Interactive Web Visualization of an Earth to Moon Mission

By Daniel A. O'Neil

Introduction

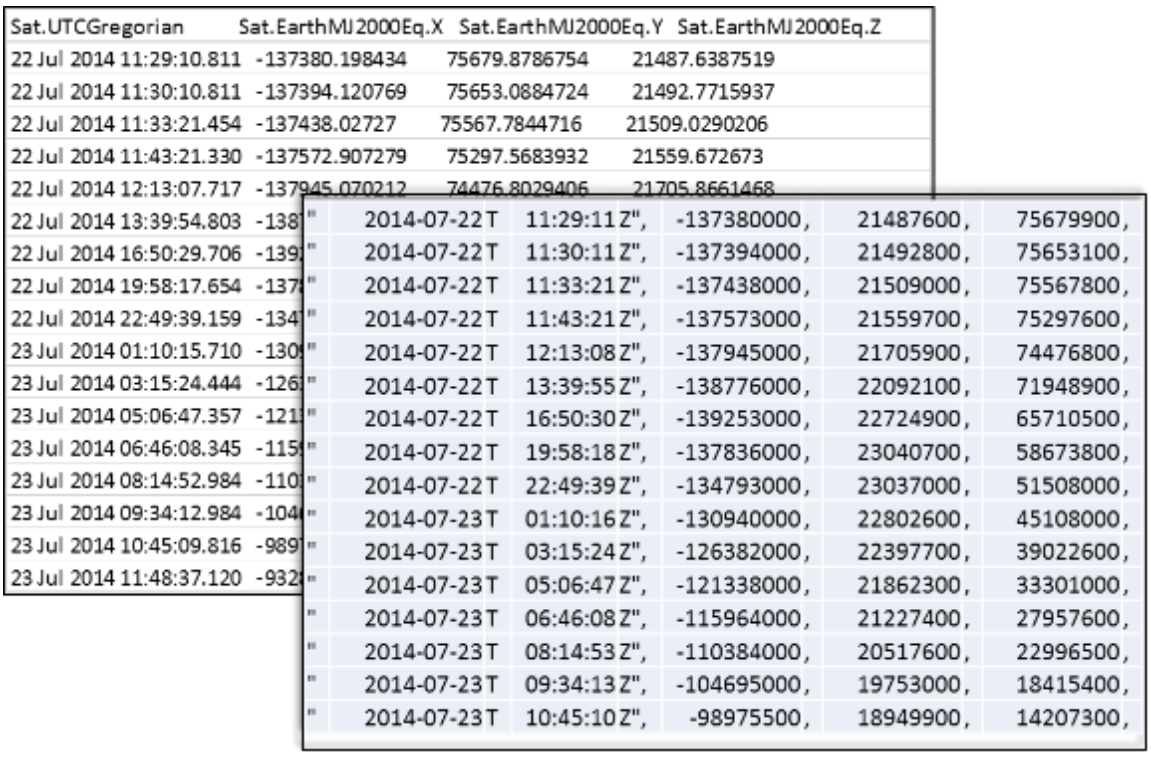
Analytical Graphics Inc. (AGI) provides a free digital globe and open source JavaScript code library, named Cesium. With Cesium, a developer can create interactive web-based space mission visualizations. This tutorial explains how to convert time-stamped Cartesian coordinate data, exported from the General Mission Analysis Tool (GMAT), into a Cesium compatible format. Additionally, this tutorial explains JavaScript code that calls functions in the Cesium Application Programming Interface (API) for loading models and tracking entities as they follow a trajectory.

Prerequisites and Objectives

The demonstration developed for this tutorial uses data generated from a GMAT tutorial. Refer to the GMAT documentation for the tutorials. In GMAT, lines can be inserted in the scripts to write trajectory time-stamped Cartesian coordinate data into a text file. The GMAT webpage, https://gmat.gsfc.nasa.gov/, provides links for downloading the application and a support forum.

