VisSurvey.js – A Web Based Javascript Application for visualisation Evaluation User Studies

Roberts, Jonathan C.; Jackson, James

Published: 01/10/2017

VisSurvey.js – A Web Based Javascript Application for visualisation Evaluation User Studies

James R. Jackson*
Bangor University, UK

Jonathan C. Roberts†
Bangor University, UK

ABSTRACT

Visualization researchers perform user studies to evaluate the effectiveness and usability of produced visualisations. While there are different styles of evaluation, some popular techniques involve comparing the result of several parameter changes, looking how results have changed and if a change is noticeable. Developers need to display many different pictures, to a wide variety of users, while guaranteeing full coverage of all potential designs across all participants. One solution is to use static screen shots of their outputs along with popular on line surveying tools, but such a solution omits any system interactivity. Alternatively they can create their own surveying application on a chosen platform. We introduce VisSurvey.js, a JavaScript library that helps developers create evaluation studies. Developers can create user studies of their application in a web browser, and easily capture the results.

Keywords: Evaluation, visualisation, JavaScript Library.

Index Terms: H.5.2 [Interfaces and Presentation]: User Interfaces—Graphical User interfaces (GUI). H.5.2 [Interfaces and Presentation]: User Interfaces—Evaluation/methodology.

1 INTRODUCTION

Researchers often wish to conduct user studies to evaluate the effectiveness of their tools. They are pragmatic in their decision making, and will choose a tool that they can easily applied for and to their situation. Therefore, researchers need tools to help them create surveys and evaluate their system in the quickest and easiest way. Consequently, many modern studies are conducted using on-line surveys such as Google Forms and Survey Monkey. While these systems are simple to use, and are approachable whatever level of technical competence of the researcher, there are two important challenges.:

It is difficult to control how the data is captured and where it is stored. Many online evaluation systems store the data in a separate spreadsheet or remote file. The data within the files are then extracted and analysed, after the evaluation has completed. It would be better to dynamically capture the data, and provide methods to integrate the data with the analysis directly.

It is difficult to control how the images are presented to the participants. Usually the researcher needs to create static images (or screenshots) of their tool and load them into the questionnaire. It would be better to integrate the actual output of the tool into the evaluation. This not only allows the actual tool to be evaluated, but saves time in creating screenshots. In addition, the data could also be used by the visualisation itself, where participants' answers could affect what types of questions are asked.

In this short paper, we introduce VisSurvey.js, a web based library that is used to create on-line user studies of interactive systems. While the library is designed to evaluate visualisation research, it has the potential to display any web-based output. With VisSurvey.js, researchers can readily construct comparative surveys, such to perform JND experiments, the developer can control how the data is captured, and directly render output from a visualisation tool.

*e-mail: j.r.jackson@bangor.ac.uk
†e-mail: j.c.roberts@bangor.ac.uk

IEEE VIS 2017
October 1–6, Phoenix, Arizona, USA

2 BACKGROUND & RELATED WORK

There is a wide range of techniques and methods used to evaluate visualisations [2, 3], ranging from paper-based data collection and compute-based surveys, to evaluating final tools using talk aloud sessions [4] and perceptual studies that compare different images. Because evaluation is widely applied, many tools and products have been created to help researchers capture user sessions and ask post-test questions. E.g., OvoLogger, Morae and Camtasia, Userlytics capture screen interactins, while other tools track clicks, and datalog [7] user actions, such to evaluate usability. Our focus is, however, to focus on user perception and judgement tasks. There are many studies whereby the participant is asked to compare or judge which design is best, such to evaluate which picture is more aesthetically pleasing. Pairs of different pictures are shown to the user and the user selects the best, or they grade the image on a Likert scale. By changing the parameters of the design, and running through many hundreds of versions, the researcher can conclude which parameters control the result to be significantly different.

Unfortunately, the current practice of creating evaluation questionnaires is often separate from the creation of the visualisations (or visualisation tools) themselves. Practically, most researchers will take screenshots and set-up a questionnaire that is separate to their visualization application they are evaluating. For instance, recently Turton et al. [6] present a JavaScript toolkit to help researchers more easily create surveys of different images. Certainly products such as Amazon's Mechanical Turk or SurveyMonkey can be easily used to deliver the questionnaire to a wide quantity of participants. But, the researcher who creates the survey must construct all of the questions in advance, and each participant is presented with the same information. It is difficult for the researcher to introduce random questions, or customise the survey based previous user answers.

What is required is a system that will allow the researcher to embed any interactive visualization tool, determine where the visualisations are placed on the screen, keep record of who has seen each picture, and store the results of any questions, which can be output into a file or connected to a database. The system could be used by researchers or students [5]. For our solution we focus on developing a JavaScript library. The advantage of JavaScript is that can be easily extended, many visualisations are already created for the web, use JavaScript (such as D3.js), and it can be readily displayed on desktop and mobile platforms.

3 VISSURVEY.JS

Our overarching vision is to create a lightweight, cross platform library to facilitate researchers to develop dynamic evaluations and surveys. A further goal is to use modern web technologies such that any visualisation tool can be integrated into the survey, and be operational within the evaluation. The ease of use is demonstrated in Listing 1, which has three main parts: setup, the list of items to compare, and the question format.

To use the system a developer would need to install the

JavaScript library, by cloning the GitHub[1] repository or by copying the necessary files into a project. The next stage is to define an appropriate JavaScript object to describe the settings for the application, a truncated example is shown in Listing 1.

