A tool to help lay out Multiple View Visualisations guided by view analysis

Hayder M. Al-maneea[1,2] and Jonathan C. Roberts[1]
[1]Bangor University, [2]University of Basrah

EuroVis 2020 Eurographics Eurovis 2020

ABSTRACT

In this poster, we present 'Layouts for Multiple View' (LMV), a tool that helps users build, control and save multiple view visualisations simply and easily using a bespoke grammar.

The tool incorporates template multiple view layout strategies as quantified from prior research on view analysis, and the user can build different layouts by defining the grammar, or through the linked visual interface.

LMV saves the multiple view layout as a JSON file, including all the details of the layout and attributes of the visualisation, which can be subsequently loaded and adapted or used to create dashboards.

LMV guides the user to:
(i) Design and control the multiple view layout.
(ii) Add data and allocate a specific visualisation technique for each view.
(iii) Adapt specific appearance properties of the layout.

CONTACTS

<Hayder M. Al-maneea>
Email: h.m.almaneea@bangor.ac.uk

<Jonathan C. Roberts>
Email: j.c.roberts@bangor.ac.uk

INTRODUCTION

Visualization tools, libraries and systems all help users create visualisations. While there are many ways to create a visualisation, controlling the layout of multiple view systems is still difficult. This is not the case for websites, where there are many design tools to help users lay out their websites.

Why can we not have the same idea in visualisation? Visualisation developers would likewise benefit from a similar system. Templates to help users follow 'typical' layout strategies, and methods to graphically design different layouts.

At the start of our research, we asked several questions: How can we develop a system to help visualisation users lay out their views? How can we allow users to quickly lay out their views, and then easily change their design? How do we map data to a view and easily change the appearance of the layout viewer?

Our motivation is to give the users the ability to create juxtaposed view layouts in a simple and easy way. We also wanted to allow users to create 'typical' layouts, and consequently we drew on recent work by Almaneea and Roberts [1,2], on quantifying and identifying typical and frequent layout strategies.

TOOL DESIGN

Our tool has four main view panels, each addressing a different task: template viewer, grammar panel, visualisation panel, and property panel. The template viewer (shown in Figure 1) is the default first view, and allows a pre-built design to be selected. Users can also search and filter designs. After selecting a starting template (which could be blank) the three-part viewer opens.

The second screen has three main parts (Figure 2), the grammar panel, visualisation editor, and property panel. The grammar panel allows users to edit a language description of the layout. The visualisation editor shows either a wireframe layout editor (without visualisations), or the visualisation view.

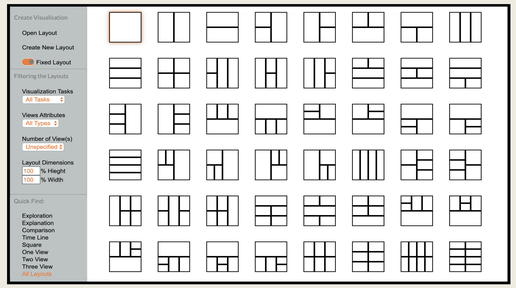

Figure 1: 'Layouts for Multiple View' (LMV) tool. First screen, Template viewer. Users can choose a starting layout, search for a specific view quantity. They can later edit the template.

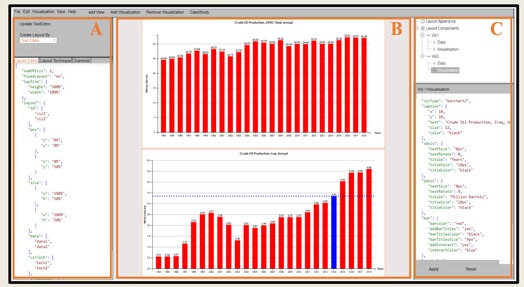

Figure 2: LMV tool, with three linked views: (A) grammar panel to edit the grammar (either shorthand e.g., v(50,50) or full JSON, shown), (B) Visualisation panel of either the wireframe editor or visualisation editor (shown) and (C) property panel.

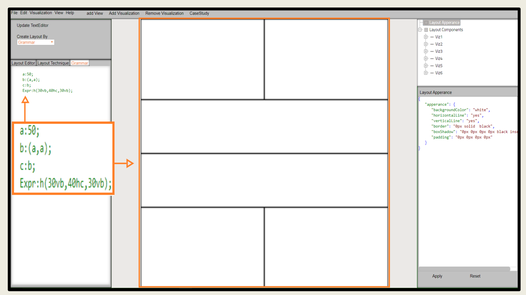

Figure 3: The grammar panel, the basic idea is based on hierarchical cuts. Cut one view horizontally (h) or vertically (v) to produce two views, and so on. E.g., h(50,50) creates an equal sized side-by-side view. The value 50 is representative, h(20;20) would provide the same result. Complex cuts can be easily created, e.g., h(50v(75,25),50) creates layout with three views, a long bottom view, with the top split 75% across.

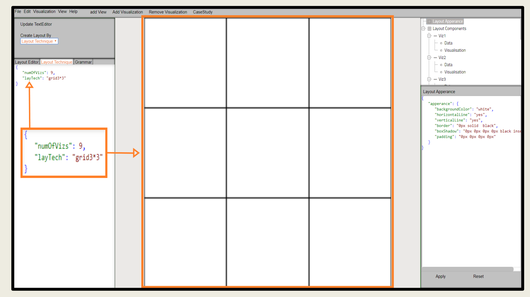

Figure 4: Layout-Technique can quickly define a nine-grid view ("laytech":"Grid3*3") or place six views in a golden ratio with a centre in the fourth quarter ("laytech":"GoldenRatioV6Q4").

The wireframe editor allows users to add, delete and change the size and the position of the views, and snap wireframe views together.

The grammar description is dynamically updated on the left panel, allowing users to jump between grammar or layout descriptions. The property panel allows users to change the appearance of the layout (border width, background colour, etc.) and how the visualisations and data are mapped to panels.

Developers can fill in any visualisation technology they require, for instance, a Google Map view, D3 view, highcharts view, etc. E.g., the example in Figure 2 uses D3.

DISCUSSION AND CONCLUSIONS

We have designed and built a tool to help users lay out multiple view systems. Users an select a template-layout, or edit the layout visually or control the layout through a grammar (Figures 3, 4).

The grammar is used to save/load view layouts, and each view is linked, such when the user controls the wireframe editor the grammar updates.

We have developed and improved the prototypes with heuristic feedback. We have demonstrated its use with D3, and it has potential to be used with other visualisation languages and tools.

We are still developing and improving LMV, and are planning a more in-depth user evaluation.

REFERENCES

1. AL-MANEEA H. M., ROBERTS J. C.: Study of Multiple View Layout Strategies in Visualisation. In Posters presented at the IEEE Conference on Visualization (IEEE VIS 2018), Berlin, Germany (Oct. 2018). URL:http://ieeevis.org/year/2018/welcome.
2. AL-MANEEA H. M., ROBERTS J. C.: Towards quantifying multiple view layouts in visualisation as seen from research publications. In 2019 IEEE Visualization Conference (VIS) (Oct 2019), pp. 121–121. doi:10.1109/VISUAL.2019.8933655.