Data Conversion

Figure 1 presents two versions of time-stamped trajectory data in Excel spreadsheets. The back spreadsheet presents the data as it appears when exported from GMAT. The spreadsheet in the front presents the data in a Cesium compatible format. The reformatting was done manually. Reformatting steps included splitting the date-timestamps into separate columns, adding columns for quotes, the T and Z, and commas. The numbers were multiplied by a thousand to convert them from kilometers to meters. Date and time-stamps are expressed in ISO 8601format. https://en.wikipedia.org/wiki/ISO_8601



Sat.UTCGregorian	Sat.EarthMJ2000Eq.X	Sat.EarthMJ2000Eq.Y	Sat.EarthMJ2000Eq.Z
22 Jul 2014 11:29:10.811	-137380.198434	75679.8786754	21487.6387519
22 Jul 2014 11:30:10.811	-137394.120769	75653.0884724	21492.7715937
22 Jul 2014 11:33:21.454	-137438.02727	75567.7844716	21509.0290206
22 Jul 2014 11:43:21.330	-137572.907279	75297.5683932	21559.672673
22 Jul 2014 12:13:07.717	-137945.070212	74476.8029406	21705.8661468
22 Jul 2014 13:39:54.803	-138		
22 Jul 2014 16:50:29.706	-139		
22 Jul 2014 19:58:17.654	-137		
22 Jul 2014 22:49:39.159	-134		
23 Jul 2014 01:10:15.710	-130		
23 Jul 2014 03:15:24.444	-126		
23 Jul 2014 05:06:47.357	-121		
23 Jul 2014 06:46:08.345	-115		
23 Jul 2014 08:14:52.984	-110		
23 Jul 2014 09:34:12.984	-104		
23 Jul 2014 10:45:09.816	-989		
23 Jul 2014 11:48:37.120	-932		

"	2014-07-22T 11:29:11Z",	-137380000,	21487600, 75679900,
"	2014-07-22T 11:30:11Z",	-137394000,	21492800, 75653100,
"	2014-07-22T 11:33:21Z",	-137438000,	21509000, 75567800,
"	2014-07-22T 11:43:21Z",	-137573000,	21559700, 75297600,
"	2014-07-22T 12:13:08Z",	-137945000,	21705900, 74476800,
"	2014-07-22T 13:39:55Z",	-138776000,	22092100, 71948900,
"	2014-07-22T 16:50:30Z",	-139253000,	22724900, 65710500,
"	2014-07-22T 19:58:18Z",	-137836000,	23040700, 58673800,
"	2014-07-22T 22:49:39Z",	-134793000,	23037000, 51508000,
"	2014-07-23T 01:10:16Z",	-130940000,	22802600, 45108000,
"	2014-07-23T 03:15:24Z",	-126382000,	22397700, 39022600,
"	2014-07-23T 05:06:47Z",	-121338000,	21862300, 33301000,
"	2014-07-23T 06:46:08Z",	-115964000,	21227400, 27957600,
"	2014-07-23T 08:14:53Z",	-110384000,	20517600, 22996500,
"	2014-07-23T 09:34:13Z",	-104695000,	19753000, 18415400,
"	2014-07-23T 10:45:10Z",	-98975500,	18949900, 14207300,

Figure 1 Spreadsheets with time-stamped trajectory data

Visualization with Cesium

A Cesium based visualization of an Earth to Moon mission, depicted in Figure 2, includes trajectories for the moon and a lunar probe, a speed-control widget, and buttons for selecting various viewpoints. Presented in the inset image is the lunar probe. The following link leads to the demo. http://daoneil.github.io/spacemission/Apps/EarthToMoon_Demo.html

