name>DME-LEVC.kml</name>
<Style id="inline0">
   <LineStyle>
         <color>ff00ffff</color>
         <width>6</width>
   </LineStyle>
   <IconStyle>
         <scale>1.5</scale>
         <Icon><href>http://maps.google.com/mapfiles/kml/paddle/wht-circle.png</href></Icon>
   </IconStyle>
   <LabelStyle>
```

```
            <color>ffffeeee</color>
            <scale>1.8</scale>
      </LabelStyle>
   </Style>
   <Style id="inline1">
      <LineStyle>
            <color>ff00ffff</color>
            <width>6</width>
      </LineStyle>
      <IconStyle>
            <scale>1</scale>
            <Icon><href>http://maps.google.com/mapfiles/kml/shapes/triangle.png</href></Icon>
      </IconStyle>
      <LabelStyle>
            <color>ff0000ff</color>
            <scale>1</scale>
      </LabelStyle>
   </Style>
   <Placemark>
      <name>LEVC-LEMD</name>
      <styleUrl>#inline0</styleUrl>
      <MultiGeometry>
            <Point>
                  <altitudeMode>clampToGround</altitudeMode>
                  <coordinates>-1.431903,39.900717,30000.000000</coordinates>
            </Point>
            <LineString>
                  <extrude>0</extrude>
                  <tessellate>1</tessellate>
                  <altitudeMode>clampToGround</altitudeMode>
                  <coordinates>
                  -0.481667,39.489444,0.000000
                  -1.432003,39.900617,30000.000000
                  -2.010342,40.147489,40000.000000
                  -3.560833,40.472222,0.000000
                  </coordinates>
            </LineString>
      </MultiGeometry>
   </Placemark>
   <Placemark>
      <name>LEVC</name>
      <description>wp1</description>
      <styleUrl>inline1</styleUrl>
      <Point>
            <altitudeMode>clampToGround</altitudeMode>
            <coordinates>-0.481667,39.489444,0.000000 </coordinates>
      </Point>
   </Placemark>
   <Placemark>
      <name>CENTA</name>
      <description>wp2</description>
      <styleUrl>inline1</styleUrl>
      <Point>
            <altitudeMode>clampToGround</altitudeMode>
```

```
            <coordinates>-1.432003,39.900617,30000.000000 </coordinates>
      </Point>
   </Placemark>
   <Placemark>
      <name>PRADO</name>
      <description>wp3</description>
      <styleUrl>inline1</styleUrl>
      <Point>
            <altitudeMode>clampToGround</altitudeMode>
            <coordinates>-2.010342,40.147489,40000.000000 </coordinates>
      </Point>
   </Placemark>
   <Placemark>
      <name>LEMD</name>
      <description></description>
      <styleUrl>inline1</styleUrl>
      <Point>
            <altitudeMode>clampToGround</altitudeMode>
            <coordinates>-3.560833,40.472222,0.000000 </coordinates>
      </Point>
   </Placemark>
   </Document>
   </kml>
```

Circles

Circles must be generated as a polyline In Matlab with a number of points.

```
function [xp,yp]=circle(x,y,r,phi1,phi2,n)
% It creates a circle (or circular sector) on the surface of the earth
% defined by N points.
% It returns an array [xp, yp] with the coordinates of the points.
% It takes into account the strain of the arch with the latitude.
% Used for example to draw DME arcs in Google Earth.
% Parameters:
%     x, y: coordinates of the center of the circle.
%           Can be longitudes and latitudes in radians
%     r: radius of the circle.
%        May be an angular distance D / R, where R is the radius of the earth
%     Phi1: Start angle circular sector
%           0 degrees is NORTH.
%     Phi2: Final angle of the circular sector
%           Anles measured clockwise
%     N: number of points
% There is a function with an example circle_example
% In the examples-kml_write file.
```

It can be used as follows:

```
[x,y]=circle(Manises.lon, Manises.lat,r/R,phi1,phi2,NP);

for i=1:NP
    wp(i).name='';
    wp(i).desc='';
    wp(i).lon=x(i);
```

```
    wp(i).lat=y(i);
    wp(i).alt=0;
end
```

See function **circle_example** in file **examples_kmlwrite.m** for more details.

3. THE KMLWRITE_POLYGON AND KMLWRITE_VOLUME FUNCTIONS

The functions described here are used in the same way that **kmlwrite_polylin**e. The main difference is the visual attributes.

KMLWRITE_POLYGON

- **function kmlwrite_polygon**(wp, filename,attr);

This function draws a polygon defined by a number of points. The following is an example of use that you can find in in file **examples_kmlwrite.m** for drawing the LED65 of Marines.

Figure 5

