Code Cities in Virtual and Augmented Reality

Rainer Koschke und Marcel Steinbeck
Universität Bremen
koschke@uni-bremen.de marcel@informatik.uni-bremen.de

Abstract

In this paper, we summarize existing research on visualizing software as code cities with a focus on virtual and augmented reality technology (VR/AR). The review of existing knowledge shows that several of the hopes in VR/AR have not been fulfilled yet. A possible reason for that may be that existing means of visualization and interaction cannot simply be transferred one-to-one from desktop to VR/AR environments as our own research suggests. Moreover, we view the true potential of VR/AR in supporting the communication among multiple developers. All studies we are aware of have had a single developer trying to solve a given task. VR/AR can bridge spatial gaps and more research should be targeted at that particular advantage for distributed teams.

1 Code-City Visualization

Maintaining software systems is a challenging tasks due to their complexity and versatility. In order to assist developers in understanding large scale systems, tools for (semi-)automatic software analysis have been developed in the last decades. However, processing analysis results, in turn, became difficult because a large amount of data accumulates. Software visualization tries to close the gap between data collection and data evaluation by mapping software-related attributes to visual components—leveraging the human ability to recognize patterns in visual data.

In the attempt to assist developers in understanding complex software systems, several analysis tools have been developed. Results gathered by such tools can be modeled by different kinds of graphs, allowing to express, for instance, the hierarchical structure of a software system and dependencies between elements. These graphs are the basis for many visualizations [17]. Due to the variety of the usage scenarios and requirements, many kinds of visualization techniques exist. For a broader overview of software visualization, we refer the reader to more comprehensive surveys [5, 40, 2, 1, 17, 6, 8, 9, 23].

In this paper, we focus on software visualization based on the code-city metaphor, originally introduced by Wettel and Lanza [41, 42, 43], which can be viewed as three-dimensional *Treemaps*. The idea of treemapping is to recursively subdivide a rectangular shape into subareas according to the hierarchy of the data that is to be visualized [14], for example, source code files in directories. The size of the resultant leaf areas visually encodes a given property, for instance, lines of code, complexity, or change frequency. This allows to compare elements and identify peculiarities. By applying a color or texture, an additional property can be encoded, so that hotspots regarding this property can be determined easily. Utilizing 3D space, yet another property can be depicted by mapping its value onto the height of the leaves, leading to three-dimensional blocks. Due to perceiving these blocks as buildings of a city, this technique is known as *CodeCity*

Treemaps and *CodeCities* are designed to make optimal use of the available space. However, the generated layout is not flexible enough to visualize the evolution of a software system (elements need to be added, removed, and relocated), leading to the issue that a city's structure can change drastically from one version to another—although it must be mentioned that researchers have started to work on that particular problem recently [33]. Furthermore, the compact layout offers very little distinct patterns and, thus, hinders humans in recollecting visited places. To overcome these shortcomings, Steinbrückner and Lewerentz [39, 38] proposed *EvoStreets* as an alternative visualization technique in which a software's hierarchy is depicted using road junctions rather than subdivided areas. Each level of the hierarchy is mapped onto a street whose width visually encodes the nesting level—the lower the level of the hierarchy is, the thinner the corresponding street gets. Similar to *CodeCities*, leaves are represented as three-dimensional blocks. *EvoStreets*, on the one hand, require more space but, one the other hand, allow parts of the city to grow and shrink without effecting the entire layout, which helps in maintaining a beholder's mental map.

Along with the hierarchical structure of a software, relations between elements may be of interest. A common technique to visualize relations in *Treemaps*, *CodeCities*, and *EvoStreets* is to connect the corresponding areas (*Treemaps*) or blocks (*CodeCity* and *EvoStreets*) with edges. Yet, if drawn as straight lines, edges easily create visual clutter due to crossing each other. Holten proposed hierarchical edge bundling, an approach which reduces some of the clutter by drawing edges as B-Splines [12, 13]. The location of the

control points of the B-Splines is based on the hierarchical structure of the visualized elements. By sharing control points among elements with similar nesting, edges with similar ends are bundled analogously to a cable tie.