The options object allows the developer to provide three key functions: `itemFunction` is the code which renders the visualization (e.g., this could be to load an `img` tag or embed a Java Applet), `saveItemDataFunction` dictates how, where and when data is stored. `selectionFunction` describes how the questions are presented to the participant, and what types of questions are used (such as a choice, or a scale). For each of these key functions we provide a default solution should the developer decide not to write their own custom functions. This allows a researcher to quickly start using the library, that loads the images from a repository, and displays the images at random.

```
var settings = {
  setup:{
    itemFunction: loadImageFunction,
    saveItemDataFunction: saveDataPointFunction,
    selectionFunction: selectionFunction  },
  items: [
    { src: "images/01.png" },
    { src: "images/02.png" },
    { src: "images/03.png" }, ... ,  ],
  questionFormat: {
    numberPerQuestion: 2,
    question: "Which visualisation is better?",
    options: [
      { name: "A", value: 0 },
      { name: "B", value: 1 }  ]
  }
};
```

Listing 1: settingsExample.js

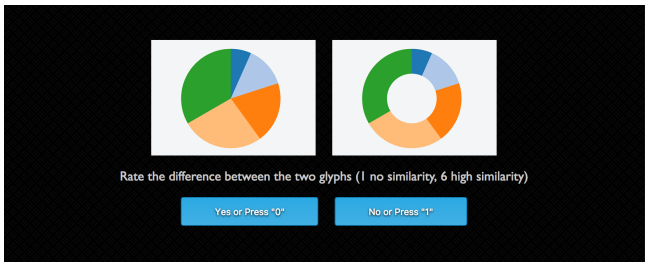

Fig. 1: Screenshot of VisSurvey, showing two images in a visualisation survey.

The developer would then create an HTML application which would use the survey. The library includes its own event listeners and builds its core HTML so a developer can run the survey by invoking the `loadSurvey(options)` function. This can be built using responsive development principles to allow for a better mobile user experience.

The system is designed in such a way that the developer is not tied to a single output type. By default a JSON file is built as the participant completes the evaluation. The JS object is then pushed to a Firebase database. It is also possible to persist data for each survey question. This data acquisition method can be overridden by a developer supplied function. By providing this option, we are enable to supply a robust and inclusive system. The overridden method could, for example, be used to make an AJAX request to a server and store object values into an SQL database. The overridable function returns the object that is being stored. This allows the possibility of instant visual confirmation of results to the user. Applications for this returned data could include browser-based visualisations (such as D3.js or Google Charts) or parameters into the visualisation algorithm itself.

[1] https://github.com/jamesjacko/visSurvey

4 Discussion & Conclusions

The library has been tested with both Canvas-based visualisations and static images. Due to the fact that it is browser-based it can be quickly shared on social media, and uptake of surveys can be large. As the system is developed on standard web technologies, using Semantic UI, it is possible to present the evaluation on mobile devices. Because modern browser API's record the device type, allowing researchers to isolate results from mobile only browsers. In addition, device information could be further harnessed in the option selection process.

Potentially different types of visualisations could be displayed (not only images or JavaScript visualisation tools such as D3.js). For example, it would be possible to display Java based tools, by embedding Java applet within the presentation function. However it may not be possible to embed all types of tools and output. In such cases the researcher would need to take screenshots and include images to be evaluated. We believe that these cases are very much fringe cases and that our system would allow the fall back to traditional exported image based surveys.

Further development of the code would allow for the integration of the acquired data into the visualisation selection process itself. We envisage a situation where the survey presents different visual depictions based on previous answers to the survey, or the data could be fed into the visualization program, which also would be useful to evaluate genetic algorithms in the Artificial Intelligence domain [1]. We would like to further explore the mobile visualisation domain and get the library working on wearable devices. Our ideas extend to being able to conduct surveys on devices such as Apple Watches or Google Wear devices. This would place our system in a unique position where researchers are able to quickly and easily conduct surveys on target devices with little extra development required.

5 Acknowledgements

This work is funded by KESS 2. Knowledge Economy Skills Scholarships (KESS 2) is a pan-Wales higher level skills initiative led by Bangor University on behalf of the HE sector in Wales. It is part funded by the Welsh Governments European Social Fund (ESF) convergence programme for West Wales and the Valleys.

References

[1] G. I. Henshall, C. J. Headleand, W. J. Teahan, and L. Ap Cenydd. Towards crowd-sourced parameter optimisation for procedural animation. In *Cyberworlds 2015*, pages 379–381. IEEE, 2015.

[2] T. Isenberg, P. Isenberg, J. Chen, M. Sedlmair, and T. Möller. A systematic review on the practice of evaluating visualization. *IEEE Trans. Vis. Comput. Graphics*, 19(12):2818–2827, 2013.

[3] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Trans. Vis. Comput. Graphics*, 18(9):1520–1536, Sept 2012.

[4] C. North. Toward measuring visualization insight. *IEEE Comput. Graph. Appl.*, 26(3):6–9, 2006.

[5] J. C. Roberts, P. D. Ritsos, J. R. Jackson, and C. Headleand. The Explanatory Visualization Framework: An active learning framework for teaching creative computing using explanatory visualizations. *IEEE Trans. Vis. and Comp. Graph. (InfoVis'17)*, Jan. 2018. To appear.

[6] T. L. Turton, A. S. Berres, D. H. Rogers, and J. Ahrens. ETK: An Evaluation Toolkit for Visualization User Studies. In *EuroVis 2017 - Short Papers*. The Eurographics Association, 2017.

[7] S. J. Westerman, S. Hambly, C. Alder, C. W. Wyatt-Millington, N. M. Shryane, C. M. Crawshaw, and G. R. J. Hockey. Investigating the human-computer interface using the datalogger. *Behavior Research Methods, Instruments, & Computers*, 28(4):603–606, 1996.