```
function polygon_example
ft2m=0.3048;
fl2m=ft2m*100;
nm2m=1852;
```

```
nm2km=1.852;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% LED65 MARINES (Sagunto)
% 394700N 0003600W;
% 394700N 0003100W;
% 394100N 0003100W;
% 394100N 0003600W;
% 394700N 0003600W.


led65(1).lat = sex2dec2('394700N');
led65(1).lon= sex2dec2('0003600W');

led65(2).lat = sex2dec2('394700N');
led65(2).lon= sex2dec2('0003100W');

led65(3).lat = sex2dec2('394100N');
led65(3).lon= sex2dec2('0003100W');

led65(4).lat = sex2dec2('394100N');
led65(4).lon= sex2dec2('0003600W');

led65(5).lat = sex2dec2('394700N');
led65(5).lon= sex2dec2('0003600W');

for i=1:length(led65)
    led65(i).alt=12500*ft2m;
end

%-------------------------------------------------------------------------
attr.label='LED65';
attr.labelscale=1;
attr.labelcolor='ffffeeee';
attr.iconurl='http://maps.google.com/mapfiles/kml/paddle/wht-circle.png';
attr.iconscale=1.0;
attr.extrude=true;

% Upper surface
attr.color='225500ff';
attr.fill=true;
attr.altmode='relativeToGround'; %'clampToGround', 'relativeToGround'

% Edges
attr.edgewidth=1;
attr.edgecolor='ffffffff';

kmlwrite_polygon(led65,'led65.kml',attr);

end
```

KMLWRITE_POLYGON

- `function kmlwrite_volume(wpu, wpl, filename,attr);`

- This function draws a volume defined by two surfaces (polygons) **`wpu`** and **`wpl`** with the same number of edges. **`wpu`** is the set of points which define the *upper* polygon and **`wpl`** is the set of points which define the *lower* polygon.

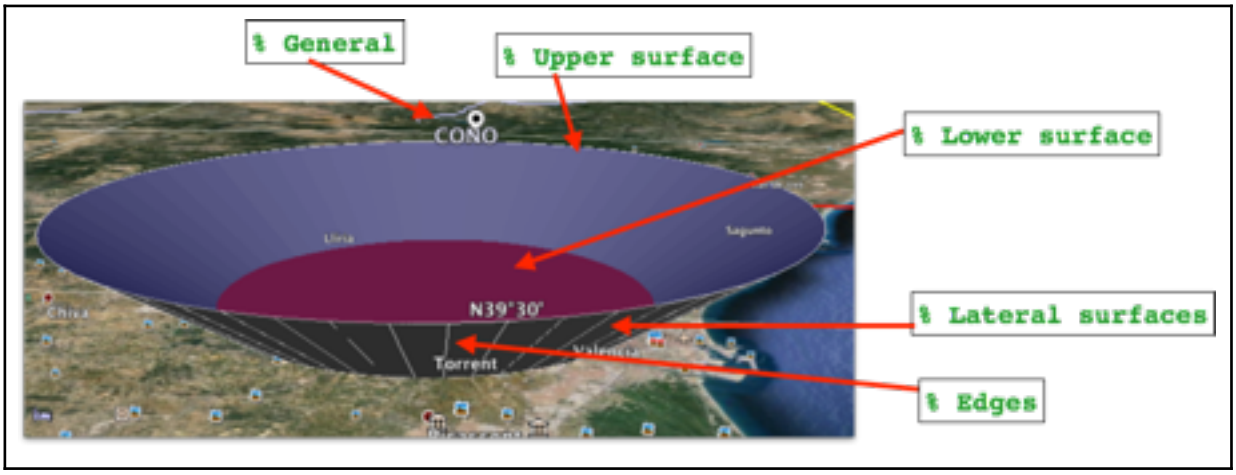

Figure 6

The following is an example of use that you can find in in file **`examples_kmlwrite.m`** for drawing a cone centered in Valencia Airport.

