# Processing.R: Make Visualization Great Again for R Users

**Ce Gao**
gaoce270863799@sjtu.edu.cn

## Abstract

Processing.R is a mode for Processing that enables users to write Processing sketches using the R language. In this paper, we first introduce the mode. We then show the functionality of the mode. Based on the functionality, we next introduce the design and implementation of Processing.R. Finally we conclude that R language users could benefit from the new mode.

## 1 Introduction

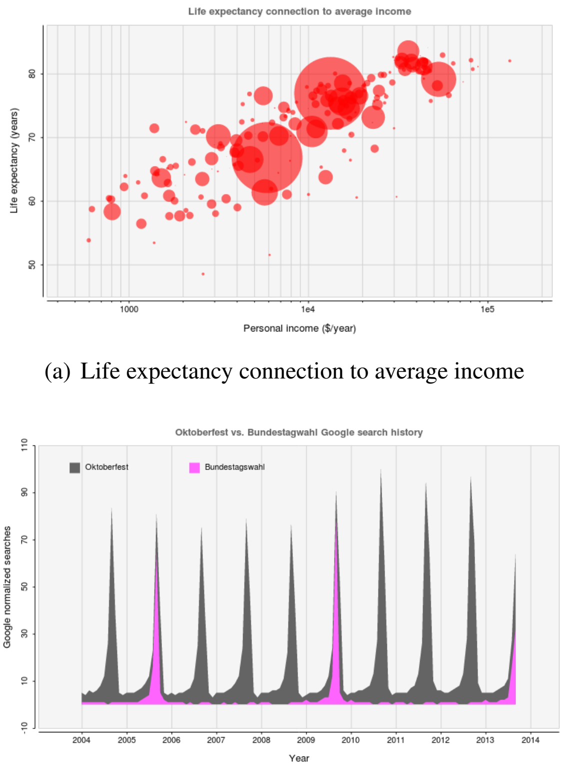
Processing is a flexible software sketchbook and a language for learning how to code within the context of the visual arts. R is a free software environment for statistical computing and graphics. We implements a mode to combine two languages. In particular, we make the following contributions:

- a general-purpose visualization tool for R users.
- A way to integrate Java with R (Processing mode is written in Java)

To show its applicability, we use Processing.R to implement three demos in different contexts. The project is open sourced in GitHub (`https://github.com/gaocegege/Processing.R`).

## 2 Functionality

Processing is a mature software project and it supports two-dimensional and three-dimensional programming. Processing.R is a mode for Processing so it supports most of the functionalities in Processing. Besides this, users could extend the functionality of Processing.R by installing the Processing libraries in Processing Contribution Manager or R packages in CRAN. We show the functionalities via some case studies.



(a) Life expectancy connection to average income

(b) Oktoberfest vs. Bundestagswahl Google Search History

Figure 1: Case Study: Statistical Graphics

### 2.1 Case Study: Statistical Graphics

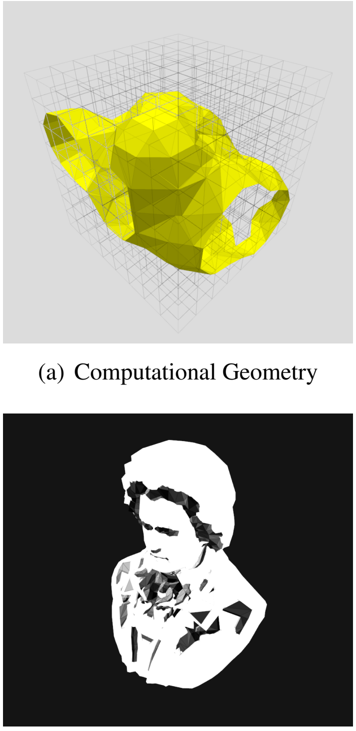
grafica is a Processing library to create simple and configurable 2D interactive plots with Processing. Processing.R is compatible with the library, so it is simple to plot in Processing.R.

Figure 1(a) and Figure 1(b) are two examples in grafica but written in R.

