Scientific Visualization System on a Chip with Tangible User Interface*

Konstantin Ryabinin[1][0000−0002−8353−7641] and
Mariia Kolesnik[2][0000−0002−4424−4723]

[1] Perm State University, Bukireva Str. 15, 614990, Perm, Russia
kostya.ryabinin@gmail.com
[2] Perm Regional Museum / branch Museum of Permian Antiquities, Monastyrskaya Str. 11, 614000, Perm, Russia
kolesnik.ma@outlook.com

Abstract. This paper is devoted to the development of the ontology-driven standalone scientific visualization station based on a single-board microcomputer with custom tangible user interface. Such a station can be used as a powerful demonstration tool in various scenarios including interactive museum exhibitions. According to the approach proposed, the particular instance of a software scientific visualization system is generated automatically by a high-level platform SciVi that was been developed earlier. Previously, ontology-driven software generation mechanisms within SciVi were tested on the firmware generation for the microcontroller units. Currently we present a generalization of this technique to the case of systems on chips like Raspberry Pi or Orange Pi. Data preprocessing and rendering capabilities of SciVi are reused without modifications from the previous stages of development, while the new mechanisms of taking into account the specifics of systems on chips software and hardware organization are introduced via extending the appropriate SciVi ontologies. The generalized technique is tested in practice by creating a set of interactive museum items for the "Transmutations" exhibition within Kidsmuseum, branch of Perm Regional Museum.

Keywords: Scientific Visualization, System on a Chip, Tangible User Interface, Smart Museum, Ontology Engineering.

1 Introduction

Particular tasks of presenting scientific data often require standalone scientific visualization stations, which are equipped with the monitor, rendering system and custom user interface. Normally, regular computers are sufficient for this, however sometimes specific circumstances prevent this traditional scenario. One of these specific cases are interactive mobile museum exhibitions within a Smart Museum concept [13]. The visualization means used in these exhibitions should meet the following requirements:

* Publication is supported by RFBR grant 20-01-00358.

1. Fast and easy building (visualization and its user interface should be modular, comprising interoperable building blocks).
2. Fast and easy deploy (mobile museum exhibitions have to be installed, reconfigured and uninstalled as fast as possible).
3. Fast and easy startup and shutdown (ideally it should be enough to switching on / off the power to start and stop the entire hardware and software system).
4. Standalone functioning (there should be no dependencies on the computing infrastructure like external Internet access, local network requirements, etc.).
5. Intuitive interface (visualization interactions should be obvious even for the technically inexperienced visitors).
6. Low cost (mobile museum exhibitions often have limited budget).
7. Exchangeable parts (in case of hardware failure, the modules should be easy testable to find the broken part, and any part should be well-spread enough to replace it as soon as possible).

Traditional computers do not meet at least the criterion (6). The criterion (3) is met only partially, since shutting the computer down just by cutting off the power can harm the data on the disks and cause severe system failure. But this criterion is indeed important, since preparing the exhibition in the mornings and shutting everything down in the evenings are the tasks of the museum hall keepers, who need to do it as fast as possible, and it would be problematic for them to perform special operations on individual exhibits.

To meet all the mentioned criteria at once, we propose using systems on chips (SoCs).

SoCs became a popular form of portable computers [10], which are often utilized as fog nodes [14] within ubiquitous computing ecosystems [11]. They are small single-board computing modules with a single processor, comprising both CPU and GPU functionality. Despite small size and low energy consumption, these systems reach fair performance to support multimedia processing [12] and even processing of scientific data [9]. The most popular types of SoCs are smartphones, tablets and microcomputers like Raspberry Pi [2] or Orange Pi [1]. In the present work, we focus on the single-board microcomputers, since they have hardware interfaces leveraging easy integration with sensor-based tangible user interfaces (TUI) [8].

As seen in the literature, SoCs are widely used in the scientific and cultural projects. For example, the popular scientific visualization library VTK is ported to Raspberry Pi [7], enabling high-quality rendering of scientific data in various formats right on the microcomputers. Sometimes, SoCs leverage creation of entire scientific devices [5]. There are also some projects like [15] utilizing Raspberry Pi as a brain of tangible exhibits dedicated for the wide audience including visually impaired people. The other projects make use of microcomputers and microcontrollers to create autonomous tour guide robots [6].