```
function volume_example

ft2m=0.3048;
fl2m=ft2m*100;
nm2m=1852;
nm2km=1.852;

R=6367.445;       % Radius of Earth
r=6*nm2km;        % A DME arch of 6 NM
phi1=0;;          % Starting angle of DME arch
phi2=359;         % Ending angle of DME arch
NP=60;            % Number of points of arch

Manises.lat=39.489444444444445;
Manises.lon=-0.481666666666667;

[x,y]=circle(Manises.lon, Manises.lat,r/R,phi1,phi2,NP);
%plot(x,y);

for i=1:NP
    wpl(i).lon=x(i);
    wpl(i).lat=y(i);
    wpl(i).alt=1000;
end
wpl(NP+1)=wpl(1);

r=10*nm2km;       % A DME arch of 20 NM
```

```
[x,y]=circle(Manises.lon, Manises.lat,r/R,phi1,phi2,NP);
%hold on;
%plot(x,y);

for i=1:NP
    wpu(i).lon=x(i);
    wpu(i).lat=y(i);
    wpu(i).alt=6000;
end
wpu(NP+1)=wpu(1);

%-------------------------------------------------------------------------
vol_attr=create_attr();

vol_attr.label='CONO';
vol_attr.labelscale=1.8;
vol_attr.labelcolor='ffffeeee';
vol_attr.iconurl='http://maps.google.com/mapfiles/kml/paddle/wht-circle.png';
vol_attr.iconscale=1.5;
vol_attr.extrude=false;

% Upper surface
vol_attr.color='55ff0000';
vol_attr.fill=true;
vol_attr.altmode='absolute';

% Lower surface
vol_attr.color2='ff0000ff';
vol_attr.fill2=true;
vol_attr.altmode2='relativeToGround';

% Lateral surfaces
vol_attr.latcolor='ff999999';
vol_attr.latfill=true;

% Edges
vol_attr.edgewidth=1;
vol_attr.edgecolor='88ffffff';

kmlwrite_volume(wpu,wpl,'volume-ex.kml',vol_attr);
%-------------------------------------------------------------------------
end
```

4. THE ROUTE_READ FUNCTION

This function is used to represent air routes produced by Route Finder.

- function `wp=route_read(file)`

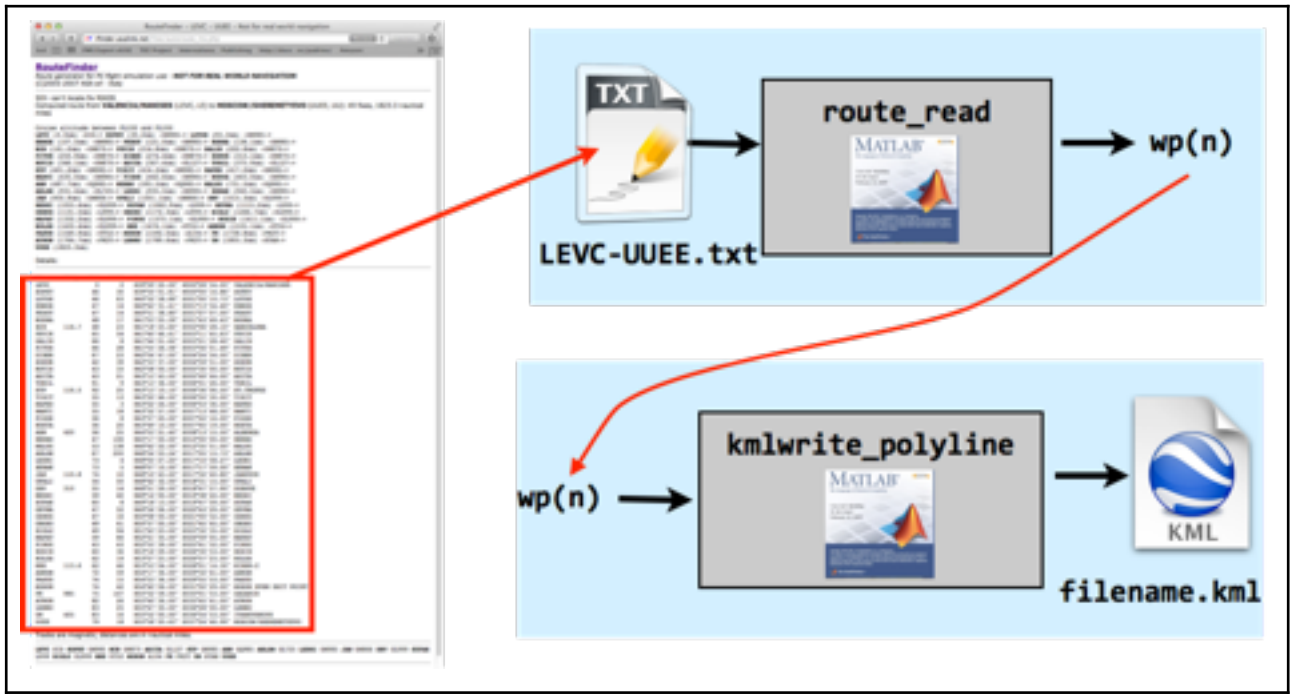

Figure 7

It reads a text file produced by Route Finder defining a route and produces a list of corresponding waypoints for Matlab with the following attributes:

```
wp(n).lon
wp(n).lat
wp(n).alt
wp(n).gspeed
wp(n).desc
```

The text file with the Route Finder output has to be generated by coping the part of the output of the web browser shown in figure 7 and pasting it in an ASCII text file using an editor (for example Matlab's editor).

The `wp` data structure produced by `route_read` can be used by `kmlwrite_polyline` to represent the air route as shown in figure 8.

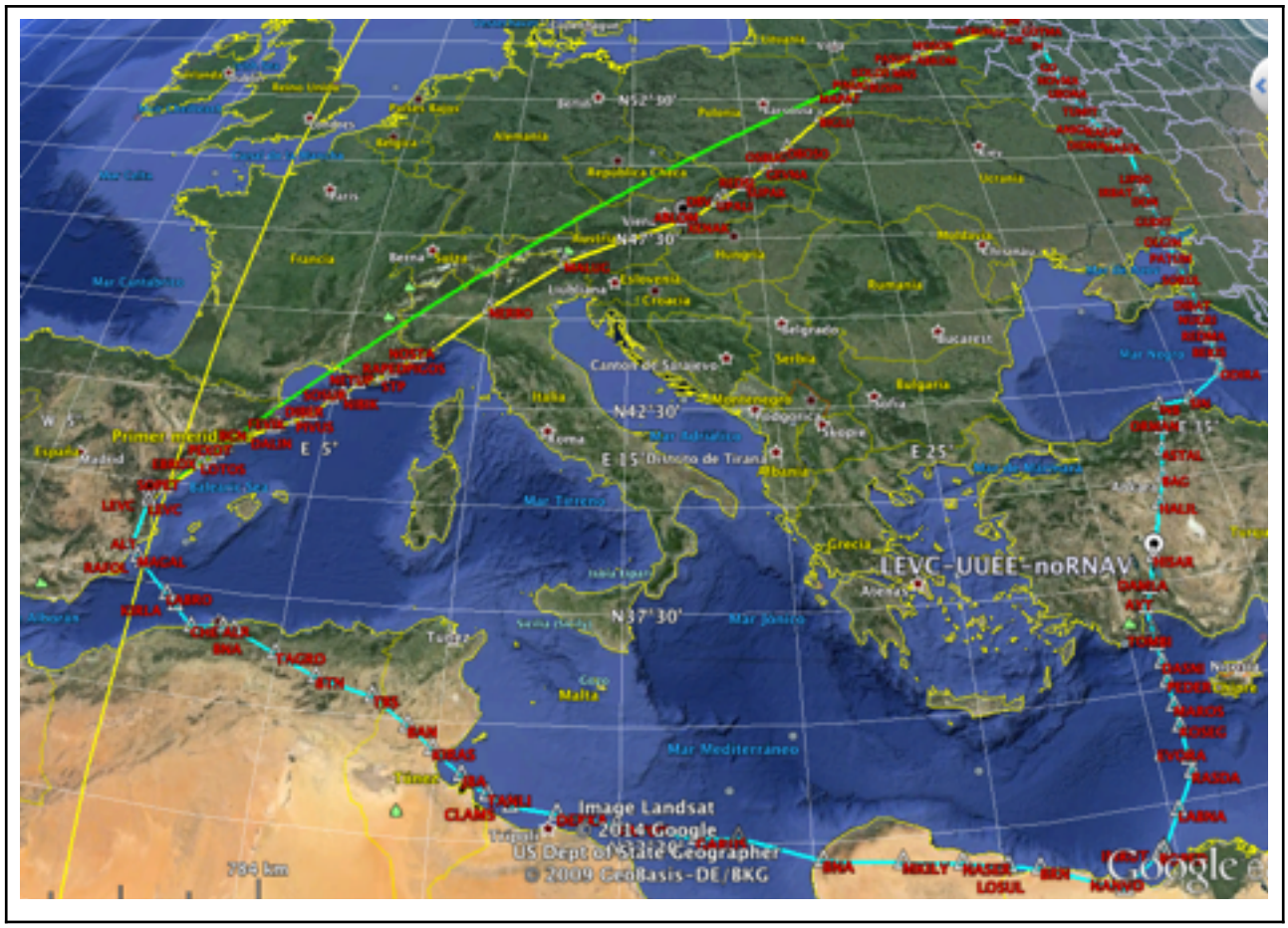
LEVC-UUEE-noRNAV
N42°30'
N37°30'
E 5°
E 15°
E 25°
Image Landsat
© 2014 Google
US Dept of State Geographer
© 2009 GeoBasis-DE/BKG
Google

Figure 8