### 2.2 Case Study: Three-dimensional Graphics

Processing supports three-dimensional programming natively, so it is easy to write three-dimensional meshes in Processing.R, which also provides users with instant feedback through interaction.

Figure 2(a) and Figure 2(b) are two examples



(a) Computational Geometry

(b) Three-dimensional Mesh

Figure 2: Case Study: Three-dimensional Graphics

to show the three-dimensional support in Processing.R.

## 3 Design and Implementation

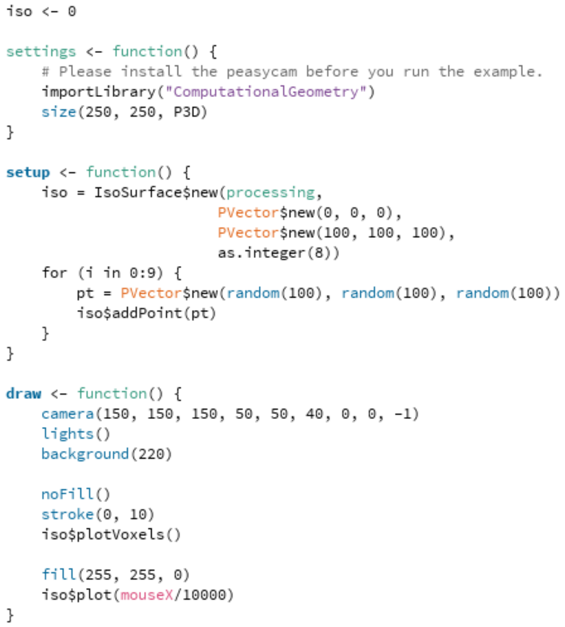
We use Figure 2(a) as an example to show the design and implementation of Processing.R.

Processing.R follows the syntax of Processing, which has built-in functions: *settings*, *setup* and *draw*. And it relies on renjin, which is a JVM interpreter for the R language, to evaluate R code.

Figure 3 is the source code for Figure 2(a). When Processing.R receives a command to run the code, it evaluates the functions and variables in root context of R ahead of time, and the process is called pre-processing. After the pre-processing phase, Processing.R tries to find the definition of the built-in functions and run these functions as usual. The implementation is simple but works well, most functions in Processing are available in Processing.R.

To support Processing libraries, we import a new primitive *importLibrary*, which accepts the name of library and import all the Java classes in the library into R context.

To support R packages, we implements similar logics in renjin's command line mode. Processing.R automatically downloads the corresponding packages according to the package installation statements in R. We do not support all CRAN packages now, because some of the packages are coupled with C++ code, and it does not supported in renjin. We rely on the renjin community to migrate more packages from CRAN to renjin and then the new packages are available in Processing.R, too.



```
iso <- 0

settings <- function() {
    # Please install the peasycam before you run the example.
    importLibrary("ComputationalGeometry")
    size(250, 250, P3D)
}

setup <- function() {
    iso = IsoSurface$new(processing,
                         PVector$new(0, 0, 0),
                         PVector$new(100, 100, 100),
                         as.integer(8))
    for (i in 0:9) {
        pt = PVector$new(random(100), random(100), random(100))
        iso$addPoint(pt)
    }
}

draw <- function() {
    camera(150, 150, 150, 50, 50, 40, 0, 0, -1)
    lights()
    background(220)

    noFill()
    stroke(0, 10)
    iso$plotVoxels()

    fill(255, 255, 0)
    iso$plot(mouseX/10000)
}
```

Figure 3: Source Code for Figure 2(a)

## 4 Conclusion

We have presented Processing.R, the new mode for Processing that supports most of Processing libraries and some R packages in CRAN with the help of renjin.

The demos that we presented show that Processing.R is a general-purpose visualization tool and it explores new possibilities of the combination between the statistical computing and visual arts.

Since its first release in May 2017, Processing.R has drawn increasing attention from Processing users and R users. The number of its GitHub stars is 76 as of October 20, 2017.