Interestingly, many research groups exploring *CodeCities* are in fact located in Germany: University of Cotbus, where *EvoStreets* were invented [39, 38], Hasso-Plattner Institute in Potsdam, where many improvements to *CodeCities* were proposed and explored [18], University of Kiel, who are among the German pioneers of using *CodeCities* in virtual reality (VR) [11], University of Stuttgart, where the particular effects of using VR for *CodeCities* are researched [22, 20, 24, 21], and our own research group, who currently explores orientation and navigation behaviour of developers exploring *CodeCities* in VR [32, 35, 36].

2 Code-Cities in VR/AR

Several implementations of *CodeCities* and *EvoStreets* exist for two- and three-dimensional (also referred to as 2D and 2.5D) rendering on regular two-dimensional displays. Recently, in the attempt to continuously improve these techniques and in the hope that advantages observed outside of software engineering [7] also apply to software visualization, researchers begun to develop systems that make use of immersive three-dimensional virtual realty. Visualization techniques using 2.5D and VR have been explored for quite some time. As early as 2000, Knight and Munro gave an overview of software visualization in VR [16]. Since then, 2.5D and VR environments have been used to visualize static [4, 19, 26, 27, 11, 15, 21, 34] as well as dynamic information [25, 10, 24]. There is already a great body of knowledge on 2.5D and VR visualization outside of computer science [3, 29, 28]. Studies have shown that head-mounted displays (HMDs) may have a positive effect on the orientation of human beholders in three-dimensional environments [7]. Sousa Santos et al., on the other hand, found in an experiment on navigation that, although being generally satisfied with VR, participants performed actually better in the desktop environment. Ruddle et al. studied the navigation in computer-simulated worlds using HMDs and regular desktop environments [30, 31]. In their first experiment, where participants had to navigate through several rooms within virtual buildings [30], they found that the HMD increased the speed of the participants. In their second experiment, Ruddle et al. looked into proprioceptive feedback and its influence on navigation within a virtual maze [31]. According to their findings, viewing the mazes using an HMD had little effect. The conflicting results of Sousa Santos et al.'s and Ruddle et al.'s experiment is a subject of further research.

Our research group conducted experiments in which we compared time and correctness of solving tasks between the 2D, 2.5D and VR representations of *CodeCities* [35]. We could not find enough evidence that task performance is different between the three environments. Yet, we found in a follow up study [36] that the path length (the distance one moves within *EvoStreets*), average speed (the length of a path put in relation to the time that was required to move along this path), and occupied volume (the convex hull of the movement trajectory) differ significantly between the 2.5D and VR environments for some of the tasks, indicating that movement is less extensive in VR. That being said, more research on how human beholders interact with *EvoStreets* in different environments is necessary, to identify visualization and user interaction concepts that adapt to the characteristics of a particular environment. Due to the design of our experiment, that is, all environments were using the same visualization engine [32] and applied the same visual mappings to the components of the *EvoStreets*, the threat that differences observed between the 2.5D and VR environments result from different *EvoStreets* implementations is minimized. Thus, we suspect that insights into differences specific to these environments, if there are any, can be gathered by analyzing the available data in more details. That is why we analyzed the results of this experiment in more details in another paper [37], to study if not only movement is affected by these environments, but also the way how *EvoStreets* are observed. Although we could not find enough evidence that the number of viewpoints and their duration differ significantly, we found indications that in virtual reality viewpoints are located closer to the *EvoStreets* and that the distance between viewpoints is shorter.

3 Conclusions

The above review of existing knowledge shows that several of the hopes in VR/AR have not been fulfilled yet. One reason for that may be that the particular characteristics of VR/AR have not been recognized and, thus, been neglected in the design of visualization techniques and means of interaction. Our own research suggests, for instance, that orientation and navigation behaviour differs between 2D and 2.5D versus VR. In order to develop techniques adapting to the circumstance of a particular environment, more research in this field is necessary. Moreover, we view the true potential of VR/AR in supporting the communication among multiple developers. All studies we are aware of have had a single developer trying to solve a given task. VR/AR can bridge spatial gaps and more development and research should be targeted at allowing developers to explore software data collaboratively. Many organizations work in spatially distributed teams and existing technology for screen-sharing and video conferencing have their limitations when it comes to viewing software together.

References

[1] S. Bassi and R. K. Keller. Software visualization tools: Survey and analysis. In *International Workshop on Program Comprehension*, pages 7–17. IEEE, 2001.