We propose using high-level scientific visualization platform SciVi [17] as a software part and Raspberry / Orange Pi, combined with the touchscreen and TUI, as a hardware part of standalone scientific visualization station. Thereby a particular stack of technologies for building the interactive museum exhibitions is suggested.

The motivation for this work was a need to create several interactive museum items for the "Transmutations" exhibition within Kidsmuseum, branch of Perm Regional Museum. Since it was a real project involved in the Perm Regional Museum, within this work we were able to do both a theoretical research and the tests of the proposed approach on the practical use case. The interactive items were successfully created and integrated in the "Transmutations" exhibition available for visitors in Perm.

2 Background

During the previous research we developed a multiplatform adaptive scientific visualization platform SciVi based on the ontology engineering methods and means that was used in various application domains to present different scientific data in a comprehensive observable form [16]. Next, we proposed a set of technologies to automate creation of custom TUI to solve particular scientific visualization tasks within SciVi [17]. Finally, we applied the developed methods and means in museum application domain to automate creation of interactive exhibits based on scientific visualization and Internet of Things (IoT) [18].

The interactive museum exhibit consists of two main parts: museum item and visitor interface. Museum item is the most important part that has its own value and in practice can be exposed even without interactivity. But the visitor interface spices up the museum item with the new visitors' experience beyond regular watching, allowing not just passively look at the item, but actively communicate with it. Smart Museum concept assumes turning the regular museum exhibits into the interactive ones to bring the museum visiting to the new qualitative level, not just exposing the items, but deeply immersing the visitors into the exhibition.

While creating the museum items (even the multimedia ones) is the task familiar to the museum research workers, assembling and tuning the visitor interface is hard due to the lack of the corresponding high-level tools for that. Therefore interactive exhibits are still an issue in many museums.

SciVi scientific visualization platform provides efficient tools to automate all the main steps of creating the interactive exhibits [18]:

1. Configuring the TUI functionality and integration of TUI with the exhibit.
2. Generating the firmware for the microcontroller units (MCUs) the hardware TUI consists of.
3. Generating the middleware to ensure communication with the hardware TUI.
4. Generating the visualization modules (if needed) controlled by TUI.
5. Calibrating created TUI to ensure its proper functioning.

All the above steps are driven by ontologies, which ensure flexibility and extensibility of SciVi. The configuration step (1) is supported by the built-in data flow diagram (DFD) editor that enables the user to declare particular data processing, visualization and TUI functioning algorithms in an intuitive visual form.

The previous work of creating IoT-based museum exhibits was been focused on utilizing programmable MCUs [18], which (in combination with different sensors and actuators) can be used as a hardware basis of TUI. However the MCUs are unable to

perform data rendering due to the low performance, so the image synthesis should take place elsewhere, for example, on a kiosk with a full-fledged computer, or on the visitors' mobile devices. While this is a viable scenario (that was, for example, implemented in practice in the State Darwin Museum, Moscow, Russia [18]), sometimes having an external rendering device is not an option. First of all, relaying on just the visitors' mobile devices is considered a bad practice, since some of the visitors may have no required gadgets or may be unable to use them in a way desired in an exhibition. Therefore, rendering kiosk appears to be mandatory (just like it has been done in the State Darwin Museum), but it takes additional place and requires appropriate budget, so it is not always affordable. The other problem is the network that is required to maintain connection of the TUI with the kiosk or visitors' mobile devices. To be stable enough, it requires a set of WiFi-routers in the museum, and this, in turn, is not always affordable too.

In the present work we tackle the problem of creating low-budget ***standalone*** visualization system that does not require any additional digital infrastructure around it (no kiosks and no network).

3 SciVi on a Chip

The distinctive feature of the present work is that we propose to merge TUI and rendering hardware into a single device. In fact, we propose building scientific visualization system on a chip equipped with a sensor-based TUI. The main requirement to the chip is a support of graphics rendering hardware acceleration. The most popular single-board computers capable of data processing and rendering are Raspberry Pi and Oragne Pi. While they are very similar, each model has its own features and tuning nuances. But thanks to the ontology-driven architecture, there is no problem in extending SciVi in a way it takes into account platform-dependent particularities during the process of generating needed TUI- and rendering software modules.