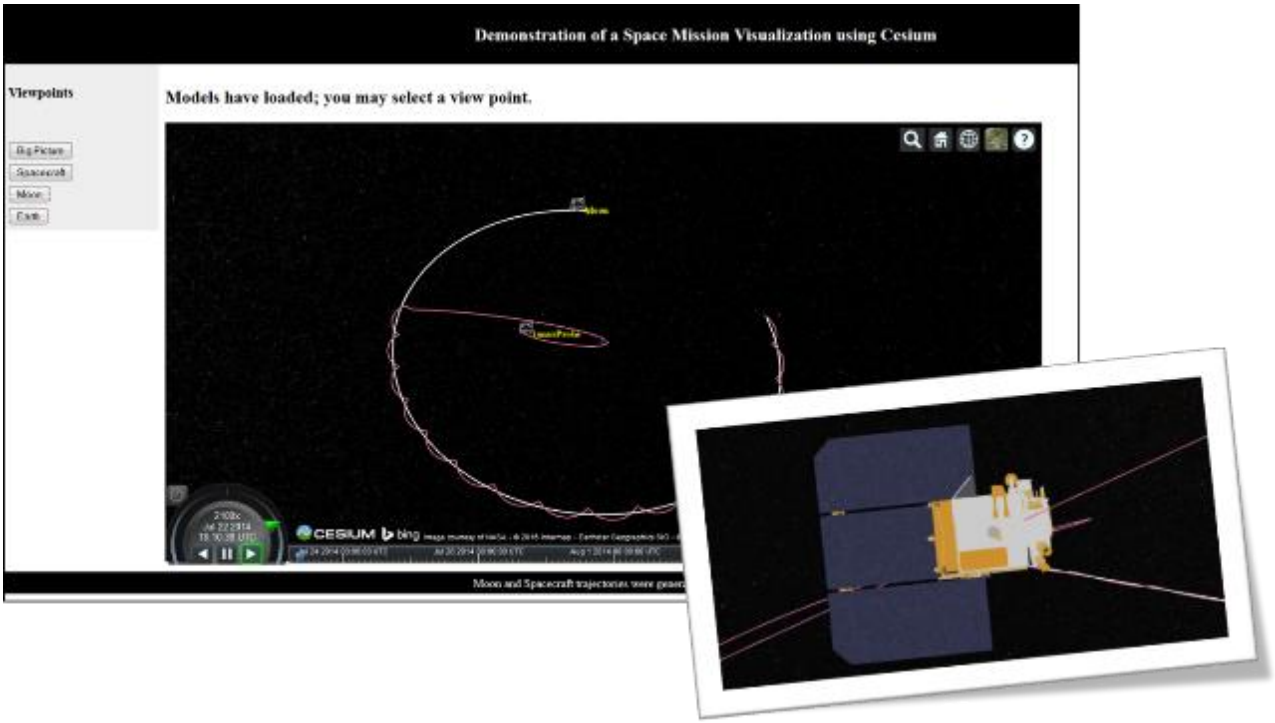


Figure 2 A Cesium based visualization of an Earth to Moon mission

A graphics industry consortium, the Khronos Group, developed a 3D file format suited for data transmission. Cesium supports the glTF format for 3D models. Originally, the lunar probe model was a Lunar Reconnaissance Orbiter (LRO) model in the Light Wave Object (LWO) format. Using the modeling application, Blender, the file was converted to the COLLADA format. A web-app at the Cesium website converts COLLADA to glTF via a drag-and-drop area.

- NASA 3D Resources, LRO model http://nasa3d.arc.nasa.gov/detail/lro-full
- COLLADA to glTF drag-and-drop converter https://cesiumjs.org/convertmodel.html
- Information about glTF https://www.khronos.org/gltf

Cesium Modeling Language (CZML)

Cesium has a JavaScript Object Notation (JSON) based modeling language, CZML, for defining entities, associated attributes, and coordinate data for paths. A CZML Guide provides an overview of the characteristics and CZML content pages provide detailed documentation about specifying entities, attributes, and behavior.

