plyLoadR, a new htmlwidget for mesh evolution visualization

Lucio Cornejo, Evelyn Gutierrez, Benjamín Castañeda, Sylvie Treuillet

Keywords: rgl, ply, WebGL, 3D visualization.

Abstract

Despite the great capabilities of R's rgl package for creating 3D graphics, that library has some considerable limitations, for example, when displaying transparent graphics (extending significantly the rendering time of knitr and Pandoc) or whenever the created graphics require custom user interactivity. The plyLoadR package solves those issues via the use of the JavaScript 3D library Three.js. We demonstrate the advantage of using our proposed widget (plyLoadR) over rgl, in three scenarios for 3D mesh rendering in the context of chronic wounds evolution.

Introduction

RGL is an R library that allows real-time creation of interactive 3D visualizations.
This library is useful for the display of 3D statistical graphics, as well as for the interactive visualization of 3D models.
In particular, in the STANDUP project, we have used the rgl package to interactively explore 3D models of patient wounds. However, the following drawbacks have been observed when using it to render large 3D models:

1. When transparent non lightweight geometries are inserted into an rgl scene, the graphic in the resulting HTML page starts to lag considerably, making it unusable.
2. In order to include an rgl scene into an HTML page, the information for that scene (geometries and such) gets included into the HTML. However, such inclusion substantially increases the rendering time of the R Markdown (or Quarto) file where the rgl scene is created, even if the cache property was used for chunks.
3. Despite the existence of JavaScript methods for manipulating the graphics created with rgl, their official documentation is quite poor. Besides this, the potential interactivity of rgl graphics falls short when compared to the capabilities of Three.js.

plyLoadR

plyLoadR is an R package created to address the limitations of rgl described above.
Our solution consists of wrapping, via the htmlwidgets R package, JavaScript's most complete 3D library, Three.js, into a widget that can be used directly from R.
That way, the plyLoadR widget can load geometries stored in ply files, some of which were created and then locally stored using the rgl package.

The proposed package, plyLoadR, offers the following advantages:

1. **Faster rendering of translucent 3D meshes**
   For lightweight graphics, the rgl package works fine when displaying semi or fully transparent geometries. However, whenever the geometry displayed is complex/heavy enough, the rgl graphics become laggy. The proposed pipeline implemented on plyLoadR allows for no delay due to semi- or fully transparent graphics.
2. **Faster rendering overall**
   When recreating a rgl scene with plyLoadR, the rendering time of the .Rmd or .qmd file decreases significantly; in some cases, from 5 minutes, to 10 seconds.
3. **Greater amount of tools for interactive graphics**
   plyLoadR graphics can be manipulated with all the default tools of the Three.js library, the most powerful JavaScript 3D graphics library in existence.

A complete description of the plyLoadR package can be found on its official repository.

Use cases

Three different scenes are shown in this website to provide an example of the advantages of the proposed library, plyLoadR, over rgl. The 3D meshes displayed belong to the STANDUP project, in which foot wounds of diabetic patients were 3D scanned with photos from portable devices in order to be used for wound assessment.

- **Case 1: Using rgl**
  The first scene was created with rgl, including two translucent geometries.
  Each of those geometries corresponds to a 3D scan of the wound of the patient, at the dates displayed in the buttons in the HTML page.
- **Case 2: plyLoadR and rgl direct comparison**
  The second scene was created with plyLoadR, as a recreation of the first scene. However, as the user interacts with the second scene, it can be confirmed that no lag occurs due to translucent geometries.
- **Case 3: Mesh evolution**
  This scene is an example of how the plyLoadR widget can be used for visualizing the evolution of meshes, especially if such meshes have been properly oriented, like in this scene.

Acknowledgment

This work was supported by the European Union’s Horizon 2020 research and innovation program under the Marie Sklodowska-Curie action [Grant number N° 777661, STANDUP project].

Lucio Cornejo
Pontificia Universidad Católica del Perú
`lucio.cornejo@pucp.edu.pe`