On the MCUs, SciVi generates the firmware as a solid binary image comprising all the functionality (and, if necessary, defines fuse-bits to fine-tune the MCU functioning). The features of MCUs and related hardware modules are described in the ontology of electronic components that is a part of SciVi knowledge base [17]. Targeting the SoCs, in general case SciVi should generate a set of programs, which will perform all needed operations within particular SoC taking into account its hardware specifics and its operating system's (OS) features. In this regard, the ontology of electronic components was extended with appropriate knowledge about the SoCs, including their hardware and software distinctiveness.

First, we decided to use Web-based applications as a target software format, because it would have level the differences between Raspberry Pi and Orange Pi. But the testing showed, that the rendering performance of WebGL is insufficient to deliver smooth animations in 3D scenes. The scene associated with the interactive exhibits we created (see the section 4) contains ca. 5 000 vertices connected in ca. 10 000 polygons and is rendered to the resolution of 1024×600 pixels. Google Chromium (the default browser on most OS for SoCs) was able to maintain the refresh rate of just 9 frames per second (FPS). In contrary, the native application written in C++ utilizing OpenGL 3.3 (that

is emulated on top of OpenGL ES 3 on SoC) ensures 16 FPS. Both refresh rates are quite low, so further optimizations of rendering are required. However, in terms of user experience, 16 FPS lay above the border, when the average human starts to recognize fluent motion instead of individual images, while 9 FPS are below that line [4].

The proposed stack of technologies is shown in the fig. 1.

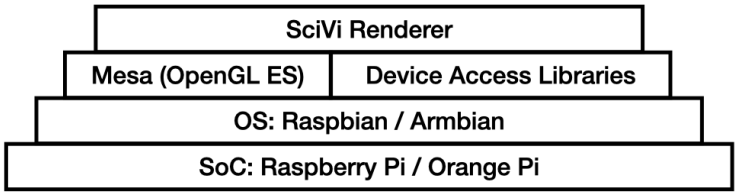

Fig. 1. Stack of technologies used in SciVi on a Chip.

SciVi Renderer is a standalone native application, which source code is generated by SciVi in C++ according to the particular settings made by the user. It utilizes OpenGL graphics rendering API (available on most OS for SoCs via Mesa library) and various device access libraries to receive / transmit data from / to TUI hardware (e. g. sensors and actuators connected via to the SoC via I^2C, SPI, UART, OneWire, etc.).

Significant difference of SoCs compared to MCUs is the presence of full-fledged OS (e. g. Raspbian or Armbian) that, in turn, brings concepts of users, console sessions and permissions. To meet the requirement of fast startup (mentioned in the Introduction), auto-login should be switched on (which is a default behavior of SoCs), as well as the autostart of target application and tuning of the device permissions (for example, to access SPI bus that by default requires root privilege level) should be set up properly.

Fast shutdown is still an issue, because cutting off the power is generally unsafe for the SoCs filesystem. So, ideally hardware shutdown button should be mounted that will cause the execution of `poweroff` command. However, even if the special button cannot be installed and the SoC goes down without involving proper cleanup mechanism, the probability of failure is fairy low in case of visualization system. This is because the visualization system does not perform a lot of disk write operations, so the filesystem normally does not run out of sync and sudden halt should not bring it into the inconsistent state. To further reduce the failure probability, swapping is disabled and root filesystem is mounted in the read-only mode. Should the failure ever happen, it is relatively easy to repair the SoC since all its data are on SD card that, in turn, is easy to back up and restore.

Currently, SciVi automatically generates the source code of the Renderer application along with a configuration shell script to make it easy to build and install the application on the target SoC (via the standard `./configure`, `make` and `make install` command sequence). The SciVi infrastructure allows cross-compiling the software directly for the target platform. This feature is not yet set up for SoCs, but it is a matter of one-time extending the special compiler's ontology and tuning the Docker-container with the needed compilers installed inside it. We plan to do it as an upcoming

development step. It will then be possible to generate an installation package for the SoC's OS, so the deploy will be even easier.