- Cesium Language (CZML) Guide https://github.com/AnalyticalGraphicsInc/cesium/wiki/CZML-Guide
- CZML Content https://github.com/AnalyticalGraphicsInc/cesium/wiki/CZML-Content
- CZML Content 2 https://github.com/AnalyticalGraphicsInc/cesium/wiki/CZML-Content-2

The following CZML code snippet specifies a document and lunar probe content:

```
[ {
    "id":"document",
    "name":"Lunar Transfer Trajectory",
    "version":"1.0",
   "clock":{
      "interval":"2014-07-22T11:00:00Z/2014-08-12T22:07:27Z",
      "currentTime":"2014-07-22T11:00:00Z",
      "multiplier":2100,
      "range":"LOOP_STOP",
      "step":"SYSTEM_CLOCK_MULTIPLIER"
  }},
{  "id": "lunarProbe",
   "availability":["2014-07-22T11:00:00Z/2014-08-12T22:07:27Z"],
   "model" : {
       "gltf" : "LRO_spacecraft_with_materials.gltf",
      "scale" : 5000.0,
       "show" : [{
                "interval" : "2014-07-22T11:00:00Z/2014-08-12T22:07:27Z",
                "boolean" : true    }]
   },
    "billboard" : {
        "eyeOffset" : {
            "cartesian" : [0.0, 0.0, 0.0]
        },
        "horizontalOrigin" : "CENTER",
        "image" : "<data not shown to save space>,
        "pixelOffset" : {
            "cartesian2" : [0.0, 0.0]},
        "scale" : 0.8,
        "show" : true,
        "verticalOrigin" : "BOTTOM" },
    "label" : {
        "fillColor" : {
            "rgba" : [255, 255, 0, 255]
        },
        "font" : "bold 10pt Segoe UI Semibold",
        "horizontalOrigin" : "LEFT",
        "outlineColor" : {
            "rgba" : [0, 0, 0, 255]
        },
        "pixelOffset" : {
            "cartesian2" : [10.0, 0.0]  },
        "scale" : 1.0,
        "show" : true,
        "style" : "FILL",
        "text" : "lunarProbe",
        "verticalOrigin" : "CENTER" },
   "path" : {  "width" : 1.5,
               "material":{
               "solidColor":{ "color":{ "rgba":[ 200,100,150,255 ] } }
            },
            "show"  : true  },
   "position": {
        "referenceFrame": "INERTIAL",
        "cartesian": [
"2014-07-22T11:29:11Z",-137380198.4340,75679878.6754,21487638.7519,
"2014-07-22T11:30:11Z",-137394120.7690,75653088.4724,21492771.5937,
"2014-07-22T11:33:21Z",-137438027.2700,75567784.4716,21509029.0206,
…. ] } }]
```

Declarations in the CZML code include identifiers, a time-frame for the availability of the model, a billboard that represents the model, a label, a model attribute with a reference to the glTF file, and a path that includes the time-stamped coordinates. JavaScript code use the identifier for entity tracking.

JavaScript Code

The following JavaScript code snippet instantiates a viewer object, loads the CZML files into data sources, defines a function for a camera look-at transform based on the International Celestial Reference Frame (ICRF).

```
<script>
   var viewer = new Cesium.Viewer('cesiumContainer', {
       infoBox : false,
       selectionIndicator : false
      });
      var scene = viewer.scene;
      var clock = viewer.clock;

   var czmlDataSource1 = new Cesium.CzmlDataSource();
       czmlDataSource1.load('LunarProbeTrajectory_withBillboard.czml');
       viewer.dataSources.add(czmlDataSource1);

   var czmlDataSource2 = new Cesium.CzmlDataSource();
       czmlDataSource2.load('LunarOrbit_withTexture.czml');
       viewer.dataSources.add(czmlDataSource2);

    var camera = viewer.scene.camera;

  function icrf(scene, time) {
    if (scene.mode !== Cesium.SceneMode.SCENE3D) {   // may not be necessary
        return;
    }
    var icrfToFixed = Cesium.Transforms.computeIcrfToFixedMatrix(time);
    if (Cesium.defined(icrfToFixed)) {
        var offset = Cesium.Cartesian3.clone(camera.position);
        var transform = Cesium.Matrix4.fromRotationTranslation(icrfToFixed);
        camera.lookAtTransform(transform, offset);
    }
};
      clock.multiplier =  2100 ;           // speed of the simulation
    scene.preRender.addEventListener(icrf);  // enable Earth rotation
      var icrfSwitch = true ;               // flag for icrf event listener
//…
</script>
```