[2] S. Bassil and R. K. Keller. A qualitative and quantitative evaluation of software visualization tools. In *Proceedings of the Workshop on Software Visualization*, pages 33–37. IEEE, 2001.

[3] D. A. Bowman, E. T. Davis, L. F. Hodges, and A. N. Badre. Maintaining spatial orientation during travel in an immersive virtual environment. *Presence: Teleoper. Virtual Environ.*, 8(6):618–631, 1999.

[4] N. Capece, U. Erra, S. Romano, and G. Scanniello. Visualising a software system as a city through virtual reality. In L. T. De Paolis, P. Bourdot, and A. Mongelli, editors, *Augmented Reality, Virtual Reality, and Computer Graphics*, pages 319–327, Cham, 2017. Springer International Publishing.

[5] S. Carpendale and Y. Ghanam. A survey paper on software architecture visualization. Technical Report 2008-906-19, University of Calgary, 2008.

[6] P. Caserta and O. Zendra. Visualization of the static aspects of software: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 17(7):913–933, 2011.

[7] S. S. Chance, F. Gaunet, A. C. Beall, and J. M. Loomis. Locomotion mode affects the updating of objects encountered during travel: The contribution of vestibular and proprioceptive inputs to path integration. *Presence: Teleoper. Virtual Environ.*, 7(2):168–178, 1998.

[8] S. Diehl, editor. *Software Visualization: International Seminar Dagstuhl Castle, Germany*. Lecture Notes in Computer Science. Springer, 2001.

[9] S. Diehl. *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*. Springer, 2010.

[10] F. Fernandes, C. S. Rodrigues, and C. Werner. Dynamic analysis of software systems through virtual reality. In *Symposium on Virtual and Augmented Reality*, pages 331–340, Nov. 2017. In Spanish.

[11] F. Fittkau, A. Krause, and W. Hasselbring. Exploring software cities in virtual reality. In *Working Conference on Software Visualization*, pages 130–134. IEEE, 2015.

[12] D. H. R. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, Sep. 2006.

[13] D. H. R. Holten. *Visualization of graphs and trees for software analysis*. PhD thesis, Technical University of Delft, 2009.

[14] B. Johnson and B. Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the Conference on Visualization*, pages 284–291. IEEE Computer Society Press, 1991.

[15] P. Khaloo, M. Maghoumi, E. Taranta, D. Bettner, and J. Laviola. Code park: A new 3d code visualization tool. In *Working Conference on Software Visualization*, pages 43–53. IEEE, 2017.

[16] C. Knight and M. Munro. Virtual but visible software. In *Information Visualization, 2000. Proceedings. IEEE International Conference on*, pages 198–205. IEEE, 2000.

[17] R. Koschke. Software visualization in software maintenance, reverse engineering, and re-engineering: a research survey. *Journal on Software Maintenance and Evolution*, 15(2):87–109, 2003.

[18] D. Limberger, W. Scheibel, J. Döllner, and M. Trapp. Advanced visual metaphors and techniques for software maps. In *International Symposium on Visual Information Communication and Interaction*, pages 1–8, Sept. 2019.

[19] J. I. Maletic, J. Leigh, A. Marcus, and G. Dunlap. Visualizing object-oriented software in virtual reality. In *International Workshop on Program Comprehension*, pages 26–35, May 2001.

[20] L. Merino, A. Bergel, and O. Nierstrasz. Overcoming issues of 3d software visualization through immersive augmented reality. In *Working Conference on Software Visualization*, pages 54–64. IEEE Computer Society Press, 2018.

[21] L. Merino, J. Fuchs, M. Blumenschein, C. Anslow, M. Ghafari, O. Nierstrasz, M. Behrisch, and D. A. Keim. On the impact of the medium in the effectiveness of 3d software visualizations. In *Working Conference on Software Visualization*, pages 11–21. IEEE, 2017.

[22] L. Merino, M. Ghafari, C. Anslow, and O. Nierstrasz. Cityvr: Gameful software visualization. In *IEEE International Conference on Software Maintenance and Evolution (TD Track)*, pages 633–637, 2017.

[23] L. Merino, M. Ghafari, C. Anslow, and O. Nierstrasz. A systematic literature review of software visualization evaluation. *Journal of Systems and Software*, 144:165–180, 2018.