4 Use Case

To test the proposed approach we created a set of museum items for the "Transmutations" exhibition within Kidsmuseum, branch of Perm Regional Museum. The most significant one demonstrates so-called Allen's ecogeographical rule [3]: animals adapted to cold climates have shorter protruding body parts (ears, tails, limbs, etc.) than animals adapted to warm climates. The explanation of this rule is simple: endothermic animals aim to maintain their body temperature at a metabolically favorable level, wherein they should vary the body surface with respect to the outer temperature to either aid or impede their heat dissipation. The longer are protruding body parts, the more is contact with environment, therefore the more is heat dissipation.

The TUI of the exhibit created is shown in the fig. 2. The visitor can choose the latitude with a special handler on the simple Earth model, while the built-in visualizer generates an image of average mammal leaving in the corresponding climate zone. The 3D avatars of "most-polar" and "most-equatorial" animals are shown in the fig. 3. The corresponding key 3D models are created from scratch by one of the paper's authors using Blender 3D editor and stored in PLY format. Both models have the same topology (ca. 5 000 vertices connected into ca. 10 000 triangles) enabling trivial linear morphing between them. The intermediate forms are automatically calculated based on the key models and visualized by SciVi in real time.

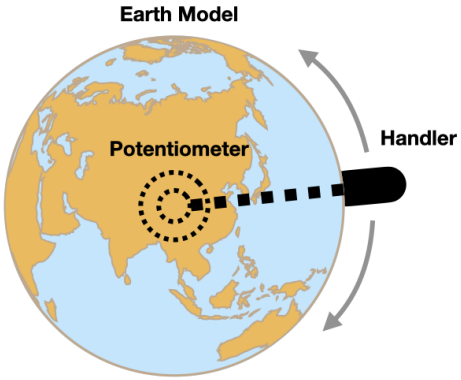

Fig. 2. Schema of the interactive exhibit TUI.

We utilize Orange Pi PC2 as a SoC with Armbian GNU/Linux as its OS, 7 inch 1024×600 pixels touchscreen connected to the SoC via HDMI, external analog-to-digital converter (ADC) MCP3201 connected via SPI (unfortunately, Orange Pi PC2 has no built-in ADC on board) and potentiometer connected to the ADC. The knob of potentiometer is attached to the handler on an Earth model. The total cost of hardware

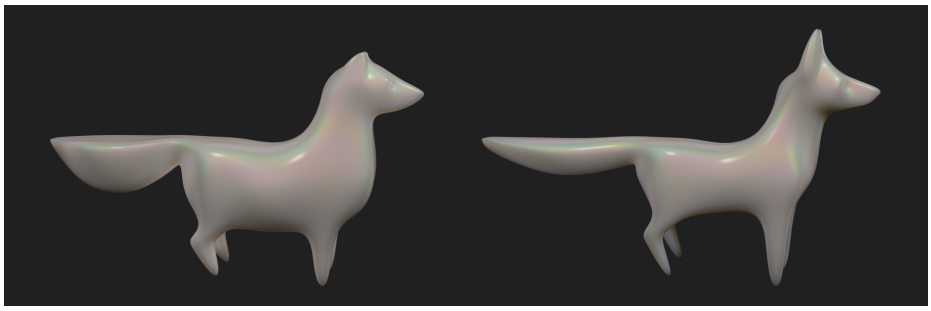

Fig. 3. Extreme forms of the mammal to present Allen's rule: "most-polar" form (left) and "most-equatorial" form (right) rendered by SciVi.

is about $130 (including the delivery of the electronic components), which is fairly low compared to the other options of reaching similar functionality.

SciVi Renderer is configured to poll the ADC and use the obtained value as a transition quotient to morph the animal model from the polar form to the equatorial and back to polar. The morphing result is rendered using the fake global illumination based on the material capture technique. The result image is showed to the visitors via the connected touchscreen, wherein the touch moves are mapped to the orbit camera rotation, so the visitors can change not only the morphing phase, but also the point of view to the 3D model.

5 Conclusion

Previously we created the scientific visualization system SciVi that is able to leverage the creation of interactive museum exhibits based on MCUs and TUI. In the present work we generalized our technique from MCUs to SoCs, which can be used as standalone scientific visualization stations equipped with TUI. Utilizing this generalized technique, we created a set of interactive exhibits based on the Orange Pi PC2 single-board microcomputers, which leverage both maintaining the TUI and rendering the result images. The created interactive museum items are presented in the "Transmutations" exhibition within Kidsmuseum, branch of Perm Regional Museum, supporting the Smart Museum concept within the cultural institutions of Perm Region. Thereby, SciVi system was applied in practice solving a real Smart Museum task.