An event listener, added to the scene preRender event calls the ICRF function to put the camera in the ICRF. A global Boolean variable, named icrfSwitch indicates whether the camera is in the ICRF or world coordinates reference frame. While in the ICRF, the Earth appears to rotate.

Buttons enable selection of a viewpoint. Clicking a button calls the function associated with the button. This code snippet displays the button for the spacecraft viewpoint and calls the showSpacecraft function in response to a click event.

```
<input type="button" id="showShip" value="Spacecraft"
onclick="showSpaceCraft();" /><br>
```

When the camera is in the ICRF, the camera's trackedEntity function does not display the intended target because the entity's coordinates are based on the Inertial reference frame as specified, in the CZML file, by the position reference frame attribute. The following code snippet is the showSpaceCraft function. The view point functions use the icrfSwitch to determine whether to remove or add an event listener to the scene.preRender event that calls the ICRF function. When viewing the spacecraft or the Moon, the icrfSwitch is true so the event listener is removed and the icrfSwitch is set to false. Functions called by the Earth or Big Picture buttons add the event listener and set the icrfSwitch to true.

```
function showSpaceCraft() {
    viewer.trackedEntity = undefined

      if (icrfSwitch) {
         scene.preRender.removeEventListener(icrf);
         icrfSwitch = false ;
         } ;
    var spaceCraft = czmlDataSource1.entities.getById('lunarProbe') ;
        viewer.trackedEntity = spaceCraft ;
}
```

Notice that spaceCraft variable is set the lunarProbe object via the getById function associated with the CZML data source entities collection. When the camera is in the Inertial frame, the viewer's trackedEntity function can follow the specified object.

Demonstration Source Code

A link to the demonstration was provided in the section "Visualization with Cesium" and Figure 2 presented a screenshot. This link leads to the GitHub repository that contains the source code, including the web-page with the embedded JavaScript and the two CZML files.
https://github.com/daoneil/spacemission

Conclusions

Topics covered in this tutorial included:

- Where to find tutorials for the General Mission Analysis Tool (GMAT)
- Where to find programming guides for the Cesium Modeling Language (CZML)
- The ISO 8601 Date and Time format used in CZML files
- How to convert GMAT generated trajectory data to CZML path coordinates
- Where to find NASA 3D Resources, i.e., http://nasa3d.arc.nasa.gov/
- Where to find information about the Khronos Group's glTF format
- How to convert a COLLADA file to glTF via a web-app with a drag-n-drop interface
- A walk-through of a CZML file that identifies a glTF model and its path coordinates
- A walk-through of the JavaScript code that loads CZML files
- How to rotate the Earth via the ICRF transform for the camera
- A walk-through of JavaScript code that toggles the ICRF to enable camera entity tracking
- Where to find the source code for the demonstration described in this tutorial

A digital globe, such as Cesium, provides a capability to visualize ground based operations, ascent and descent phases of a mission, and missions within the vicinity of the Earth. Implemented, in JavaScript means that globe works within modern web-browsers without the need for an add-on or additional software. Analytical Graphics Inc.'s decision to develop Cesium as free open source code means the user community can contribute to the project. The Cesium API and CZML offer a standardized approach to developing web-based space mission simulators.