[24] L. Merino, M. Hess, A. Bergel, O. Nierstrasz, and D. Weiskopf. Perfvis: Pervasive visualization in immersive augmented reality for performance awareness. In *ACM/SPEC International Conference on Performance Engineering*, pages 13–16. ACM Press, 2019.

[25] K. Ogami, R. G. Kula, H. Hata, T. Ishio, and K. Matsumoto. Using high-rising cities to visualize performance in real-time. In *Working Conference on Software Visualization*, pages 33–42. IEEE, 2017.

[26] T. Panas, R. Berrigan, and J. Grundy. A 3d metaphor for software production visualization. In *International Conference Information Visualization*, pages 314–319. IEEE, 2003.

[27] T. Panas, T. Epperly, D. Quinlan, A. Saebjornsen, and R. Vuduc. Communicating software architecture using a unified single-view visualization. In *Engineering Complex Computer Systems, 2007. 12th IEEE International Conference on*, pages 217–228. IEEE, 2007.

[28] J. W. Regian, W. L. Shebilske, and J. M. Monk. Virtual reality: An instructional medium for visual-spatial tasks. *Journal of Communication*, 42(4):136–149, 1992.

[29] B. E. Riecke, D. W. Cunningham, and H. H. Bülthoff. Spatial updating in virtual reality: the sufficiency of visual information. *Psychological Research*, 71(3):298–313, May 2007.

[30] R. A. Ruddle, S. J. Payne, and D. M. Jones. Navigating large-scale virtual environments: what differences occur between helmet-mounted and desktop displays? *Presence: Teleoperators & Virtual Environments*, 8(2):157–168, 1999.

[31] R. A. Ruddle and P. Péruch. Effects of proprioceptive feedback and environmental characteristics on spatial learning in virtual environments. *International Journal of Human-Computer Studies*, 60(3):299 – 326, 2004.

[32] M. Rüdel, J. Ganser, and R. Koschke. A controlled experiment on spatial orientation in vr-based software cities. In *Working Conference on Software Visualization*, pages 21–31, Sept. 2018.

[33] W. Scheibel, C. Weyand, and J. Döllner. Evocells - A treemap layout algorithm for evolving tree data. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 273–280, 2018.

[34] A. Schreiber and M. Brüggemann. Interactive visualization of software components with virtual reality headsets. In *Working Conference on Software Visualization*, pages 119–123. IEEE, 2017.

[35] M. Steinbeck, R. Koschke, and M.-O. Rüdel. Comparing the evostreet visualization technique in two- and three-dimensional environments—a controlled experiment. In *International Conference on Program Comprehension*, pages 231–242. IEEE Computer Society Press, 2019.

[36] M. Steinbeck, R. Koschke, and M.-O. Rüdel. Movement patterns and trajectories in three-dimensional software visualization. In *Conference on Source Code Analysis and Manipulation*, pages 163–174, Sept. 2019.

[37] M. Steinbeck, R. Koschke, and M.-O. Rüdel. How evostreets are observed in three-dimensional and virtual reality environments. In *IEEE International Conference on Software Analysis, Evolution and Reengineering*, page 12 pages. IEEE Computer Society Press, Feb. 2020. to appear.

[38] F. Steinbrückner. *Consistent Software Cities: supporting comprehension of evolving software systems.* PhD thesis, Brandenburgische Technische Universität Cottbus, 06 2013.

[39] F. Steinbrückner and C. Lewerentz. Representing development history in software cities. In *ACM International Symposium on Software Visualization*, pages 193–202. ACM, 2010.

[40] A. R. Teyseyre and M. R. Campo. An overview of 3d software visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):87–105, 2009.

[41] R. Wettel and M. Lanza. Visualizing software systems as cities. In *2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, pages 92–99, June 2007.

[42] R. Wettel and M. Lanza. Codecity: 3d visualization of large-scale software. In *Companion of the 30th International Conference on Software Engineering*, ICSE Companion '08, pages 921–922, New York, NY, USA, 2008. ACM.

[43] R. Wettel and M. Lanza. Visual exploration of large-scale system evolution. In *2008 15th Working Conference on Reverse Engineering*, pages 219–228, Oct 2008.