The future work includes optimizing the rendering mechanism to increase FPS rate and setting up the cross-compilation of the rendering software modules for SoCs within SciVi.

References

1. Orange Pi, http://www.orangepi.org, last accessed 9 Aug 2020
2. Raspberry Pi, https://www.raspberrypi.org, last accessed 9 Aug 2020
3. Allen, J.A.: The Influence of Physical Conditions in the Genesis of Species. Radical Review **1**, 108–140 (1877)
4. Bakaus, P.: The Illusion of Motion (2014), https://paulbakaus.com/tutorials/performance/the-illusion-of-motion/, last accessed 9 Aug 2020
5. Buscher, N., Ojeda, A., Francoeur, M., Hulyalkar, S., Claros, C., Tang, T., Terry, A., Gupta, A., Fakhraei, L., Ramanathan, D.S.: Open-Source Raspberry Pi-Based Operant Box for Translational Behavioral Testing in Rodents. Journal of Neuroscience Methods **342** (2020). https://doi.org/10.1016/j.jneumeth.2020.108761
6. Diallo, A.D., Gobee, S., Durairajah, V.: Autonomous Tour Guide Robot Using Embedded System Control. Procedia Computer Science **76**, 126–133 (2015). https://doi.org/10.1016/j.procs.2015.12.302
7. Ibanez, L.: Raspberry Pi likes VTK (2012), https://blog.kitware.com/raspberry-pi-likes-vtk/, last accessed 9 Aug 2020
8. Ishii, H.: Tangible Bits: Beyond Pixels. In: Proceedings of the 2nd International Conference on Tangible and Embedded Interaction. pp. XV–XXV (2008). https://doi.org/10.1145/1347390.1347392
9. Kent, B.R.: Science and Computing with Raspberry Pi. IOP Concise Physics (2018)
10. Lin, Y.L.S.: Essential Issues in SOC Design: Designing Complex Systems-on-Chip. Springer (2006). https://doi.org/10.1007/1-4020-5352-5
11. Neustein, A.: Advances in Ubiquitous Computing. Elsevier (2020). https://doi.org/10.1016/C2017-0-04641-4
12. Newmarch, J.: Raspberry Pi GPU Audio Video Programming. Apress, Berkeley, CA (2017). https://doi.org/10.1007/978-1-4842-2472-4
13. Piccialli, F., Chianese, A.: Designing a Smart Museum: when Cultural Heritage Joins IoT. In: Eighth International Conference on Next Generation Mobile Apps, Services and Technologies (2014). https://doi.org/10.1109/NGMAST.2014.21
14. Pisani, F., Borin, E.: Fog vs. Cloud Computing: Should I Stay or Should I Go? In: Proceedings of the Workshop on INTelligent Embedded Systems Architectures and Applications. pp. 27–32 (2018). https://doi.org/10.1145/3285017.3285026
15. Rossetti, V., Furfari, F., Leporini, B., Pelagatti, S., Quarta, A.: Enabling Access to Cultural Heritage for the Visually Impaired: an Interactive 3D Model of a Cultural Site. Procedia Computer Science **130**, 383–391 (2018). https://doi.org/10.1016/j.procs.2018.04.057
16. Ryabinin, K., Chuprina, S.: High-Level Toolset for Comprehensive Visual Data Analysis and Model Validation. Procedia Computer Science **108**, 2090–2099 (2017). https://doi.org/10.1016/j.procs.2017.05.050
17. Ryabinin, K., Chuprina, S., Belousov, K.: Ontology-Driven Automation of IoT-Based Human-Machine Interfaces Development. Lecture Notes in Computer Science **11540**, 110–124 (2019). https://doi.org/10.1007/978-3-030-22750-0_9
18. Ryabinin, K., Kolesnik, M., Akhtamzyan, A., Sudarikova, E.: Cyber-Physical Museum Exhibits Based on Additive Technologies, Tangible Interfaces and Scientific Visualization. Scientific Visualization **11**(4), 27–42 (2019). https://doi.org/10.26583/sv.11.4.03